Announcements

- Homework 3 out tonight
  - Start early!!
- Project milestones due today
  - Please email to TAs
Parameter learning for log-linear models

- Feature functions $\phi_i(C_i)$ defined over cliques

- Log linear model over undirected graph $G$
  - Feature functions $\phi_1(C_1), ..., \phi_k(C_k)$
  - Domains $C_i$ can overlap

- Joint distribution

$$P(X_1, \ldots, X_n) = \frac{1}{Z} \exp\left(\sum_i w_i^T \phi_i(C_i)\right)$$

- How do we get weights $w_i$?
Log-linear conditional random field

- Define log-linear model over outputs $Y$
  - No assumptions about inputs $X$
- Feature functions $\phi_i(C_i, x)$ defined over cliques and inputs $C_i \subseteq Y$
- Joint distribution

\[
P(Y_1, \ldots, Y_n \mid x) = \frac{1}{Z(x)} \exp\left(\sum_i w_i^T \phi_i(C_i, x)\right)
\]
Example: CRFs in NLP

Mrs. Greene spoke today in New York. Green chairs the finance committee

Classify into Person, Location or Other
Example: CRFs in vision

Saxena et al, PAMI ‘08
Gradient of conditional log-likelihood

- Partial derivative

\[
\frac{\partial \log P(D_Y | w, D_X)}{\partial w_i} = \sum_j \left[ \phi_i(c_i^{(j)}, x^{(j)}) + \sum_{c_i} P(c_i | w, x^{(j)}) \phi_i(c_i, x^{(j)}) \right]
\]

- Requires one inference per feature and per data point

  Can be very expensive

  Can do "pseudo" likelihood est. (approximate)

- Can optimize using conjugate gradient
Exponential Family Distributions

- Distributions for log-linear models

\[ P(X_1, \ldots, X_n) = \frac{1}{Z} \exp\left( \sum_i w_i^T \phi_i(C_i) \right) \]

- More generally: **Exponential family distributions**

\[ P(x) = h(x) \exp\left( w^T \phi(x) - A(w) \right) \]

- h(x): Base measure \[ \text{often constant} \]
- w: natural parameters
- \( \phi(x) \): Sufficient statistics
- A(w): log-partition function
- Hereby x can be continuous (defined over any set)
Examples

- **Exp. Family:**
  \[ P(x) = h(x) \exp\left( w^T \phi(x) - A(w) \right) \]

- **Gaussian distribution**
  \[ P(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left( -\frac{(x - \mu)^2}{2\sigma^2} \right) \]
  \[ h(x) = \frac{1}{\sqrt{2\pi}} \]
  \[ \phi(x) = \left[ -\frac{x^2}{2\sigma^2} \right] \]
  \[ A(w) = \frac{\mu^2}{2\sigma^2} - \log \sigma \]
  \[ w = \left[ \frac{1}{\sigma^2}, \frac{\mu}{\sigma^2} \right] \]

- **Other examples:** Multinomial, Poisson, Exponential, Gamma, Weibull, chi-square, Dirichlet, Geometric, ...
Moments and gradients

\[ P(x) = h(x) \exp(w^T \phi(x) - A(w)) \]

• Correspondence between moments and log-partition function (just like in log-linear models)

\[
\frac{\partial A(w)}{\partial w_i} = \int p(x \mid w) \phi_i(x) dx = \mathbb{E} [\phi_i \mid w]
\]

\[
\frac{\partial^2 A(w)}{\partial w_i \partial w_j} = \text{Cov}(\phi_i, \phi_j \mid w)
\]

• Can compute moments from derivatives, and derivatives from moments!

