Machine Learning & Data Mining
CS/CNS/EE 155

Lecture 9:
Conditional Random Fields
Announcements

• Homework 5 released
  – Skeleton code available on Moodle
  – Due in 2 weeks (2/16)

• Kaggle competition closes 2/9
  – SHORT report due 2/11 via Moodle
  – Submit as a group

• Nothing due week of 2/23
Today

• Recap of Sequence Prediction

• Conditional Random Fields
  – Sequential version of logistic regression
    • Analogous to how HMMs generalize Naïve Bayes
  – Discriminative sequence prediction
    • Learns to optimize $P(y|x)$ for sequences
Recap: Sequence Prediction

- Input: \( x = (x^1, \ldots, x^M) \)
- Predict: \( y = (y^1, \ldots, y^M) \)
  - Each \( y^i \) one of \( L \) labels.

- \( x = \) “Fish Sleep”
- \( y = (N, V) \)

- \( x = \) “The Dog Ate My Homework”
- \( y = (D, N, V, D, N) \)

- \( x = \) “The Fox Jumped Over The Fence”
- \( y = (D, N, V, P, D, N) \)

POS Tags:
Det, Noun, Verb, Adj, Adv, Prep
\( L=6 \)
Recap: General Multiclass

• $x = \text{“Fish sleep”}$
• $y = (N, V)$

• Multiclass prediction:
  – All possible length-$M$ sequences as different class
    – $(D, D), (D, N), (D, V), (D, \text{Adj}), (D, \text{Adv}), (D, \text{Pr})$
      $(N, D), (N, N), (N, V), (N, \text{Adj}), (N, \text{Adv}), \ldots$

• $L^M$ classes!
  – Length 2: $6^2 = 36!$
Recap: General Multiclass

- $x = \text{“Fish sleep”}$
- $y = (N, V)$
- Multiclass prediction:
  - All possible length-$M$ sequences as different class
    - $(D, D)$, $(D, N)$, $(D, V)$, $(D, Adj)$, $(D, Adv)$, ...
    - $(N, D)$, $(N, N)$, $(N, V)$, $(N, Adj)$, $(N, Adv)$, ...
- $L=6$ classes!
- Length 2: $6^2 = 36$
- Can Model Everything! (In Theory)
- Exponential Explosion in #Classes! (Not Tractable)

POS Tags:
Det, Noun, Verb, Adj, Adv, Prep
Recap: Independent Multiclass

x=“I fish often”

• Treat each word independently (assumption)
  – Independent multiclass prediction per word
  – Predict for x=“I” independently
  – Predict for x=“fish” independently
  – Predict for x=“often” independently
  – Concatenate predictions.

Assume pronouns are nouns for simplicity.
Recap: Independent Multiclass

x=“I fish often”

#Classes = #POS Tags
(6 in our example)

Solvable using standard multiclass prediction. But ignores context!

Assume pronouns are nouns for simplicity.
Recap: 1st Order HMM

- $x = (x^1, x^2, x^4, x^4, ..., x^M)$ (sequence of words)
- $y = (y^1, y^2, y^3, y^4, ..., y^M)$ (sequence of POS tags)
- $P(x^i | y^i)$ Probability of state $y^i$ generating $x^i$
- $P(y^{i+1} | y^i)$ Probability of state $y^i$ transitioning to $y^{i+1}$
- $P(y^1 | y^0)$ $y^0$ is defined to be the Start state
- $P(\text{End} | y^M)$ Prior probability of $y^M$ being the final state
  - Not always used
Graphical Model Representation

\[ P(x, y) = P(\text{End} \mid y^M) \prod_{i=1}^{M} P(y^i \mid y^{i-1}) \prod_{i=1}^{M} P(x^i \mid y^i) \]
HMM Matrix Formulation

\[ P(x, y) = P(END \mid y^M) \prod_{j=1}^{M} P(x^j \mid y^j)P(y^j \mid y^{j-1}) \]

\[ = A_{END,y^M} \prod_{j=1}^{M} A_{y^j,y^{j-1}}O_{y^j,x^j} \]

