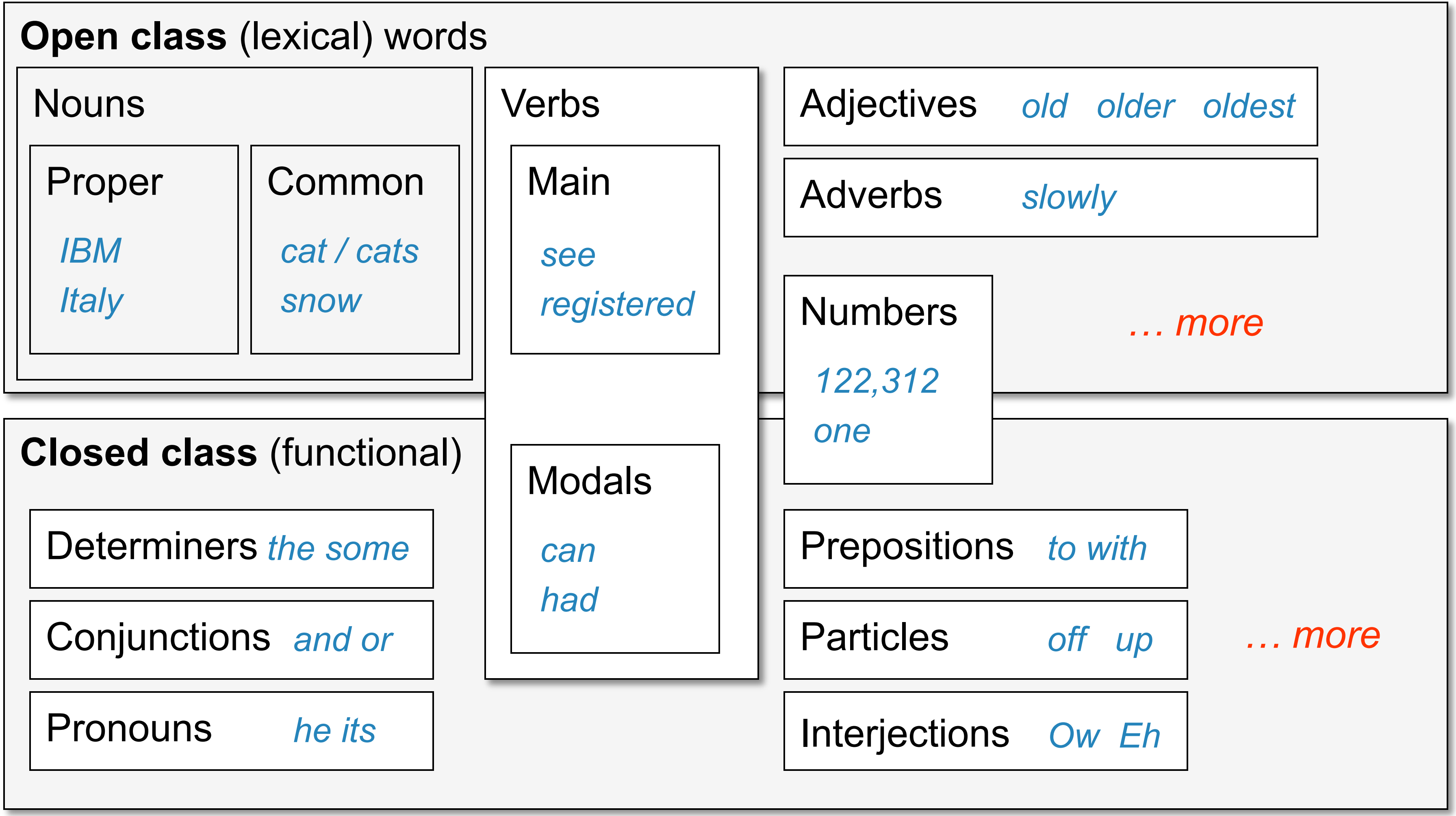
Part-of-speech tagging

Parts of Speech

- Perhaps starting with Aristotle in the West (384–322 BCE) the idea of having parts of speech
  - lexical categories, word classes, “tags”, POS
- Dionysius Thrax of Alexandria (c. 100 BCE): 8 parts of speech
  - Still with us! But his 8 aren’t exactly the ones we are taught today
    - *Thrax*: noun, verb, article, adverb, preposition, conjunction, participle, pronoun
    - *School grammar*: noun, verb, adjective, adverb, preposition, conjunction, pronoun, interjection



Open class (lexical) words
Nouns
Proper
IBM
Italy
Common
cat / cats
snow
Verbs
Main
see
registered
Modals
can
had
Adjectives old older oldest
Adverbs slowly
Numbers
122,312
one
… more
Closed class (functional)
Determiners the some
Conjunctions and or
Pronouns he its
Prepositions to with
Particles off up
Interjections Ow Eh
… more

Open vs. Closed classes

- Open vs. Closed classes
  - Closed:
    - determiners: ***a, an, the***
    - pronouns: ***she, he, I***
    - prepositions: ***on, under, over, near, by, …***
    - Why "closed"?
  - Open:
    - Nouns, Verbs, Adjectives, Adverbs.

POS Tagging

- Words often have more than one POS: *back*
  - The *back* door = JJ
  - On my *back* = NN
  - Win the voters *back* = RB
  - Promised to *back* the bill = VB
- The POS tagging problem is to determine the POS tag for a particular instance of a word.

