Probabilistic Graphical Models

Conditional Random Fields

&

Case study I: image segmentation

Eric Xing

Lecture 11, February 21, 2014

Reading: See class website

© Eric Xing @ CMU, 2005-2014
Hidden Markov Model revisit

- Transition probabilities between any two states
  \[ p(y_{i}^{t} = 1 \mid y_{i-1}^{t} = 1) = a_{i,j}, \]
  or
  \[ p(y_{i} \mid y_{i-1}^{t} = 1) \sim \text{Multinomial}(a_{i,1}, a_{i,2}, \ldots, a_{i,M}), \forall i \in I. \]

- Start probabilities
  \[ p(y_{1}) \sim \text{Multinomial}(\pi_{1}, \pi_{2}, \ldots, \pi_{M}). \]

- Emission probabilities associated with each state
  \[ p(x_{i} \mid y_{i}^{t} = 1) \sim \text{Multinomial}(b_{i,1}, b_{i,2}, \ldots, b_{i,K}), \forall i \in I. \]
  or in general: \[ p(x_{i} \mid y_{i}^{t} = 1) \sim f(\cdot \mid \theta_{i}), \forall i \in I. \]
Inference (review)

- **Forward algorithm**

\[ \alpha_t^k \overset{\text{def}}{=} \mu_{t-1 \rightarrow t}^k (k) = P(x_1, \ldots, x_{t-1}, x_t, y_t^k = 1) \]

\[ \alpha_t^k = p(x_t \mid y_t^k = 1) \sum_i \alpha_{t-1}^i a_{i,k} \]

- **Backward algorithm**

\[ \beta_t^k = \sum_i a_{k,i} p(x_{t+1} \mid y_{t+1}^i = 1) \beta_{t+1}^i \]

\[ \beta_t^k \overset{\text{def}}{=} \mu_{t-1 \leftrightarrow t} (k) = P(x_{t+1}, \ldots, x_T \mid y_T^k = 1) \]

\[ \gamma_i^t \overset{\text{def}}{=} p(y_i^t = 1 \mid x_{1:T}) \propto \alpha_i^t \beta_i^t = \sum_j \xi_{i,j}^t \]

\[ \xi_{i,j}^t \overset{\text{def}}{=} p(y_i^t = 1, y_{i+1}^j = 1, x_{1:T}) \]

\[ \propto \mu_{t-1 \rightarrow t}(y_i^t = 1) \mu_{t \leftarrow t+1}(y_i^j = 1) p(x_{t+1} \mid y_{i+1}) p(y_{t+1} \mid y_i) \]

\[ \xi_{i,j}^t = \alpha_i^t \beta_{t+1}^j a_{i,j} p(x_{t+1} \mid y_{t+1}^j = 1) \]

The matrix-vector form:

\[ B_t(i) \overset{\text{def}}{=} p(x_t \mid y_t^i = 1) \]

\[ A(i, j) \overset{\text{def}}{=} p(y_{t+1}^j = 1 \mid y_t^i = 1) \]

\[ \alpha_t = (A^T \alpha_{t-1}) \ast B_t \]

\[ \beta_t = A(\beta_{t+1} \ast B_{t+1}) \]

\[ \xi_t = (\alpha_t (\beta_{t+1} \ast B_{t+1})^T) \ast A \]

\[ \gamma_t = \alpha_t \ast \beta_t \]
Learning HMM

- **Supervised learning**: estimation when the “right answer” is known
  - **Examples**:
    - GIVEN: a genomic region $x = x_1 \ldots x_{1,000,000}$ where we have good (experimental) annotations of the CpG islands
    - GIVEN: the casino player allows us to observe him one evening, as he changes dice and produces 10,000 rolls

- **Unsupervised learning**: estimation when the “right answer” is unknown
  - **Examples**:
    - GIVEN: the porcupine genome; we don’t know how frequent are the CpG islands there, neither do we know their composition
    - GIVEN: 10,000 rolls of the casino player, but we don’t see when he changes dice

- **QUESTION**: Update the parameters $\theta$ of the model to maximize $P(x|\theta)$ -- Maximal likelihood (ML) estimation
Learning HMM: two scenarios

- Supervised learning: if only we knew the true state path then ML parameter estimation would be trivial
  - E.g., recall that for complete observed tabular BN:

  $\theta_{ijk}^{ML} = \frac{n_{ijk}}{\sum_{i,j,k} n_{ijk}}$

  - What if y is continuous? We can treat $\{(x_{n,t}, y_{n,t}): t=1:T, n=1:N\}$ as $N \times T$ observations of, e.g., a GLIM, and apply learning rules for GLIM ...

