Modeling Sequence Data

CS478 – Machine Learning
Spring 2008
Thorsten Joachims
Cornell University

Reading:
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

Markov Model

• Definition
  – Set of States: \( s_1, \ldots, s_k \)
  – Start probabilities: \( P(S=s) \)
  – Transition probabilities: \( P(S=s' | Sprev=s) \)

• Random walk on graph
  – Start in state \( s \) with probability \( P(S=s) \)
  – Move to next state with probability \( P(S=s' | Sprev=s) \)

• Assumptions
  – Limited dependence: Next state depends only on previous state, but no other state (i.e. first order Markov model)
  – Stationary: \( P(S=s | Sprev=s) \) does not change

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/VBP often/RB called/VBN a DT star/NN that/IN revolve/RB caused/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)

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

• Average of ~2 parts of speech for each word

Why is POS Tagging Hard?

• 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>POS</th>
<th>(, , , , , , )</th>
<th>( , , , , )</th>
<th>( , , )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prep</td>
<td>( , , , , )</td>
<td>( , , )</td>
<td>( , )</td>
</tr>
<tr>
<td>Prepos</td>
<td>( , , , , )</td>
<td>( , , )</td>
<td>( , )</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: \( y \in \{ s_1, \ldots, s_k \} \)
- Outputs symbols: \( x \in \{ o_1, \ldots, o_m \} \)
- Starting probability \( P(Y_1 = y_1) \)
  - Specifies where the sequence starts
- Transition probability \( P(Y_i = y_i | Y_{i-1} = y_{i-1}) \)
  - Probability that one states succeeds another
- Output/Emission probability \( P(X_i = x_i | Y_i = y_i) \)
  - Probability that word is generated in this state

\[ P(x_1, x_2, y_1, y_2) = \prod_i P(y_i | y_{i-1}) \prod_i P(x_i | y_i) P(y_1) \]

HMM Decoding: Viterbi Algorithm

- Question: What is the most likely state sequence given an output sequence
- Given fully specified HMM:
  - \( P(Y_1 = y_1) \)
  - \( P(Y_{i-1} = y_{i-1}) \)
  - \( P(X_i = x_i | Y_i = y_i) \)
- Find

\[ \max_{y_1, y_2, \ldots, y_n} \prod_i P(y_i | y_{i-1}) \prod_i P(x_i | y_i) P(y_1) \]

- “Viterbi” algorithm has runtime linear in length of sequence
- Example: find the most likely tag sequence for a given sequence of words

Viterbi Example

| \( P(X=x | Y=y) \) | I | bank | at | CFCU | ge | 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 |
| Y 0.01 0.4 0.01 0.01 0.55 0.01 0.01 |

| \( P(Y=y) \) | \( P(Y|y_{i-1}) \) | DET | PRP | N | PREP | V |
|-----------|----------------|-----|-----|---|------|---|
| DET 0.3 0.01 0.96 0.01 0.01 0.01 |
| PRP 0.3 0.01 0.01 0.01 0.2 0.77 |
| N 0.1 0.01 0.2 0.3 0.19 |
| PREP 0.1 0.3 0.19 0.3 0.19 |
| V 0.2 0.2 0.19 0.3 0.3 0.01 |

Estimating the Probabilities

- Given: Fully observed data
  - Pairs of output sequence with their state sequence
- Estimating transition probabilities \( P(S_{t+1} | S_t) \)

\[ P(Y_{i+1} | Y_i) = \frac{\text{# of Times State B Occurs}}{\text{# of Times State A Follows State B}} \]

- Estimating emission probabilities \( P(W | S) \)

\[ P(x_i | y_i) = \frac{\text{# of Times Output A Is Observed In State B}}{\text{# of Times State B Occurs}} \]

- Smoothing the estimates
  - Laplace smoothing -> uniform prior
  - See naïve Bayes for text classification
- Partially observed data: Expectation Maximization (EM)
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

\[ p(s|x) = \frac{p(x|s)p(s)}{p(x)} \]

Generative:

- Make assumptions about \( p(x|s)p(s) \)
- Estimate parameters of the two distributions

Discriminative:

- Define set of prediction rules (i.e. hypotheses) \( H \)
- Find \( h \) in \( H \) that best approximates

\[ p(s|x) = \arg\max_{s} p(x|s)p(s) \]

Question: Can we train HMM’s discriminately?

Idea for Discriminative Training of HMM

Bayes Rule

\[ p(s|x) = \frac{p(x|s)p(s)}{p(x)} \]

Model \( p(x|y=s) \) with \( \phi(x,y) \) so that

\[ (\arg\max_{s} p(x|s)p(s)) = (\arg\max_{s} \sum_{y} \phi(x,y)) \]

Intuition:

- Tune \( \phi \) so that correct \( y \) has the highest value of \( \sum_{x} \phi(x,y) \)
- \( \phi(x,y) \) is a feature vector that describes the match between \( x \) and \( y \)

Structural Support Vector Machine

- Joint features \( \phi(x,y) \) describe match between \( x \) and \( y \)
- Learn weights \( \theta \) so that \( \theta^T \phi(x,y) \) is max for correct \( y \)

\[ \sum_{x} \phi(x,y) \]