POS Tagging

Penn Treebank POS tags

- Input: Plays well with others
- Ambiguity: NNS/VBZ UH/JJ/NN/RB IN NNS
- Output: Plays/VBZ well/RB with/IN others/NNS
- Uses:
  - MT: reordering of adjectives and nouns (say from Spanish to English)
  - Text-to-speech (how do we pronounce "lead"?)
  - Can write regexps like (Det) Adj* N+ over the output for phrases, etc.
  - Input to a syntactic parser

The Penn TreeBank Tagset

Tag	Description	Example	Tag	Description	Example
CC	coordin. conjunction	*and, but, or*	SYM	symbol	+,%, &
CD	cardinal number	*one, two*	TO	“to”	*to*
DT	determiner	*a, the*	UH	interjection	*ah, oops*
EX	existential ‘there’	*there*	VB	verb base form	*eat*
FW	foreign word	*mea culpa*	VBD	verb past tense	*ate*
IN	preposition/sub-conj	*of, in, by*	VBG	verb gerund	*eating*
JJ	adjective	*yellow*	VBN	verb past participle	*eaten*
JJR	adj., comparative	*bigger*	VBP	verb non-3sg pres	*eat*
JJS	adj., superlative	*wildest*	VBZ	verb 3sg pres	*eats*
LS	list item marker	*1, 2, One*	WDT	wh-determiner	*which, that*
MD	modal	*can, should*	WP	wh-pronoun	*what, who*
NN	noun, sing. or mass	*llama*	WP$	possessive wh-	*whose*
NNS	noun, plural	*llamas*	WRB	wh-adverb	*how, where*
NNP	proper noun, sing.	*IBM*	$	dollar sign	*$*
NNPS	proper noun, plural	*Carolinas*	#	pound sign	*#*
PDT	predeterminer	*all, both*	“	left quote	‘ or “
POS	possessive ending	*’s*	”	right quote	’ or ”
PRP	personal pronoun	*I, you, he*	(	left parenthesis	[, (, {, <
PRP$	possessive pronoun	*your, one’s*	)	right parenthesis	],), }, >
RB	adverb	*quickly, never*	,	comma	,
RBR	adverb, comparative	*faster*	.	sentence-final punc	. ! ?
RBS	adverb, superlative	*fastest*	:	mid-sentence punc	: ; ... – -
RP	particle	*up, off*			

Penn Treebank tags

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

There/EX are/VBP 70/CD children/NNS **there/RB**

Preliminary/JJ findings/NNS were/VBD **reported/VBN** in/IN today/NN **’s/POS** New/NNP England/NNP Journal/NNP of/IN Medicine/NNP ./.

POS tagging performance

- How many tags are correct? (Tag accuracy)
  - About 97% currently
  - But baseline is already 90%
    - Baseline is performance of stupidest possible method
      - Tag every word with its most frequent tag
      - Tag unknown words as nouns
  - Partly easy because
    - Many words are unambiguous
    - You get points for them (*the*, *a*, etc.) and for punctuation marks!

Deciding on the correct part of speech can be difficult even for people

- Mrs/NNP Shaefer/NNP never/RB got/VBD around/RP to/TO joining/VBG
- All/DT we/PRP gotta/VBN do/VB is/VBZ go/VB around/IN the/DT corner/NN
- Chateau/NNP Petrus/NNP costs/VBZ around/RB 250/CD

How difficult is POS tagging?

- About 11% of the word types in the Brown corpus are ambiguous with regard to part of speech
- But they tend to be very common words. E.g., *that*
  - I know *that* he is honest = IN
  - Yes, *that* play was nice = DT
  - You can’t go *that* far = RB
- 40% of the word tokens are ambiguous

Sources of information

- What are the main sources of information for POS tagging?
  - Knowledge of neighboring words
    - Bill saw that man yesterday
    - NNP NN DT NN NN
    - VB VB(D) IN VB NN
  - Knowledge of word probabilities
    - *man* is rarely used as a verb….
- The latter proves the most useful, but the former also helps

More and Better Features ➔ Feature-based tagger

- Can do surprisingly well just looking at a word by itself:
  - Word — the: the → DT
  - Lowercased word — Importantly: importantly → RB
  - Prefixes — unfathomable: un- → JJ
  - Suffixes — Importantly: -ly → RB
  - Capitalization — Meridian: CAP → NNP
  - Word shapes — 35-year: d-x → JJ
- Then build a classifier to predict tag
  - Maxent $P(t|w)$: 93.7% overall / 82.6% unknown

Overview: POS Tagging Accuracies

- Rough accuracies:
  - Most freq tag: ~90% / ~50%
  - Trigram HMM: ~95% / ~55%
  - Maxent P(t|w): 93.7% / 82.6%
  - TnT (HMM++): 96.2% / 86.0%
  - MEMM tagger: 96.9% / 86.9%
  - Bidirectional dependencies: 97.2% / 90.0%
  - Upper bound: ~98% (human agreement)

Most errors on unknown words

POS tagging as a sequence classification task

- We are given a sentence (an "observation" or "sequence of observations")
  - Secretariat is expected to race tomorrow
  - She promised to back the bill
- What is the best sequence of tags which 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 w1…wn.

How do we apply classification to sequences?

Sequence Labeling as Classification

- Classify each token independently but use as input features, information about the surrounding tokens (sliding window).

John saw the saw and decided to take it to the table.

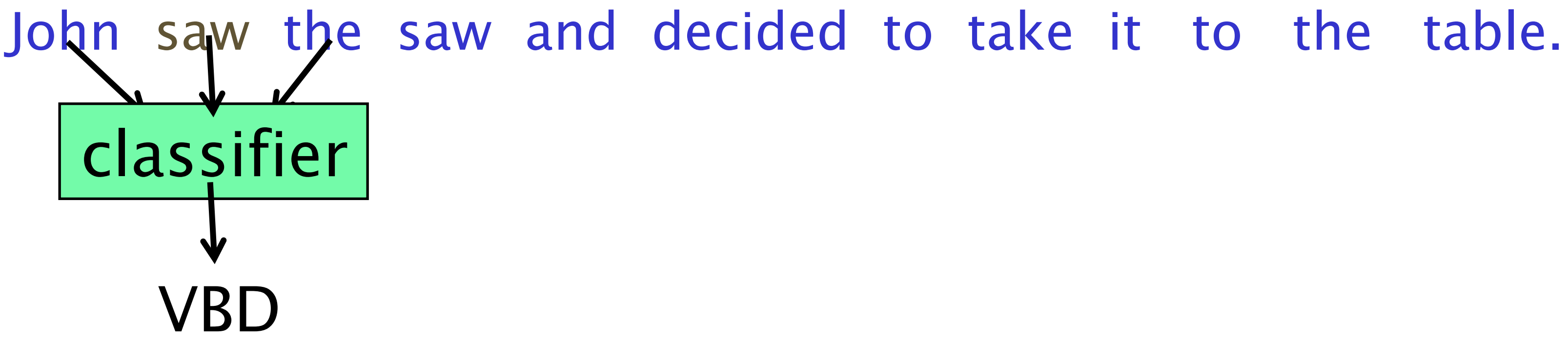
classifier

NNP

Slide from Ray Mooney

Sequence Labeling as Classification

- Classify each token independently but use as input features, information about the surrounding tokens (sliding window).



