

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

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

- Assumption (l words in document)

$$P(\mathbf{X}=\vec{x}|Y=+1) = P(\mathbf{W}=\mathbf{w}_1|Y=+1) \prod_{i=2}^l P(\mathbf{W}=\mathbf{w}_i|W_{prev}=\mathbf{w}_{i-1}, Y=+1)$$

$$P(\mathbf{X}=\vec{x}|Y=-1) = P(\mathbf{W}=\mathbf{w}_1|Y=-1) \prod_{i=2}^l P(\mathbf{W}=\mathbf{w}_i|W_{prev}=\mathbf{w}_{i-1}, Y=-1)$$

- Decision Rule

$$y_{\text{class}}(\vec{x}) = \underset{y \in \{+1, -1\}}{\text{argmax}} \left\{ P(Y=y) P(\mathbf{W}=\mathbf{w}_1|Y=y) \prod_{i=2}^l P(\mathbf{W}=\mathbf{w}_i|W_{prev}=\mathbf{w}_{i-1}, Y=y) \right\}$$

Markov Model

- **Definition**
 - Set of States: s_1, \dots, s_k
 - Start probabilities: $P(S=s)$
 - Transition probabilities: $P(S=s / S_{prev}=s')$
- **Random walk on graph**
 - Start in state s with probability $P(S=s)$
 - Move to next state with probability $P(S=s / S_{prev}=s')$
- **Assumptions**
 - Limited dependence: Next state depends only on previous state, but no other state (i.e. first order Markov model)
 - Stationary: $P(S=s / S_{prev}=s')$ 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**/NOUN end?
 - I **bank**/VB at CFCU.
 - Go to the **bank**/NOUN!
- **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
$\vec{s}_1 = (I, \text{bank}, \text{at}, \text{CFCU})$	$\vec{y}_1 = (\text{PRP}, \text{V}, \text{PREP}, \text{N})$
$\vec{s}_2 = (\text{Go}, \text{to}, \text{the}, \text{bank})$	$\vec{y}_2 = (\text{V}, \text{PREP}, \text{DET}, \text{N})$

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: $y \in \{s_1, \dots, s_k\}$
 - Outputs symbols: $x \in \{o_1, \dots, o_m\}$
 - 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
- => Every output + state sequence has a probability

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

HMM Decoding: Viterbi Algorithm

- **Question: What is the most likely state sequence given an output sequence**

- Given fully specified HMM:

- $P(Y_i = y_i)$,
- $P(Y_i = y_i | Y_{i-1} = y_{i-1})$,
- $P(X_i = x_i | Y_i = y_i)$

- Find

$$\max_{P(y_1, \dots, y_n, x_1, \dots, x_n)} = \max_{(y_1, \dots, y_n)} \left[P(y_1) P(x_1 | y_1) \prod_{i=2}^n P(y_i | y_{i-1}) P(x_i | y_i) P(x_i | y_{i-1}) \right]$$

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

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

$P(Y=y)$		$P(Y Y_{prev})$	DET	PRP	N	PREP	V
DET	0.3	DET	0.01	0.01	0.96	0.01	0.01
PRP	0.3	PRP	0.01	0.01	0.01	0.2	0.77
N	0.1	N	0.01	0.2	0.3	0.3	0.19
PREP	0.1	PREP	0.3	0.2	0.3	0.19	0.01
V	0.2	V	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 $P(S_i | S_{i-1})$**

$$P(y_a | y_b) = \frac{\#ofTimesStateAFollowsStateB}{\#ofTimesStateBOccurs}$$

- **Estimating emission probabilities $P(W_i | S_i)$**

$$P(x_a | y_b) = \frac{\#ofTimesOutputAIsObservedInStateB}{\#ofTimesStateBOccurs}$$

- **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

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

$$h_{\text{bayes}}(\mathbf{x}) = \underset{y \in Y}{\text{argmax}} [P(Y = y | X = \mathbf{x})]$$

$$= \underset{y \in Y}{\text{argmax}} [P(X = \mathbf{x} | Y = y)P(Y = y)]$$

Generative:

- Make assumptions about $P(X = \mathbf{x} | Y = y), P(Y = y)$
- Estimate parameters of the two distributions

Discriminative:

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

$$h_{\text{bayes}}(\mathbf{x}) = \underset{y \in Y}{\text{argmax}} [P(Y = y | X = \mathbf{x})]$$

Question: Can we train HMM's discriminately?