• MLE ⇔ moment matching
Conjugate priors in Exponential Family

\[ P(x \mid w) = h(x) \exp(w^T \phi(x) - A(w)) \]

Any exponential family likelihood has a conjugate prior

\[ P(w|\alpha, \beta) = \exp(\alpha^T w - \beta A(w) - B(\alpha, \beta)) \]

\[ P(x \mid w) P(w|\alpha, \beta) \propto \exp(w^T(\phi(x) + \alpha) - (\beta + 1) A(w)) \]
Exponential family graphical models

- So far, only defined graphical models over discrete variables.
- Can define GMs over continuous distributions!
- For exponential family distributions:

\[
p(X_1, \ldots, X_n) = \prod_i h_i(C_i) \exp \left( \sum_i w_i^T \phi_i(C_i) - A(w) \right)
\]

\[
\exp A(w) = \int \int \ldots \int \prod_i h_i(C_i) \exp \left( \sum_i w_i^T \phi_i(C_i) - A(w) \right) dx_1 \ldots dx_n
\]

- Can do much of what we discussed (VE, JT, parameter learning, etc.) for such exponential family models
- Important example: **Gaussian Networks**
Multivariate Gaussian distribution

\[ N(x; \Sigma, \mu) = \frac{1}{(2\pi)^{n/2} \sqrt{|\Sigma|}} \exp \left( -\frac{1}{2} (x - \mu)^T \Sigma^{-1} (x - \mu) \right) \]

\[ \Sigma = \begin{pmatrix}
\sigma_1^2 & \sigma_{12} & \cdots & \sigma_{1n} \\
\sigma_{21} & \sigma_2^2 & \cdots & \vdots \\
\vdots & \vdots & \ddots & \vdots \\
\sigma_{n1} & \sigma_{n2} & \cdots & \sigma_n^2
\end{pmatrix} \quad \mu = \begin{pmatrix}
\mu_1 \\
\mu_2 \\
\vdots \\
\mu_n
\end{pmatrix} \]

\[ \Sigma^{\top} = \Sigma \quad \text{positive definite} \quad x^\top \Sigma x \geq 0 \quad \forall x \]

- Joint distribution over \( n \) random variables \( P(X_1, \ldots, X_n) \)
- \( \sigma_{jk} = E[(X_j - \mu_j)(X_k - \mu_k)] = \text{Cov}(X_j, X_k) \)
- \( X_j \) and \( X_k \) independent \( \iff \sigma_{jk} = 0 \)
Marginalization

- Suppose \((X_1, \ldots, X_n) \sim N(\mu, \Sigma)\)
- What is \(P(X_1)??\)

\[
\begin{align*}
P(X_i) &= \int \int \cdots \int p(x_1, \ldots, x_n) dx_{k_2} \cdots dx_{k_m} \\
P(X_i) &= N(x_i; \mu, \sigma^2)
\end{align*}
\]

More generally: Let \(A=\{i_1, \ldots, i_k\} \subseteq \{1, \ldots, N\}\)

- Write \(X_A = (X_{i_1}, \ldots, X_{i_k})\)

\[
X_A \sim N(\mu_A, \Sigma_{AA})
\]

\[
\Sigma_{AA} = \\
\begin{pmatrix}
\sigma^2_{i_1} & \sigma_{i_1i_2} & \cdots & \sigma_{i_1i_k} \\
\vdots & \sigma_{i_2i_1} & \cdots & \vdots \\
\sigma_{i_ki_1} & \sigma_{i_ki_2} & \cdots & \sigma^2_{i_k}
\end{pmatrix}
\]

\[
\mu_A = \\
\begin{pmatrix}
\mu_{i_1} \\
\mu_{i_2} \\
\vdots \\
\mu_{i_k}
\end{pmatrix}
\]
Suppose \((X_1, \ldots, X_n) \sim N(\mu, \Sigma)\)

Decompose as \((X_A, X_B)\)

What is \(P(X_A | X_B)??\)

\[
P(X_A = x_A | X_B = x_B) = N(x_A; \mu_{A|B}, \Sigma_{A|B})
\]

where

\[
\Sigma = \begin{pmatrix}
    \Sigma_{AA} & \Sigma_{AB} \\
    \Sigma_{AB} & \Sigma_{BB}
\end{pmatrix}
\]

\[
\mu_{A|B} = \mu_A + \Sigma_{AB} \Sigma_{BB}^{-1} (x_B - \mu_B)
\]

\[
\Sigma_{A|B} = \Sigma_{AA} - \Sigma_{AB} \Sigma_{BB}^{-1} \Sigma_{BA}
\]

Computable using linear algebra!