Sequence Labeling as Classification

- Classify each token independently but use as input features, information about the surrounding tokens (sliding window).

John saw the saw and decided to take it to the table.

classifier

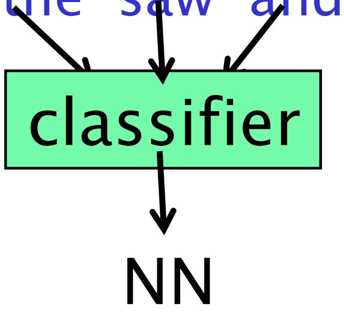
DT

Slide from Ray Mooney

Sequence Labeling as Classification

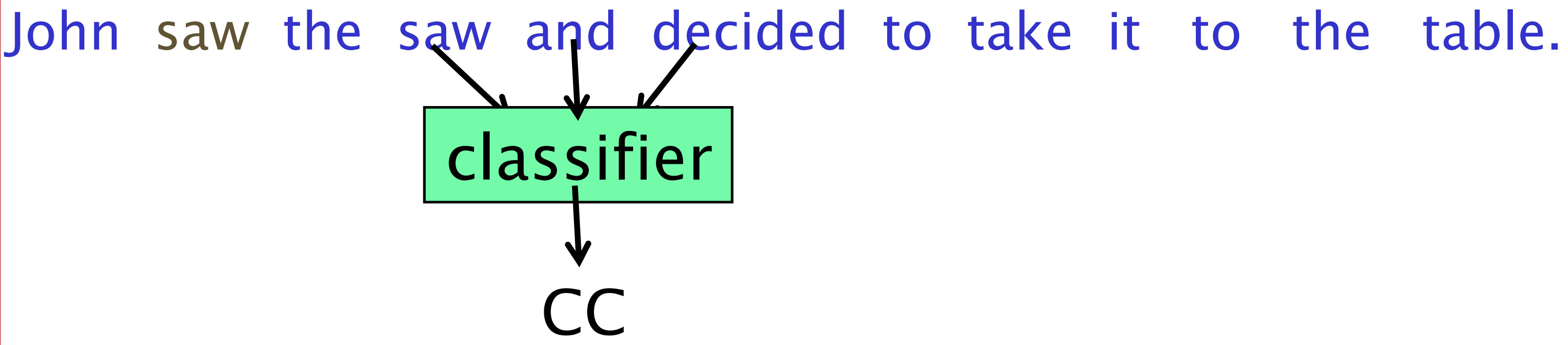
- Classify each token independently but use as input features, information about the surrounding tokens (sliding window).

John saw the saw and decided to take it to the table.



Sequence Labeling as Classification

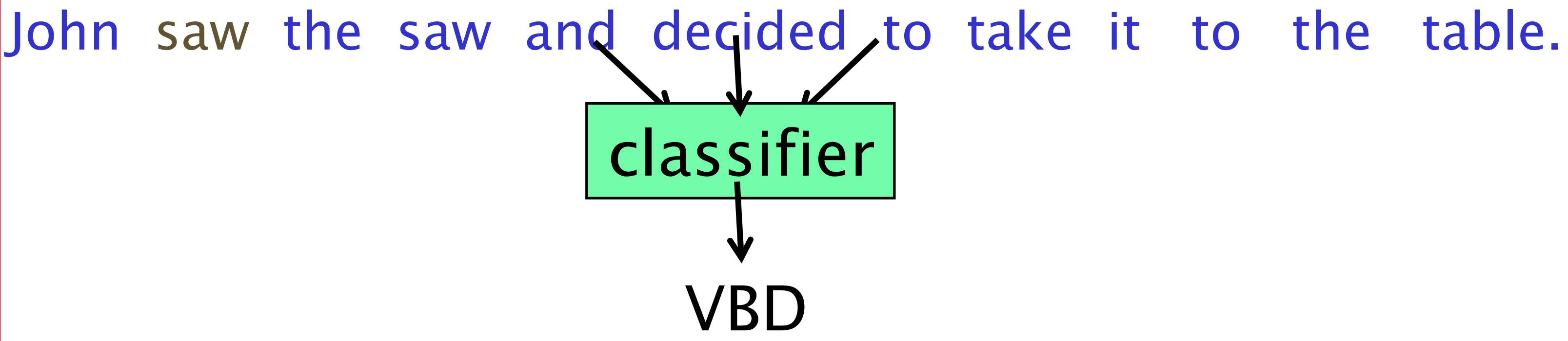
- Classify each token independently but use as input features, information about the surrounding tokens (sliding window).



Slide from Ray Mooney

Sequence Labeling as Classification

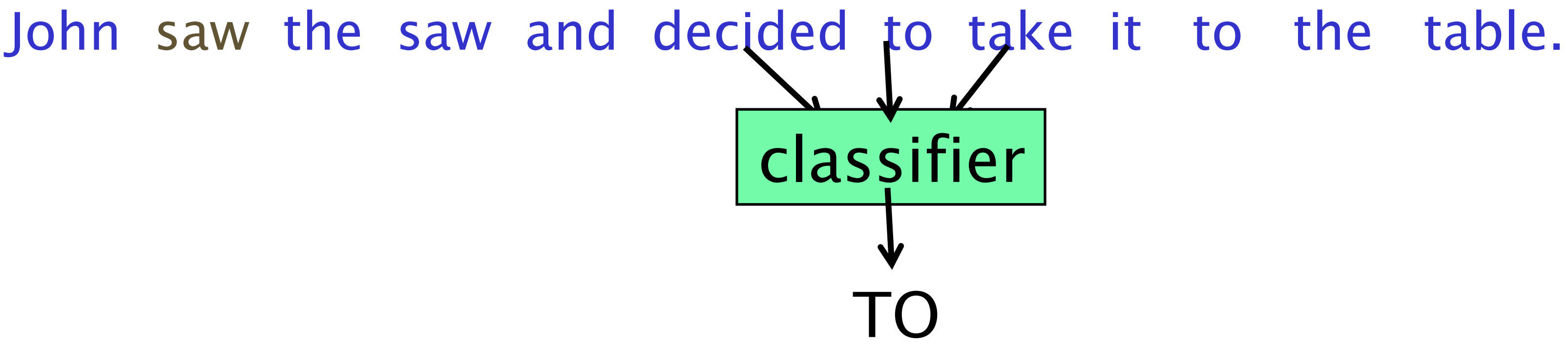
- Classify each token independently but use as input features, information about the surrounding tokens (sliding window).



