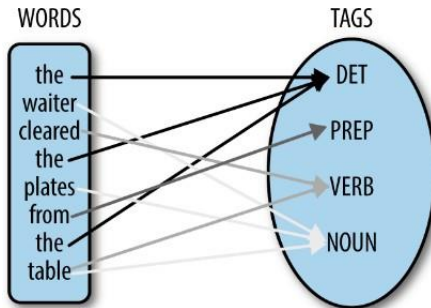


Part-of-speech (POS) tagging

Part-of-speech (POS) tagging

Task

The process of marking up a word in a text (corpus) as corresponding to a particular part of speech, based on both its *definition* and its *context*.



Parts of Speech: How many?

During our school days: **Nine** parts of speech in English

- Noun
- Verb
- Article
- Adjective
- Preposition
- Pronoun
- Adverb
- Conjunction
- Interjection.

PARTS OF SPEECH

Tom appeared on the sidewalk with a bucket of whitewash and a long-handled brush. He busily surveyed the fence and all gladness left him and a deep melancholy settled down upon his spirit. "Oh man! This will take all day," he said.

NOUNS identify PEOPLE- family, man, daughter PLACES- town, street, school THINGS- markers, bananas, hair IDEAS- thoughts, love, friendship	PRONOUNS used in place of a noun in a sentence I - you - he - she - they him - her - them - who that - those - this - these	ADJECTIVES describe nouns WHICH ONE- this, that WHAT KIND- happy, naughty HOW MANY- three, more, few
PREPOSITIONS precede a noun & refers to position ANY WAY A NOUN CAN BE: above - across - before - below beside - down - during - following inside - into - near - next to onto - over - towards - under	ARTICLES precede a noun and are either DEFINITE- the chair, the book, the apple or INDEFINITE- a chair, a book, an apple	CONJUNCTIONS are used to connect single words or groups of words in a sentence and, but, or, either neither, nor
VERBS what someone or something is doing, an action word run, jump, skip, swim, fly balance, move, eat, hug OR a state of being like am, was, is, are, were	ADVERBS add meaning to a verb, adjective or sentence HOW? happily, easily, quickly, sadly HOW OFTEN? always, often, never WHEN? after, before, today, soon WHERE? here, there, away, near	INTERJECTIONS Can be a single word or phrase which describes feeling or emotion. Look! Look out! Ouch! Oh! Help! Oh my goodness! Gosh! Oh dear! Ah! Aww!

Open vs. Closed class

Open class words (content words)

- nouns, verbs, adjectives, adverbs
- mostly content-bearing: they refer to objects, actions, and features in the world
- *open class*, since new words are added all the time

Closed class words

- pronouns, determiners, prepositions, connectives, ...
- there is a limited number of these
- *mostly functional*: to tie the concepts of a sentence together

POS tagging

Choosing a tagset

- The school time tagset is very coarse (only nine categories).
- There are clearly many more categories and sub-categories.
- Such as the plural, possessive, singular, case, and tense forms.

In real scenarios, *pick a standard tagset* which is more fine grained such as “**UPenn TreeBank tagset**” containing 48 tags.

- Paper: <https://dl.acm.org/citation.cfm?id=972470.972475>
- It is based on annotated Brown corpus (87 tags).

UPenn TreeBank POS tagset

48 tags = 36 POS + 12 other tags

The Penn Treebank POS tagset.

1. CC	Coordinating conjunction	25. TO	<i>to</i>
2. CD	Cardinal number	26. UH	Interjection
3. DT	Determiner	27. VB	Verb, base form
4. EX	Existential <i>there</i>	28. VBD	Verb, past tense
5. FW	Foreign word	29. VBG	Verb, gerund/present participle
6. IN	Preposition/subordinating conjunction	30. VBN	Verb, past participle
7. JJ	Adjective	31. VBP	Verb, non-3rd ps. sing. present
8. JJR	Adjective, comparative	32. VBZ	Verb, 3rd ps. sing. present
9. JJS	Adjective, superlative	33. WDT	<i>wh</i> -determiner
10. LS	List item marker	34. WP	<i>wh</i> -pronoun
11. MD	Modal	35. WP\$	Possessive <i>wh</i> -pronoun
12. NN	Noun, singular or mass	36. WRB	<i>wh</i> -adverb
13. NNS	Noun, plural	37. #	Pound sign
14. NNP	Proper noun, singular	38. \$	Dollar sign
15. NNPS	Proper noun, plural	39. .	Sentence-final punctuation
16. PDT	Predeterminer	40. ,	Comma
17. POS	Possessive ending	41. :	Colon, semi-colon
18. PRP	Personal pronoun	42. (Left bracket character
19. PP\$	Possessive pronoun	43.)	Right bracket character
20. RB	Adverb	44. "	Straight double quote
21. RBR	Adverb, comparative	45. '	Left open single quote
22. RBS	Adverb, superlative	46. "	Left open double quote
23. RP	Particle	47. '	Right close single quote
24. SYM	Symbol (mathematical or scientific)	48. "	Right close double quote