---

\[
\text{Does not depend on } x_B
\]
Conditional linear Gaussians

\[ \mu_{A|B} = \mu_A + \Sigma_{AB} \Sigma_{BB}^{-1} (x_B - \mu_B) \]

\[ \Sigma_{A|B} = \Sigma_{AA} - \Sigma_{AB} \Sigma_{BB}^{-1} \Sigma_{BA} \]

\[ x_{A|B} = \mu_A + \mathbf{W} x_B - \mathbf{W} \mu_B + \varepsilon \sim \mathcal{N}(0, \Sigma_{A|B}) \]

\[ = \mathbf{W} x_B + b + \varepsilon \sim \text{noise} \]

\[ \mathbf{W} = \Sigma_{AB} \Sigma_{BB}^{-1} \]

\[ b = \mu_A - \mathbf{W} \mu_B \]
Canonical Representation

\[
p(X_1, \ldots, X_n) = \frac{1}{(2\pi)^{n/2}\sqrt{|\Sigma|}} \exp \left( -\frac{1}{2} (x - \mu)^T \Sigma^{-1} (x - \mu) \right) \\
\propto \exp(\eta^T x - \frac{1}{2} x^T \Lambda x) = \exp \left( \sum_{k} \eta_k x_k - \frac{1}{2} \sum_{k,j} \Sigma_{kj} x_k x_j \right)
\]

- Multivariate Gaussians in exponential family!

- Standard vs canonical form:
  \[
  \mu = \Lambda^{-1} \eta \\
  \Sigma = \Lambda^{-1}
  \]
Gaussian Networks

\[ p(X_1, \ldots, X_n) \propto \exp\left( -\frac{1}{2} \sum_{i,j} \lambda_{i,j} x_i x_j + \sum_i \eta_i x_i \right) \]

\[ \phi_i(x_i) \]

\[ \phi_{i,j}(x_i, x_j) \]

\[ \phi_{i,j}(x_i, x_j) = -\frac{1}{2} x_i x_j \quad \phi_{i}(x_i) = x_i \]

No edge between \( X_i, X_j \)

\[ \lambda_{i,j} = 0 \]

Zeros in precision matrix \( \Lambda \) indicate missing edges in log-linear model!
Inference in Gaussian Networks

- Can compute marginal distributions in $O(n^3)$.
- For large numbers $n$ of variables, still intractable.
- If Gaussian Network has low treewidth, can use variable elimination / JT inference!

Need to be able to multiply and marginalize factors!

$$g = \int_{x_i} \prod_{j} f_j$$
Multiplying factors in Gaussians

\[ P(X_{A}) = \mathcal{N}(x_{A} ; \Lambda_{1}, \eta_{1}) \]
\[ P(X_{B} | X_{A}) \propto \mathcal{N}(x_{B} ; x_{A}, \Lambda_{2}, \eta_{2}) \]

\[ P(X_{A}, X_{B}) = P(X_{A}) P(X_{B} | X_{A}) \]
\[ \propto \exp(x_{A}^{T} \Lambda_{1} x_{A} + \eta_{1}^{T} x_{A}) \exp((x_{A}^{T} x_{B})^{T} \Lambda_{2} (x_{A}^{T} x_{B}) + \eta_{2}^{T} x_{A}) \]
\[ = \mathcal{N}(x_{A}, x_{B} ; \Lambda, \eta) \]

\[ \Lambda = \Lambda_{1} + \Lambda_{2} = \begin{pmatrix} 1 & 0 \\ 0 & \Lambda_{2} \end{pmatrix} \]
\[ \eta = \eta_{1} + \eta_{2} \]
Conditioning in canonical form