- Unsupervised learning: when the true state path is unknown, we can fill in the missing values using inference recursions.
  - The Baum Welch algorithm (i.e., EM)
    - Guaranteed to increase the log likelihood of the model after each iteration
    - Converges to local optimum, depending on initial conditions

© Eric Xing @ CMU, 2005-2014
The Baum Welch algorithm

- The complete log likelihood

\[ \ell_c(\theta; x, y) = \log p(x, y) = \log \prod_n \left( \prod_{t=2}^T p(y_{n,t} | y_{n,t-1}) \prod_{t=1}^T p(x_{n,t} | x_{n,t}) \right) \]

- The expected complete log likelihood

\[ \langle \ell_c(\theta; x, y) \rangle = \sum_n \left( \langle y_{n,1}^i \rangle \log \pi_i \right) + \sum_n \sum_{t=2}^T \left( \langle y_{n,t-1}^i y_{n,t}^j \rangle \log a_{i,j} \right) + \sum_n \sum_{t=1}^T x_{n,t}^k \langle y_{n,t}^i \rangle \log b_{i,k} \]

- EM

  - The E step

  \[ \gamma_{n,t}^i = \langle y_{n,t}^i \rangle = p(y_{n,t}^i = 1 | x_n) \]

  \[ \xi_{n,t}^{i,j} = \langle y_{n,t-1}^i y_{n,t}^j \rangle = p(y_{n,t-1}^i = 1, y_{n,t}^j = 1 | x_n) \]

  - The M step ("symbolically" identical to MLE)

\[ \pi_i^{ML} = \frac{\sum_n \gamma_{n,1}^i}{N} \quad a_{ij}^{ML} = \frac{\sum_n \sum_{t=2}^T \xi_{n,t}^{i,j}}{\sum_n \sum_{t=1}^{T-1} \gamma_{n,t}^i} \quad b_{ik}^{ML} = \frac{\sum_n \sum_{t=1}^T \gamma_{n,t}^i x_{n,t}^k}{\sum_n \sum_{t=1}^{T-1} \gamma_{n,t}^i} \]
Shortcomings of Hidden Markov Model (1): locality of features

- HMM models capture dependences between each state and only its corresponding observation
  - NLP example: In a sentence segmentation task, each segmental state may depend not just on a single word (and the adjacent segmental stages), but also on the (non-local) features of the whole line such as line length, indentation, amount of white space, etc.

- Mismatch between learning objective function and prediction objective function
  - HMM learns a joint distribution of states and observations $P(Y, X)$, but in a prediction task, we need the conditional probability $P(Y|X)$
Solution:
Maximum Entropy Markov Model (MEMM)

- Models dependence between each state and the full observation sequence explicitly
  - More expressive than HMMs
- Discriminative model
  - Completely ignores modeling $P(X)$: saves modeling effort
  - Learning objective function consistent with predictive function: $P(Y|X)$

\[
P(y_{1:n} | x_{1:n}) = \prod_{i=1}^{n} P(y_i | y_{i-1}, x_{1:n}) = \prod_{i=1}^{n} \frac{\exp(w^T f(y_i, y_{i-1}, x_{1:n}))}{Z(y_{i-1}, x_{1:n})}
\]
Then, shortcomings of MEMM (and HMM) (2): the Label bias problem

What the local transition probabilities say:
- State 1 almost always prefers to go to state 2
- State 2 almost always prefer to stay in state 2
MEMM: the Label bias problem

Probability of path 1 -> 1 -> 1 -> 1:
• $0.4 \times 0.45 \times 0.5 = 0.09$
MEMM: the Label bias problem

