Sequence Models 4

CS 287
procedure BackwardViterbi

\[
\pi \in \mathbb{R}^{(n+1) \times C} \text{ initialized to } -\infty
\]
\[
\pi[n + 1, \langle /s \rangle] = 0
\]

for \( i = n \) to 1 do
  for \( c_i \in C \) do
    \[
    \pi[i, c_i] = \max_{c_{i+1}'} \pi[i + 1, c_{i+1}'] + \log \hat{y}(c_i) c_{i+1}'
    \]
  
return \( \max_{c_1 \in C} \pi[1, c_1] \)
Review: Edge Marginal

Mayor DeBlasio from New York
Assume we are not given $c_{1:i-1}$ and $c_{i+1:n}$.

The best completed sequence, i.e.

$$p(y_i = \delta(c_i)|x)$$
Answer: Marginalization

- Similar idea. Score involving \( c_i \) are local \((i-1\) and \(i+1)\).

\[
p(y_i = \delta(c'_i)|x) = \sum_{c_{i-1}:c_i+1:n} p(y_i = \delta(c'_i), y_{1:i-1,i+1:n}|x) \\
= \sum_{c_{i-1}} p(y_{1:i-1}|x)p(y_i = \delta(c'_i)|y_{i-1}, x) \\
\times \sum_{c_{i+1}:n} p(y_{i+1}|y_i = , x)p(y_{i+1:n}|x) \\
= \sum_{c_{i-1}} \hat{y}(c_i-1)_{c'_i} \prod_{j=1}^{i-1} \hat{y}(c_j-1)_{c_j} \\
\times \sum_{c_{i+1}:n} \hat{y}(c_{i+1})_{c_{i+1}} \prod_{j=i+1}^{n} \hat{y}(c_j)_{c_{j+1}}
\]
Answer: Marginalization

- Similar idea. Score involving $c_i$ are local ($i - 1$ and $i + 1$).

$$p(y_i = \delta(c'_i)|x) = \sum_{c_{1:i-1}:c_{i+1:n}} p(y_i = \delta(c'_i), y_{1:i-1,i+1:n}|x)$$

$$= \sum_{c_{1:i-1}} p(y_{1:i-1}|x)p(y_i = \delta(c'_i)|y_{i-1}, x)$$

$$\times \sum_{c_{i+1:n}} p(y_{i+1}|y_i = , x)p(y_{i+1:n}|x)$$

$$= \sum_{c_{1:i-1}} \hat{y}(c_{i-1}) c'_i \prod_{j=1}^{i-1} \hat{y}(c_{j-1}) c_j$$

$$\times \sum_{c_{i+1:n}} \hat{y}(c'_i) c_{i+1} \prod_{j=i+1}^{n} \hat{y}(c_j) c_{j+1}$$
Review: Edge Marginals

\[
\hat{y}(c_i') c'_{i+1} \times \sum_{c_{1:i-1}} \hat{y}(c_{i-1}) c'_i \prod_{j=1}^{i-1} \hat{y}(c_{j-1}) c_j \\
\times \sum_{c_{i+2:n}} \hat{y}(c_{i+1}) c_{i+1} \prod_{j=i+1}^{n} \hat{y}(c_j) c_{j+1}
\]

- Compute \( \alpha \) using Forward
- Compute \( \beta \) using Backward
- Multiply in the edge

\[
\hat{y}(c_i') c'_{i+1} \times \alpha[i, c_i'] \times \beta[i + 1, c'_{i+1}]
\]
Quiz

Last class we looked at discriminative sequence models $p(y|x)$. Consider now a generative model (such as an HMM), where we model $p(y, x)$. Unlike a discriminative model, we can use to compute the probability of a specific $x$ by marginalizing out $y$, $p(x) = \sum_{c_{1:n}} p(y = \delta(c_{1:n}), x)$.

▶ How do you compute this?

▶ What value should this same algorithm give you in the discriminative case?
\[ p(x) = \sum_{c_1:n} p(y = \delta(c_{1:n}), x) \]

Return value here.

