

# CSE 842 Natural Language Processing

## Lecture 5: Part-of-Speech Tagging

1/28/2009

CSE842, Spring 2009, MSU

1

## Part of Speech Tagging

- What is POS tagging?
- Associate with each word a lexical tag
  - 45 classes from Penn Treebank
  - 87 classes from Brown Corpus
  - 146 classes from C7 tagset (CLAWS system)

1/28/2009

CSE842, Spring 2009, MSU

2

## Why is POS Tagging Useful?

- First step of a vast number of practical tasks
- Speech synthesis
  - How to pronounce "lead"?
  - INsult                    inSULT
  - OBJect                    obJECT
  - OVERflow                overFLOW
  - DIScount                disCOUNT
  - CONtent                 conTENT
- Parsing
  - Need to know if a word is an N or V before you can parse
- Information extraction
  - Finding names, relations, etc.

1/28/2009

CSE842, Spring 2009, MSU

3

## POS Tagging

The/DT grand/JJ jury/NN commented/VBD on/IN a/DT number/NN  
of/IN other/JJ topics/NNS ./.

We have:

- A set of tags (tagset)
- A dictionary with all possible tags for each word
- Input text

1/28/2009

CSE842, Spring 2009, MSU

4

## Tagging Method

- Rule-based tagging
- Statistical tagging
- Transformation-based tagging

1/28/2009

CSE842, Spring 2009, MSU

5

## Statistical Tagging

- Using an HMM to do POS tagging is a special case of *Bayesian inference*
- It is also related to the "noisy channel" model that's the basis for ASR, OCR and MT

1/28/2009

CSE842, Spring 2009, MSU

6

## HMM Tagger

- HMM Tagger: Choose the most probable sequence of tags for each sentence
  - Given the sentence  $W = w_1, w_2, \dots, w_n$
  - Compute the most probable sequence of tags  $T = t_1, t_2, \dots, t_n$ , so that it maximizes:
    - $\text{argmax } P(t_1, t_2, \dots, t_n | w_1, w_2, \dots, w_n)$
- Apply Bayes law to rewrite it to:
 
$$P(t_1, \dots, t_n | w_1, \dots, w_n) = \frac{P(w_1, \dots, w_n | t_1, \dots, t_n) P(t_1, \dots, t_n)}{P(w_1, \dots, w_n)}$$

1/28/2009

CSE842, Spring 2009, MSU

7

## HMM Tagger (Cont.)

- It becomes finding  $t_1, \dots, t_n$  that maximizes:
 
$$P(w_1, \dots, w_n | t_1, \dots, t_n) P(t_1, \dots, t_n)$$
- Approximation 1: Independence assumption
 
$$P(w_1, \dots, w_n | t_1, \dots, t_n) \cong \prod_{i=1}^n P(w_i | t_i)$$
- Approximation 2: Bigram
 
$$P(t_1, \dots, t_n) \cong \prod_{i=1}^n P(t_i | t_{i-1})$$
- Now it becomes finding  $t_1, \dots, t_n$  that maximizes:
 
$$\prod_{i=1}^n P(t_i | t_{i-1}) * P(w_i | t_i)$$

1/28/2009

CSE842, Spring 2009, MSU

8

## An Example

Example: suppose  $w_i = \text{race}$ , a verb (VB) or a noun (NN)? Assume that other mechanism has already done the best tagging to the preceding words.

- 1) Secretariat/NNP is/VBZ expected/VBN to/TO **race/?** tomorrow
- 2) People/NNS continue/VBP to/TO inquire/VB the/DT reason/NN for/IN the/DT **race/?** for outer space

Bigram  $t_i = \text{argmax}_j P(t_j | t_{i-1}) P(w_i | t_j)$

Simplify the problem:  
to/To race/???  
the/DT race/???

P(VB|TO) P(race | VB)  
P(NN|TO) P(race | NN)

1/28/2009

CSE842, Spring 2009, MSU

9

## Where is the data?

Look at the Brown and Switchboard corpora

$$P(\text{NN} | \text{TO}) = 0.021$$

$$P(\text{VB} | \text{TO}) = 0.34$$

If we are expecting a verb, how likely it would be “race”

$$P(\text{race} | \text{NN}) = 0.00041$$

$$P(\text{race} | \text{VB}) = 0.00003$$

Finally:

$$P(\text{NN} | \text{TO}) P(\text{race} | \text{NN}) = 0.000007$$

$$P(\text{VB} | \text{TO}) P(\text{race} | \text{VB}) = 0.00001$$

1/28/2009

CSE842, Spring 2009, MSU

10

## POS Tagging as Sequence Classification

- We are given a sentence (an “observation” or “sequence of observations”)
  - *Secretariat is expected to race tomorrow*
- What is the best sequence of tags that corresponds to this sequence of observations?
- Probabilistic view:
  - Consider all possible sequences of tags
  - Out of this universe of sequences, choose the tag sequence which is most probable given the observation sequence of  $n$  words  $w_1 \dots w_n$ .