State 1

Observation 1  Observation 2  Observation 3  Observation 4

State 2

State 3

State 4

State 5

Probability of path 2->2->2->2 :
• 0.2 X 0.3 X 0.3 = 0.018

Other paths:
1-> 1-> 1-> 1: 0.09

© Eric Xing @ CMU, 2005-2014
MEMM: the Label bias problem

Probability of path 1->2->1->2:
• 0.6 X 0.2 X 0.5 = 0.06

Other paths:
1->1->1->1: 0.09
2->2->2->2: 0.018
MEMM: the Label bias problem

Probability of path 1->1->2->2:
• 0.4 X 0.55 X 0.3 = 0.066

Other paths:
1->1->1->1: 0.09
2->2->2->2: 0.018
1->2->1->2: 0.06
MEMM: the Label bias problem

Most Likely Path: 1-> 1-> 1-> 1

• Although **locally** it seems state 1 wants to go to state 2 and state 2 wants to remain in state 2.

• **why?**
MEMM: the Label bias problem

Most Likely Path: 1-> 1-> 1-> 1

- State 1 has only two transitions but state 2 has 5:
  - Average transition probability from state 2 is lower
MEMM: the Label bias problem

Label bias problem in MEMM:
• Preference of states with lower number of transitions over others
Solution:
Do not normalize probabilities locally

From local probabilities ....
Solution:
Do not normalize probabilities locally

From local probabilities to local potentials

• States with lower transitions do not have an unfair advantage!
From MEMM ....

\[ P(y_{1:n} | x_{1:n}) = \prod_{i=1}^{n} P(y_i | y_{i-1}, x_{1:n}) = \prod_{i=1}^{n} \frac{\exp(w^T f(y_i, y_{i-1}, x_{1:n}))}{Z(y_{i-1}, x_{1:n})} \]
From MEMM to CRF

- CRF is a partially directed model
  - Discriminative model like MEMM
  - Usage of global normalizer $Z(x)$ overcomes the label bias problem of MEMM
  - Models the dependence between each state and the entire observation sequence (like MEMM)

\[
P(y_{1:n}|x_{1:n}) = \frac{1}{Z(x_{1:n})} \prod_{i=1}^{n} \phi(y_i, y_{i-1}, x_{1:n}) = \frac{1}{Z(x_{1:n}, w)} \prod_{i=1}^{n} \exp(w^T f(y_i, y_{i-1}, x_{1:n}))
\]
Conditional Random Fields

- General parametric form:

\[
P(y|x) = \frac{1}{Z(x, \lambda, \mu)} \exp(\sum_{i=1}^{n} \left( \sum_{k} \lambda_k f_k(y_i, y_{i-1}, x) + \sum_{l} \mu_l g_l(y_i, x) \right))
\]

\[
= \frac{1}{Z(x, \lambda, \mu)} \exp(\sum_{i=1}^{n} (\lambda^T f(y_i, y_{i-1}, x) + \mu^T g(y_i, x)))
\]

where

\[
Z(x, \lambda, \mu) = \sum_{y} \exp(\sum_{i=1}^{n} (\lambda^T f(y_i, y_{i-1}, x) + \mu^T g(y_i, x)))
\]
CRFs: Inference

- Given CRF parameters $\lambda$ and $\mu$, find the $y^*$ that maximizes $P(y|x)$

$$y^* = \arg \max_y \exp \left( \sum_{i=1}^{n} (\lambda^T f(y_i, y_{i-1}, x) + \mu^T g(y_i, x)) \right)$$

- Can ignore $Z(x)$ because it is not a function of $y$

- Run the max-product algorithm on the junction-tree of CRF:

Same as Viterbi decoding used in HMMs!
CRF learning

- Given \{\{(x_d, y_d)\}_{d=1}^N\}, find \(\lambda^*, \mu^*\) such that

\[
\lambda^*, \mu^* = \arg \max_{\lambda, \mu} L(\lambda, \mu) = \arg \max_{\lambda, \mu} \prod_{d=1}^N P(y_d|x_d, \lambda, \mu)
\]

\[
= \arg \max_{\lambda, \mu} \prod_{d=1}^N \frac{1}{Z(x_d, \lambda, \mu)} \exp\left(\sum_{i=1}^n (\lambda^T f(y_{d,i}, y_{d,i-1}, x_d) + \mu^T g(y_{d,i}, x_d))\right)
\]

\[
= \arg \max_{\lambda, \mu} \sum_{d=1}^N \left(\sum_{i=1}^n (\lambda^T f(y_{d,i}, y_{d,i-1}, x_d) + \mu^T g(y_{d,i}, x_d)) - \log Z(x_d, \lambda, \mu)\right)
\]