**procedure** `FORWARD`

\[ \alpha \in \mathbb{R}^{\{0,\ldots,n\} \times C} \text{ initialized to } -\infty \]

\[ \alpha[0, \{s\}] = 0 \]

for \( i = 1 \) to \( n \) do

for \( c_i \in C \) do

\[ \alpha[i, c_i] = \sum_{c_{i-1}} \alpha[i - 1, c_{i-1}] \cdot \hat{y}(c_{i-1}) c_i \]

return \( \sum_{c_n \in C} \alpha[n, c_n] \)

- In the discriminative case, sums to 1 (nice unit test)
Sequence Models Zoology

- Generative versus Discriminative Model
- Local versus Sequence Prediction
- Probabilistic versus Non-probabilistic Objective
- Markov versus Non-Markov Model
- Linear versus Non-Linear Model
Examples of discriminative sequence model with local prediction

<table>
<thead>
<tr>
<th>Linear</th>
<th>MEMM</th>
<th>LR with global features</th>
</tr>
</thead>
<tbody>
<tr>
<td>Non-Linear (NN)</td>
<td>NNLM</td>
<td>RNN (transducer)</td>
</tr>
</tbody>
</table>

Markov \( \hat{y}(c_{i-1}) \)  
Non-Markov \( \hat{y}(c_1, \ldots, c_{i-1}) \)
Examples of linear discriminative models

\[ p(y|x) \]

<table>
<thead>
<tr>
<th></th>
<th>Local</th>
<th>Sequence</th>
</tr>
</thead>
<tbody>
<tr>
<td>Probabilistic</td>
<td>MEMM</td>
<td>CRF (new)</td>
</tr>
<tr>
<td>Non-Probabilistic</td>
<td>N/A</td>
<td>Structured Perceptron/SVM</td>
</tr>
</tbody>
</table>
Examples of linear, generative probabilistic models

\[ p(x, y) \]

<table>
<thead>
<tr>
<th>Local</th>
<th>Sequence</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linear</td>
<td>HMM</td>
</tr>
</tbody>
</table>
Contents

Local Prediction in Sequence Models

Conditional Random Fields

Training
Benefits of Local Prediction Markov Models

- Relatively easy to train (multi-class)
- Particularly convenient to use with NN ($\hat{y}(c_i)$)
- Can use same decoding algorithms (Viterbi, forward, backward)
Review: Entropy of a Distribution

- Recall: entropy of distribution

\[ H(y) = - \sum_i p(y_i) \log p(y_i) \]
Issue: Label Bias (Bottou, 1991)

- Local normalization can lead to pathological sequence scores $f$.
- Issue: low-entropy (spiky) transitions $y(c_{i-1})$
- Can cause the model to ignore input $x_i$
Label Bias Example 1

The United Nations will meet

- Correct example, should have a high score.

\[ f(x, c_{1,n}) \]
Label Bias Example 2

The United Nations will meet

- Very incorrect example, should have a low score.

\[ f(x, c_{1,n}) \]
Label Bias Example 3

The United Nations will meet

- Correct example, should have a high score.

\[ f(x, c_{1,n}) = \log(0.5) + \log(1.0) + \log(1.0) + \log(1.0) \]
Label Bias Example 4

The United Nations will meet

- Correct example, should have a high score.

\[ f(x, c_{1,n}) = \log(0.5) + \log(1.0) + \log(1.0) + \log(1.0) \]

- Very incorrect example, should have a low score.

\[ f(x, c_{1,n}) = \log(0.5) + \log(1.0) + \log(1.0) + \log(1.0) \]
Issue: Local Normalization

The United Nations will meet

- At I-LOC, we only have 4 choices, 2 of which have 0 prob.
- Of the option only I-LOC makes sense (definitely not O).
- Local model, cannot report current path is wrong
- Effectively ignores input Nations
Further Issues

- Note: this is a modeling issue, not a search issue.
- i.e. failure even with exact search.
- Related issue: Exposure Bias.
- Training never condition on incorrect decisions.
Contents

Local Prediction in Sequence Models

Conditional Random Fields

Training
Issues with Multiclass for Sequences (3rd time!)