- Transition Probabilities
- Emission Probabilities (Observation Probabilities)
Recap: 1\textsuperscript{st}-Order Sequence Models

- **General multiclass:**
  - Unique scoring function per entire seq.
  - Very intractable

- **Independent multiclass**
  - Scoring function per token, apply to each token in seq.
  - Ignores context, low accuracy

- **First-order models**
  - Scoring function per pair of tokens.
  - "Sweet spot" between fully general & ind. multiclass
Recap: Naïve Bayes & HMMs

- Naïve Bayes:
  \[ P(x, y) = P(y) \prod_{d=1}^{D} P(x^d | y) \]

- Hidden Markov Models:
  \[ P(x, y) = P(End | y^M) \prod_{j=1}^{M} P(y^j | y^{j-1}) \prod_{i=1}^{M} P(x^i | y^j) \]

- HMMs ≈ 1\textsuperscript{st} order variant of Naïve Bayes!
  
  (just one interpretation...)

"Naïve" Generative Independence Assumption
Recap: Generative Models

- Joint model of \((x,y)\):
  - Compact & easy to train...
  - ...with ind. assumptions
    - E.g., Naïve Bayes & HMMs

- Maximize Likelihood Training:

- Mismatch w/ prediction goal:
  - But hard to maximize \(P(y|x)\)
Learn Conditional Prob.?

• Weird to train to maximize:

\[
\text{argmax}_\theta \prod_{i=1}^{N} P(x_i, y_i)
\]

• When goal should be to maximize:

\[
\text{argmax}_\theta \prod_{i=1}^{N} P(y_i | x_i) = \text{argmax}_\theta \prod_{i=1}^{N} \frac{P(x_i, y_i)}{P(x_i)}
\]

Breaks independence!
Can no longer use count statistics

\[
P(x^d = a | y = z) = \frac{\sum_{i=1}^{N} 1\{y_i = z\} \{x_i^d = a\}}{\sum_{i=1}^{N} 1\{y_i = z\}}
\]

Both HMMs & Naïve Bayes suffer this problem!
Learn Conditional Prob.?

• Weird to train to maximize:

\[
\sum_{i=1}^{N} \log \frac{P(x_i, y_i)}{P(x_i)}
\]

In general, you should maximize the likelihood of the model you define!

So if you define joint model \(P(x, y)\), then maximize \(P(x, y)\) on training data.

Both HMMs & Naïve Bayes suffer this problem!
Generative vs Discriminative

• Generative Models:
  – Joint Distribution: \( P(x,y) \)
  – Uses Bayes’s Rule to predict: \( \arg\max_y P(y|x) \)
  – Can generate new samples \((x,y)\)

• Discriminative Models:
  – Conditional Distribution: \( P(y|x) \)
  – Can directly to predict: \( \arg\max_y P(y|x) \)

• Both trained via Maximum Likelihood
First Try
(for classifying a single y)

• Model $P(y|x)$ for every possible $x$

| $P(y=1|x)$ | $x^1$ | $x^2$ |
|------------|-------|-------|
| 0.5        | 0     | 0     |
| 0.7        | 0     | 1     |
| 0.2        | 1     | 0     |
| 0.4        | 1     | 1     |

• Train by counting frequencies

• **Exponential in # input variables!**
  – Need to assume something... what?
Log Linear Models!
(Logistic Regression)

\[ P(y \mid x) = \frac{\exp\left\{ w_y^T x - b_y \right\}}{\sum_k \exp\left\{ w_k^T x - b_k \right\}} \quad x \in \mathbb{R}^D \]
\[ y \in \{1, 2, \ldots, L\} \]

• “Log-Linear” assumption
  – Model representation to linear in \( x \)
  – Most common discriminative probabilistic model