- Computing the gradient w.r.t \(\lambda\):

\[
\nabla_\lambda L(\lambda, \mu) = \sum_{d=1}^N \left(\sum_{i=1}^n f(y_{d,i}, y_{d,i-1}, x_d) - \sum_{y} P(y|x_d) \sum_{i=1}^n f(y_{d,i}, y_{d,i-1}, x_d))\right)
\]

Gradient of the log-partition function in an exponential family is the expectation of the sufficient statistics.

© Eric Xing @ CMU, 2005-2014
Computing the model expectations:

- Requires exponentially large number of summations: Is it intractable?

\[ \sum_{d=1}^{N} \left( \sum_{i=1}^{n} f(y_{d,i}, y_{d,i-1}, x_d) - \sum_{\mathbf{y}} \left( P(\mathbf{y} | \mathbf{x}_d) \sum_{i=1}^{n} f(y_i, y_{i-1}, x_d) \right) \right) \]

- Tractable!
  - Can compute marginals using the sum-product algorithm on the chain

\[ \sum_{\mathbf{y}} \left( P(\mathbf{y} | \mathbf{x}_d) \sum_{i=1}^{n} f(y_i, y_{i-1}, x_d) \right) = \sum_{i=1}^{n} \sum_{\mathbf{y}} f(y_i, y_{i-1}, x_d) P(\mathbf{y} | \mathbf{x}_d) \]

\[ = \sum_{i=1}^{n} \sum_{y_i, y_{i-1}} f(y_i, y_{i-1}, x_d) P(y_i, y_{i-1} | x_d) \]

Expectation of \( f \) over the corresponding marginal probability of neighboring nodes!!
CRF learning

- Computing marginals using junction-tree calibration:

- Junction Tree Initialization:

- After calibration:

$$P(y_i, y_{i-1} | x_d) \propto \alpha(y_i, y_{i-1})$$

$$\Rightarrow P(y_i, y_{i-1} | x_d) = \frac{\alpha(y_i, y_{i-1})}{\sum_{y_i, y_{i-1}} \alpha(y_i, y_{i-1})} = \alpha'(y_i, y_{i-1})$$

Also called forward-backward algorithm
CRF learning

- Computing feature expectations using calibrated potentials:
  \[
  \sum_{y_i, y_{i-1}} f(y_i, y_{i-1}, x_d) P(y_i, y_{i-1} | x_d) = \sum_{y_i, y_{i-1}} f(y_i, y_{i-1}, x_d) \alpha'(y_i, y_{i-1})
  \]

- Now we know how to compute \( r_\lambda L(\lambda, \mu) \):

  \[
  \nabla_\lambda L(\lambda, \mu) = \sum_{d=1}^{N} \left( \sum_{i=1}^{n} f(y_{d,i}, y_{d,i-1}, x_d) - \sum_{y} (P(y|x_d) \sum_{i=1}^{n} f(y_i, y_{i-1}, x_d)) \right)
  \]

  \[
  = \sum_{d=1}^{N} \left( \sum_{i=1}^{n} f(y_{d,i}, y_{d,i-1}, x_d) - \sum_{y_i, y_{i-1}} \alpha'(y_i, y_{i-1}) f(y_i, y_{i-1}, x_d) \right)
  \]

- Learning can now be done using gradient ascent:

  \[
  \lambda^{(t+1)} = \lambda^{(t)} + \eta \nabla_\lambda L(\lambda^{(t)}, \mu^{(t)})
  \]

  \[
  \mu^{(t+1)} = \mu^{(t)} + \eta \nabla_\mu L(\lambda^{(t)}, \mu^{(t)})
  \]
CRF learning

- In practice, we use a Gaussian Regularizer for the parameter vector to improve generalizability.

\[
\lambda^*, \mu^* = \arg \max_{\lambda, \mu} \sum_{d=1}^{N} \log P(y_d | x_d, \lambda, \mu) - \frac{1}{2\sigma^2} (\lambda^T \lambda + \mu^T \mu)
\]