1/28/2009

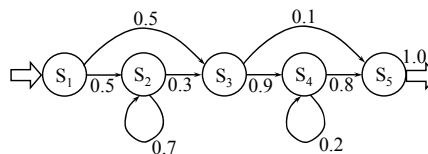
CSE842, Spring 2009, MSU

11

## Markov Model

A Markov Model (Markov Chain) is:

- similar to a finite-state automata, with *probabilities* of transitioning from one state to another:



- transition from state to state at discrete time intervals
- can only be in 1 state at any given time

1/28/2009

CSE842, Spring 2009, MSU

12

## Elements of a Markov Model

- Time  
 $t = \{1, 2, 3, \dots, T\}$
- $N$  states  
 $Q = \{1, 2, 3, \dots, N\}$
- $N$  events (corresponding to observations)  
 $E = \{e_1, e_2, e_3, \dots, e_N\}$
- initial probabilities  
 $\pi_j = P[q_1 = j] \quad 1 \leq j \leq N$
- transition probabilities (first-order)  
 $a_{ij} = P[q_t = j | q_{t-1} = i] \quad 1 \leq i, j \leq N$

1/28/2009

CSE842, Spring 2009, MSU

13

## Elements of a Markov Model

- the (potentially) occupied state at time  $t$  is called  $q_t$
- the occupied state referred to by its index:  $q_t = j$
- 1 event corresponds to 1 state:  
At each time  $t$ , the occupied state outputs (“emits”) its corresponding event which can be observed. (i.e., the observation is the same as the state).
- Markov model is *generator* of events.

1/28/2009

CSE842, Spring 2009, MSU

14

## First-order Observable Markov Model

- a set of states  $Q = 1, 2, \dots, N$ ; the state at time  $t$  is  $q_t$
- Current state only depends on previous state  
 $P(q_t | q_1 \dots q_{t-1}) = P(q_t | q_{t-1})$
- Transition probability matrix  $A$   
 $a_{ij} = P(q_t = j | q_{t-1} = i) \quad 1 \leq i, j \leq N$
- Special initial probability vector  $\pi$   
 $\pi_i = P(q_1 = i) \quad 1 \leq i \leq N$
- Constraints:

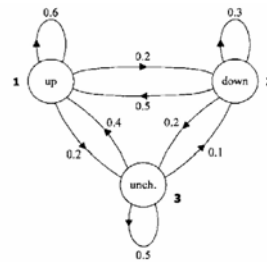
$$\sum_{j=1}^N a_{ij} = 1; \quad 1 \leq i \leq N \quad \sum_{j=1}^N \pi_j = 1$$

1/28/2009

CSE842, Spring 2009, MSU

15

## Markov Model for Dow Jones



Initial state probability matrix

$$\pi = (\pi_i) = \begin{pmatrix} 0.5 \\ 0.2 \\ 0.3 \end{pmatrix}$$

State-transition probability matrix

$$A = \{a_{ij}\} = \begin{bmatrix} 0.6 & 0.2 & 0.2 \\ 0.5 & 0.3 & 0.2 \\ 0.4 & 0.1 & 0.5 \end{bmatrix}$$

1/28/2009

CSE842, Spring 2009, MSU

16

## Markov Model for Dow Jones

- What is the probability of 5 consecutive up days?

1/28/2009

CSE842, Spring 2009, MSU

17

## Markov Model for Dow Jones

- What is the probability of 5 consecutive up days?
- Sequence is up-up-up-up-up  
– i.e., state sequence is 1 1 1 1 1  
–  $P(1,1,1,1,1) =$   
–  $\pi_1 a_{11} a_{11} a_{11} a_{11} = 0.5 \times (0.6)^4 = 0.0648$

1/28/2009

CSE842, Spring 2009, MSU

18

## Hidden Markov Model

- For Markov models, the output symbols are the same as the states.
  - See the market is **up**: we're in state **up**
- But in part of speech tagging (and other things)
  - The output symbols are **words**
  - But the hidden states are **part-of-speech tags**
- So we need an extension!
- A **Hidden Markov Model** is an extension of a Markov model in which the output symbols are not the same as the states.
- This means **we don't know which state we are in.**

1/28/2009

CSE842, Spring 2009, MSU

19

## Hidden Markov Model

- more than 1* event are associated with each state.
- all events have some *probability* of emitting at each state.
- given a sequence of observations, we can't determine exactly the state sequence.
- We *can* compute the *probabilities* of different state sequences given an event/observation sequence.
- Doubly stochastic (probabilities of both emitting events and transitioning between states); exact state sequence is "hidden."