Prediction: \( \arg\max_y P(y \mid x) \) ← Match! → Training: \( \arg\max_{\Theta} \prod_{i=1}^{N} P(y_i \mid x_i) \)
Naïve Bayes vs Logistic Regression

• Naïve Bayes:
  – Strong ind. assumptions
  – Super easy to train...
  – ...but mismatch with prediction

• Logistic Regression:
  – "Log Linear" assumption
    • Often more flexible than Naïve Bayes
  – Harder to train (gradient desc.)...
  – ...but matches prediction

\[ P(x, y) = A_y \prod_{d=1}^{D} O_{x^d, y} \]

\[ P(y) \quad P(x \mid y) \]

\[ P(y \mid x) = \frac{\exp \left\{ w_y^T x - b_y \right\}}{\sum_k \exp \left\{ w_k^T x - b_k \right\}} \]

\[ x \in \mathbb{R}^D \]
\[ y \in \{1, 2, \ldots, L\} \]
# Naïve Bayes vs Logistic Regression

- NB has L parameters for $P(y)$ (i.e., A).
- LR has L parameters for bias $b$.
- NB has $L \times D$ parameters for $P(x|y)$ (i.e., O).
- LR has $L \times D$ parameters for $w$.
- **Same number of parameters!**

<table>
<thead>
<tr>
<th>Naïve Bayes</th>
<th>Logistic Regression</th>
</tr>
</thead>
<tbody>
<tr>
<td>$P(x, y) = A_y \prod_{d=1}^{D} O_{x^d, y}^{d}$</td>
<td>$P(y</td>
</tr>
<tr>
<td>$P(y) \quad P(x</td>
<td>y)$</td>
</tr>
</tbody>
</table>
Naïve Bayes vs Logistic Regression

Intuition:
Both models have same “capacity”
NB spends a lot of capacity on P(x)
LR spends all of capacity on P(y|x)

No Model Is Perfect!
(Especially on finite training set)
NB will trade off P(y|x) with P(x)
LR will fit P(y|x) as well as possible
Conditional Random Fields
Sequential Version of Logistic Regression
"Log-Linear" 1\textsuperscript{st} Order Sequential Model

\[ P(y \mid x) = \frac{1}{Z(x)} \exp \left\{ \sum_{j=1}^{M} \left( A_{y^j, y^{j-1}} + O_{y^j, x^j} \right) \right\} \]

\[ Z(x) = \sum_{y'} \exp \{ F(y', x) \} \quad \text{aka "Partition Function"} \]

\[ F(y, x) \equiv \sum_{j=1}^{M} \left( A_{y^j, y^{j-1}} + O_{y^j, x^j} \right) \]

Scoring Function

Scoring transitions  Scoring input features

\[ P(y \mid x) = \frac{\exp \{ F(y, x) \} }{Z(x)} \]

\[ \log P(y \mid x) = F(y, x) - \log(Z(x)) \]

\( y^0 \) = special start state, excluding end state
\[ P(y \mid x) = \frac{1}{Z(x)} \exp \left\{ \sum_{j=1}^{M} \left( A_{y^j,y^{j-1}} + O_{y^j,x^j} \right) \right\} \]

- \( x = \text{"Fish Sleep"} \)
- \( y = (N,V) \)

\[
P(N,V \mid \text{"Fish Sleep"}) = \frac{1}{Z(x)} \exp \left\{ A_{N,\text{Start}} + O_{N,\text{Fish}} + A_{V,N} + O_{V,\text{Sleep}} \right\} = \frac{1}{Z(x)} \exp \left\{ 4 \right\} \approx 0.66
\]

\[
Z(x) = \text{Sum}\left(\begin{array}{c}
A_{N,V} \\
A_{*,N} \\
A_{*,V} \\
A_{*,\text{Start}}
\end{array}\right)
\]

