

CS 3750 Machine Learning

Latent variable models Variational approximations.

Milos Hauskrecht
milos@cs.pitt.edu
5329 Sennott Square

CS 2750 Machine Learning

Variational methods

Variational methods have been used in applied math, physics, statistics, control theory, economics.

In this course:

- Used to support approximate inference and estimation when exact methods become hard
- An alternative to Monte Carlo method

Basic idea:

Assume a complex function of \mathbf{x} , $f(\mathbf{x})$

And a simpler function q with a set of parameters λ , $q(\mathbf{x}, \lambda)$

Approximate $f(\mathbf{x})$ as:

$$f(\mathbf{x}) \sim q(\mathbf{x}, \lambda^*)$$

where λ^* are parameters that yield the best approximation

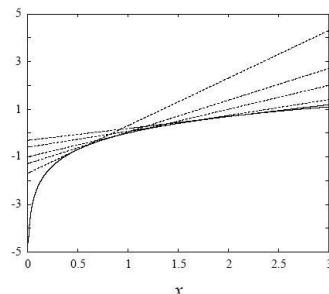
Complex $f(\mathbf{x})$ represented by a simpler $q(\mathbf{x}, \lambda) + \text{optimization}$

CS 2750 Machine Learning

Variational methods

Example:

- Assume a concave function $\ln(x)$
- It can be upper bounded by a ‘simpler’ linear function
$$\ln(x) \leq \lambda x - \ln(\lambda) - 1$$
- λ - a variational parameter that helps to approximate $\ln(x)$.

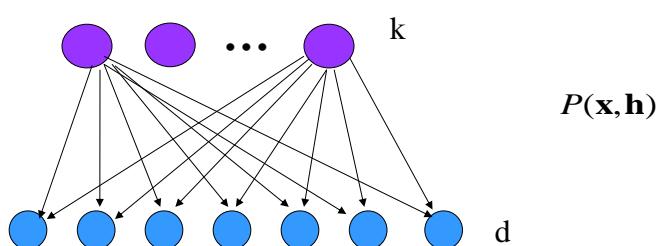


CS 2750 Machine Learning

Variational methods: probabilistic inference

Support approximate inference and estimation

Example: Assume we have a probabilistic model with two sets of variables \mathbf{x} and \mathbf{h}



(1) Assume we want to model $P(\mathbf{h} | \mathbf{x})$ for all configurations of \mathbf{h}

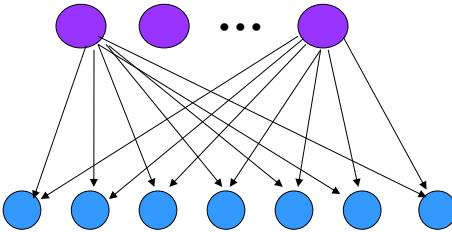
If \mathbf{h} are binary then we need to model/calculate 2^k probabilities

(2) Assume we want to calculate $P(\mathbf{x}) = \sum_{\{\mathbf{h}\}} P(\mathbf{x}, \mathbf{h})$
It requires to sum 2^k terms

CS 2750 Machine Learning

Cooperative vector quantizer

Latent variables (s): binary vars
Dimensionality k



**Observed variables x : real valued vars
Dimensionality d**

CS 2750 Machine Learning

Cooperative vector quantizer

Model:

Latent vars:

\sim Bernoulli distribution
parameter: π_i

$$P(s_i | \pi_i) = \pi_i^{s_i} (1 - \pi_i)^{1-s_i}$$

Observable variables x:

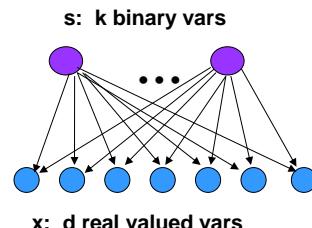
~ Normal distribution
parameters: \mathbf{W}, Σ

$$P(\mathbf{x} | \mathbf{s}) \equiv N(\mathbf{W}\mathbf{s} | \boldsymbol{\Sigma})$$

We assume $\Sigma = \mathbf{I}^d$

Joint for one instance of x and s :

$$P(\mathbf{x}, \mathbf{s} | \Theta) = (2\pi)^{-d/2} \sigma^{-d/2} \exp \left\{ -\frac{1}{2\sigma^2} (\mathbf{x} - \mathbf{Ws})^T (\mathbf{x} - \mathbf{Ws}) \right\} \prod_{i=1}^k \pi_i^{s_i} (1-\pi_i)^{(1-s_i)}$$



CS 2750 Machine Learning

Cooperative vector quantizer

Our objective:

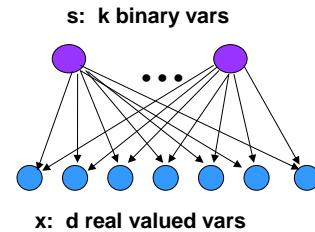
- Learn the parameters of the model \mathbf{W}, π, σ
- One can use the data likelihood or loglikelihood and optimize ..