1/28/2009

CSE842, Spring 2009, MSU

20

## Elements of Hidden Markov Model

- Time  $t = \{1, 2, 3, \dots, T\}$
- $N$  states  $Q = \{1, 2, 3, \dots, N\}$
- $M$  events/observations  $E = \{e_1, e_2, e_3, \dots, e_M\}$
- initial probabilities  $\pi_j = P[q_1 = j] \quad 1 \leq j \leq N$
- transition probabilities  $a_{ij} = P[q_t = j | q_{t-1} = i] \quad 1 \leq i, j \leq N$
- Observation probabilities (Emission probabilities)  $b_j(k) = P[o_t = e_k | q_t = j] \quad 1 \leq k \leq M$   
 $b_j(o) = P[o_t = e_k | q_t = j] \quad 1 \leq k \leq M$
- $A$  = matrix of  $a_{ij}$  values,  $B$  = set of observation probabilities,  $\pi$  = vector of  $\pi_j$  values.

Entire Model:  $\Phi = (A, B, \pi)$

1/28/2009

CSE842, Spring 2009, MSU

21

## Hidden Markov Models

- a set of states  $Q = 1, 2, \dots, N$ ; the state at time  $t$  is  $q_t$
- Transition probability matrix  $A = \{a_{ij}\}$ 

$$a_{ij} = P(q_t = j | q_{t-1} = i) \quad 1 \leq i, j \leq N$$
- Output probability matrix  $B = \{b_i(k)\}$ 

$$b_i(k) = P(X_t = o_k | q_t = i) \quad 1 \leq k \leq M$$
- Special initial probability vector  $\pi$ 

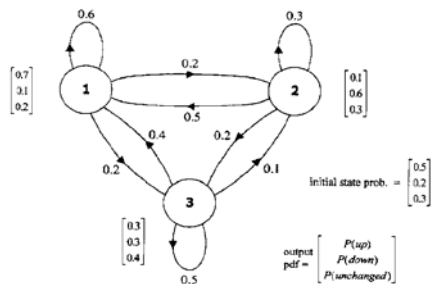
$$\pi_i = P(q_1 = i) \quad 1 \leq i \leq N$$
- Constraints:
 
$$\sum_{j=1}^N a_{ij} = 1; \quad 1 \leq i \leq N \quad \sum_{k=1}^M b_i(k) = 1 \quad \sum_{j=1}^N \pi_j = 1$$

1/28/2009

CSE842, Spring 2009, MSU

22

## HMM for Dow Jones



1/28/2009

CSE842, Spring 2009, MSU

23

## Back to POS Tagging

Suppose we have

- A set of tags (tagset)
- A training data set where each sentence is annotated with a sequence of correct tags

The/DT grand/JJ jury/NN commented/VBD on/IN a/DT number/NN of/IN other/JJ topics/NNS ./.

The goal: automatically recognize the underlying sequence of POS for any new unseen sentence.

1/28/2009

CSE842, Spring 2009, MSU

24

## An Example

Example:

*flies like a flower,*

Given

Word sequence  $w_1, \dots, w_N$

POS tags  $t_1, \dots, t_M, t_i \in [V, N, P, ART]$

Find:

Most likely sequence of POS tags  $T_1, \dots, T_N$   
for the word sequence that maximizes:

$P(w_1, \dots, w_N | t_1, \dots, t_M)$

1/28/2009

CSE842, Spring 2009, MSU

25

## Example: Bigram of Tags from a Corpus

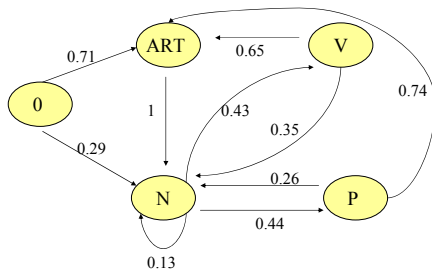
Cat	# at i	Pair	# at i, i+1	Bigram	Estimate
0	300	0, ART	213	Prob(ART 0)	0.71
0	300	0, N	87	Prob(N 0)	0.29
ART	558	ART, N	558	Prob(N   ART)	1
N	833	N, V	358	Prob(V   N)	0.43
N	833	N, N	108	Prob(N   N)	0.13
N	833	N, P	366	Prob(P   N)	0.44
V	300	V, N	75	Prob(N   V)	0.35
V	300	V, ART	194	Prob(ART   V)	0.65
P	307	P, ART	226	Prob(ART   P)	0.74
P	307	P, N	81	Prob(N   P)	0.26

