CSCI 5832
Natural Language Processing

Lecture 10

Jim Martin

Today: 2/20

- **Review**
  - **POS Tagging**
  - **HMMs and Viterbi**
- **Break**
- **Syntax and Context-free grammars**

Review

- **Parts of Speech**
  - **Basic syntactic/morphological categories that words belong to**
- **Part of Speech tagging**
  - **Assigning parts of speech to all the words in a sentence**

Probabilities

- **We want the best set of tags for a sequence of words (a sentence)**
- **W is a sequence of words**
- **T is a sequence of tags**

$$\arg\max P(T \mid W) = P(W \mid T)P(T)$$

So…

- **We start with**

$$\arg\max P(T \mid W) = P(W \mid T)P(T)$$

- **And get**

$$\arg\max \prod_{i=2}^{n} P(w_i \mid t_i) * P(t1) * \prod_{i=2}^{n} P(t_i \mid t_{i-1})$$

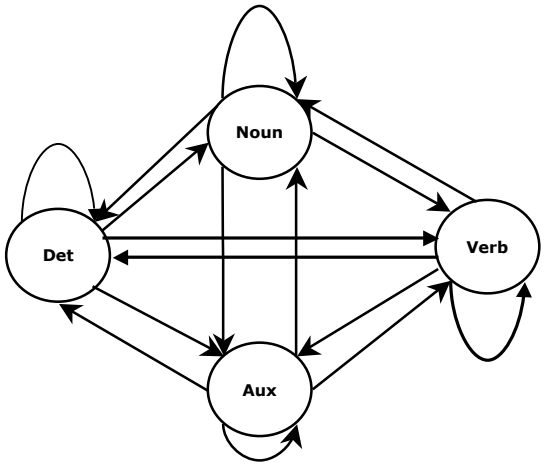
HMMs

- **This is an HMM**

$$\arg\max \prod_{i=2}^{n} P(w_i \mid t_i) * P(t_1) * \prod_{i=2}^{n} P(t_i \mid t_{i-1})$$

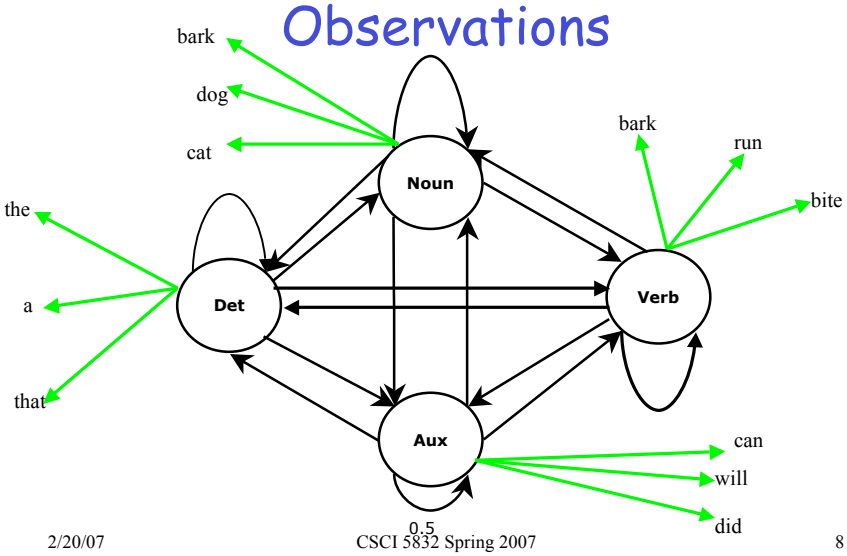
- **The states in the model are the tags, and the observations are the words.**
  - **The state to state transitions are driven by the bigram statistics**
  - **The observed words are based solely on the state that you're in**