- In practice, gradient ascent has very slow convergence.
  - Alternatives:
    - Conjugate Gradient method
    - Limited Memory Quasi-Newton Methods
CRFs: some empirical results

- Comparison of error rates on synthetic data

Data is increasingly higher order in the direction of arrow

CRFs achieve the lowest error rate for higher order data
CRFs: some empirical results

- Parts of Speech tagging

<table>
<thead>
<tr>
<th>model</th>
<th>error</th>
<th>oov error</th>
</tr>
</thead>
<tbody>
<tr>
<td>HMM</td>
<td>5.69%</td>
<td>45.99%</td>
</tr>
<tr>
<td>MEMM</td>
<td>6.37%</td>
<td>54.61%</td>
</tr>
<tr>
<td>CRF</td>
<td>5.55%</td>
<td>48.05%</td>
</tr>
<tr>
<td>MEMM+</td>
<td>4.81%</td>
<td>26.99%</td>
</tr>
<tr>
<td>CRF+</td>
<td>4.27%</td>
<td>23.76%</td>
</tr>
</tbody>
</table>

+ Using spelling features

- Using same set of features: HMM >=< CRF > MEMM
- Using additional overlapping features: CRF+ > MEMM+ >> HMM
Other CRFs

- So far we have discussed only 1-dimensional chain CRFs
  - Inference and learning: exact
- We could also have CRFs for arbitrary graph structure
  - E.g: Grid CRFs
  - Inference and learning no longer tractable
  - Approximate techniques used
    - MCMC Sampling
    - Variational Inference
    - Loopy Belief Propagation
  - We will discuss these techniques soon
Image Segmentation

- Image segmentation (FG/BG) by modeling of interactions btw RVs
  - Images are noisy.
  - Objects occupy continuous regions in an image.

\[ Y^* = \arg \max_{y \in \{0,1\}^n} \sum_{i \in S} V_i(y_i, X) + \sum_{i \in S} \sum_{j \in N_i} V_{i,j}(y_i, y_j). \]

Y*: labels
X*: data (features)
S: pixels
N_i: neighbors of pixel i
Undirected Graphical Models
(with an Image Labeling Example)

- Image can be represented by 4-connected 2D grid.

- MRF / CRF with image labeling problem
  - $X=\{x_i\}_{i \in S}$: observed data of an image.
  - $x_i$: data at $i$-th site (pixel or block) of the image set $S$
  - $Y=\{y_i\}_{i \in S}$: (hidden) labels at $i$-th site. $y_i \in \{1, \ldots, L\}$.

- Object: maximize the conditional probability $Y^* = \arg\max_Y P(Y|X)$
  - $y_i = 0$ (BG)
  - $y_i = 1$ (FG)
MRF (Markov Random Field)

- Definition: $Y = \{y_i\}_{i \in S}$ is called Markov Random Field on the set $S$, with respect to neighborhood system $N$, iff for all $i \in S$,

$$P(y_i | y_{S-\{i\}}) = P(y_i | y_{N_i}).$$

- The posterior probability is

$$P(Y | X) = \frac{P(X, Y)}{P(X)} \propto P(X | Y)P(Y) = \prod_{i \in S} P(x_i | y_i) \cdot P(Y)$$

  - (1) Very strict independence assumptions for tractability: Label of each site is a function of data only at that site.
  - (2) $P(Y)$ is modeled as a MRF

$$P(Y) = \frac{1}{Z} \prod_{c \in C} \psi_c(y_c)$$
CRF

- Definition: Let $G = (S, E)$, then $(X, Y)$ is said to be a Conditional Random Field (CRF) if, when conditioned on $X$, the random variables $y_i$ obey the Markov property with respect to the graph.

$$P(y_i | X, y_{S-\{i\}}) = P(y_i | X, y_{Ni})$$

MRF: $$P(y_i | y_{S-\{i\}}) = P(y_i | y_{Ni})$$

- Globally conditioned on the observation $X$
CRF vs MRF

- MRF: two-step generative model
  - Infer likelihood $P(X|Y)$ and prior $P(Y)$
  - Use Bayes theorem to determine posterior $P(Y|X)$

$$P(Y | X) = \frac{P(X, Y)}{P(X)} \propto P(X | Y)P(Y) = \prod_{i \in S} P(x_i | y_i) \cdot \frac{1}{Z} \prod_{c \in C} \psi_c (y_c)$$