Using the UPenn tagset

Example Sentence

The grand jury commented on a number of other topics.

Using the UPenn tagset

Example Sentence

The grand jury commented on a number of other topics.

POS tagged sentence

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

Why is POS tagging hard?

Words often have more than one POS: *back*

- The back door:

Why is POS tagging hard?

Words often have more than one POS: *back*

- The back door: *back/JJ*
- On my back:

Why is POS tagging hard?

Words often have more than one POS: *back*

- The back door: *back/JJ*
- On my back: *back/NN*
- Win the voters back:

Why is POS tagging hard?

Words often have more than one POS: *back*

- The back door: *back/JJ*
- On my back: *back/NN*
- Win the voters back: *back/RB*
- Promised to back the bill:

Why is POS tagging hard?

Words often have more than one POS: *back*

- The back door: *back/JJ*
- On my back: *back/NN*
- Win the voters back: *back/RB*
- Promised to back the bill: *back/VB*

Why is POS tagging hard?

Words often have more than one POS: *back*

- The back door: *back/JJ*
- On my back: *back/NN*
- Win the voters back: *back/RB*
- Promised to back the bill: *back/VB*

POS tagging problem

To determine the POS tag for a particular instance of a word

Ambiguous word types in the Brown Corpus

Ambiguity in the Brown corpus

- 40% of word tokens are ambiguous
- 12% of word types are ambiguous
- Breakdown of ambiguous word types:

Unambiguous (1 tag)	35,340
Ambiguous (2–7 tags)	4,100
2 tags	3,760
3 tags	264
4 tags	61
5 tags	12
6 tags	2
7 tags	1 (“still”)

How bad is the ambiguity problem?

- One tag is usually more likely than the others.
In the Brown corpus, *race* is a noun 98% of the time, and a verb 2% of the time
- A tagger for English that simply chooses the most likely tag for each word can achieve good performance
- Any new approach should be compared against the unigram baseline (assigning each token to its most likely tag).
- Sometimes it can be difficult even for people to distinguish.

Relevant knowledge for POS tagging

Signals from the word itself

- Some words may only be nouns, e.g. *arrow*
- Some words are ambiguous, e.g. *like*, *flies*
- Probabilities may help, if one tag is more likely than another

Relevant knowledge for POS tagging

Signals from the word itself

- Some words may only be nouns, e.g. *arrow*
- Some words are ambiguous, e.g. *like*, *flies*
- Probabilities may help, if one tag is more likely than another

Signals from the local context

- Two determiners rarely follow each other.
- Two base form verbs rarely follow each other.
- Determiner is almost always followed by adjective or noun.

POS tagging: Two approaches

Rule-based Approach

- Assign each word in the input a list of potential POS tags
- Then winnow down this list to a single tag using hand-written rules

Rule-Based Part-of-Speech Tagging

- The rule-based approach uses handcrafted sets of rules to tag input sentence.
- There are two stages in rule-based taggers:
 - **First Stage:** Uses a dictionary to assign each word a list of potential parts-of-speech.
 - **Second Stage:** Uses a large list of handcrafted rules to window down this list to a single part-of-speech for each word.
- The ENGTWOL is a rule-based tagger
 - In the first stage, uses a two-level lexicon transducer
 - In the second stage, uses hand-crafted rules (about 1100 rules).
 - Rule-1: if (the previous tag is an article)
then eliminate all verb tags
 - Rule-2: if (the next tag is verb)
then eliminate all verb tags