State Transitions



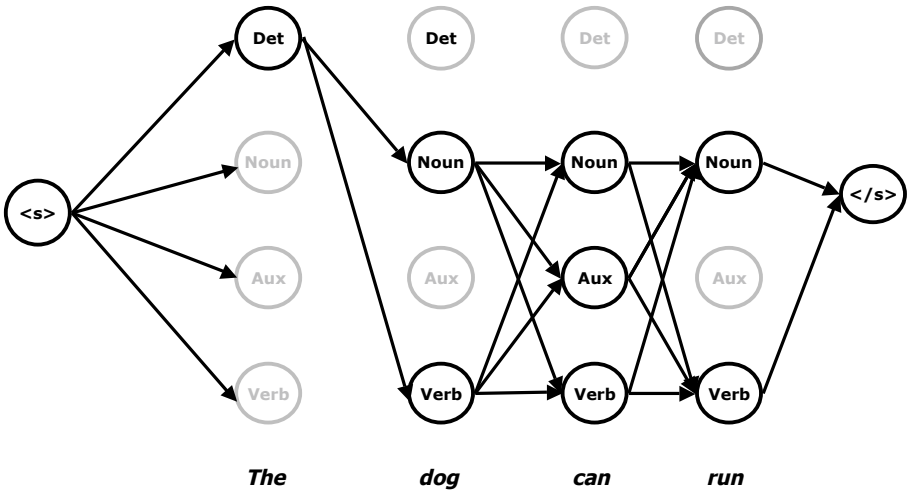
0.5

State Transitions and Observations

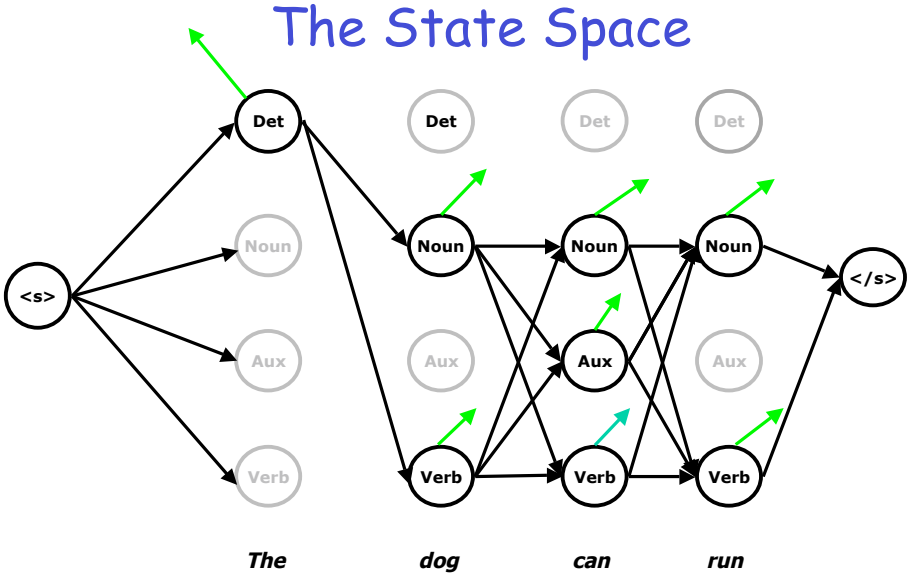


0.5

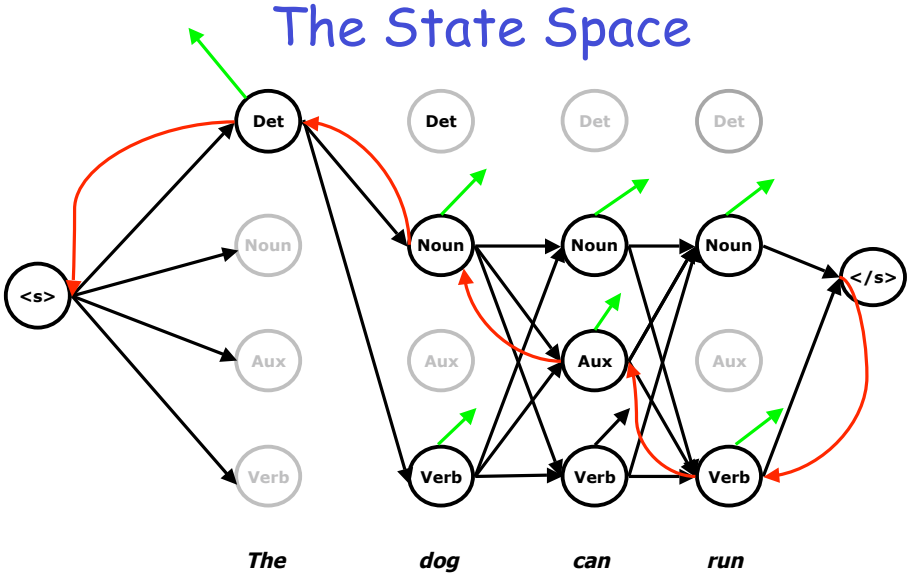
The State Space



The State Space



The State Space



Viterbi

- Efficiently return the most likely path
- Sweep through the columns multiplying the probabilities of one row, times the transition probabilities to the next row, times the appropriate observation probabilities
- And store the MAX