1/28/2009

CSE842, Spring 2009, MSU

26

## A Markov Chain



1/28/2009

CSE842, Spring 2009, MSU

27

## Sparse Data

What about bigrams that are not seen from the data?

In practice, we should use smoothing

In this example, for illustration purpose, we use a simplification: any bigram data that is not listed in the table will be assumed to have a transitional probability of 0.0001.

1/28/2009

CSE842, Spring 2009, MSU

28

## Word Counts

	N	V	ART	P	Total
flies	21	23	0	0	44
fruit	49	5	1	0	55
like	10	30	0	21	61
a	1	0	201	0	202
the	1	0	300	2	303
flower	53	15	0	0	68
flowers	42	16	0	0	58
bird	64	1	0	0	65
others	592	210	56	284	1142
Total	833	300	558	307	1998

1/28/2009

CSE842, Spring 2009, MSU

$P(\text{the} | \text{ART})?$

29

## The Emission Probability

Some examples:

$$P(\text{the} | \text{ART}) = 300/558 = 0.54$$

$$P(\text{flies} | \text{N}) = 0.025$$

$$P(\text{flies} | \text{V}) = 0.076$$

$$P(\text{like} | \text{V}) = 0.1$$

$$P(\text{like} | \text{P}) = 0.068$$

$$P(\text{like} | \text{N}) = 0.012$$

$$P(a | \text{ART}) = 0.360$$

$$P(a | \text{N}) = 0.001$$

$$P(\text{flower} | \text{N}) = 0.063$$

$$P(\text{flower} | \text{V}) = 0.05$$

$$P(\text{bird} | \text{N}) = 0.076$$

1/28/2009

CSE842, Spring 2009, MSU

30

## Viterbi Algorithm

Given

Word sequence  $w_1, \dots, w_N$

Set of POS tags  $t_1, \dots, t_M$

Emission Probability:  $P(w_i | t_j)$

Transition probability:  $P(t_i | t_j)$

Find:

Most likely sequence of POS tags  $T_1, \dots, T_N$   
for the word sequence

1/28/2009

CSE842, Spring 2009, MSU

31

## Viterbi Algorithm

### Initialization Step

For  $i = 1$  to  $M$  do

VSCORE( $i, 1$ ) =  $P(w_1 | t_i) * P(t_i | \emptyset)$ ;

BACKPTR( $i, 1$ ) = 0

### Iteration Step

For  $j = 2$  to  $N$

for  $i = 1$  to  $M$

VSCORE( $i, j$ ) =  $\text{Max}_{k=1}^M (\text{VSCORE}(k, j-1) * P(t_i | t_k)) P(w_j | t_i)$

BACKPTR( $i, j$ ) = index of  $k$  that gave the max above

### Sequence Identification Step

$C(N)$  =  $i$  that maximizes VSCORE( $i, N$ )

For  $j = N - 1$  to 1 do

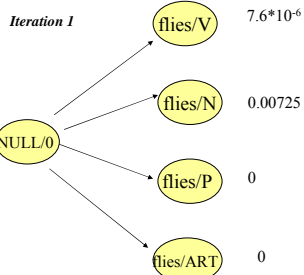
$C(j)$  = BACKPTR( $C(j+1), j+1$ )