Slide from Ray Mooney

Sequence Labeling as Classification

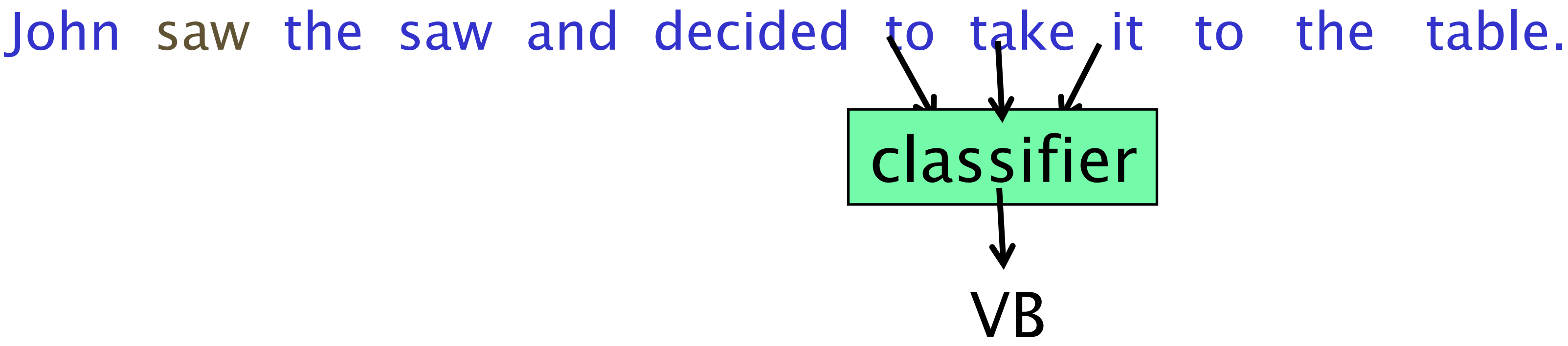
- Classify each token independently but use as input features, information about the surrounding tokens (sliding window).



Slide from Ray Mooney

Sequence Labeling as Classification

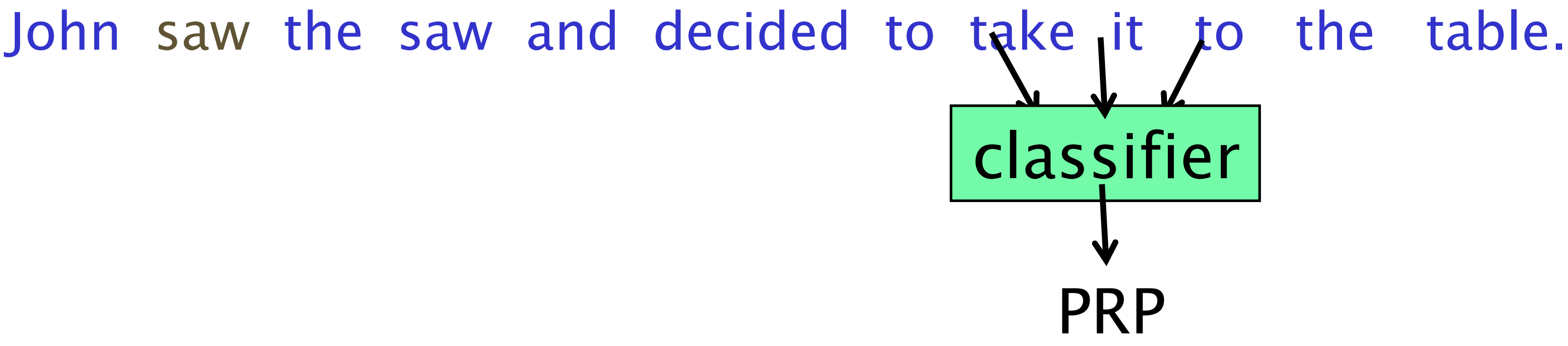
- Classify each token independently but use as input features, information about the surrounding tokens (sliding window).



Slide from Ray Mooney

Sequence Labeling as Classification

- Classify each token independently but use as input features, information about the surrounding tokens (sliding window).

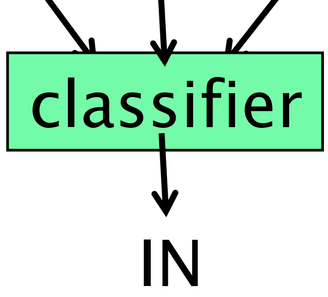


Slide from Ray Mooney

Sequence Labeling as Classification

- Classify each token independently but use as input features, information about the surrounding tokens (sliding window).