Learning if \mathbf{x} and \mathbf{s} are observable

Log likelihood:

$$\begin{aligned} \sum_{n=1}^N \log P(\mathbf{x}^{(n)}, \mathbf{s}^{(n)} | \Theta) = \\ \sum_{n=1}^N -d \log \sigma - \frac{1}{2\sigma^2} (\mathbf{x}^{(n)} - \mathbf{W}\mathbf{s}^{(n)})^T (\mathbf{x}^{(n)} - \mathbf{W}\mathbf{s}^{(n)}) + \sum_{i=1}^k s_i^{(n)} \log \pi_i + (1 - s_i^{(n)}) \log (1 - \pi_i) + c \end{aligned}$$

Solution: nice and easy



CS 2750 Machine Learning

Cooperative vector quantizer

Our objective:

- Learn the parameters of the model \mathbf{W}, π, σ
- One can use the data likelihood or loglikelihood and optimize ..

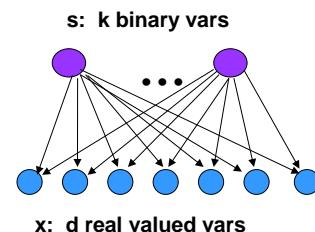
Learning if only \mathbf{x} are observable

Log likelihood of data:

$$\log P(D | \Theta) = \sum_{n=1}^N \log P(\mathbf{x}^{(n)} | \Theta) = \sum_{n=1}^N \log \sum_{\{\mathbf{s}^{(n)}\}} P(\mathbf{x}^{(n)}, \mathbf{s}^{(n)} | \Theta)$$

Solution: does not let us benefit from the decomposition

EM: used to work in such cases ...



CS 2750 Machine Learning

EM

Let H – be a set of all variables with hidden or missing values

$$P(H, D | \Theta, \xi) = P(H | D, \Theta, \xi)P(D | \Theta, \xi)$$

$$\log P(H, D | \Theta, \xi) = \log P(H | D, \Theta, \xi) + \log P(D | \Theta, \xi)$$

$$\log P(D | \Theta, \xi) = \log P(H, D | \Theta, \xi) - \log P(H | D, \Theta, \xi)$$



Log-likelihood of data

Average both sides with $P(H | D, \Theta', \xi)$ for Θ'

$$E_{H|D,\Theta'} \log P(D | \Theta, \xi) = E_{H|D,\Theta'} \log P(H, D | \Theta, \xi) - E_{H|D,\Theta'} \log P(H | D, \Theta, \xi)$$

$$\underbrace{\log P(D | \Theta, \xi)}_{\text{Log-likelihood of data}} = F(\Theta | \Theta') = E(\Theta | \Theta') + H(\Theta | \Theta')$$

Log-likelihood of data

CS 2750 Machine Learning

EM algorithm

Algorithm (general formulation)

Initialize parameters Θ

Repeat

Set $\Theta' = \Theta$

1. Expectation step

$$E(\Theta | \Theta') = \langle \log P(H, D | \Theta, \xi) \rangle_{P(H|D,\Theta')}$$

2. Maximization step

$$\Theta = \arg \max \Theta$$

until no or small improvement in Θ ($\Theta = \Theta'$)

Problem: posterior $P(H | D, \Theta', \xi)$ is defined over 2^k probabilities

CS 2750 Machine Learning

EM algorithm

Posterior $P(H | D, \Theta', \xi)$ for our model

$$P(H | D, \Theta') = \prod_{n=1}^N P(s^{(n)} | x^{(n)}, \Theta')$$

- Each data point $n=1, \dots, N$ requires us to calculate 2^k probabilities
- If k is larger then this is a bottleneck!!!

CS 2750 Machine Learning

Variational approximation

Let H – be a set of all variables with hidden or missing values

Derivation

$$\log P(D | \Theta, \xi) = \log P(H, D | \Theta, \xi) - \log P(H | D, \Theta, \xi)$$

 **Log-likelihood of data**

Average both sides with

$$Q(H | \lambda)$$

$$\begin{aligned} E_{H|\lambda} \log P(D | \Theta, \xi) &= E_{H|\lambda} \log P(H, D | \Theta, \xi) - E_{H|\lambda} \log P(H | \Theta, \xi) \\ &\quad + E_{H|\lambda} \log Q(H | \lambda) - E_{H|\lambda} \log Q(H | \lambda) \end{aligned}$$

$$\begin{aligned} E_{H|\lambda} \log P(D | \Theta, \xi) &= E_{H|\lambda} \log P(H, D | \Theta, \xi) - E_{H|\lambda} \log Q(H | \lambda) \\ &\quad + E_{H|\lambda} \log Q(H | \lambda) - E_{H|\lambda} \log P(H | \Theta, \xi) \end{aligned}$$

$$\log P(D | \Theta, \xi) = F(P, Q) + KL(Q, P)$$

Log-likelihood of data

CS 2750 Machine Learning

Variational approximation

$$\begin{aligned} E_{H|\lambda} \log P(D | \Theta, \xi) &= E_{H|\lambda} \log P(H, D | \Theta, \xi) - E_{H|\lambda} \log Q(H | \lambda) \\ &\quad + E_{H|\lambda} \log Q(H | \lambda) - E_{H|\lambda} \log P(H | \Theta, \xi) \end{aligned}$$

$$\log P(D | \Theta, \xi) = F(Q, \Theta) + KL(Q, P)$$

$$F(Q, \Theta) = \sum_{\{H\}} Q(H | \lambda) \log P(H, D | \Theta, \xi) - \sum_{\{H\}} Q(H | \lambda) \log Q(H | \lambda)$$

$$KL(Q, P) = \sum_{\{H\}} Q(H | \lambda) [\log Q(H | \lambda) - \log P(H | D, \Theta)]$$