Rule-Based Part-of-Speech Tagging: Example

- Example: He had a fly.
- The first stage:
 - he **he/pronoun**
 - had **have/verbpast** have/auxliarypast
 - a **a/article**
 - fly **fly/verb** fly/noun
- The second stage:
 - apply rule: if (the previous tag is an article)
 then eliminate all verb tags
 - he **he/pronoun**
 - had **have/verbpast** have/auxliarypast
 - a **a/article**
 - fly fly/verb **fly/noun**

Transformation-Based Tagging

- Transformation-based tagging is also known as **Brill Tagging**.
- **Brill Tagging uses transformation rules** and rules are learned from a tagged corpus.
- Then these learned rules are used in tagging.
- Before the rules are applied, the tagger labels every word with its most likely tag.
 - We get these most likely tags from a tagged corpus.

Transformation-Based Tagging: Example

- Example:
 - He is expected to race tomorrow
 - he/PRN is/VBZ expected/VBN to/TO race/NN tomorrow/NN
- After selecting most-likely tags, we apply transformation rules.
 - Change NN to VB when the previous tag is TO
 - This rule converts **race/NN** into **race/VB**
- This may not work for every case
 - According to race

POS tagging: Two approaches

Rule-based Approach

- Assign each word in the input a list of potential POS tags
- Then winnow down this list to a single tag using hand-written rules

Statistical tagging (supervised/unsupervised)

- Probabilistic: Find the most likely sequence of tags T for a sequence of words W

POS tagging: Two approaches

Rule-based Approach

- Assign each word in the input a list of potential POS tags
- Then winnow down this list to a single tag using hand-written rules

Statistical tagging (supervised/unsupervised)

- Probabilistic: Find the most likely sequence of tags T for a sequence of words W
- Get a training corpus of tagged text, give it to a machine learning algorithm and it will learn the tagging rules.

Probabilistic Tagging: Two different families of models

Problem at hand

We have some data $\{(d, c)\}$ of paired observations d and hidden classes c .

Different instances of d and c

- **Part-of-Speech Tagging:** words are observed and tags are hidden.
- **Text Classification:** sentences/documents are observed and the category is hidden.
Categories can be positive/negative for sentiments ..
sports/politics/business for documents ...

What gives rise to the two families?

Whether they generate the observed data from hidden stuff or the hidden structure given the data?

Generative vs. Conditional Models

Generative (Joint) Models

Generate the observed data from hidden stuff, i.e. put a probability over the observations given the class: $P(d, c)$ in terms of $P(d|c)$

e.g. Naïve Bayes' classifiers, Hidden Markov Models etc.

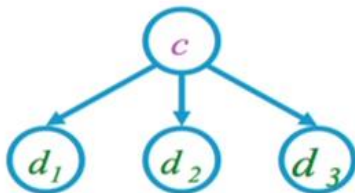
Discriminative (Conditional) Models

Take the data as given, and put a probability over hidden structure given the data: $P(c|d)$

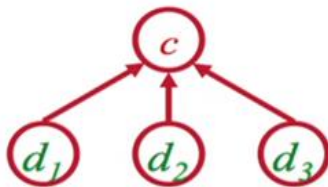
e.g. Logistic regression, maximum entropy models, conditional random fields

SVMs, perceptron, etc. are discriminative classifiers but not directly probabilistic

Generative vs. Discriminative Models



Naive Bayes



Logistic Regression

Joint vs. conditional likelihood

- A *joint* model gives probabilities $P(d, c)$ and tries to maximize this joint likelihood.
- A *conditional* model gives probabilities $P(c|d)$, taking the data as and modeling only the conditional probability of the class.

Where is **Parts of Speech** used?