- Say there are $C$ tags and sequence length is $n$
- There are $d_{out} = O(C^n)$ sequences!
- Just naively computing the softmax is exponential in length.
- Even if you could compute the softmax, $\mathbf{W} \in \mathbb{R}^{d_{in} \times d_{out}}$ would be impossible to train.
(Linear Chain) Conditional Random Field (Lafferty et al, 2001)

- Model consists of unnormalized weights

\[ \log \hat{y}(c_{i-1})_{c_i} = \text{feat}(x, c_{i-1}) \mathbf{W} + \mathbf{b} \]

- Out of log space,

\[ \hat{y}(c_{i-1})_{c_i} = \exp(\text{feat}(x, c_{i-1}) \mathbf{W} + \mathbf{b}) \]

- Score of the sequence, (same as last few classes)

\[ f(x, c_{1:n}) = \sum_{i=1}^{n} \log \hat{y}(c_{i-1})_{c_i} \]

- Objective is based on global NLL of this sequence distribution

\[ \mathbf{z}_{c_{1:n}} = f(x, c_{1:n}) \]
Distribution over Sequences

- How do we compute the probability of sequences?
- Softmax over scores,

\[
p(y = \delta(c_1:n) | x) = \text{softmax}(f(x, c_1:n))
\]

- What does this look like?

\[
p(y = \delta(c_1:n) | x) = \frac{\prod_{i=1}^{n} \hat{y}(c_{i-1})_{c_i}}{\sum_{c_{1:n}'} \prod_{i=1}^{n} \hat{y}(c_{i-1}')_{c_i'}}
\]

\[
= \frac{\exp \left( \sum_{i=1}^{n} \log \hat{y}(c_{i-1})_{c_i} \right)}{\sum_{c_{1:n}'} \exp \left( \sum_{i=1}^{n} \log \hat{y}(c_{i-1}')_{c_i'} \right)}
\]
Computing the Softmax

Want to compute:

\[
p(y = \delta(c_{1:n})|x) = \frac{\prod_{i=1}^{n} \hat{y}(c_{i-1})_{c_i}}{\sum_{c'_{1:n}} \prod_{i=1}^{n} \hat{y}(c'_{i-1})_{c'_i}}
\]

- \(\prod_{i=1}^{n} \hat{y}(c_{i-1})_{c_i}\); easy to compute

- \(\sum_{c'_{1:n}} \prod_{i=1}^{n} \hat{y}(c'_{i-1})_{c'_i}\); can use forward algorithm.

Softmax goes from \(O(|C|^n)\) to \(O(|C|^2)\).
Forward Algorithm

\begin{align*}
&\textbf{procedure} \text{ FORWARD} \\
&\alpha \in \mathbb{R}^{\{0,\ldots,n\} \times C} \\
&\alpha[0, \langle s \rangle] = 1 \\
&\text{for } i = 1 \text{ to } n \text{ do} \\
&\quad \text{for } c_i \in C \text{ do} \\
&\quad\quad \alpha[i, c_i] = \sum_{c_{i-1}} \alpha[i - 1, c_{i-1}] \times \hat{y}(c_{i-1}) c_i \\
&\text{return } \alpha
\end{align*}
Computing Marginals

Want to compute:

\[ p(y_i = \delta_{c_i} | x) = \sum_{c_{1:i-1}, c_{i+1:n}} p(y | x) \]

\[ = \left( \sum_{c_{1:i-1}} \prod_{j=1}^{i-1} y(c_{j-1}) c_j \right) \left( \sum_{c_{i+1:n}} \prod_{j=i+1}^{n} y(c_{j-1}) c_j \right) \]

\[ = \sum_{c'_{1:n}} \prod_{i=1}^{n} \hat{y}(c'_{i-1}) c'_i \]

\[ \sum_{i=1}^{i-1} \prod_{j=1}^{i-1} y(c_{j-1}) c_j; \text{ forward algorithm} \]

\[ \sum_{i=i+1}^{n} \prod_{j=i+1}^{n} y(c_{j-1}) c_j; \text{ backward algorithm} \]
Local Prediction in Sequence Models

Conditional Random Fields

Training
How do you fit these models?