Approximation: jointly maximize $F(Q, \Theta)$

Parameters: Θ, λ

Why? $\log P(D | \Theta, \xi) \geq F(Q, \Theta)$

Maximization of F pushes up the lower bound on the log-likelihood

CS 2750 Machine Learning

Variational approximation

• Comparison:

- EM uses true posterior $P(H | D, \Theta', \xi)$
- Variational EM uses a surrogate posterior $Q(H | \lambda)$

EM:

$$\begin{aligned} \log P(D | \Theta, \xi) &= E_{H|D, \Theta'} \log P(H, D | \Theta, \xi) - E_{H|D, \Theta'} \log P(H | D, \Theta, \xi) \\ \log P(D | \Theta, \xi) &= F(\Theta | \Theta') \end{aligned}$$

Variational EM:

$$\begin{aligned} E_{H|\lambda} \log P(D | \Theta, \xi) &= E_{H|\lambda} \log P(H, D | \Theta, \xi) - E_{H|\lambda} \log Q(H | \lambda) \\ &\quad + E_{H|\lambda} \log Q(H | \lambda) - E_{H|\lambda} \log P(H | \Theta, \xi) \end{aligned}$$

$$\log P(D | \Theta, \xi) = F(P, Q) + KL(Q, P)$$

$$\log P(D | \Theta, \xi) \geq F(P, Q)$$

CS 2750 Machine Learning

Variational EM

Let \mathbf{H} – be a set of all variables with hidden or missing values

- **E step:**

- Optimize

$F(Q, \Theta)$ with respect to λ while keeping Θ fixed

- **M step**

- Optimize

$F(Q, \Theta)$ with respect to Θ while keeping λ s

Note: if $Q(\mathbf{H})$ is the posterior then the variational EM reduces to the standard EM

CS 2750 Machine Learning

Variational EM

- So what is the deal?
 - Why should we use the variational EM?
- Hope:
 - If we choose $Q(H | \lambda)$ well the optimization of both λ and Θ will become easy !!!
- A well behaved choice for $Q(H | \lambda)$
 - the mean field approximation

$$Q(H | \lambda) = \prod_i Q_i(H_i | \lambda_i)$$

CS 2750 Machine Learning

Mean Field Approximation

Assumption:

- $Q(H|\lambda)$ is the mean field approximation.
 - Variables in the $Q(H)$ distribution are independent variables H_i .
 - Q is completely factorized:
- $$Q(H | \lambda) = \prod_i Q_i(H_i | \lambda_i)$$

- For our CVQ model

- Hidden variables are binary sources

$$Q(\mathbf{H} | \lambda) = \prod_{n=1,..N} Q(\mathbf{s}^{(n)} | \lambda^{(n)})$$

$$Q(\mathbf{s}^{(n)} | \lambda^{(n)}) = \prod_{i=1,..d} Q(s_i^{(n)} | \lambda_i^{(n)})$$

$$Q(s_i^{(n)} | \lambda_i^{(n)}) = \lambda_i^{(n)s_i^{(n)}} (1 - \lambda_i^{(n)})^{1-s_i^{(n)}}$$

CS 2750 Machine Learning

Mean Field Approximation

Functional F for the mean field:

$$F(Q, \Theta) = \sum_{\{H\}} Q(H | \lambda) \log P(H, D | \Theta, \xi) - \sum_{\{H\}} Q(H | \lambda) \log Q(H | \lambda)$$

Assume just one data point \mathbf{x} and corresponding \mathbf{s} :

$$F(Q, \Theta) = \langle \log P(\mathbf{x}, \mathbf{s} | \Theta) \rangle_{Q(s|\lambda)} - \langle \log Q(\mathbf{s} | \lambda) \rangle_{Q(s|\lambda)}$$

$$= \left\langle -d \log \sigma - \frac{1}{2\sigma^2} (\mathbf{x} - \mathbf{W}\mathbf{s})^T (\mathbf{x} - \mathbf{W}\mathbf{s}) \right\rangle_{Q(s|\lambda)} \quad (1)$$

$$+ \left\langle \sum_{i=1}^k s_i \log \pi_i + (1-s_i) \log (1-\pi_i) \right\rangle_{Q(s|\lambda)} \quad (2)$$

$$- \left\langle \sum_{i=1}^k s_i \log \lambda_i + (1-s_i) \log (1-\lambda_i) \right\rangle_{Q(s|\lambda)} \quad (3)$$

CS 2750 Machine Learning

Mean Field Approximation

Functional F. Part 1:

$$\begin{aligned}
 & \left\langle -d \log \sigma - \frac{1}{2\sigma^2} (\mathbf{x} - \sum_{i=1}^k s_i \mathbf{w}_i)^T (\mathbf{x} - \sum_{i=1}^k s_i \mathbf{w}_i) \right\rangle_{Q(s|\lambda)} = \\
 &= \left\langle -d \log \sigma - \frac{1}{2\sigma^2} (\mathbf{x} - \sum_{i=1}^k s_i \mathbf{w}_i)^T (\mathbf{x} - \sum_{i=1}^k s_i \mathbf{w}_i) \right\rangle_{Q(s|\lambda)} \\
 &= \left\langle -d \log \sigma - \frac{1}{2\sigma^2} \left[\mathbf{x}^T \mathbf{x} - 2 \sum_{i=1}^k (s_i \mathbf{w}_i)^T \mathbf{x} + \sum_{i=1}^k \sum_{j=1}^k s_i s_j \mathbf{w}_i^T \mathbf{w}_j \right] \right\rangle_{Q(s|\lambda)} \\
 &= -d \log \sigma - \frac{1}{2\sigma^2} \left[\mathbf{x}^T \mathbf{x} - 2 \sum_{i=1}^k \langle s_i \rangle_{Q(s_i|\lambda_i)} \mathbf{w}_i^T \mathbf{x} + \sum_{i=1}^k \sum_{j=1}^k \langle s_i s_j \rangle_{Q(s|\lambda)} \mathbf{w}_i^T \mathbf{w}_j \right] \\
 & \quad \langle s_i \rangle_{Q(s_i|\lambda_i)} = \lambda_i \qquad \qquad \langle s_i s_j \rangle_{Q(s|\lambda)} = \lambda_i \lambda_j + \delta_{ij}(\lambda_i - \lambda_i^2)
 \end{aligned}$$

CS 2750 Machine Learning

Mean Field Approximation

Functional F. Part 2:

$$\begin{aligned}
 & \left\langle \sum_{i=1}^k s_i \log \pi_i + (1-s_i) \log(1-\pi_i) \right\rangle_{Q(s|\lambda)} = \sum_{i=1}^k \langle s_i \rangle_{Q(s_i|\lambda_i)} \log \pi_i + (1 - \langle s_i \rangle_{Q(s_i|\lambda_i)}) \log(1-\pi_i) \\
 &= \sum_{i=1}^k \lambda_i \log \pi_i + (1 - \lambda_i) \log(1-\pi_i)
 \end{aligned}$$

Functional F. Part 3:

$$\left\langle \sum_{i=1}^k s_i \log \lambda_i + (1-s_i) \log(1-\lambda_i) \right\rangle_{Q(s|\lambda)} = \sum_{i=1}^k \lambda_i \log \lambda_i + (1 - \lambda_i) \log(1-\lambda_i)$$

CS 2750 Machine Learning

Mean Field Approximation

Functional F:

$$\begin{aligned}
 F(Q, \Theta) &= \langle \log P(\mathbf{x}, \mathbf{s} | \Theta) \rangle_{Q(s|\lambda)} - \langle \log Q(\mathbf{s} | \lambda) \rangle_{Q(s|\lambda)} \\
 &= -d \log \sigma - \frac{1}{2\sigma^2} \left[\mathbf{x}^T \mathbf{x} - 2 \sum_{i=1}^k \lambda_i \mathbf{w}_i \right] \mathbf{x} + \sum_{i=1}^k \sum_{j=1}^k \left[\lambda_i \lambda_j + \delta_{ij} (\lambda_i - \lambda_i^2) \right] \mathbf{w}_i^T \mathbf{w}_j \\
 &\quad + \sum_{i=1}^k \lambda_i \log \pi_i + (1 - \lambda_i) \log(1 - \pi_i) \\
 &\quad + \sum_{i=1}^k \lambda_i \log \lambda_i + (1 - \lambda_i) \log(1 - \lambda_i)
 \end{aligned}$$

Parameters: $\mathbf{W}, \boldsymbol{\pi}, \sigma$

Mean field parameters: λ

CS 2750 Machine Learning

Mean Field Approximation

Functional F (for all data points):

$$\begin{aligned}
 F(Q, \Theta) &= \sum_{n=1}^N \langle \log P(\mathbf{x}^{(n)}, \mathbf{s}^{(n)} | \Theta) \rangle_{Q(s^{(n)}|\lambda^{(n)})} - \langle \log Q(\mathbf{s}^{(n)} | \lambda^{(n)}) \rangle_{Q(s^{(n)}|\lambda^{(n)})} \\
 &= -d \log \sigma - \frac{1}{2\sigma^2} \left[\mathbf{x}^{(n)T} \mathbf{x}^{(n)} - 2 \sum_{i=1}^k \lambda_i^{(n)} \mathbf{w}_i \right] \mathbf{x}^{(n)} + \sum_{i=1}^k \sum_{j=1}^k \left[\lambda_i^{(n)} \lambda_j^{(n)} + \delta_{ij} (\lambda_i^{(n)} - \lambda_i^{(n)2}) \right] \mathbf{w}_i^T \mathbf{w}_j \\
 &\quad + \sum_{i=1}^k \lambda_i^{(n)} \log \pi_i + (1 - \lambda_i^{(n)}) \log(1 - \pi_i) \\
 &\quad + \sum_{i=1}^k \lambda_i^{(n)} \log \lambda_i^{(n)} + (1 - \lambda_i^{(n)}) \log(1 - \lambda_i^{(n)})
 \end{aligned}$$

Parameters: $\mathbf{W}, \boldsymbol{\pi}, \sigma$

Mean field parameters: $\lambda = \lambda^{(1)}, \lambda^{(2)}, \dots, \lambda^{(N)}$

CS 2750 Machine Learning

Variational EM: E step

Optimization of the functional F with respect to λ :

$$\frac{\partial}{\partial \lambda_u} F = \frac{1}{\sigma^2} (\mathbf{x} - \sum_{j \neq u} \lambda_j \mathbf{w}_j)^T \mathbf{w}_u - \frac{1}{2\sigma^2} \mathbf{w}_u^T \mathbf{w}_u + \log \frac{\pi_u}{1-\pi_u} - \log \frac{\lambda_u}{1-\lambda_u}$$

set $\frac{\partial}{\partial \lambda_u} F = 0$

$$\lambda_u = g\left(\frac{1}{\sigma^2} (\mathbf{x} - \sum_{j \neq u} \lambda_j \mathbf{w}_j)^T \mathbf{w}_u - \frac{1}{2\sigma^2} \mathbf{w}_u^T \mathbf{w}_u + \log \frac{\pi_u}{1-\pi_u}\right)$$
$$g(x) = \frac{1}{1+e^{-x}}$$

Defines a fixed point equation

Iterate a set fixed point equations for all indexes $u=1..k$ and for all n

CS 2750 Machine Learning

Variational EM: M step

Optimization of the functional F with respect to Θ .

