Cornell University

Modeling Sequence Data

CS4780/5780 – Machine Learning
Fall 2011

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)
(http://www.comp.leeds.ac.uk/roger/HiddenMarkovModels/html_dev/main.html)

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

Cornell University

- Example: Classify sentences as insulting/ not insulting

text	Insult?
$\bar{x}_1 = (Peter, is, nice, and, not, stupid)$	-1
$\bar{x}_2 = (Peter, is, not, nice, and, stupid)$	+1

- Assumption (l words in document)

$$P(X{=}\bar{x}|Y{=}+1) = P(W{=}w_i|Y{=}+1)\prod_{i=2}^{l} P(W{=}w_i|W_{prev}{=}w_{i-1}, Y{=}+1)$$

$$P(X{=}\bar{x}|Y{=}-1) = P(W{=}w_i|Y{=}-1)\prod_{i=2}^{l} P(W{=}w_i|W_{prev}{=}w_{i-1}, Y{=}-1)$$

- Decision Rule

$$h_{less}(\bar{x}) = argmax_{y\in\{+1,-1\}}\left\{P(Y = y)P(W{=}w_i|Y{=}y)\prod_{i=2}^{l} P(W{=}w_i|W_{prev}{=}w_{i-1}, Y{=}y)\right\}$$

Cornell University

Markov Model

- Definition
  - Set of States: $s_1, \ldots, s_k$
  - Start probabilities: $P(S=s)$
  - Transition probabilities: $P(S=s \mid S_{prev}=s')$
- Random walk on graph
  - Start in state s with probability $P(S=s)$
  - Move to next state with probability $P(S=s \mid S_{prev}=s')$
- Assumptions
  - Limited dependence: Next state depends only on previous state, but no other state (i.e. first order Markov model)
  - Stationary: $P(S=s \mid S_{prev}=s')$ 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/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

sentence	POS
$\bar{x}_1 = (I, bank, at, CFCU)$	$\bar{y}_1 = (PRP, V, PREP, N)$
$\bar{x}_2 = (Go, to, the, bank)$	$\bar{y}_2 = (V, PREP, DET, N)$

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 $P(Y = y_i \mid Y_{i-1} = y_{i-1})$
  - Probability that one states 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

$$P(\bar{x},\bar{y}) = P(x_1,\ldots,x_l,y_1,\ldots,y_l)$$

$$= \left[P(y_1)P(x_1|y_1)\prod_{i=2}^{l} P(x_i|y_i)P(y_i|y_{i-1}) \right]$$

Estimating the Probabilities

- Given: Fully observed data
  - Pairs of output sequence with their state sequence
- Estimating transition probabilities $P(Y_t|Y_{t-1})$

$$P(Y_i = a | Y_{i-1} = b) = \frac{\#\text{ of times state a follows state b}}{\#\text{ of times state b occurs}}$$

- Estimating emission probabilities $P(X_t|Y_t)$

$$P(X_i = a | Y_i = b) = \frac{\#\text{ of times output a is observed in state b}}{\#\text{ of times state b occurs}}$$

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

Viterbi Example

P(X=x|Y=y)	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

P(Y=y)	
DET	0.3
PRP	0.3
N	0.1
PREP	0.1
V	0.2

$P(Y\|Y_{prev})$	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 $\bar{y} = \underset{(y_1,\ldots,y_l)}{\operatorname{argmax}} P(x_1,\ldots,x_l,y_1,\ldots,y_l)$

$$= \underset{(y_1,\ldots,y_l)}{\operatorname{argmax}} \left\{ P(y_1)P(x_1|y_1)\prod_{i=2}^{l} P(x_i|y_i)P(y_i|y_{i-1}) \right\}$$

  - “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

Tagger	Accuracy	Training time	Prediction time
HMM	96.80%	20 sec	18.000 words/s
TBL Rules	96.47%	9 days	750 words/s

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

Discriminative vs. Generative

- Bayes Rule

$$\begin{aligned} h_{bayes}(x) &= argmax_{y \in Y} \left[P(Y = y | X = x) \right] \\ &= argmax_{y \in Y} \left[P(X = x | Y = y) P(Y = y) \right] \end{aligned}$$

- Generative:
  - Make assumptions about $P(X = x | Y = y), P(Y = y)$
  - Estimate parameters of the two distributions
- Discriminative:
  - Define set of prediction rules (i.e. hypotheses) H
  - Find h in H that best approximates

$$h_{bayes}(x) = argmax_{y \in Y} \left[P(Y = y | X = x) \right]$$

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