Viterbi

function VITERBI(*observations* of len T, *state-graph*) **returns** *best-path*

num-states ← NUM-OF-STATES(*state-graph*)
Create a path probability matrix *viterbi[num-states+2,T+2]*
viterbi[0,0] ← 1.0
for each time step t **from** 1 **to** T **do**
  for each state s **from** 1 **to** *num-states* **do**
    $viterbi[s,t] \leftarrow \max_{1 \le s' \le num\text{-}states} viterbi[s',t-1] * a_{s',s} * b_s(o_t)$
    $backpointer[s,t] \leftarrow \operatorname{argmax}_{1 \le s' \le num\text{-}states} viterbi[s',t-1] * a_{s',s}$
Backtrace from highest probability state in final column of *viterbi[]* and return path

Break

- **Changing the schedule a bit…**
  - **We're going to move on to Chapters 11 and 12 starting today.**
  - **We'll then go back to cover relevant aspects of Chapter 6.**
  - **Next quiz will cover 5, 6, 11, 12 and 13**

Talks

- CS Colloquium (Thursday 3:30)
  - Fernando Pereira -- Penn
    - Learning to Analyze Sequences
      - Basically Chapter 6 on steroids
- ICS Colloquium (Friday noon)
  - Christer Samuelsson
    - A Computational Linguist on Wall Street
      - How to use HMMs to do market prediction
        - » Using Chapter 6 to make gobs of money

Syntax

- By syntax (or grammar) I mean the kind of implicit knowledge of your native language that you had mastered by the time you were 2 or 3 years old without explicit instruction
- Not the kind of stuff you were later taught in school.

Syntax

- **Why should you care?**
  - **Grammar checkers**
  - **Question answering**
  - **Information extraction**
  - **Machine translation**

Search?

On Friday, PARC is announcing a deal that underscores that strategy. It is licensing a broad portfolio of patents and technology to a well-financed start-up with an ambitious and potentially lucrative goal: to build a search engine that could some day rival Google.The start-up, Powerset, is licensing PARC's natural language technology - the art of making computers understand and process languages like English... Powerset hopes the technology will be the basis of a new search engine that allows users to type queries in plain English, rather than using keywords.

Search

For a lot of things, keyword search works well, said Barney Pell, chief executive of Powerset. But I think we are going to look back in 10 years and say, remember when we used to search using keywords.

Search

In a November interview, Marissa Mayer, Google's vice president for search and user experience, said: "Natural language is really hard. I don't think it will happen in the next five years."

Search

"My general feeling about natural-language processing in search is that I'm a bit of a skeptic in the sense that even the best systems, and I include there the systems from PARC, make many mistakes," said Mr. Pereira of the University of Pennsylvania.

Context-Free Grammars

- **Capture constituency and ordering**
  - **Ordering is easy**
    What are the rules that govern the ordering of words and bigger units in the language
  - **What's constituency?**
    How words group into units and how the various kinds of units behave

CFG Examples

- S -> NP VP
- NP -> Det NOMINAL
- NOMINAL -> Noun
- VP -> Verb
- Det -> *a*
- Noun -> *flight*
- Verb -> *left*

CFGs

- S -> NP VP
  - This says that there are units called S, NP, and VP in this language
  - That an S consists of an NP followed immediately by a VP
  - Doesn't say that that's the only kind of S
  - Nor does it say that this is the only place that NPs and VPs occur

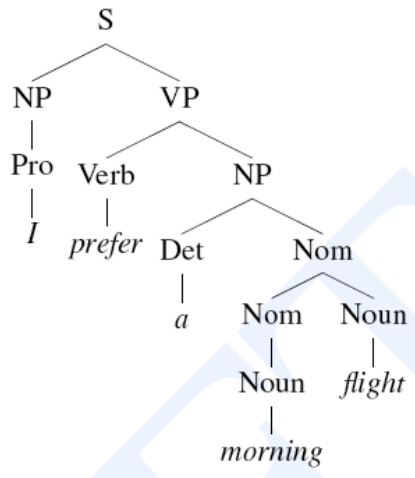
Generativity

- As with FSAs and FSTs you can view these rules as either analysis or synthesis machines
  - Generate strings in the language
  - Reject strings not in the language
  - Impose structures (trees) on strings in the language

Derivations

- A derivation is a sequence of rules applied to a string that accounts for that string
  - Covers all the elements in the string
  - Covers only the elements in the string

Derivations as Trees



Parsing

- Parsing is the process of taking a string and a grammar and returning a (many?) parse tree(s) for that string
- It is completely analogous to running a finite-state transducer with a tape
  - It's just more powerful
    - Remember this means that there are languages we can capture with CFGs that we can't capture with finite-state methods

Other Options

- Regular languages (expressions)
  - Too weak
- Context-sensitive or Turing equiv
  - Too powerful (maybe)

Context?