- Grammatical Analysis of a text.
 - Spell-checkers

Probabilistic Tagging

- $W = w_1 \dots w_n$ - words in the corpus (observed)
- $T = t_1 \dots t_n$ - the corresponding tags (unknown)

Tagging: Probabilistic View

Find

$$\begin{aligned}\hat{T} &= \operatorname{argmax}_T P(T | W) \\ &= \operatorname{argmax}_T \frac{P(W|T)P(T)}{P(W)} \\ &= \operatorname{argmax}_T P(W|T)P(T) \\ &= \operatorname{argmax}_T \prod_i P(w_i | w \dots w_{i-1}, t_1 \dots t_i) P(t_i | t_1 \dots t_{i-1})\end{aligned}$$

Further simplifications

$$\hat{T} = \underset{i}{\operatorname{argmax}_T} P(w_i | w_1 \dots w_{i-1}, t_1 \dots t_i) P(t_i | t_1 \dots t_{i-1})$$

- The probability of a word appearing depends only on its own POS tag

Further simplifications

$$\hat{T} = \underset{i}{\operatorname{argmax}_T} P(w_i|w_1 \dots w_{i-1}, t_1 \dots t_i) P(t_i|t_1 \dots t_{i-1})$$

- The probability of a word appearing depends only on its own POS tag
 $P(w_i|w_1 \dots w_{i-1}, t_1 \dots t_i) \approx P(w_i|t_i)$
- Bigram assumption: the probability of a tag appearing depends only on the previous tag

Further simplifications

$$\hat{T} = \underset{i}{\operatorname{argmax}_T} P(w_i|w_1 \dots w_{i-1}, t_1 \dots t_i) P(t_i|t_1 \dots t_{i-1})$$

- The probability of a word appearing depends only on its own POS tag
 $P(w_i|w_1 \dots w_{i-1}, t_1 \dots t_i) \approx P(w_i|t_i)$
- Bigram assumption: the probability of a tag appearing depends only on the previous tag

$$P(t_i|t_1 \dots t_{i-1}) \approx P(t_i|t_{i-1})$$

Further simplifications

$$\hat{T} = \underset{i}{\operatorname{argmax}_T} P(w_i|w_1 \dots w_{i-1}, t_1 \dots t_i) P(t_i|t_1 \dots t_{i-1})$$

- The probability of a word appearing depends only on its own POS tag
 $P(w_i|w_1 \dots w_{i-1}, t_1 \dots t_i) \approx P(w_i|t_i)$
- Bigram assumption: the probability of a tag appearing depends only on the previous tag

$$P(t_i|t_1 \dots t_{i-1}) \approx P(t_i|t_{i-1})$$

- Using these simplifications:

$$\hat{T} = \underset{i}{\operatorname{argmax}_T} P(w_i|t_i) P(t_i|t_{i-1})$$

Computing the probability values

Tag Transition probabilities $p(t_i|t_{i-1})$

$$P(t_i|t_{i-1}) = \frac{C(t_{i-1}, t_i)}{C(t_{i-1})}$$

$$P(NN|DT) = \frac{C(DT, NN)}{C(DT)} = \frac{56,509}{116,454} = 0.49$$

Computing the probability values

Tag Transition probabilities $p(t_i|t_{i-1})$

$$P(t_i|t_{i-1}) = \frac{C(t_{i-1}, t_i)}{C(t_{i-1})}$$

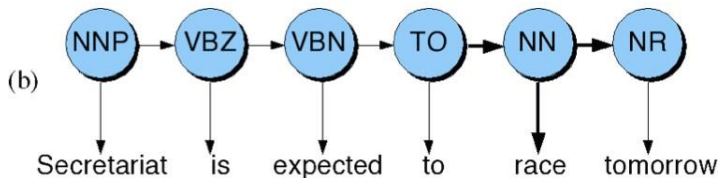
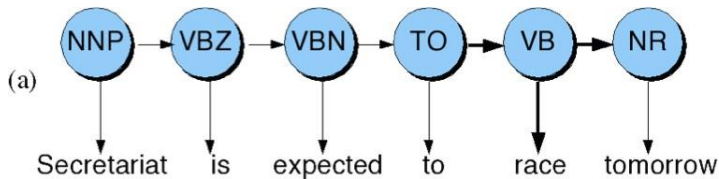
$$P(NN|DT) = \frac{C(DT, NN)}{C(DT)} = \frac{56,509}{116,454} = 0.49$$

Word Likelihood probabilities $p(w_i|t_i)$

$$P(w_i|t_i) = \frac{C(t_i, w_i)}{C(t_i)}$$

$$P(is|VBZ) = \frac{C(VBZ, is)}{C(VBZ)} = \frac{10,073}{21,627} = 0.47$$

What is this model?