- Joint distribution \((X_A, X_B) \sim N(\eta_{AB}, \Lambda_{AB})\)

- Conditioning: \(P(X_A \mid X_B = x_B) = N(x_A; \eta_{A \mid B = x_B}, \Lambda_{A \mid B = x_B})\)

\[
\eta_{A \mid B = x_B} = \eta_A - \Lambda_{AB} x_B
\]

\[
\Lambda_{A \mid B = x_B} = \Lambda_{AA}
\]
Marginalizing in canonical form

- Recall conversion formulas
  - $\mu = \Lambda^{-1} \eta$
  - $\Sigma = \Lambda^{-1}$

- Marginal distribution

\[
\eta^m_A = \eta_A - \Lambda_{AB} \Lambda^{-1}_{BB} \eta_B
\]

\[
\Lambda^m_{AA} = \Lambda_{AA} - \Lambda_{AB} \Lambda^{-1}_{BB} \Lambda_{BA}
\]
### Standard vs. canonical form

<table>
<thead>
<tr>
<th>Standard form</th>
<th>Canonical form</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Marginalization</strong></td>
<td></td>
</tr>
<tr>
<td>$\mu^m_A = \mu_A$</td>
<td>$\eta^m_A = \eta_A - \Lambda_{AB} \Lambda^{-1}_{BB} \eta_B$</td>
</tr>
<tr>
<td>$\Sigma^m_{AA} = \Sigma_{AA}$</td>
<td>$\Lambda^m_{AA} = \Lambda_{AA} - \Lambda_{AB} \Lambda^{-1}<em>{BB} \Lambda</em>{BA}$</td>
</tr>
<tr>
<td><strong>Conditioning</strong></td>
<td></td>
</tr>
<tr>
<td>$\mu_{A</td>
<td>B=x_B} = \mu_A + \Sigma_{AB} \Sigma^{-1}_{BB} (x_B - \mu_B)$</td>
</tr>
<tr>
<td>$\Sigma_{A</td>
<td>B=x_B} = \Sigma_{AA} - \Sigma_{AB} \Sigma^{-1}<em>{BB} \Sigma</em>{BA}$</td>
</tr>
</tbody>
</table>

- In standard form, marginalization is easy
- In canonical form, conditioning is easy!
In Gaussian Markov Networks, Variable elimination = Gaussian elimination (fast for low bandwidth = low treewidth matrices)
Dynamical models
HMMs / Kalman Filters

- Most famous Graphical models:
  - Naïve Bayes model
  - Hidden Markov model
  - Kalman Filter

- Hidden Markov models
  - Speech recognition
  - Sequence analysis in comp. bio

- Kalman Filters control
  - Cruise control in cars
  - GPS navigation devices
  - Tracking missiles..

- Very simple models but very powerful!!
HMMs / Kalman Filters

- $X_1, \ldots, X_T$: Unobserved (hidden) variables
- $Y_1, \ldots, Y_T$: Observations
- HMMs: $X_i$ Multinomial, $Y_i$ arbitrary
- Kalman Filters: $X_i$, $Y_i$ Gaussian distributions
  - Non-linear KF: $X_i$ Gaussian, $Y_i$ arbitrary
HMMs for speech recognition

Infer spoken words from audio signals

Phoneme

Words

“He ate the cookies on the couch”

Infer spoken words from audio signals
Hidden Markov Models

- Inference:
  - In principle, can use VE, JT etc.
  - New variables $X_t, Y_t$ at each time step $\Rightarrow$ need to rerun

- Bayesian Filtering:
  - Suppose we already have computed $P(X_t \mid y_{1,\ldots,t})$
  - Want to efficiently compute $P(X_{t+1} \mid y_{1,\ldots,t+1})$
Bayesian filtering