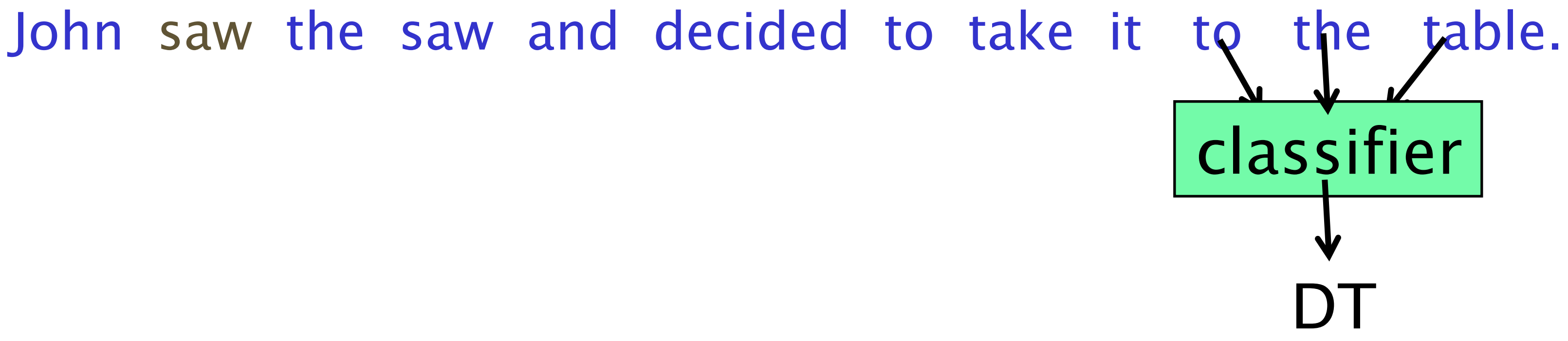
John saw the saw and decided to take it to the table.



Slide from Ray Mooney

Sequence Labeling as Classification

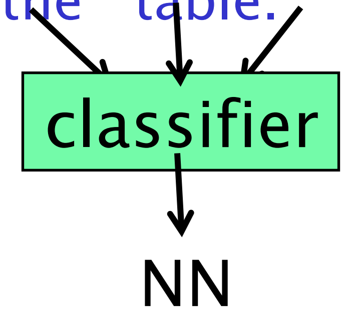
- Classify each token independently but use as input features, information about the surrounding tokens (sliding window).



Sequence Labeling as Classification

- Classify each token independently but use as input features, information about the surrounding tokens (sliding window).

John saw the saw and decided to take it to the table.

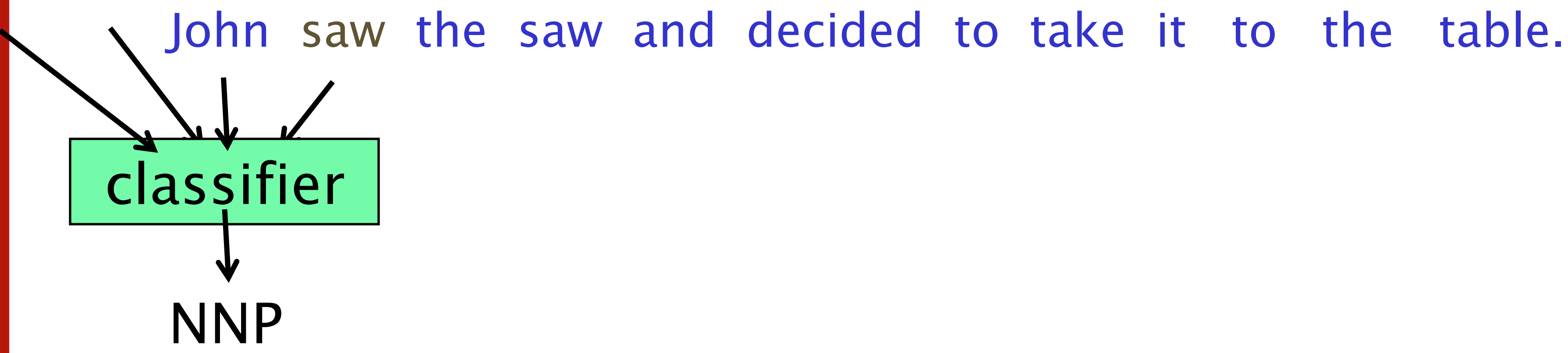


Sequence Labeling as Classification Using Outputs as Inputs

- Better input features are usually the categories of the surrounding tokens, but these are not available yet.
- Can use category of either the preceding or succeeding tokens by going forward or back and using previous output.