Disambiguating “race”

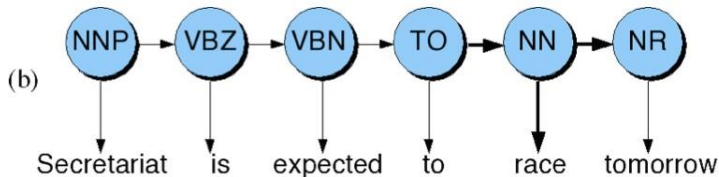
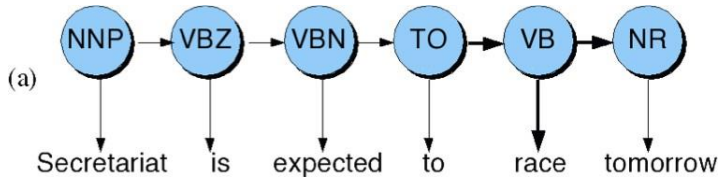
Difference in probability due to

- $P(VB|TO)$ vs. $P(NN|TO)$
- $P(race|VB)$ vs. $P(race|NN)$
- $P(NR|VB)$ vs. $P(NR|NN)$

After computing the probabilities

- $P(NN|TO)P(NR|NN)P(race|NN) = 0.0047 \times 0.0012 \times 0.00057 = 0.00000000032$
- $P(VB|TO)P(NR|VB)P(race|VB) = 0.83 \times 0.0027 \times 0.00012 = 0.000000027$

What is this model?



This is a Hidden Markov Model

Before describing HMM, let us quickly see Markov Chains, or observable Markov Model.

Hidden Markov Models

- Tag Transition probabilities $p(t_i|t_{i-1})$
- Word Likelihood probabilities (emissions) $p(w_i|t_i)$
- What we have described with these probabilities is a hidden markov model.
- Let us quickly introduce the Markov Chain, or observable Markov Model.

Markov Chain = First-order Markov Model

A chain of events, where current event depends only on the previous event.

Markov Chain = First-order Markov Model

A chain of events, where current event depends only on the previous event.

Weather Prediction Problem



- Three types of weather: *sunny*, *rainy*, *foggy*
- q_n : variable denoting the weather on the n^{th} day
- We want to find the following conditional probabilities:

$$P(q_n | q_{n-1}, q_{n-2}, \dots, q_1)$$

Markov Chain = First-order Markov Model

A chain of events, where current event depends only on the previous event.

Weather Prediction Problem



- Three types of weather: *sunny*, *rainy*, *foggy*
- q_n : variable denoting the weather on the n^{th} day
- We want to find the following conditional probabilities:





$$P(q_n | q_{n-1}, q_{n-2}, \dots, q_1)$$

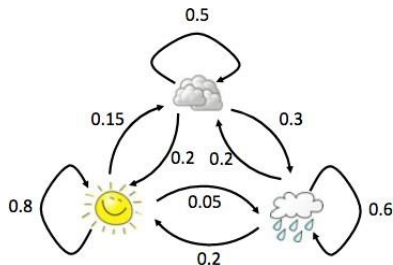
First-order Markov Assumption

$$P(q_n | q_{n-1}, q_{n-2}, \dots, q_1) = P(q_n | q_{n-1})$$

Markov Chain Transition Table

Table 1: Probabilities $p(q_{n+1}|q_n)$ of tomorrow's weather based on today's weather

	Tomorrow's weather		
Today's weather			
	0.8	0.05	0.15
	0.2	0.6	0.2
	0.2	0.3	0.5



Using Markov Chain

- Given that today the weather is sunny, what is the probability that tomorrow is sunny and day after is rainy?