- CRF: one-step discriminative model
  - Directly Infer posterior $P(Y|X)$

- Popular Formulation

**MRF**

$$P(Y | X) = \frac{1}{Z} \exp\left( \sum_{i \in S} \log p(x_i | y_i) + \sum_{i \in S} \sum_{i' \in N_i} V_2(y_i, y_{i'}) \right)$$

**CRF**

$$P(Y | X) = \frac{1}{Z} \exp\left( -\sum_{i \in S} V_1(y_i, X) + \sum_{i \in S} \sum_{i' \in N_i} V_2(y_i, y_{i'} | X) \right)$$

Assumption

Potts model for $P(Y)$ with only pairwise potential

Only up to pairwise clique potentials

© Eric Xing @ CMU, 2005-2014
Example of CRF – DRF

- A special type of CRF
  - The unary and pairwise potentials are designed using local discriminative classifiers.
  - Posterior

\[
P(Y | X) = \frac{1}{Z} \exp\left(\sum_{i \in S} A_i(y_i, X) + \sum_{i \in S} \sum_{j \in N_i} I_{ij}(y_i, y_j, X)\right)
\]

- Association Potential
  - Local discriminative model for site \( i \): using logistic link with GLM.

\[
A_i(y_i, X) = \log P(y_i | f_i(X)) \quad P(y_i = 1 | f_i(X)) = \frac{1}{1 + \exp(-w^T f_i(X))} = \sigma(w^T f_i(X))
\]

- Interaction Potential
  - Measure of how likely site \( i \) and \( j \) have the same label given \( X \)

\[
I_{ij}(y_i, y_j, X) = ky_i y_j + (1 - k)(2\sigma(y_i y_j \mu_{ij}(X)) - 1))
\]

(1) Data-independent smoothing term (2) Data-dependent pairwise logistic function

Task: Detecting man-made structure in natural scenes.
- Each image is divided in non-overlapping 16x16 tile blocks.

An example

- Logistic: No smoothness in the labels
- MRF: Smoothed False positive. Lack of neighborhood interaction of the data

Example of CRF – Body Pose Estimation

- Task: Estimate a body pose.
  - Need to detect parts of human body
  - Appearance + Geometric configuration.
  - A large number of DOFs

- Use CRF to model a human body
  - Nodes: Parts (head, torso, upper/lower left/right arms).
    \[ L = (l_1, \ldots, l_6), \quad l_i = [x_i, y_i, \theta_i] \]
  - Edges: Pairwise linkage between parts
  - Tree vs. Graph

Example of CRF – Body Pose Estimation

- Posterior of configuration
  
  \[ P(L \mid I) \propto \exp(\sum_i \Phi(l_i) + \sum_{(i,j) \in E} \Psi(l_i, l_j)) \]

  - \( \psi(l_i, l_j) \): relative position with geometric constraints
  - \( \phi(l_i) \): local image evidence for a part in a particular location
  - If \( E \) is a tree, exact inference is efficiently performed by BP.

- Example of unary and pairwise terms
  
  - Unary term: appearance feature
  
  - Pairwise term: kinematic layout

  \[ \text{HOG of lower arm template (learned)} \]
  
  \[ \text{HOG of image} \]
  
  \[ \text{L2 Distance} \]
  
  \[ \text{Truncated quadratic} \]

  [Zisserman 2010]
Example of CRF – Results of Body Pose Estimation

- Examples of results

[Ramanan 2006]

- Datasets and codes are available.
  - http://www.robots.ox.ac.uk/~vgg/research/pose_estimation/

[Ferrari et al. 2008]
Summary

- Conditional Random Fields are partially directed discriminative models
- They overcome the label bias problem of MEMMs by using a global normalizer
- Inference for 1-D chain CRFs is exact
  - Same as Max-product or Viterbi decoding
- Learning also is exact
  - Globally optimum parameters can be learned
  - Requires using sum-product or forward-backward algorithm
- CRFs involving arbitrary graph structure are intractable in general
  - E.g.: Grid CRFs
  - Inference and learning require approximation techniques
    - MCMC sampling
    - Variational methods
    - Loopy BP