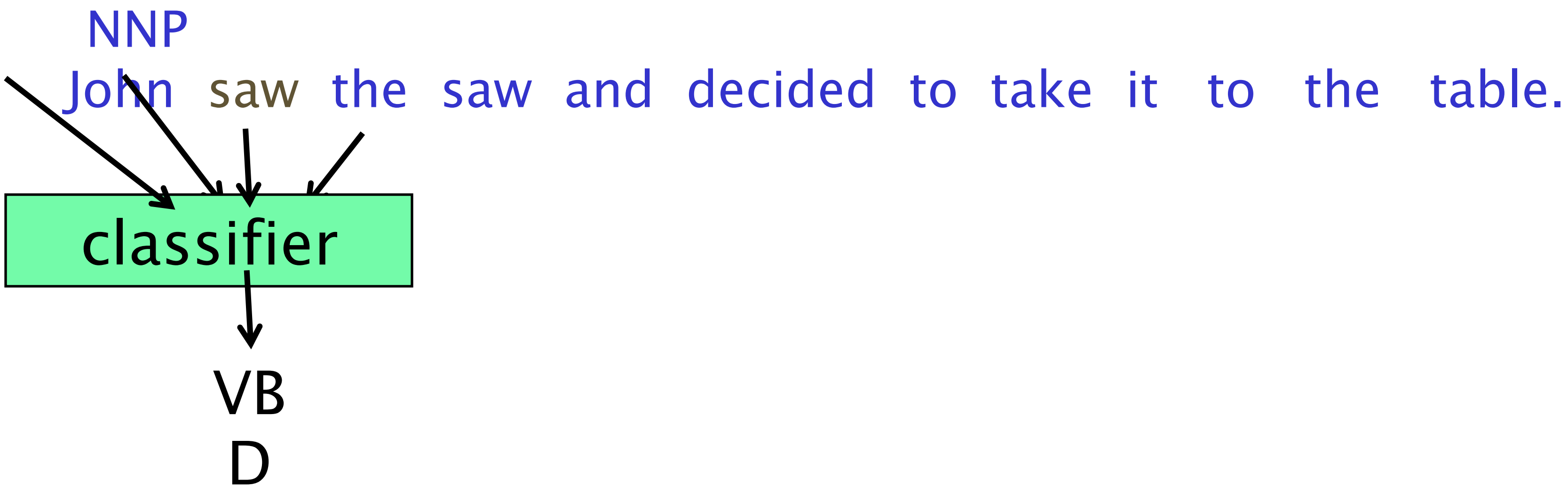
Slide from Ray Mooney

Forward Classification



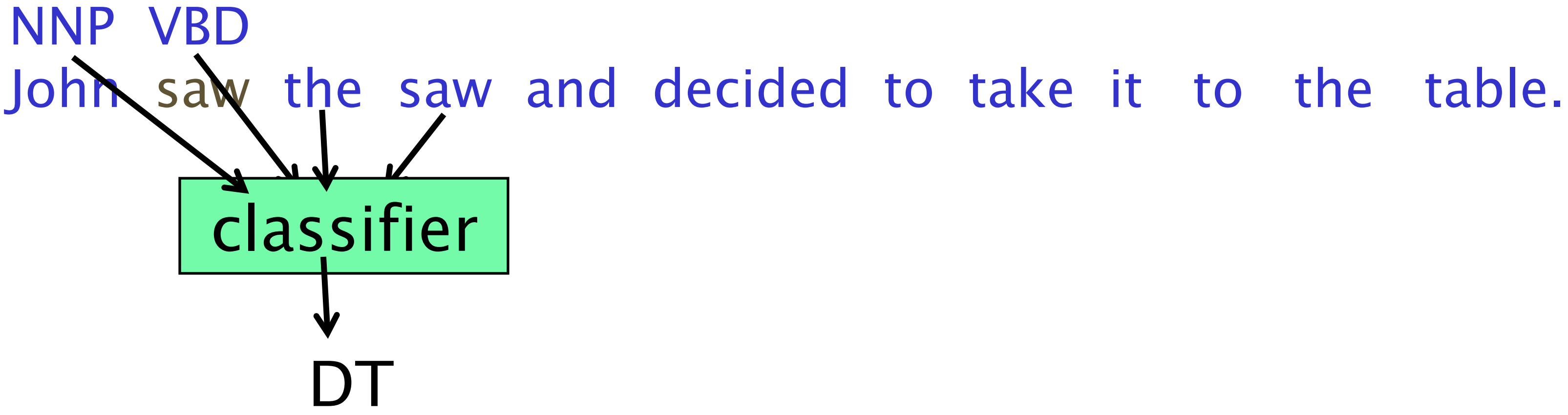
Slide from Ray Mooney

Forward Classification



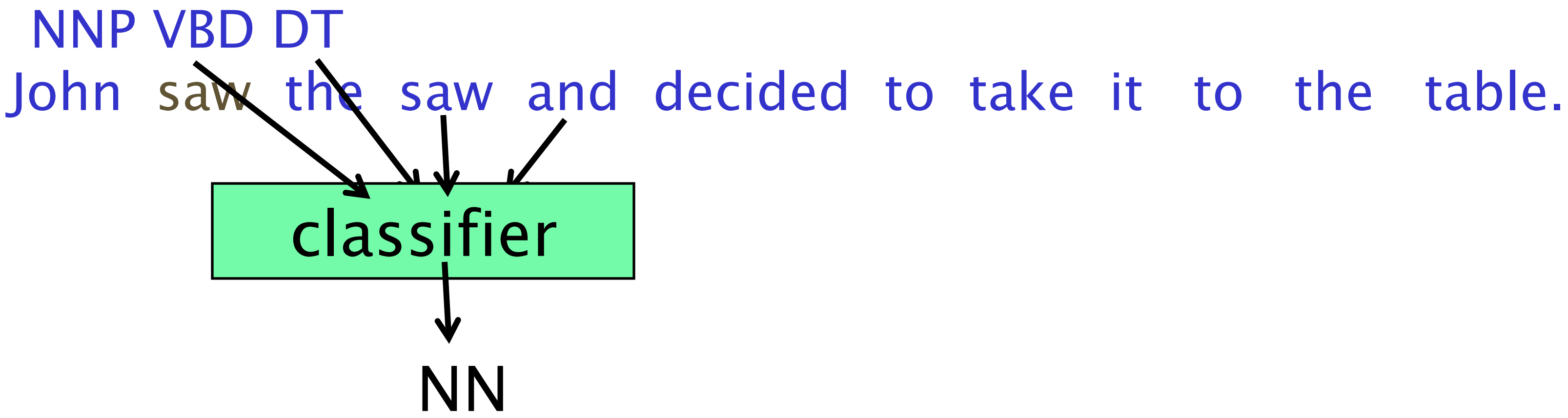
Slide from Ray Mooney

Forward Classification



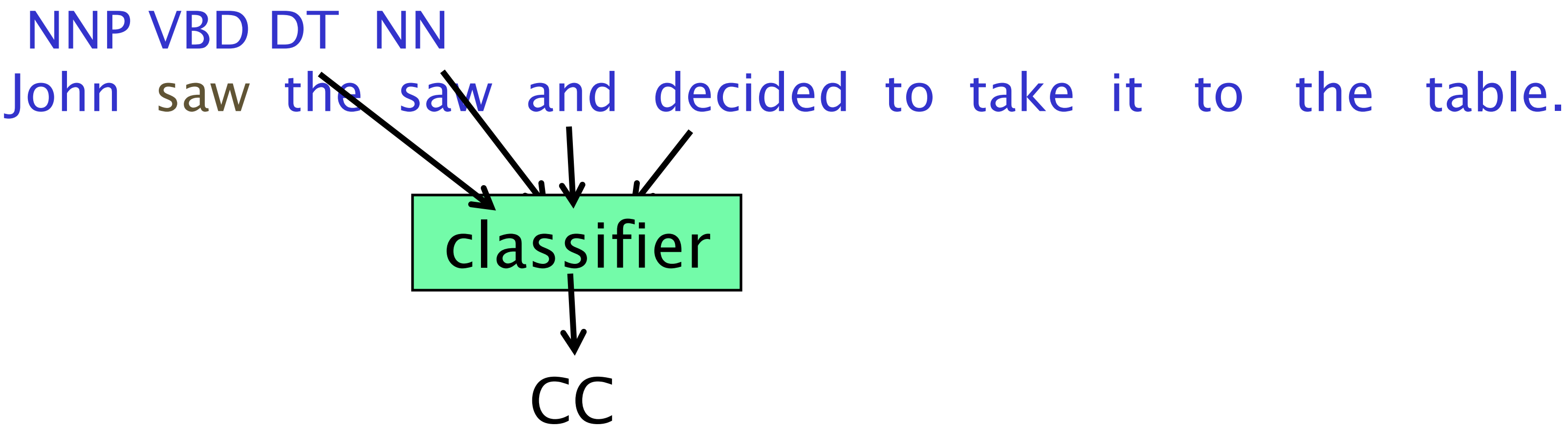
Slide from Ray Mooney

Forward Classification



Slide from Ray Mooney