<table>
<thead>
<tr>
<th>y</th>
<th>exp(F(y,x))</th>
</tr>
</thead>
<tbody>
<tr>
<td>(N,N)</td>
<td>exp(1+2-2+1) = exp(2)</td>
</tr>
<tr>
<td>(N,V)</td>
<td>exp(1+2+2+0) = exp(4)</td>
</tr>
<tr>
<td>(V,N)</td>
<td>exp(-1+1+2+1) = exp(3)</td>
</tr>
<tr>
<td>(V,V)</td>
<td>exp(-1+1-2+0) = exp(-2)</td>
</tr>
</tbody>
</table>
• $x = \text{“Fish Sleep”}$
• $y = (N,V)$

\[
P(N,V | \text{“Fish Sleep”}) = \frac{1}{Z(x)} \exp\{F(x,y)\}
\]

*hold other parameters fixed*
Basic Conditional Random Field

• Directly models $P(y \mid x)$
  – Discriminative
  – Log linear assumption
  – Same #parameters as HMM
  – 1st Order Sequential LR

• How to Predict?
• How to Train?
• Extensions?

CRF spends all model capacity on $P(y \mid x)$, rather than $P(x, y)$
Predict via Viterbi

$$\arg\max_y P(y \mid x) = \arg\max_y \log P(y \mid x) = \arg\max_y F(y, x)$$

$$\begin{align*}
= \arg\max_y \sum_{j=1}^{M} \left( A_{j, y^{j-1}} + O_{y^j, x^j} \right)
\end{align*}$$

Scoring transitions

Scoring observations

<table>
<thead>
<tr>
<th>Maintain length-k prefix solutions</th>
<th>( \hat{Y}^k(T) = \left( \arg\max_{y^{1:k-1}} F(y^{1:k-1} \oplus T, x^{1:k}) \right) \oplus T )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Recursively solve for length-(k+1) solutions</td>
<td>( \hat{Y}^{k+1}(T) = \left( \arg\max_{y^{1:k} \in {\hat{Y}^k(T)}_T} F(y^{1:k} \oplus T, x^{1:k+1}) \right) \oplus T )</td>
</tr>
<tr>
<td></td>
<td>( = \left( \arg\max_{y^{1:k} \in {\hat{Y}^k(T)}<em>T} F(y^{1:k}, x^{1:k}) + A</em>{T, y^k} + O_{T, x^{k+1}} \right) \oplus T )</td>
</tr>
<tr>
<td>Predict via best length-M solution</td>
<td>( \arg\max_y F(y, x) = \arg\max_{y \in {\hat{Y}^M(T)}_T} F(y, x) )</td>
</tr>
</tbody>
</table>
Solve: \[ \hat{Y}^2(V) = \left( \arg\max_{y^1 \in \hat{Y}^1(T)} F(y^1, x^1) + A_{V,y^1} + O_{V,x^2} \right) \oplus V \]

Store each \( \hat{Y}^1(T) \) & \( F(\hat{Y}^1(T),x) \)

\( \hat{Y}^1(V) \) \( y^1=V \) \( \rightarrow \) \( \hat{Y}^2(V) \)

\( \hat{Y}^1(D) \) \( y^1=D \) \( \rightarrow \) \( \hat{Y}^2(D) \)

\( \hat{Y}^1(N) \) \( y^1=N \) \( \rightarrow \) \( \hat{Y}^2(N) \)

\( \hat{Y}^1(T) \) is just \( T \)
Store each $\hat{Y}^1(T)$ & $F(\hat{Y}^1(T), x^1)$

$\hat{Y}^1(V)$  $\hat{Y}^2(V)$

$\hat{Y}^1(D)$  $\hat{Y}^2(D)$

$\hat{Y}^1(N)$  $\hat{Y}^2(N)$

$y^1 = N$

$\hat{Y}^1(T)$ is just $T$

Ex: $\hat{Y}^2(V) = (N, V)$
Solve:  \[ \hat{Y}^3(V) = \left( \arg\max_{y^1} F(y^{1:2}, x^{1:2}) + A_{y^2} + O_{y^3} \right) \oplus V \]