- Same objective as for MEMMs.
- Minimize sequence NLL,

\[ \mathcal{L}(\theta) = -\sum_{j=1}^{J} \log p(y^{(j)}|x^{(j)}; \theta) \]

- However, very different training procedure.
Recall: Deriving Logistic Regression update

\[ \mathcal{L}(\theta) = - \sum_{j=1}^{J} \log p(y^{(j)}|x^{(j)}; \theta) \]

And define

\[ p(y|x; \theta) = \hat{y} = \text{softmax}(z) \]

Where \( z \in \mathbb{R}^{|C|} \) was the score of each class.
Recall: Log-likelihood and softmax partials

- Partial of $L(y, \hat{y})$ for all $j \in \{1, \ldots, d_{out}\}$ and $y_c = 1$
  \[
  \frac{\partial L(y, \hat{y})}{\partial \hat{y}_j} = \begin{cases} 
  -\frac{1}{\hat{y}_j} & j = c \\
  0 & \text{o.w.}
  \end{cases}
  \]

- Partial of $\hat{y} = \text{softmax}(z)$
  \[
  \frac{\partial \hat{y}_j}{\partial z_i} = \begin{cases} 
  \hat{y}_i(1 - \hat{y}_i) & i = j \\
  -\hat{y}_i\hat{y}_j & i \neq j
  \end{cases}
  \]

- Partial
  \[
  \frac{\partial L(y, \hat{y})}{\partial z_i} = \begin{cases} 
  -(1 - p(y = \delta(i))) & i = c \\
  p(y = \delta(i)) & i \neq c
  \end{cases}
  \]
CRF update

\[
\mathcal{L}(\theta) = - \sum_{j=1}^{J} \log p(y^{(j)}|x^{(j)}; \theta)
\]

Define

\[
p(y|x; \theta) = \hat{y} = \text{softmax}(z)
\]

Where \( z \in \mathbb{R}^{|C|^n} \) was the score of each sequence, i.e. \( z_{c_{1:n}} \)

And let \( c_{1:n} \) be correct sequence
What is happening here?

- Partials for all sequences $d_{1:n} \in \mathcal{C}^n$,

$$\frac{\partial L}{\partial z_{d_{1:n}}} = \begin{cases} -(1 - \hat{y}_i) & d_{1:n} = c_{1:n} \\ \hat{y}_i & d_{1:n} \neq c_{1:n} \end{cases}$$

- Partials for all edges

$$\frac{\partial z_{d_{1:n}}}{\partial \log \hat{y}(c'_{i-1}) c'_i} = \begin{cases} 1 & c'_{i-1} = d_{i=1} \land c'_i = d_i \\ 0 & o.w. \end{cases}$$
Final Gradients

\[
\frac{\partial L}{\partial \log \hat{y}_i(c'_{i-1})} = \sum_{d_{1:n}} \frac{\partial z_{d_{1:n}}}{\partial \log \hat{y}_i(c'_{i-2})} \frac{\partial L}{\partial z_{d_{1:n}}} \\
= \sum_{c'_{1:i-2}, c'_{i+1:n}} \frac{\partial L}{\partial z_{c'_{1:n}}} \\
= p(y_{i-1} = c'_{i-1}, y_i = c'_i | x) - \mathbf{1}(c'_{i-1} = c_{i-1} \land c'_i = c_i)
\]

- First term, marginals of the CRF.
- Second term, indicator of whether edge is in gold.
CRF Training Algorithm

- Compute forward algorithm
- Compute partition
- Compute backward algorithm
- Compute the edge marginals
- Compute and backprop gradients to each $\log y(c_i)$. 

$\hat{y}$
The United Nations will meet
Model Choices

Discriminative, Markov Models

<table>
<thead>
<tr>
<th>Normalization</th>
<th>Local</th>
<th>Global</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linear</td>
<td>MEMM</td>
<td>CRF</td>
</tr>
<tr>
<td>Non-Linear</td>
<td>NN-MM</td>
<td>NN-CRF</td>
</tr>
</tbody>
</table>