- Start with $P(X_1)$
- At time $t$
  - Assume we have $P(X_t \mid y_{1\ldots t-1})$
  - Condition: $P(X_t \mid y_{1\ldots t})$

  $$P(X_t \mid y_{1\ldots t}) \propto P(X_t \mid y_{1\ldots t-1}) \frac{P(Y_t \mid X_t, y_{1\ldots t-1})}{\text{cond. ind. } P(Y_t \mid X_t)}$$

  - Prediction: $P(X_{t+1}, X_t \mid y_{1\ldots t})$
    $$P(X_{t+1}, X_t \mid y_{1\ldots t}) = P(X_t \mid y_{1\ldots t}) \cdot P(X_{t+1} \mid X_t, y_{1\ldots t})$$
    "Rollup"

  - Marginalization: $P(X_{t+1} \mid y_{1\ldots t})$
    $$P(X_{t+1} \mid y_{1\ldots t}) = \sum_{X_t} P(X_{t+1}, X_t \mid y_{1\ldots t})$$
Parameter learning in HMMs

- Assume we have labels for hidden variables
- Assume stationarity
  - \( P(X_{t+1} | X_t) \) is same over all time steps
  - \( P(Y_t | X_t) \) is same over all time steps
  - Violates parameter independence (parameter “sharing”)

Example: compute parameters for \( P(X_{t+1} = x | X_t = x') \)

\[
\log P(x, \ldots, x_1, y_1, \ldots, y_T | \theta) = \log P(x_1) \prod_{t=2}^{T} P(x_t | x_{t-1}, \theta_t) \prod_{t=1}^{T} P(y_t | x_t, \theta)
\]

\[
= \log P(x_1 | \theta_1) + \sum_{t} \log P(x_t | x_{t-1}, \theta_t) + \sum_{t} \log P(y_t | x_t, \theta_t)
\]

\[
\theta_{x_t = x | x_{t-1} = x'} = \frac{\text{Count}(x_1; x)}{t-1}
\]

- What if we don’t have labels for hidden vars?
  - Use EM (later this course)
Kalman Filters (Gaussian HMMs)

- \( X_1, \ldots, X_T \): Location of object being tracked
- \( Y_1, \ldots, Y_T \): Observations
- \( P(X_1) \): Prior belief about location at time 1
- \( P(X_{t+1} \mid X_t) \): “Motion model”
  - How do I expect my target to move in the environment?
  - Represented as CLG: \( X_{t+1} = A X_t + N(0, \Sigma_M) \)
- \( P(Y_t \mid X_t) \): “Sensor model”
  - What do I observe if target is at location \( X_t \)?
  - Represented as CLG: \( Y_t = H X_t + N(0, \Sigma_O) \)
Understanding Motion model

\[ X_t = \begin{pmatrix} L_t \\ V_t \end{pmatrix} \quad X_{t+1} = A X_t + \text{noise} \]

\[ A = \begin{pmatrix} 1 & \Delta t \\ 0 & 1 \end{pmatrix} \]

P(X_i) \quad P(X_2) \quad P(X_3) \quad P(X_4)

P(X_i) = \int P(X_2 | X_i) P(X_i) \Delta x_i
Understanding sensor model

\[ X_t = \begin{pmatrix} L_t \\ V_t \end{pmatrix} \]

\[ Y_t = \begin{pmatrix} 1 & 0 \end{pmatrix} X_t + \text{noise} \]

\[ P(X_t | y_{t-1}) \]

Posterior \[ P(X_t | y_t, \ldots, y_{t-1}) \]

Observer in \[ Y_t \]
Bayesian Filtering for KFs

- Can use Gaussian elimination to perform inference in “unrolled” model

- Start with prior belief $P(X_1)$

- At every timestep have belief $P(X_t | y_{1:t-1})$
  - Condition on observation: $P(X_t | y_{1:t})$
  - Predict (multiply motion model): $P(X_{t+1}, X_t | y_{1:t})$
  - “Roll-up” (marginalize prev. time): $P(X_{t+1} | y_{1:t})$
Implementation