Store each \[ \hat{Y}^1(T) \& F(\hat{Y}^1(T),x^1) \] Store each \[ \hat{Y}^2(Z) \& F(\hat{Y}^2(Z),x) \]

\[ \hat{Y}^1(V) \rightarrow \hat{Y}^2(V) \rightarrow \hat{Y}^3(V) \]

\[ \hat{Y}^1(D) \rightarrow \hat{Y}^2(D) \rightarrow \hat{Y}^3(D) \]

\[ \hat{Y}^1(N) \rightarrow \hat{Y}^2(N) \rightarrow \hat{Y}^3(N) \]

\[ \hat{Y}^1(Z) \text{ is just } Z \]

Ex: \[ \hat{Y}^2(V) = (N, V) \]
Solve: \[ \hat{Y}^M(V) = \left( \operatorname{argmax}_{y^{M-1} \in \{\hat{Y}^M(T)\}_T} F(y_1^{M-1}, x_1^{M-1}) + A_{V,y^{M-1}} + O_{V,x^M} \right) \bigoplus V \]

Store each \( \hat{Y}^1(Z) \) & \( F(\hat{Y}^1(Z), x^1) \)

Store each \( \hat{Y}^2(T) \) & \( F(\hat{Y}^2(T), x) \)

Store each \( \hat{Y}^3(T) \) & \( F(\hat{Y}^3(T), x) \)

\( \hat{Y}^1(T) \) is just \( T \)

Ex: \( \hat{Y}^2(V) = (N, V) \)

Ex: \( \hat{Y}^3(V) = (D, N, V) \)
Computing $P(y \mid x)$

• Viterbi doesn’t compute $P(y \mid x)$
  – Just maximizes the numerator $F(y, x)$

\[
P(y \mid x) = \frac{\exp\{F(y, x)\}}{\sum_{y'} \exp\{F(y', x)\}} \equiv \frac{1}{Z(x)} \exp\{F(y, x)\}
\]

• Also need to compute $Z(x)$
  – aka the “Partition Function”

\[
Z(x) = \sum_{y'} \exp\{F(y', x)\}
\]
Computing Partition Function

• Naive approach is iterate over all $y'$
  – Exponential time, $L^M$ possible $y'$!

\[
Z(x) = \sum_{y'} \exp \{F(y', x)\} \quad \text{and} \quad F(y, x) = \sum_{j=1}^{M} \left(A_{y^j, y_{j-1}} + O_{y^j, x^j}\right)
\]

• Notation: \(G^j(b, a) = \exp \{A_{b, a} + O_{b, x^j}\}\) Suppressing dependency on $x$ for simpler notation

\[
P(y \mid x) = \frac{1}{Z(x)} \prod_{j=1}^{M} G^j(y^j, y^{j-1})
\]

\[
Z(x) = \sum_{y'} \prod_{j=1}^{M} G^j(y'^j, y'^{j-1})
\]

Matrix Semiring

\[ Z(x) = \sum \prod_{j=1}^{M} G^j(y'^j, y'^{j-1}) \]

\[ G^j(b, a) = \exp\left\{ A_{b,a} + O_{a,x^j} \right\} \]

\[ G^{1:2}(b, a) = \sum_c G^2(b, c) G^1(c, a) \]

\[ G^{i:j}(b, a) = G^i_{i:j} = G^j \quad G^{j-1} \quad \ldots \quad G^{i+1} \quad G^i \]

Path Counting Interpretation

• Interpretation $G_1^1(b,a)$
  – L+1 start & end locations
  – Weight of path from ‘a’ to ‘b’ in step 1

• $G_1^{1:2}(b,a)$
  – Weight of all paths
    • Start in ‘a’ beginning of Step 1
    • End in ‘b’ after Step 2