Start with π :

For N data points

$$\frac{\partial}{\partial \pi_u} F = \sum_{n=1}^N \lambda_u^{(n)} \log \frac{1}{\pi_u} - (1 - \lambda_u^{(n)}) \log \frac{1}{1-\pi_u}$$

set $\frac{\partial}{\partial \pi_u} F = 0$

$$\pi_u = \frac{\sum_{n=1}^N \lambda_u^{(n)}}{N}$$

Closed form solution

CS 2750 Machine Learning

Variational EM: M step

Optimization of the functional F with respect to Θ .

Parameters \mathbf{w} :

$$\frac{\partial}{\partial w_{uv}} F = \sum_{n=1}^N -\frac{1}{2\sigma^2} \left[\lambda_v^{(n)} x_u^{(n)} + 2 \sum_{j \neq v} \lambda_v^{(n)} \lambda_j^{(n)} w_{uj} + 2 \lambda_v^{(n)} w_{uv} \right] = 0$$

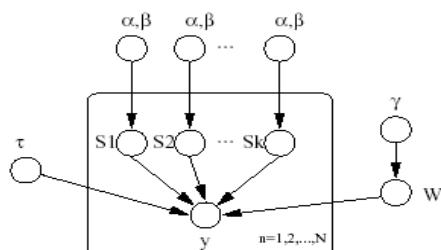
$$\mathbf{W} = \begin{pmatrix} w_{11} & w_{12} & \dots & w_{1k} \\ w_{21} & & & \\ \dots & & & \\ w_{d1} & \dots & \dots & w_{dk} \end{pmatrix} \quad \mathbf{W} = (\mathbf{w}_1 \ \mathbf{w}_2 \ \dots \ \mathbf{w}_k)$$

For each variable v :

The equations define a set of k linear equations that can be solved

CS 2750 Machine Learning

Bayesian CVQ Model



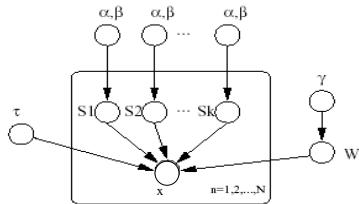
$$y = \sum_{k=1}^K s_k w_k + \varepsilon$$

Bayesian model:
Distributions over parameters

$$P(y | S, \theta) \sim N \left(\sum_{k=1}^K s_k w_k, \tau^{-1} I \right)$$

CS 2750 Machine Learning

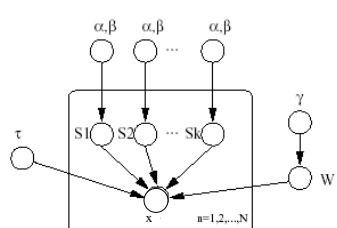
Model Specification



$X = \{x_1, x_2, \dots, x_n\}$	observed data
$S = \{s_1, \dots, s_k\}$	latent sources
$\pi = \{\pi_1, \pi_2, \dots, \pi_k\}$	probability of $s_k = 1$
$W = \{w_1, w_2, \dots, w_k\}$	DxK weight matrix
$\gamma = \{\gamma_1, \gamma_2, \dots, \gamma_k\}$	Variance of W
τ	Precision of noise

CS 2750 Machine Learning

Priors



$$P(\boldsymbol{\pi}) = \prod_{k=1}^K Beta(\pi_k | \alpha, \beta)$$

$$P(\mathbf{W}) = \prod_{k=1}^K N(w_k | 0, \gamma_k)$$

$$P(\boldsymbol{\gamma}) = \prod_{k=1}^K Gamma(\gamma_k | a_\gamma, b_\gamma)$$

$$P(\tau) = Gamma(\tau | c_\tau, d_\tau)$$

CS 2750 Machine Learning

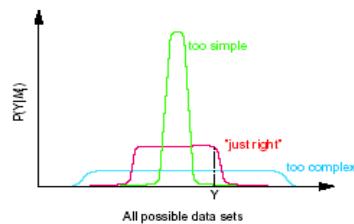
Why the Bayesian model ?