- The notion of context in CFGs has nothing to do with the ordinary meaning of the word context in language.
- All it really means is that the non-terminal on the left-hand side of a rule is out there all by itself (free of context)

  A -> B C

  Means that I can rewrite an A as a B followed by a C regardless of the context in which A is found

Key Constituents (English)

- Sentences
- Noun phrases
- Verb phrases
- Prepositional phrases

Sentence-Types

- Declaratives: A plane left
  S -> NP VP
- Imperatives: Leave!
  S -> VP
- Yes-No Questions: Did the plane leave?
  S -> Aux NP VP
- WH Questions: When did the plane leave?
  S -> WH Aux NP VP

Recursion

- We'll have to deal with rules such as the following where the non-terminal on the left also appears somewhere on the right (directly).

  Nominal -> Nominal PP [[flight] [to Boston]]

  VP -> VP PP [[departed Miami] [at noon]]

Recursion

- Of course, this is what makes syntax interesting

  flights from Denver

  Flights from Denver to Miami

  Flights from Denver to Miami in February

  Flights from Denver to Miami in February on a Friday

  Flights from Denver to Miami in February on a Friday under $300

  Flights from Denver to Miami in February on a Friday under $300 with lunch

Recursion

- Of course, this is what makes syntax interesting

  [[flights] [from Denver]]

  [[[Flights] [from Denver]] [to Miami]]

  [[[[Flights] [from Denver]] [to Miami]] [in February]]

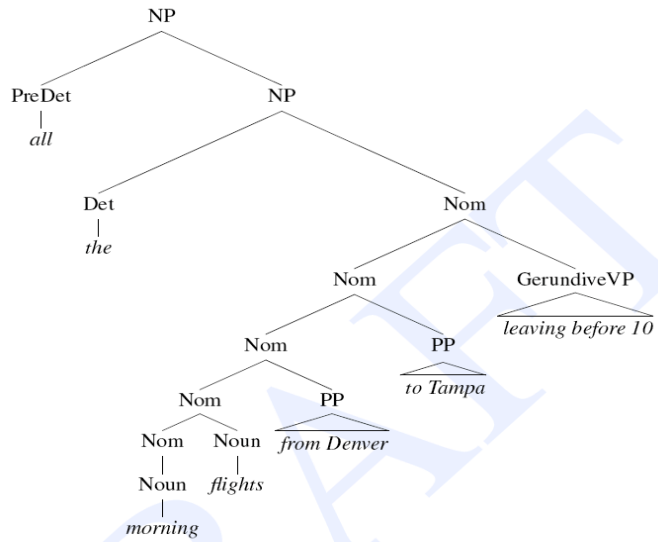
  [[[[[Flights] [from Denver]] [to Miami]] [in February]] [on a Friday]]

  Etc.

The Point

- If you have a rule like
  - VP -> V NP
  - It only cares that the thing after the verb is an NP. It doesn't have to know about the internal affairs of that NP

The Point



Conjunctive Constructions

- S -> S and S
  - John went to NY and Mary followed him
- NP -> NP and NP
- VP -> VP and VP
- …
- In fact the right rule for English is
  X -> X and X

Problems

- Agreement
- Subcategorization
- Movement (for want of a better term)

Agreement

- This dog
- Those dogs
- *This dogs
- *Those dog

- This dog eats
- Those dogs eat
- *This dog eat
- *Those dogs eats

Subcategorization

- Sneeze: John sneezed
- Find: Please find [a flight to NY]$_{NP}$
- Give: Give [me]$_{NP}$[a cheaper fare]$_{NP}$
- Help: Can you help [me]$_{NP}$[with a flight]$_{PP}$
- Prefer: I prefer [to leave earlier]$_{TO\text{-}VP}$
- Told: I was told [United has a flight]$_{S}$
- …

Subcategorization

- *John sneezed the book
- *I prefer United has a flight
- *Give with a flight

- Subcat expresses the constraints that a predicate (verb for now) places on the number and syntactic types of arguments it wants to take (occur with).

So?

- So the various rules for VPs overgenerate.
  - They permit the presence of strings containing verbs and arguments that don't go together
  - For example
  - VP -> V NP therefore

    Sneezed the book is a VP since "sneeze" is a verb and "the book" is a valid NP

Next Time

- We're now covering Chapters 11 and 12.
- Next time…
  - Project discussion. Come prepared to talk about project ideas. We'll break into preliminary groups on Thursday.