Structural SVM Training Problem

Hard-margin optimization problem:

\[ \min_{\theta} \frac{1}{2} \theta^T \theta \]

s.t. \( \forall y \in Y \setminus \{y\}: \theta^T \phi(x,y) \geq \theta^T \phi(x,y) + 1 \)

- Training Set: \((x_1,y_1), \ldots, (x_m,y_m) ~ P(X,Y)\)
- Prediction Rule: \( \hat{y}_{\text{disc}}(x) = \arg\max_{y} \theta^T \phi(x,y) \)
- Optimization:
  - Correct label \( y \) must have higher value of \( \theta^T \phi(x,y) \) than any incorrect label \( y \)
  - Find weight vector with smallest norm
  - Polynomial time algorithm (e.g. SVM-struct)
**Experiment: Part-of-Speech Tagging**

- **Task**: Given a sequence of words $x$, predict sequence of tags $y$.

  - The dog chased the cat → Det→ N→ V→ Det→ N

- Dependencies from tag-tag transitions in Markov model.

- **Model**: Markov model with one state per tag and words as emissions

  - Each word described by ~250,000 dimensional feature vector (all word suffixes/prefixes, word length, capitalization …)

- **Experiment (by Dan Fleisher)**

  - Train/test on 7966/1700 sentences from Penn Treebank

- **Input**

  - One feature for each possible output in each possible state

  - One feature for each possible transition

  - One feature for each possible start state

- **Loss Functions**: Soft-Margin Struct SVM

  - Loss function $\Delta(x, y)$ measures match between target and prediction.

- **Soft-Margin Structural SVM**

  - Soft-margin optimization problem:

    \[
    \min_{\theta} \frac{1}{2} \sum_{i,j} c_{ij} \theta_i \theta_j + \sum_i \xi_i
    \]

    \[
    \text{subject to } \sum_{j} \theta_j = 1, \theta_j \geq 0, \xi_i \geq 0
    \]

  - Lemma: The training loss is upper bounded by

    \[
    \text{Err}(\theta) = \frac{1}{n} \sum_{i} \Delta(x_i, y_i) \leq \frac{1}{n} \sum_{i} \xi_i
    \]

- **Loss Functions**: Soft-Margin Struct SVM

  - One feature for each possible output in each possible state

  - One feature for each possible transition

  - One feature for each possible start state

  - Feature values are counts

- **Sparse Approximation Algorithm for Structural SVM**

  - Input $(x_1, y_1), \ldots, (x_n, y_n), c_i$

  - $\mathcal{S} = \emptyset$

  - **REPEAT**

    - **FOR** $i = 1, \ldots, n$

      - compute $g_i = \arg\min_{g} \left\{ \Delta(x_i, y_i, g) + \frac{1}{2} \sum_{j} \theta_j g_j \right\}$

      - if $\Delta(x_i, y_i, g_i) > 0 + \varepsilon$

        - $g = g_i$

        - Add constraint to working set

      - **ENDIF**

    - **ENDFOR**

  - **UNTIL** $\mathcal{S}$ has not changed during iteration

- **NE Identification**

  - **Identify all named locations, named persons, named organizations, dates, times, monetary amounts, and percentages.**

  - The delegation, which included the commander of the army troops, went to the Swiss embassy to sign a new agreement for talks with Bosnian war leader Radovan Karadzic.

  - Esta ha sido el primer encuentro público de la presidenta Chávez respecto a la crisis de Haití. Minube, cuyo secretario de Estado, Warren Christopher, decidió regresar temporalmente para supervisar el proceso de paz y la concatenación de efectivas fuerzas en el país.
Experiment: Named Entity Recognition

- **Data**
  - Spanish Newswire articles
  - 300 training sentences
  - 9 tags
    - no-name,
    - beginning and continuation of person name, organization, location, misc name
  - Output words are described by features (e.g. starts with capital letter, contains number, etc.)
- **Error on test set (% mislabeled tags):**
  - Generative HMM: 9.36%
  - Support Vector Machine HMM: 5.08%

General Problem: Predict Complex Outputs

- **Supervised Learning from Examples**
  - Find function from input space $X$ to output space $Y$
    $$ h : X \rightarrow Y $$
  - such that the prediction error is low.
- **Typical**
  - Output space is just a single number
    - Classification: $-1, +1$
    - Regression: some real number
- **General**
  - Predict outputs that are complex objects

Examples of Complex Output Spaces

- **Natural Language Parsing**
  - Given a sequence of words $x$, predict the parse tree $y$.
  - Dependencies from structural constraints, since $y$ has to be a tree.

Examples of Complex Output Spaces

- **Multi-Label Classification**
  - Given a (bag-of-words) document $x$, predict a set of labels $y$.
  - Dependencies between labels from correlations between labels (“iraq” and “oil” in newswire corpus)

Examples of Complex Output Spaces

- **Noun-Phrase Co-reference**
  - Given a set of noun phrases $x$, predict a clustering $y$.
  - Structural dependencies, since prediction has to be an equivalence relation.
  - Correlation dependencies from interactions.