Very useful for Bayesian Model Selection

- Assume we do not know the number of sources
- Bayesian score tells us how good the structure is

Benefits of the Bayesian score:

- Embodies Occam's Razor
- Prevents overfit



$$P(M_i | D) = \frac{P(D | M_i)P(M_i)}{P(D)}$$

$$P(D | M_i) = \int_{\theta} P(D | \theta, M_i)P(\theta | M_i) \leftarrow \text{Marginal likelihood}$$

CS 2750 Machine Learning

Variational approximation

- Approximation: loglikelihood of data

$$\begin{aligned} \log P(X) &= \log \int_{\theta} P(X, \theta) d\theta \\ &= \log \int_{\theta} \sum_H P(X, H, \theta) d\theta \\ &= \log \int_{\theta} \sum_H P(X, H | \theta) P(\theta) d\theta \\ &\geq \int_{\theta} \sum_H Q(H, \theta) \log \frac{P(X, H | \theta) P(\theta)}{Q(H, \theta)} d\theta = F(Q) \end{aligned}$$

Where Q is a distribution with different parameterization

CS 2750 Machine Learning

Variational approximation

- Approximation: loglikelihood of observable data

$$\log P(X) = F(Q) + KL(Q(H, \theta), P(H, \theta))$$

- Optimization of $F(Q)$ is pushing up the lower bound on the loglikelihood of observable data
- How to choose Q ?

$$Q(H, \theta) = Q_\theta(\theta)Q_H(H)$$

- Then:

$$F(Q) = \int_{\theta} Q_\theta(\theta) \left[\sum_H Q_H(H) \log \frac{P(X, H | \theta)}{Q_H(H)} \right] d\theta$$
$$+ \int_{\theta} Q_\theta(\theta) \log \frac{Q_\theta(\theta)}{P(\theta)} d\theta \quad \leftarrow \text{KL distance}$$

CS 2750 Machine Learning

Variational Bayes approximation

- Evaluation of $Q(H, \theta)$ is intractable
- Meanfield approximation

$$Q(H, \theta) = \prod_{k=1}^K Q(H_k) \prod_{i=1}^P (\theta_i)$$

- Allows analytical evaluation of $F(Q)$

CS 2750 Machine Learning

VB learning

Learn Model with an EM like algorithm

(1) VBE – Optimize Q(H)

Estimate state of latent variables

$$Q^*_H(H) \propto \exp \langle \log P(D, H | \theta) \rangle_{Q_\theta(\theta)}$$

(2) VBM – Optimize Q(Θ)

Estimate parameters

$$Q^*_\theta(\theta) \propto P(\theta) \exp \langle \log P(D, H | \theta) \rangle_{Q_H(H)}$$

See Ghahramani & Beal 2004 for the details

CS 2750 Machine Learning

VBE

$$\begin{aligned} \log \frac{\lambda_k}{1-\lambda_k} = & \left\langle \log \frac{\pi_k}{1-\pi_k} \right\rangle_{Q_\theta(\pi)} + y^T \langle \tau \rangle_{Q_\theta(\tau)} \langle w_k \rangle_{Q_\theta(W)} \\ & - \sum_{j \neq k} \lambda_j \langle \tau \rangle_{Q_\theta(\tau)} \text{tr} \left(\langle w_j w_k^T \rangle_{Q_\theta(W)} \right) \\ & - \frac{1}{2} \langle \tau \rangle_{Q_\theta(\tau)} \text{tr} \left(\langle w_k w_k^T \rangle_{Q_\theta(W)} \right) \end{aligned}$$

CS 2750 Machine Learning

VBM

$$Q_{\pi}(\boldsymbol{\pi}) = \prod_{k=1}^K Beta\left(\pi_k | \tilde{\alpha}_k, \tilde{\beta}_k\right)$$

$$Q_W(\mathbf{W}) = \prod_{d=1}^D N\left(\mathbf{w}_k | \tilde{\mathbf{m}}_w^{(d)}, \tilde{\Sigma}_w^{(d)}\right)$$

$$Q_{\gamma}(\boldsymbol{\gamma}) = \prod_{k=1}^K Gamma\left(\gamma_k | \tilde{a}_{\gamma k}, \tilde{b}_{\gamma k}\right)$$

$$Q_{\tau}(\tau) = Gamma\left(\tau | \tilde{c}_{\tau}, \tilde{d}_{\tau}\right)$$

CS 2750 Machine Learning

VBM (cont')

$$\tilde{\alpha}_k = \alpha_k + \sum_{n=1}^N \left\langle s_k^{(n)} \right\rangle$$

$$\tilde{\beta}_k = \beta_k + N - \sum_{n=1}^N \left\langle s_k^{(n)} \right\rangle$$

$$\tilde{\Sigma}_{\mathcal{W}^{(d)}} = \left(diag\left(\langle \boldsymbol{\gamma} \rangle\right) + \left\langle \tau \right\rangle \sum_{n=1}^N \left\langle \mathbf{s}^n \mathbf{s}^{nT} \right\rangle \right)^{-1}$$

$$\tilde{\mathbf{m}}_w^{(d)} = \tilde{\Sigma}_{\mathcal{W}^{(d)}} \left\langle \tau \right\rangle \sum_{n=1}^N \left\langle \mathbf{s}^n \right\rangle \mathbf{y}_d^n$$

CS 2750 Machine Learning

VBM (cont')

$$\tilde{a}_{jk} = a_{jk} + \frac{D}{2}$$

$$\tilde{b}_{jk} = b_{jk} + \frac{\langle ||\mathbf{w}_k||^2 \rangle}{2}$$

$$\tilde{c}_\tau = c_\tau + \frac{ND}{2}$$

$$\tilde{d}_\tau = d_\tau + \frac{1}{2} \sum_{n=1}^N \left\{ ||\mathbf{y}||^2 - 2\mathbf{y}^n{}^T \langle \mathbf{W} \rangle \langle \mathbf{s}^n \rangle \right.$$

$$\left. + \text{tr} \left(\langle \mathbf{W}^T \mathbf{W} \rangle \langle \mathbf{s}^n \mathbf{s}^n{}^T \rangle \right) \right\}$$