### Computing Partition Function

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
</table>
| **Consider Length-1 (M=1)** | \[ Z(x) = \sum_{b} G^1(b, \text{Start}) \]  
  Sum column ‘Start’ of \( G^1 \)!
| **M=2** | \[ Z(x) = \sum_{a,b} G^2(b,a)G^1(a, \text{Start}) = \sum_{b} G^{1:2}(b, \text{Start}) \]  
  Sum column ‘Start’ of \( G^{1:2} \)!
| **General M** | Sum column ‘Start’ of \( G^{1:M} \)!
  - Do \( M (L+1) \times (L+1) \) matrix computations to compute \( G^{1:M} \)
  - \( Z(x) = \) sum column ‘Start’ of \( G^{1:M} \)

Computing Partition Function

- Consider Length-1 (M=1)
  \[ Z(x) = \sum_{b, \text{Start}} G^1(b, \text{Start}) \]

- M=2

- General M
  - Do M \((L+1) \times (L+1)\) matrix computations to compute \(G^{1:M}\)
  - \(Z(x) = \text{sum column ‘Start’ of } G^{1:M}\)

Numerical Instability Issues! (See Course Notes)

Train via Gradient Descent

• Similar to Logistic Regression
  – Gradient Descent on negative log likelihood (log loss)

\[
\arg\min_{\Theta} \sum_{i=1}^{N} - \log P(y_i \mid x_i) = \arg\min_{\Theta} \sum_{i=1}^{N} -F(y_i, x_i) + \log(Z(x_i))
\]

$\Theta$ often used to denote all parameters of model

• First term is easy:
  – Recall:

\[
F(y, x) \equiv \sum_{j=1}^{M} \left( A_{y^j, y^{j-1}} + O_{y^j, x^j} \right)
\]

\[
\partial_{\text{A}_b} - F(y, x) = - \sum_{j=1}^{M} 1_{(y^j, y^{j-1})=(b,a)}
\]

\[
\partial_{\text{O}_a} - F(y, x) = - \sum_{j=1}^{M} 1_{(y^j, x^j)=(a,z)}
\]

Differentiating Log Partition

\[ \partial_{A_{ba}} \log(Z(x)) = \frac{1}{Z(x)} \partial_{A_{ba}} Z(x) = \frac{1}{Z(x)} \partial_{A_{ba}} \sum_{y'} \exp\{F(y', x)\} \]

\[ = \frac{1}{Z(x)} \sum_{y'} \partial_{A_{ba}} \exp\{F(y', x)\} \]

\[ = \frac{1}{Z(x)} \sum_{y'} \exp\{F(y', x)\} \partial_{A_{ba}} F(y', x) = \sum_{y'} \frac{\exp\{F(y', x)\}}{Z(x)} \partial_{A_{ba}} F(y', x) \]

Definition of \( P(y' \mid x) \)

\[ = \sum_{y'} P(y' \mid x) \partial_{A_{ba}} F(y', x) = \sum_{y'} \left[ P(y' \mid x) \sum_{j=1}^M 1_{(y'^j, y'^{j-1})=(b,a)} \right] \]

\[ = \sum_{j=1}^M \sum_{y'} P(y' \mid x) 1_{(y'^j, y'^{j-1})=(b,a)} = \sum_{j=1}^M P(y^j = b, y^{j-1} = a \mid x) \]

Forward-Backward!

Marginalize over all \( y' \)

Optimality Condition

\[
\arg\min_{\Theta} \sum_{i=1}^{N} -\log P(y_i \mid x_i) = \arg\min_{\Theta} \sum_{i=1}^{N} -F(y_i, x_i) + \log(Z(x))
\]

- Consider one parameter:

\[
\partial_{A_{ba}} \sum_{i=1}^{N} -F(y_i, x_i) = -\sum_{i=1}^{N} \sum_{j=1}^{M_i} \mathbb{1}[y_i^j,y_i^j-1=(b,a)]
\]

\[
\partial_{A_{ba}} \sum_{i=1}^{N} \log(Z(x)) = \sum_{i=1}^{N} \sum_{j=1}^{M_i} P(y_i^j = b, y_i^j-1 = a \mid x_i)
\]