Forward Classification



Slide from Ray Mooney

Forward Classification

NNP VBD DT NN CC

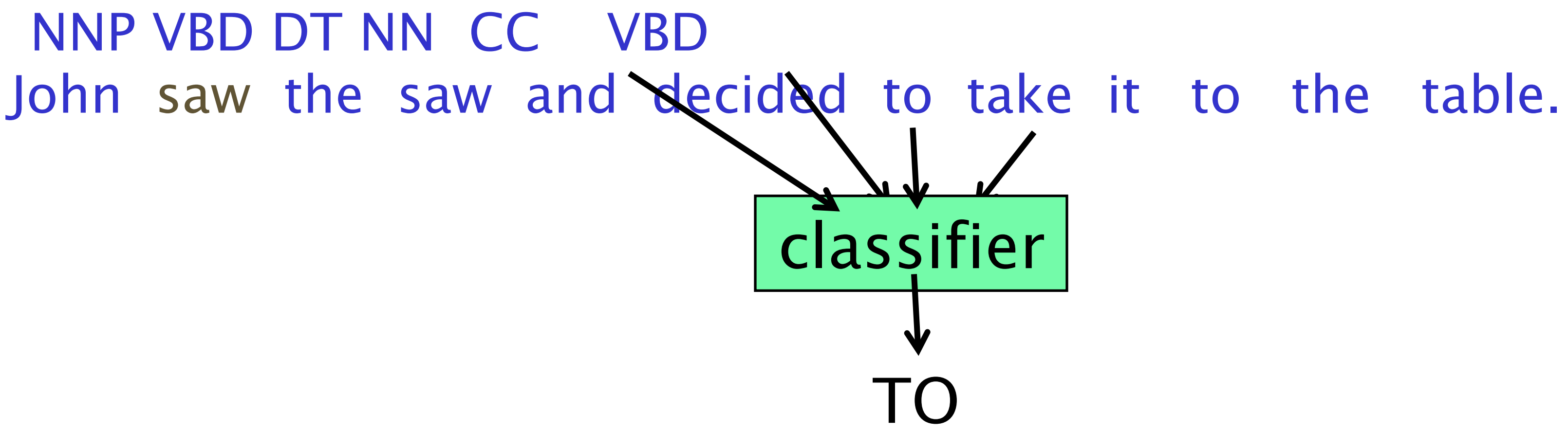
John saw the saw and decided to take it to the table.

classifier

VBD

Slide from Ray Mooney

Forward Classification



Slide from Ray Mooney

Forward Classification

NNP VBD DT NN CC VBD TO

John saw the saw and decided to take it to the table.