CS 2750 Machine Learning

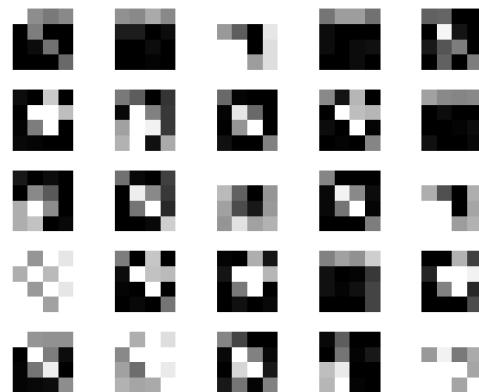
Image Separation Experiment

- 8 sources images projected to \mathbf{x}
- A binary switch defines whether the image is projected or not



CS 2750 Machine Learning

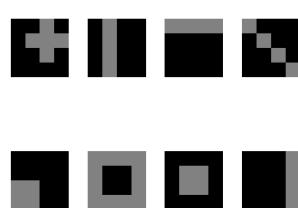
Mixed images



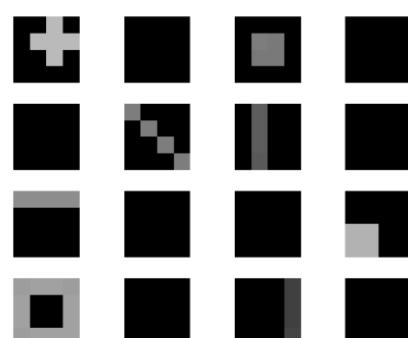
CS 2750 Machine Learning

Recovered sources

Initial sources



Recovered sources



CS 2750 Machine Learning

The Noisy-OR Vector Quantizer

Tomas Singliar and Milos Hauskrecht

CS 2750 Machine Learning

Outline

- High-dimensional data models
- Latent Variable Models
- Noisy-or model (for binary data)
- Learning LVM with EM
- The approximate learning algorithm
- Experiments: Data mining with Noisy-OR VQ
- Conclusions

CS 2750 Machine Learning

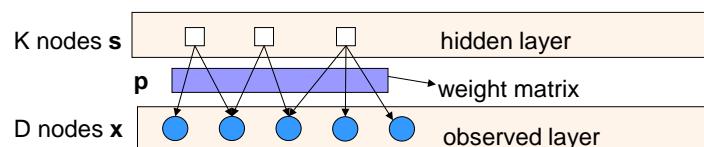
Modeling high-dimensional data

- High-dimensional data
 - sensor networks
 - document repositories
- typically variables are dependent
- How to model dependencies?
 - Full model intractable, overfitting
 - All-independent unrealistic
 - Middle-of-the road models
 - capture dependencies
 - efficient – representation, reasoning, learning

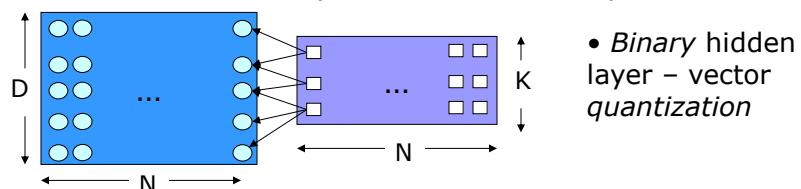
CS 2750 Machine Learning

Latent Factor Model

- Captures dependencies via latent factors and their combinations



- $K \ll D$: Dimensionality reduction, data compression



CS 2750 Machine Learning

Compact Parameterization

- CPT specifies $P(X|\text{parents}(X))$
 - *full model* would have 2^K entries
 - Binary nodes – Noisy-OR parameterization
 - K parameters for each observed node p_{1j}, \dots, p_{kj}
 - p_{ij} “strength of influence” of s_i on observable x_j
- $$P(X_j = 0 | \mathbf{s}) = \prod_{i=1}^K (1 - p_{ij})^{s_i}$$
- $$P(X_j = 1 | \mathbf{s}) = 1 - P(X_j = 0 | \mathbf{s}) = 1 - \prod_{i=1}^K (1 - p_{ij})^{s_i}$$
-

CS 2750 Machine Learning

Exponential Reparameterization

- Exponential form easier to work with
- New parameters $\theta_{ij} = -\log(1 - p_{ij})$

$$P(X_j = 0 | \mathbf{s}) = \prod_{i=1}^K (1 - p_{ij})^{s_i} = \prod_{i=1}^K \exp(-\theta_{ij}s_i)$$

$$P(X_j = 1 | \mathbf{s}) = 1 - P(X_j = 0 | \mathbf{s}) = 1 - \prod_{i=1}^K \exp(-\theta_{ij}s_i)$$

$$= 1 - \exp\left[-\sum_i \theta_{ij}s_i\right] = \exp\left(\log\left(1 - \exp\left[-\sum_i \theta_{ij}s_i\right]\right)\right)$$

