Modeling Sequence Data

CS4780/5780 – Machine Learning
Fall 2012

Thorsten Joachims
Cornell University

Reading:
Manning/Schuetze, Sections 9.1-9.3 (except 9.3.1)
Leeds Online HMM Tutorial (except Forward and Forward/Backward Algorithm)
Outline

• Markov Models in Classification
  – A “less naïve” Bayes for text classification

• Hidden Markov Models
  – Part-of-speech tagging
  – Viterbi Algorithm
  – Estimation with fully observed training data
“Less Naïve” Bayes Classifier

• Example: Classify sentences as insulting / not insulting

<table>
<thead>
<tr>
<th>text</th>
<th>Insult?</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \overline{x}_1 ) = (Peter, is, nice, and, not, stupid)</td>
<td>-1</td>
</tr>
<tr>
<td>( \overline{x}_2 ) = (Peter, is, not, nice, and, stupid)</td>
<td>+1</td>
</tr>
</tbody>
</table>

• Assumption (l words in document)
  
  - \( P(X = x | Y = +1) \)
    
    \[ = P(W_1 = w_1 | Y = +1) \prod_{i=2}^{l} P(W_i = w_i | W_{i-1} = w_{i-1}, Y = +1) \]
  
  - \( P(X = x | Y = -1) \)
    
    \[ = P(W_1 = w_1 | Y = -1) \prod_{i=2}^{l} P(W_i = w_i | W_{i-1} = w_{i-1}, Y = -1) \]

• Decision Rule

\[
h_{\text{less}}(x) = \arg\max_{y \in \{+1, -1\}} \left\{ P(Y = y)P(W_1 = w_1 | Y = y) \prod_{i=2}^{l} P(W_i = w_i | W_{i-1} = w_{i-1}, Y = y) \right\}
\]
Markov Model

• Definition
  – Set of States: \( s_1, \ldots, s_k \)
  – Start probabilities: \( P(S_1 = s) \)
  – Transition probabilities: \( P(S_i = s \mid S_{i-1} = s') \)

• Random walk on graph
  – Start in state \( s \) with probability \( P(S_1 = s) \)
  – Move to next state with probability \( P(S_i = s \mid S_{i-1} = s') \)

• Assumptions
  – Limited dependence: Next state depends only on previous state, but no other state (i.e. first order Markov model)
  – Stationary: \( P(S_i = s \mid S_{i-1} = s') \) is the same for all \( i \)
Part-of-Speech Tagging Task

• Assign the correct part of speech (word class) to each word in a document
  “The/DT planet/NN Jupiter/NNP and/CC its/PRP moons/NNS are/VBP in/IN effect/NN a/DT mini-solar/JJ system/NN ,/, and/CC Jupiter/NNP itself/PRP is/VBZ often/RB called/VBN a/DT star/NN that/IN never/RB caught/VBN fire/NN ./.”

• Needed as an initial processing step for a number of language technology applications
  – Information extraction
  – Answer extraction in QA
  – Base step in identifying syntactic phrases for IR systems
  – Critical for word-sense disambiguation (WordNet apps)
  – ...
Why is POS Tagging Hard?

• Ambiguity
  – He will race/VB the car.
  – When will the race/NN end?
  – I bank/VB at CFCU.
  – Go to the bank/NN!

• Average of ~2 parts of speech for each word
  – The number of tags used by different systems varies a lot. Some systems use < 20 tags, while others use > 400.
The POS Learning Problem

- Example

<table>
<thead>
<tr>
<th>sentence</th>
<th>POS</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\bar{x}_1 = (I, bank, at, CFCU)$</td>
<td>$\bar{y}_1 = (PRP, V, PREP, N)$</td>
</tr>
<tr>
<td>$\bar{x}_2 = (Go, to, the, bank)$</td>
<td>$\bar{y}_2 = (V, PREP, DET, N)$</td>
</tr>
</tbody>
</table>
Hidden Markov Model for POS Tagging

• States
  – Think about as nodes of a graph
  – One for each POS tag
  – special start state (and maybe end state)

• Transitions
  – Think about as directed edges in a graph
  – Edges have transition probabilities

• Output
  – Each state also produces a word of the sequence
  – Sentence is generated by a walk through the graph
Hidden Markov Model

- States: \( y \in \{s_1, ..., s_k\} \)
- Outputs symbols: \( x \in \{o_1, ..., o_m\} \)
- Starting probability \( P(Y_1 = y_1) \)
  - Specifies where the sequence starts
- Transition probability \( P(Y_i = y_i \mid Y_{i-1} = y_{i-1}) \)
  - Probability that one state succeeds another
- Output/Emission probability \( P(X_i = x_i \mid Y_i = y_i) \)
  - Probability that word is generated in this state

=> Every output+state sequence has a probability

\[
P(x, y) = P(x_1, ..., x_l, y_1, ..., y_l) \\
= P(y_1)P(x_1|y_1) \prod_{i=2}^{l} P(x_i|y_i)P(y_i|y_{i-1})
\]
Estimating the Probabilities