- Optimality condition:

\[
\sum_{i=1}^{N} \sum_{j=1}^{M_i} \mathbb{1}[y_i^j, y_i^j-1=(b,a)] = \sum_{i=1}^{N} \sum_{j=1}^{M_i} P(y_i^j = b, y_i^j-1 = a \mid x_i)
\]

- Frequency counts = Cond. expectation on training data!
  - Holds for each component of the model
  - Each component is a “log-linear” model and requires gradient desc.
Forward-Backward for CRFs

\[
\begin{align*}
\alpha^1(a) &= G^1(a, \text{Start}) \\
\alpha^j(a) &= \sum_{a'} \alpha^{j-1}(a')G^j(a, a') \\
\beta^M(b) &= 1 \\
\beta^j(b) &= \sum_{b'} \beta^{j+1}(b')G^{j+1}(b', b)
\end{align*}
\]

\[
P(y^j = b, y^{j-1} = a \mid x) = \frac{\alpha^{j-1}(a)G^j(b, a)\beta^j(b)}{Z(x)}
\]

\[
Z(x) = \sum_{y'} \exp\{F(y', x)\} \\
F(y, x) = \sum_{j=1}^{M} \left( A_{y^j, y^{j-1}} + O_{y^j, x^j} \right) \\
G^j(b, a) = \exp\left\{ A_{b,a} + O_{b,x^j} \right\}
\]

Path Interpretation

Total Weight of paths from “Start” to “V” in 3\textsuperscript{rd} step

\[ \alpha^1 \]
\[ \alpha^1(V) \]
\[ \alpha^1(D) \]
\[ \alpha^1(N) \]

\[ \alpha^2 \]
\[ \alpha^2(V) \]
\[ \alpha^2(D) \]
\[ \alpha^2(N) \]

\[ \alpha^3 \]
\[ \alpha^3(V) \]
\[ \alpha^3(D) \]
\[ \alpha^3(N) \]

\[ G^1(V,"Start") \]
\[ G^1(N,"Start") \]

\[ \beta \] just does it backwards

x \[ G^2(D,N) \]

x \[ G^3(N,D) \]
Matrix Formulation

• Use Matrices!

• Fast to compute!

• Easy to implement!
Path Interpretation:

Forward-Backward vs Viterbi

- **Forward (and Backward) sums over all paths**
  - Computes expectation of reaching each state
  - E.g., total (un-normalized) probability of $y^3=$Verb over all possible $y^{1:2}$

- **Viterbi only keeps the best path**
  - Computes best possible path to reaching each state
  - E.g., single highest probability setting of $y^{1:3}$ such that $y^3=$Verb
Summary: Training CRFs

• Similar optimality condition as HMMs:
  – Match frequency counts of model components!
  \[
  \sum_{i=1}^{N} \sum_{j=1}^{M_i} \sum_{(y_i^j, y_{i-1}^j) = (b,a)} 1 = \sum_{i=1}^{N} \sum_{j=1}^{M_i} P(y_i^j = b, y_{i-1}^j = a | x_i)
  \]
  – Except HMMs can just set the model using counts
  – CRFs need to do gradient descent to match counts

• Run Forward-Backward for expectation
  – Just like HMMs as well
Summary: CRFs

• Log-Linear Sequential Model:

\[ P(y \mid x) = \frac{\exp\{F(y, x)\}}{Z(x)} \]

\[ F(y, x) = \sum_{j=1}^{M} \left( A_{y_j, y_{j-1}} + O_{y_j, x_j} \right) \]

\[ Z(x) = \sum_{y'} \exp\{F(y', x)\} \]

• Same #parameters as HMMs
  – Focused on learning \( P(y \mid x) \)
  – Prediction via Viterbi
  – Gradient Descent via Forward-Backward
Next Lecture

• More General Formulation of CRFs
  – More concise notation
    • Matches logistic regression notation
    • Matches course notes (later this week)
  – Easier to reason about for implementation

• General Structured Prediction