CS 2750 Machine Learning

Learning with hidden variables - EM

- Expectation – maximization: standard MLL technique
 - E-step: Compute $p(H | D, \Theta')$ - fill in the data
 - M-step: Choose new Θ to maximize *complete* LL:
$$Q(\theta | \theta') = \langle \log P(D, H | \Theta) \rangle_{P(H|D, \Theta')}$$
 - $\Theta' \leftarrow \Theta$ and repeat until convergence
- Guaranteed to improve Q every iteration
- Local optimization method
- Following EM, we want to optimize

$$\Theta = \arg \max \langle \log P(\mathbf{x}, \mathbf{s} | \Theta) \rangle_{P(\mathbf{s}|\mathbf{x}, \Theta)}$$

CS 2750 Machine Learning

Why EM won't work

- N iid samples (D-dimensional binary vectors)
- We will need:

Problem 1: not a product

- The joint distribution

$$P(\mathbf{x}, \mathbf{s}) = P(\mathbf{x} | \mathbf{s})P(\mathbf{s}) = P(\mathbf{s}) \prod_j \left(1 - \prod_{i=1}^K (1 - p_{ij})^{s_i} \right)^{x_j} \left(\prod_{i=1}^K (1 - p_{ij})^{s_i} \right)^{1-x_j}$$

- Joint over observables

$$P(\mathbf{x}) = \sum_{\mathbf{s}} P(\mathbf{x}, \mathbf{s}) = \sum_{\mathbf{s}} \left(\prod_j P(x_j | \mathbf{s}) \right) P(\mathbf{s})$$

Problem 2: summation over 2^K terms

CS 2750 Machine Learning

Decomposition

If we can express $P(\mathbf{x}|\mathbf{s})$ as $P(x_j | \mathbf{s}) = \prod_i h(x_j | s_i)$

$$\begin{aligned} P(\mathbf{x}) &= \sum_{\mathbf{s}} \left(\prod_j P(x_j | \mathbf{s}) \right) \left(\prod_i P(s_i) \right) \\ &= \sum_{\mathbf{s}} \left(\prod_j \prod_i h(x_j | s_i) \right) \left(\prod_i P(s_i) \right) \\ &= \prod_i P(s_i) \sum_{s_i} \prod_j h(x_j | s_i) \end{aligned}$$

- joint over observables decomposes along hidden factors
- how do we find $h(x_j | s_i)$?

CS 2750 Machine Learning

Decomposable lower bound

- Based on Jensen's inequality

$$\begin{aligned} P(X_j = 1 | \mathbf{s}) &= \exp \left(\log \left(1 - \exp \left[- \sum_i \theta_{ij} s_i \right] \right) \right) \\ &\geq \exp \left(\sum_i q_j(i) s_i \log [1 - e^{-\theta_{ij}/q_j(i)}] + C \right) = \prod_i h_L(x_j | s_i) \end{aligned}$$

This is a product!
Solves Problem 1

Variational
Parameters
satisfying
 $\sum_j q_j = 1$

CS 2750 Machine Learning

Inference on latent sources

- The loglikelihood bound decomposes (**Problem 1 solved**)
- We can now compute posteriors efficiently

$$\tilde{P}(s_i = 1 | \mathbf{x}) = \frac{P(s_i = 1) \prod_j h_L(x_j | s_i = 1)}{\sum_{s_i} P(s_i) \prod_j h_L(x_j | s_i)} \quad \text{Solves Problem 2}$$

- $O(D)$ operations required to perform inference

CS 2750 Machine Learning

EM vs Variational EM

Classical EM

- Iterate until convergence of F:
 - **Compute expectation over hidden variables (E-step)**
 - **Maximize expected loglikelihood (M-step)**
 - optimize w.r.t. Θ (model parameters)

As if we had additional model parameters

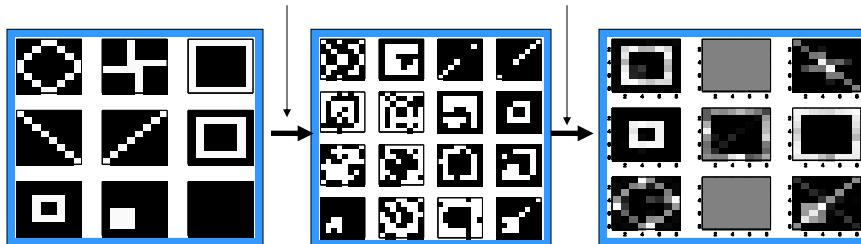
Variational EM

- Iterate until convergence of F:
 - **Compute expectation over hidden variables (E-step)**
 - **Maximize expected loglikelihood (M-step)**
 - optimize w.r.t. \boldsymbol{q} (variational parameters)
 - a fixed-point equation
 - optimize w.r.t. Θ (model parameters)
 - numerical search

CS 2750 Machine Learning

Structure recovery experiments

- Similar source “Source separation” task
- Synthetic data: small B/W pictures
 - true scramble + noise recover



CS 2750 Machine Learning

Real-world data mining with Noisy-OR VQ

- CiteSeer data: 40 ML authors, ~5000 papers
- Create dataset:
 - $M(i,j) = 1$ if document i cites author j , 0 otherwise (5000x40 mx)
- Run learning and see components:
 - 1: Friedman, Jordan, Lauritzen, Pearl
 - 3: Scholkopf, Smola, Vapnik, Burges
 - 4: Ghahramani, Jordan, Hintom, Neal, Saul, Bishop, Tipping
 - 5: Frey, Freeman, Murphy, *Lauritzen*, Pearl, Weiss, Yedidia
- Components reflect ML communities



CS 2750 Machine Learning