Using Markov Chain

- Given that today the weather is sunny, what is the probability that tomorrow is sunny and day after is rainy?

$$P(q_2 = \textit{sunny}, q_3 = \textit{rainy} | q_1 = \textit{sunny})$$

Using Markov Chain

- Given that today the weather is sunny, what is the probability that tomorrow is sunny and day after is rainy?

$$P(q_2 = \textit{sunny}, q_3 = \textit{rainy} | q_1 = \textit{sunny})$$

$$= P(q_3 = \textit{rainy} | q_2 = \textit{sunny}, q_1 = \textit{sunny}) \times P(q_2 = \textit{sunny} | q_1 = \textit{sunny})$$

Using Markov Chain

- Given that today the weather is sunny, what is the probability that tomorrow is sunny and day after is rainy?

$$P(q_2 = \text{sunny}, q_3 = \text{rainy} | q_1 = \text{sunny})$$

$$= P(q_3 = \text{rainy} | q_2 = \text{sunny}, q_1 = \text{sunny}) \times P(q_2 = \text{sunny} | q_1 = \text{sunny})$$

$$= P(q_3 = \text{rainy} | q_2 = \text{sunny}) \times P(q_2 = \text{sunny} | q_1 = \text{sunny})$$

$$= 0.05 \times 0.8$$

$$= 0.04$$

Hidden Markov Model

- For Markov chains, the output symbols are the same as the states
'sunny' weather is both observable and state

Hidden Markov Model

- For Markov chains, the output symbols are the same as the states
'sunny' weather is both observable and state
- But in our formulation for POS tagging
Output symbols: Words
Hidden states: POS tags
- A Hidden Markov Model is an extension of a Markov chain in which the output symbols are not the same as the states
- We don't know which state we are in

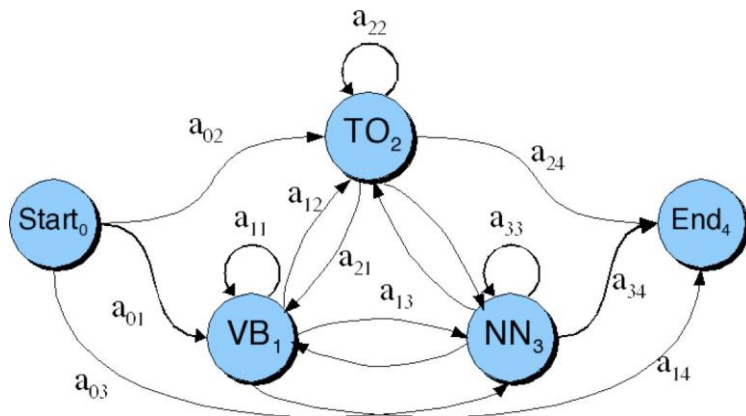
Hidden Markov Models (HMMs)

Elements of an HMM model

- A set of states (here: the tags)
- An output alphabet (here: words)
- Initial state (here: beginning of sentence)
- State transition probabilities (here: $p(t_n|t_{n-1})$)
- Symbol emission probabilities (here: $p(w_i|t_i)$)

Graphical Representation

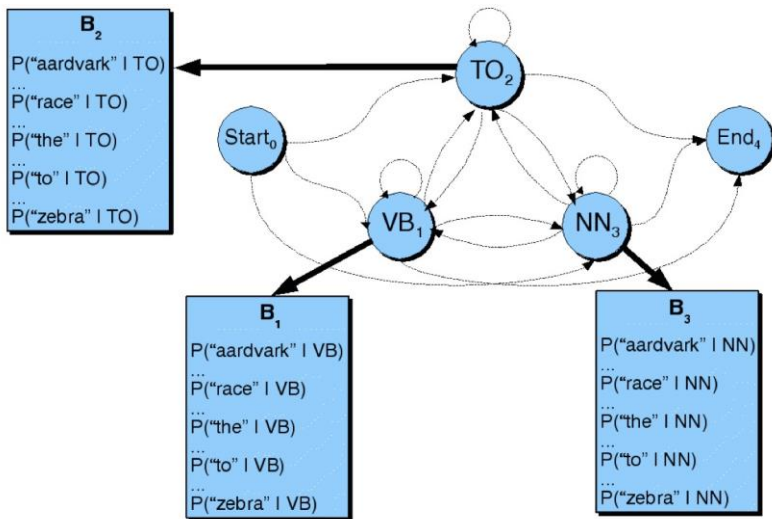
When tagging a sentence, we are walking through the state graph:



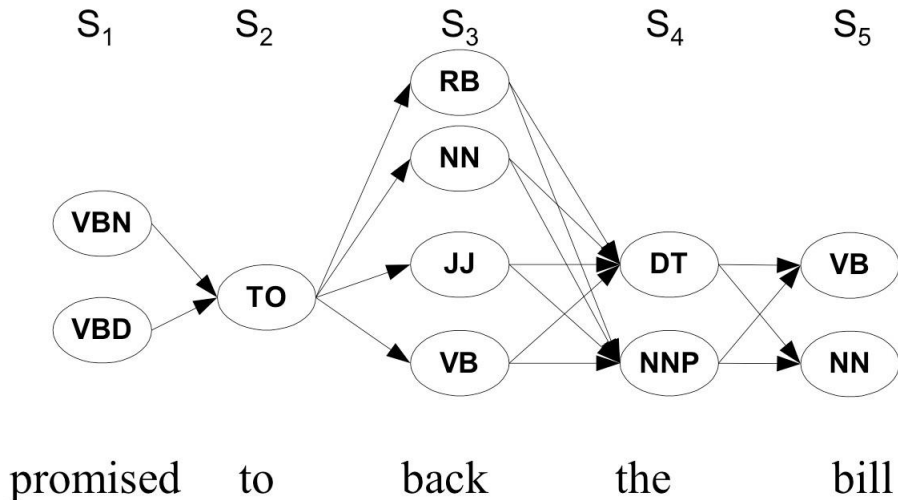
Edges are labeled with the state transition probabilities: $p(t_n|t_{n-1})$

Graphical Representation

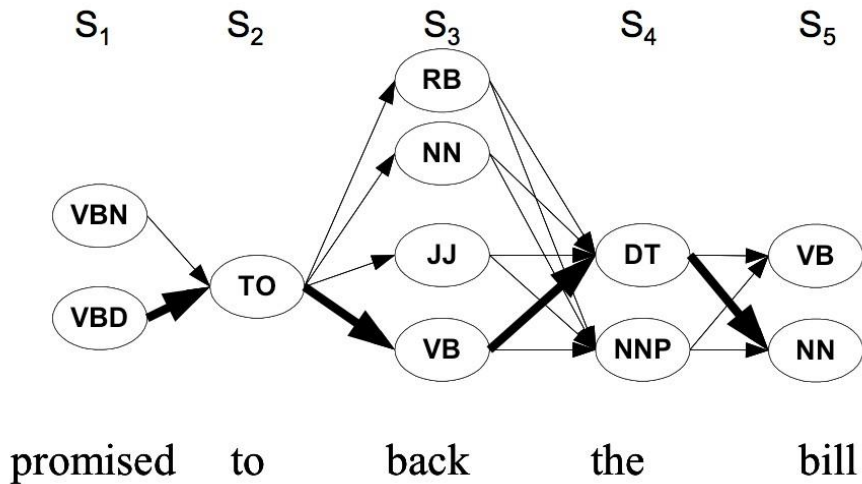
At each state we emit a word: $P(w_n | t_n)$



Walking through the states: best path



Walking through the states: best path



Finding the best path: Viterbi Algorithm

Intuition

Optimal path for each state can be recorded. We need

- Cheapest cost to state j at step s : $\delta_j(s)$
- Backtrace from that state to best predecessor $\psi_j(s)$

Computing these values

- $\delta_j(s+1) = \max_{1 \leq i \leq N} \delta_i(s) p(t_j | t_i) p(w_{s+1} | t_j)$
- $\psi_j(s+1) = \operatorname{argmax}_{1 \leq i \leq N} \delta_i(s) p(t_j | t_i) p(w_{s+1} | t_j)$

Finding the best path: Viterbi Algorithm

Intuition

Optimal path for each state can be recorded. We need

- Cheapest cost to state j at step s : $\delta_j(s)$
- Backtrace from that state to best predecessor $\psi_j(s)$

Computing these values

- $\delta_j(s+1) = \max_{1 \leq i \leq N} \delta_i(s) p(t_j | t_i) p(w_{s+1} | t_j)$
- $\psi_j(s+1) = \operatorname{argmax}_{1 \leq i \leq N} \delta_i(s) p(t_j | t_i) p(w_{s+1} | t_j)$

Best final state is $\operatorname{argmax}_{1 \leq i \leq N} \delta_i(|S|)$, we can backtrack from there