Idea for Discriminative Training of HMM

Bayes Rule

$$h_{\text{bayes}}(\mathbf{x}) = \underset{y \in Y}{\text{argmax}} [P(Y = y | X = \mathbf{x})]$$

$$= \underset{y \in Y}{\text{argmax}} [P(X = \mathbf{x} | Y = y)P(Y = y)]$$

Model $P(Y = y | X = \mathbf{x})$ with $\vec{w} \cdot \Phi(\mathbf{x}, y)$ so that

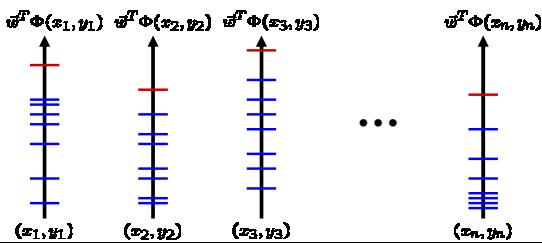
$$\left(\underset{y \in Y}{\text{argmax}} [P(Y = y | X = \mathbf{x})] \right) = \left(\underset{y \in Y}{\text{argmax}} [\vec{w} \cdot \Phi(\mathbf{x}, y)] \right)$$

Intuition:

- Tune \vec{w} so that correct y has the highest value of $\vec{w} \cdot \Phi(\mathbf{x}, y)$
- $\Phi(\mathbf{x}, y)$ is a feature vector that describes the match between x and y

Structural Support Vector Machine

- Joint features $\Phi(\mathbf{x}, \mathbf{y})$ describe match between x and y
- Learn weights \vec{w} so that $\vec{w}^T \Phi(\mathbf{x}, \mathbf{y})$ is max for correct y



Structural SVM Training Problem

Hard-margin optimization problem:

$$\min_{\vec{w}} \frac{1}{2} \vec{w}^T \vec{w}$$

$$s.t. \quad \forall y \in Y \setminus y_1 : \vec{w}^T \Phi(x_1, y_1) \geq \vec{w}^T \Phi(x_1, y) + 1$$

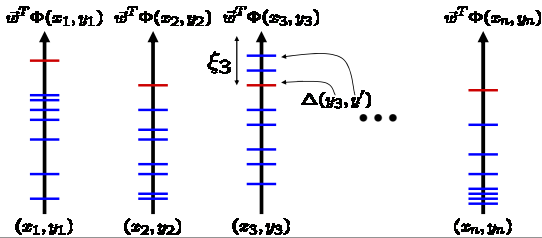
$$\dots$$

$$\forall y \in Y \setminus y_n : \vec{w}^T \Phi(x_n, y_n) \geq \vec{w}^T \Phi(x_n, y) + 1$$

- Training Set: $(x_1, y_1), \dots, (x_n, y_n) \sim P(X, Y)$
- Prediction Rule: $h_{\text{svm}}(\mathbf{x}) = \underset{y \in Y}{\text{argmax}} [\vec{w} \cdot \Phi(\mathbf{x}, y)]$
- Optimization:
 - Correct label y, must have higher value of $\vec{w} \cdot \Phi(x_i, y)$ than any incorrect label y
 - Find weight vector with smallest norm
 - Polynomial time algorithm (e.g. SVM-struct)

Loss Functions: Soft-Margin Struct SVM

- Loss function $\Delta(y_i, \hat{y}_i)$ measures match between target and prediction.