- Current belief: \( P(x_t \mid y_{1:t-1}) = N(x_t; \eta_{x_t}, \Lambda_{x_t}) \)
- Multiply sensor and motion model
  \[
P(X_{t+1} | X_t) \propto \text{Represented as CLG, canonical pams } \Lambda_m, \eta_m \]
  \[
P(X_{t+1}, X_t | y_t, \ldots y_t) = P(X_t | y_t, \ldots y_t) P(X_{t+1} | X_t) = N(-i; \eta_{x_t} + \eta_m, \Lambda_{x_t} + \Lambda_m)
  \]
- Marginalize
  \[
P(X_{t+1} | y_t, \ldots y_t) = N(-i; \eta_{A}^m, \Lambda_{B}^m)
  \]
  \[
  \eta_A^m = \eta_A - \Lambda_{AB} \Lambda_{BB}^{-1} \eta_B
  \]
  \[
  \Lambda_A^m = \Lambda_{AA} - \Lambda_{AB} \Lambda_{BB}^{-1} \Lambda_{BA}
  \]
  \[
  A = X_{t+1} | y_t, \ldots y_t \quad B = X_{t-1} | y_t, \ldots y_t
  \]
What if observations not “linear”?

- Linear observations:
  \[ Y_t = H X_t + \text{noise} \]

- Nonlinear observations:
  \[ "\text{Motion detector}" : Y_t = 1 \text{ if } X_t \in \mathbb{R} \]
  \[ = 0 \text{ otherwise} \]
Incorporating Non-gaussian Observations

- Nonlinear observation $\Rightarrow$ $P(Y_t \mid X_t)$ not Gaussian

First approach: Approximate $P(Y_t \mid X_t)$ as CLG

- Linearize $P(Y_t \mid X_t)$ around current estimate $E[X_t \mid y_{1..t-1}]$
- Known as Extended Kalman Filter (EKF)
- Can perform poorly if $P(Y_t \mid X_t)$ highly nonlinear

Second approach: Approximate $P(Y_t, X_t)$ as Gaussian

- Takes correlation in $X_t$ into account
- After obtaining approximation, condition on $Y_t = y_t$
  (now a “linear” observation)
Finding Gaussian approximations

- Need to find Gaussian approximation of $P(X_t, Y_t)$
- How?
  - Gaussians in Exponential Family $\Rightarrow$ Moment matching!!

\[
E[Y_t] = \int y P(y_t | x_t) \, dx_t = \int y P(y_t | x_t) P(x_t) \, dx_t \, dy_t
\]

\[
E[Y_t^2] = \int y^2 P(y_t | x_t) P(x_t) \, dx_t \, dy_t
\]

\[
E[X_t Y_t] = \int x y P(y_t | x_t) P(x_t) \, dx_t \, dy_t
\]
Linearization by integration

- Need to integrate product of Gaussian with arbitrary function

- Can do that by numerical integration
  - Approximate integral as weighted sum of evaluation points
    \[ \int f(x, y) \rho(x) \, dx \, dy \approx \sum_i w_i f(x^{(i)}, y^{(i)}) \]
  - Gaussian quadrature defines locations and weights of points
  - For 1 dim: **Exact** for polynomials of degree D if choosing 2D points using Gaussian quadrature
  - For higher dimensions: Need exponentially many points to achieve exact evaluation for polynomials

- Application of this is known as “Unscented” Kalman Filter (UKF)
Factored dynamical models

- So far: HMMs and Kalman filters

What if we have more than one variable at each time step?
  - E.g., temperature at different locations, or road conditions in a road network?
  - Spatio-temporal models
Dynamic Bayesian Networks

At every timestep have a Bayesian Network

Variables at each time step t called a “slice” $S_t$

“Temporal” edges connecting $S_{t+1}$ with $S_t$
Tasks

- Read Koller & Friedman Chapters 6.2.3, 15.1