1/28/2009

CSE842, Spring 2009, MSU

32

## Viterbi Algorithm - Example

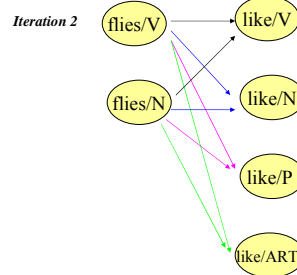


1/28/2009

CSE842, Spring 2009, MSU

33

## Viterbi Algorithm - Example

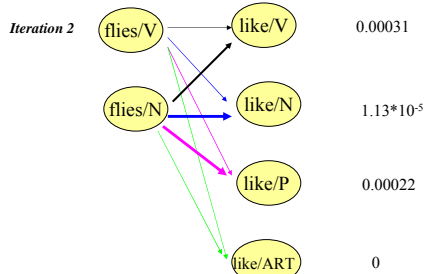


1/28/2009

CSE842, Spring 2009, MSU

34

## Viterbi Algorithm - Example

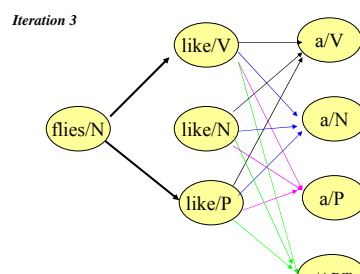


1/28/2009

CSE842, Spring 2009, MSU

35

## Viterbi Algorithm - Example



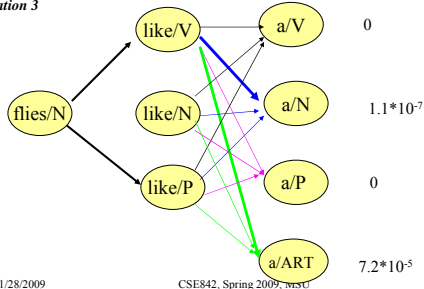
1/28/2009

CSE842, Spring 2009, MSU

36

## Viterbi Algorithm - Example

Iteration 3



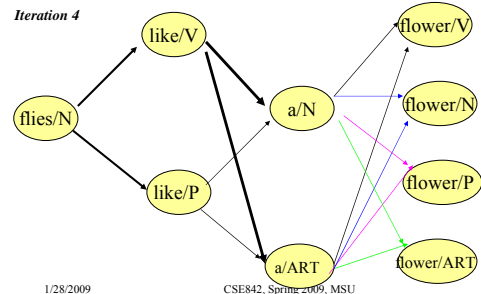
1/28/2009

CSE842, Spring 2009, MSU

37

## Viterbi Algorithm - Example

Iteration 4



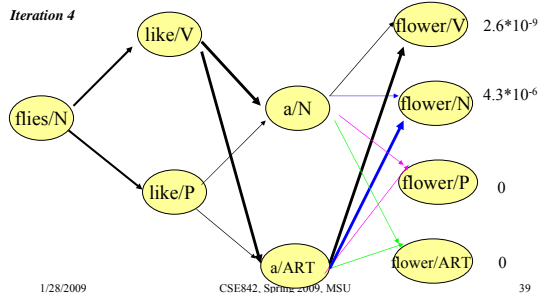
1/28/2009

CSE842, Spring 2009, MSU

38

## Viterbi Algorithm - Example

Iteration 4



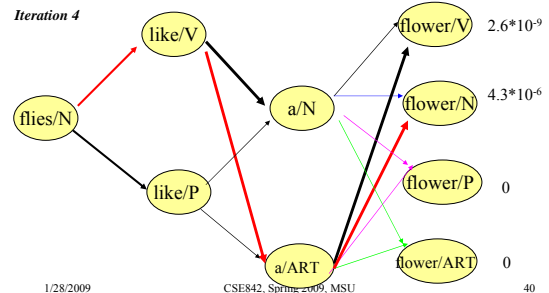
1/28/2009

CSE842, Spring 2009, MSU

39

## Viterbi Algorithm - Example

Iteration 4



1/28/2009

CSE842, Spring 2009, MSU

40

## Viterbi Algorithm for POS

- The basic operation is to sweep methodically through a two-dimensional array filling in the columns one at a time
- For each entry, the highest probability path through the array to that entry is computed and stored. (a pointer to remember where it came from is also stored)

1/28/2009

CSE842, Spring 2009, MSU

41

## Dynamic Programming

- Dynamic programming approaches operate by solving small problems once and remember the answers in a table so that they can be used to solve the overall problem.
  - Need best solutions to sub-problems
  - Need optimization criteria to indicate a given solution is the best solution to a sub-problem.
  - Need to only remember the best solution to that problem, not all the solutions.

1/28/2009

CSE842, Spring 2009, MSU

42

## Dynamic Programming

- Dynamic programming approaches operate by solving small problems once and remember the answers in a table so that they can be used to solve the overall problem.
  - Need best solutions to sub-problems
  - Need optimization criteria to indicate a given solution is the best solution to a sub-problem.
  - Need to only remember the best solution to that problem, not all the solutions.

1/28/2009

CSE842, Spring 2009, MSU

43

## Dynamic Programming

- Minimum Editing distance
  - Viterbi
- All are bottom up table filters. Each element of the table is computed from previously entered elements. Each entered element represents the best solution to the sub problem represented by that table element

1/28/2009

CSE842, Spring 2009, MSU

44