Conditional Random Fields (CRFs)

- Notation: for convenience we’ll use $x$ to refer to the sequence of input words, $x_1 \ldots x_m$, and $s$ to refer to a sequence of possible states, $s_1 \ldots s_m$. The set of possible states is $S$. We use $\mathcal{Y}$ to refer to the set of all possible state sequences (we have $|\mathcal{Y}| = |S|^m$).

- We’re again going to build a model of

$$p(s_1 \ldots s_m | x_1 \ldots x_m) = p(s | x)$$
CRFs

- We use $\Phi(x, s) \in \mathbb{R}^d$ to refer to a feature vector for an entire state sequence.
- We then build a giant log-linear model,

$$
p(s|x; w) = \frac{\exp(w \cdot \Phi(x, s))}{\sum_{s' \in \mathcal{Y}} \exp(w \cdot \Phi(x, s'))}
$$

- The model is “giant” in the sense that: 1) the space of possible values for $s$, i.e., $\mathcal{Y}$, is huge. 2) The normalization constant (denominator in the above expression) involves a sum over a huge number of possibilities (i.e., all members of $\mathcal{Y}$).
CRFs (continued)

\[ p(s|\mathbf{x}; \mathbf{w}) = \frac{\exp (\mathbf{w} \cdot \Phi(\mathbf{x}, s))}{\sum_{s' \in \mathcal{Y}} \exp (\mathbf{w} \cdot \Phi(\mathbf{x}, s'))} \]

▶ How do we define \( \Phi(\mathbf{x}, s) \)? Answer:

\[ \Phi(\mathbf{x}, s) = \sum_{j=1}^{m} \phi(\mathbf{x}, j, s_{j-1}, s_{j}) \]

where \( \phi(\mathbf{x}, j, s_{j-1}, s_{j}) \) are the same as the feature vectors used in MEMMMs.
Decoding with CRFs

- The decoding problem: find

\[
\arg \max_{s \in Y} p(s|x; w) = \arg \max_{s \in Y} \frac{\exp (w \cdot \Phi(x, s))}{\sum_{s' \in Y} \exp (w \cdot \Phi(x, s'))}
\]

\[
= \arg \max_{s \in Y} \exp (w \cdot \Phi(x, s))
\]

\[
= \arg \max_{s \in Y} w \cdot \Phi(x, s)
\]

\[
= \arg \max_{s \in Y} w \cdot \sum_{j=1}^{m} \phi(x, j, s_{j-1}, s_{j})
\]

\[
= \arg \max_{s \in Y} \sum_{j=1}^{m} w \cdot \phi(x, j, s_{j-1}, s_{j})
\]

- Again, we can use the Viterbi algorithm...
The Viterbi Algorithm for CRFs

- **Initialization:** for $s \in S$
  \[
  \pi[1, s] = w \cdot \phi(x, 1, s_0, s)
  \]
  where $s_0$ is a special “initial” state.

- For $j = 2 \ldots m$, $s = 1 \ldots k$:
  \[
  \pi[j, s] = \max_{s' \in S} \left[ \pi[j - 1, s'] + w \cdot \phi(x, j, s', s) \right]
  \]

- We then have
  \[
  \max_{s_1 \ldots s_m} \sum_{j=1}^{m} w \cdot \phi(x, j, s_{j-1}, s_j) = \max_{s} \pi[m, s]
  \]

- The algorithm runs in $O(mk^2)$ time. As before (see HMM lecture slides), we can use backpointers to recover the most likely sequence of states.
Parameter Estimation in CRFs

- To estimate the parameters, we assume we have a set of $n$ labeled examples, $\{(x^i, s^i)\}_{i=1}^n$. Each $x^i$ is an input sequence $x_1^i \ldots x_m^i$, each $s^i$ is a state sequence $s_1^i \ldots s_m^i$.
- We then proceed in exactly the same way as for regular log-linear models.
- The regularized log-likelihood function is

$$L(w) = \sum_{i=1}^n \log p(s^i | x^i; w) - \frac{\lambda}{2} ||w||^2$$

- Our parameter estimates are

$$w^* = \arg \max_{w \in \mathbb{R}^d} \sum_{i=1}^n \log p(s^i | x^i; w) - \frac{\lambda}{2} ||w||^2$$

- We find the optimal parameters using gradient-based methods.
The Structured Perceptron

- Input: labeled examples, \( \{(x^i, s^i)\}_{i=1}^n \).
- Initialization: \( \underline{w} = 0 \)
- For \( t = 1 \ldots T \), for \( i = 1 \ldots n \):
  - Use the Viterbi algorithm to calculate
    \[
    \underline{s}^* = \arg \max_{\underline{s} \in \mathcal{Y}} \underline{w} \cdot \Phi(x^i, \underline{s}) = \arg \max_{\underline{s} \in \mathcal{Y}} \sum_{j=1}^{m} \underline{w} \cdot \phi(x, j, s_j - 1, s_j)
    \]
  - Updates:
    \[
    \underline{w} = \underline{w} + \Phi(x^i, \underline{s}^i) - \Phi(x^i, \underline{s}^*)
    \]
    \[
    = \underline{w} + \sum_{j=1}^{m} \phi(x, j, s^i_{j-1}, s^i_j) - \sum_{j=1}^{m} \phi(x, j, s^*_j - 1, s^*_j)
    \]
- Return \( \underline{w} \)
The Structured Perceptron with Averaging

- Input: labeled examples, \( \{(x^i, s^i)\} \) \( n \)
  
  Initialization: \( w = 0, w_a = 0 \)

- For \( t = 1 \ldots T \), for \( i = 1 \ldots n \):
  
  Use the Viterbi algorithm to calculate

  \[
  s^* = \arg \max_{s \in Y} w \cdot \Phi(x^i, s) = \arg \max_{s \in Y} \sum_{j=1}^{m} w \cdot \phi(x, j, s_{j-1}, s_j)
  \]

- Updates:

  \[
  w = w + \Phi(x^i, s^i) - \Phi(x^i, s^*)
  \]

  \[
  = w + \sum_{j=1}^{m} \phi(x, j, s^i_{j-1}, s^i_j) - \sum_{j=1}^{m} \phi(x, j, s^*_j_{j-1}, s^*_j)
  \]

  \[
  w_a = w_a + w
  \]

- Return \( w_a/nT \)
Convergence of the Structured Perceptron

Definition: The training set \( \{(x^i, s^i)\}_{i=1}^n \) is separable with margin \( \delta > 0 \), if there exists some parameter vector \( \mathbf{w} \) such that:

1. \( \| \mathbf{w} \|^2 = 1 \)
2. For all \( i = 1 \ldots n \), for all \( s_1 \ldots s_m \) such that \( s_j \neq s_j^i \) for some \( j \),
   \[ \mathbf{w} \cdot \Phi(x^i, s^i) - \mathbf{w} \cdot \Phi(x^i, s) \geq \delta \]

Theorem: If a training set is separable with margin \( \delta \), the structured perceptron makes at most
\[ \frac{R^2}{\delta^2} \]
mistakes before convergence, where \( R \) is related to the norm of the feature vectors \( \Phi(x^i, s) \)