• Given: Fully observed data
  – Pairs of output sequence with their state sequence
• Estimating transition probabilities $P(Y_i | Y_{i-1})$
  \[ P(Y_i = a | Y_{i-1} = b) = \frac{\# \text{ of times state } a \text{ follows state } b}{\# \text{ of times state } b \text{ occurs}} \]
• Estimating emission probabilities $P(X_i | Y_i)$
  \[ P(X_i = a | Y_i = b) = \frac{\# \text{ of times output } a \text{ is observed in state } b}{\# \text{ of times state } b \text{ occurs}} \]
• Smoothing the estimates
  – Laplace smoothing -> uniform prior
  – See naïve Bayes for text classification
• Partially observed data
  – Expectation Maximization (EM)
### Viterbi Example

| $P(X_i | Y_i)$ | I | bank | at | CFCU | go | to | the |
|---------------|---|------|----|------|----|----|-----|
| DET           | 0.01 | 0.01 | 0.01 | 0.01 | 0.01 | 0.01 | 0.94 |
| PRP           | 0.94 | 0.01 | 0.01 | 0.01 | 0.01 | 0.01 | 0.01 |
| N             | 0.01 | 0.4  | 0.01 | 0.4  | 0.16 | 0.01 | 0.01 |
| PREP          | 0.01 | 0.01 | 0.48 | 0.01 | 0.01 | 0.47 | 0.01 |
| V             | 0.01 | 0.4  | 0.01 | 0.01 | 0.55 | 0.01 | 0.01 |

<table>
<thead>
<tr>
<th>$P(Y_1)$</th>
<th>DET</th>
<th>PRP</th>
<th>N</th>
<th>PREP</th>
<th>V</th>
</tr>
</thead>
<tbody>
<tr>
<td>DET</td>
<td>0.3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PRP</td>
<td>0.3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>N</td>
<td>0.1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PREP</td>
<td>0.1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>V</td>
<td>0.2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| $P(Y_i | Y_{i-1})$ | DET | PRP | N | PREP | V |
|-----------------|-----|-----|---|------|---|
| DET             | 0.01 | 0.01 | 0.96 | 0.01 | 0.01 |
| PRP             | 0.01 | 0.01 | 0.01 | 0.2  | 0.77 |
| N               | 0.01 | 0.2  | 0.3  | 0.3  | 0.19 |
| PREP            | 0.3  | 0.2  | 0.3  | 0.19 | 0.01 |
| V               | 0.2  | 0.19 | 0.3  | 0.3  | 0.01 |
HMM Decoding: Viterbi Algorithm

• Question: What is the most likely state sequence given an output sequence
  – Given fully specified HMM:
    • \( P(Y_1 = y_1) \),
    • \( P(Y_i = y_i \mid Y_{i-1} = y_{i-1}) \),
    • \( P(X_i = x_i \mid Y_i = y_i) \)
  – Find \( y^* = \arg\max_{y \in \{y_1, \ldots, y_l\}} P(x_1, \ldots, x_l, y_1, \ldots, y_l) \)
    \[
    = \arg\max_{y \in \{y_1, \ldots, y_l\}} \left\{ P(y_1) P(x_1 \mid y_1) \prod_{i=2}^{l} P(x_i \mid y_i) P(y_i \mid y_{i-1}) \right\}
    \]
  – “Viterbi” algorithm has runtime linear in length of sequence
  – Example: find the most likely tag sequence for a given sequence of words
HMM’s for POS Tagging

• Design HMM structure (vanilla)
  – States: one state per POS tag
  – Transitions: fully connected
  – Emissions: all words observed in training corpus

• Estimate probabilities
  – Use corpus, e.g. Treebank
  – Smoothing
  – Unseen words?

• Tagging new sentences
  – Use Viterbi to find most likely tag sequence
### Experimental Results

<table>
<thead>
<tr>
<th>Tagger</th>
<th>Accuracy</th>
<th>Training time</th>
<th>Prediction time</th>
</tr>
</thead>
<tbody>
<tr>
<td>HMM</td>
<td>96.80%</td>
<td>20 sec</td>
<td>18.000 words/s</td>
</tr>
<tr>
<td>TBL Rules</td>
<td>96.47%</td>
<td>9 days</td>
<td>750 words/s</td>
</tr>
</tbody>
</table>

- Experiment setup
  - WSJ Corpus
  - Trigram HMM model
  - Lexicalized
  - from [Pla and Molina, 2001]
Discriminative vs. Generative

• Bayes Rule

\[ h_{\text{bayes}}(x) = \arg\max_{y \in Y} \left[ P(Y = y|X = x) \right] \]

\[ = \arg\max_{y \in Y} \left[ P(X = x|Y = y)P(Y = y) \right] \]

• Generative:
  – Make assumptions about \( P(X = x|Y = y) \) and \( P(Y = y) \)
  – Estimate parameters of the two distributions

• Discriminative:
  – Define set of prediction rules (i.e. hypotheses) \( H \)
  – Find \( h \) in \( H \) that best approximates the classifications made by

\[ h_{\text{bayes}}(x) = \arg\max_{y \in Y} \left[ P(Y = y|X = x) \right] \]

• Question: Can we train HMM’s discriminately?
  – Later in semester: discriminative training of HMM and general structured prediction.