Soft-Margin Structural SVM

Soft-margin optimization problem:

$$\min_{\tilde{w}, \xi} \frac{1}{2} \tilde{w}^T \tilde{w} + C \sum_{i=1}^n \xi_i$$

s.t. $\forall y \in Y \setminus y_1 : \tilde{w}^T \Phi(x_1, y_1) \geq \tilde{w}^T \Phi(x_1, y) + \Delta(y_1, y) - \xi_1$
 \dots
 $\forall y \in Y \setminus y_n : \tilde{w}^T \Phi(x_n, y_n) \geq \tilde{w}^T \Phi(x_n, y) + \Delta(y_n, y) - \xi_n$

Lemma: The training loss is upper bounded by

$$Err_{TS}(h) = \frac{1}{n} \sum_{i=1}^n \Delta(y_i, h(\tilde{x}_i)) \leq \frac{1}{n} \sum_{i=1}^n \xi_i$$

Sparse Approximation Algorithm for Structural SVM

- Input: $(x_1, y_1), \dots, (x_n, y_n), C, \epsilon$

- $S \leftarrow \emptyset, \tilde{w} \leftarrow 0, \xi^* \leftarrow 0$

REPEAT

- FOR $i = 1, \dots, n$

- compute $\tilde{y} = \underset{y \in Y}{\operatorname{argmax}} \{ \Delta(y, \tilde{w}) + \tilde{w}^T \Phi(x_i, y) \}$

- IF $(\Delta(y_i, \tilde{y}) - \tilde{w}^T [\Phi(x_i, y_i) - \Phi(x_i, \tilde{y})]) > \xi_i + \epsilon$

- $S \leftarrow S \cup \{ \tilde{w}^T [\Phi(x_i, y_i) - \Phi(x_i, \tilde{y})] \geq \Delta(y_i, \tilde{y}) - \xi_i \}$

- $[\tilde{w}, \xi] \leftarrow$ optimize StructSVM over S

- ENDIF

- ENDFOR

- UNTIL S has not changed during iteration

Find most violated constraint

Violated by more than ϵ ?

Add constraint to working set

Training HMMs with Structural SVM

- Define $\Phi(x, y)$ so that model is isomorphic to HMM

- One feature for each possible start state
- One feature for each possible transition
- One feature for each possible output in each possible state
- Feature values are counts

Experiment: Part-of-Speech Tagging

Task

- Given a sequence of words x , predict sequence of tags y .

x The dog chased the cat $\rightarrow y$ Det \rightarrow N \rightarrow V \rightarrow Det \rightarrow N

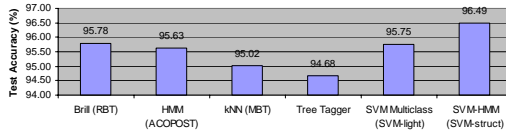
- Dependencies from tag-tag transitions in Markov model.

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



NE Identification

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

The delegation, which included the commander of the **U.N.** troops in **Bosnia**, Lt. Gen. Sir **Michael Rose**, went to the Serb stronghold of **Falč**, near **Sarajevo**, for talks with Bosnian Serb leader **Radovan Karadzic**.

Este ha sido el primer comentario publico del presidente **Clinton** respecto a la crisis de **Oriente Medio** desde que el secretario de Estado, **Warren Christopher**, decidiera regresar precipitadamente a **Washington** para impedir la ruptura del proceso de paz tras la violencia desatada en el sur de **Libano**.

- Locations
- Persons
- Organizations

Figure 1.1 Examples. Examples of correct labels for English text and for Spanish text.

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

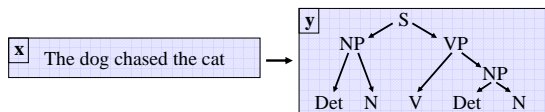
$$h : X \longrightarrow Y$$

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

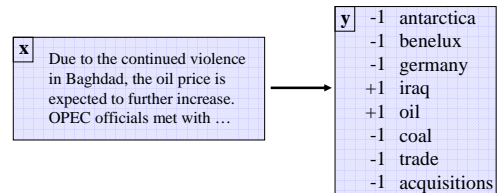
Examples of Complex Output Spaces

- **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.