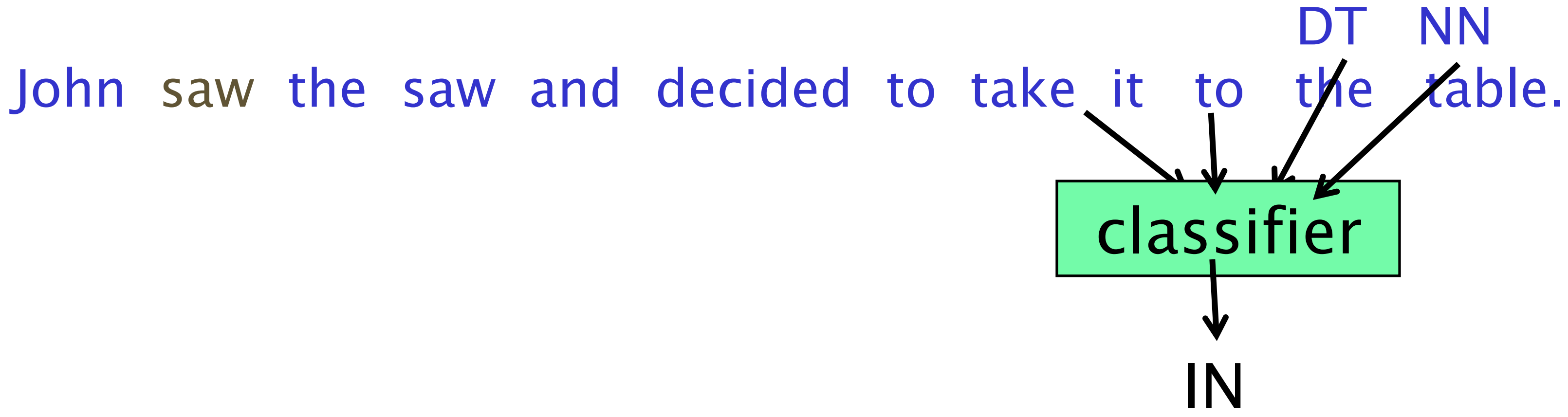
classifier

VB

Slide from Ray Mooney

Backward Classification

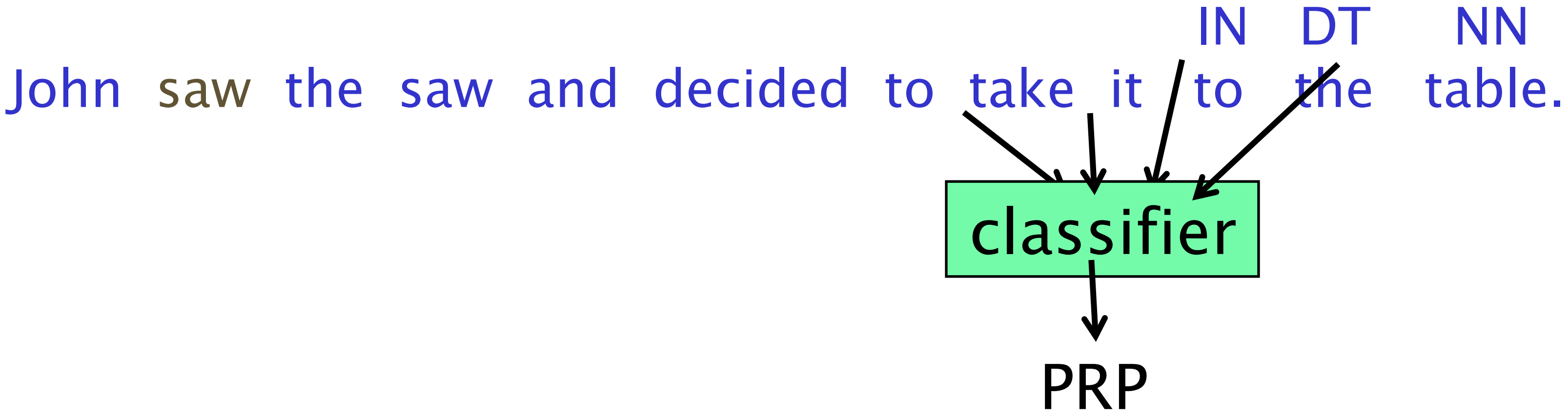
- Disambiguating “to” in this case would be even easier backward.



Slide from Ray Mooney

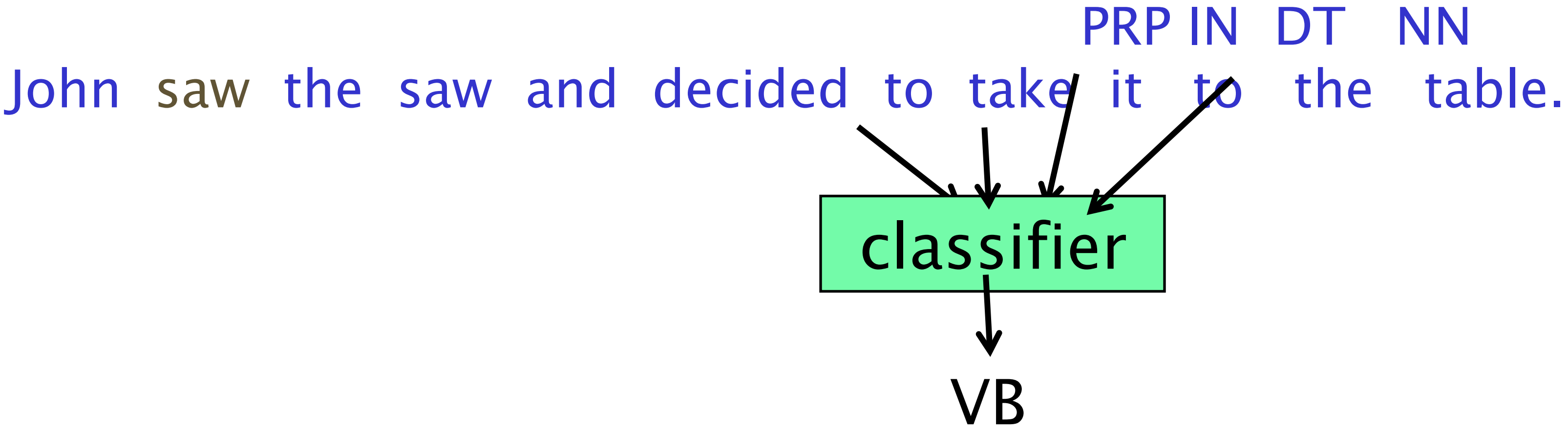
Backward Classification

- Disambiguating “to” in this case would be even easier backward.



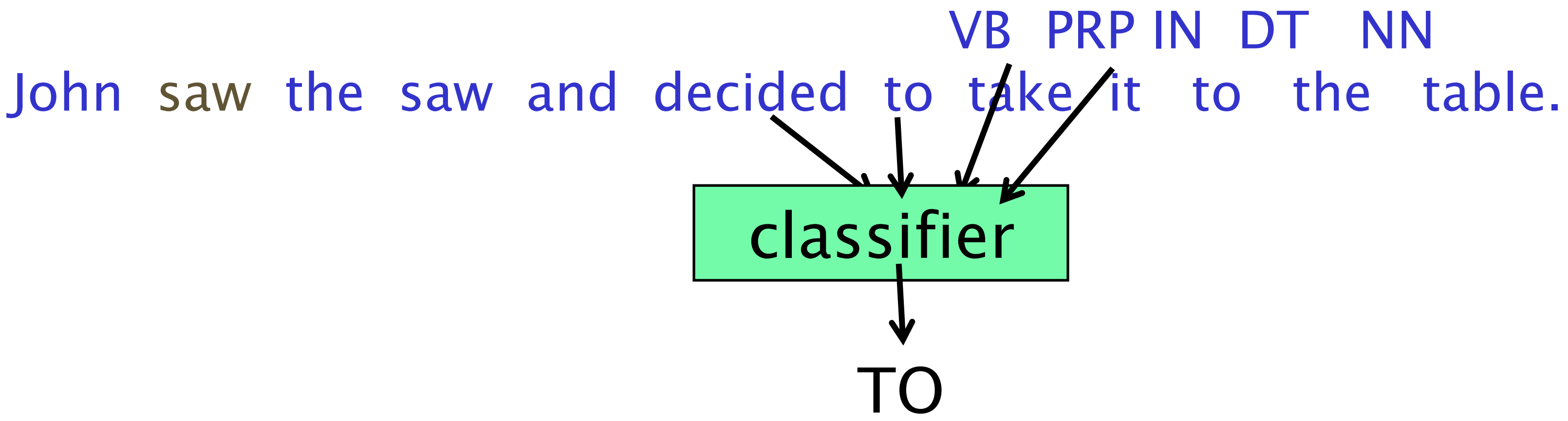
Backward Classification

- Disambiguating “to” in this case would be even easier backward.



Backward Classification

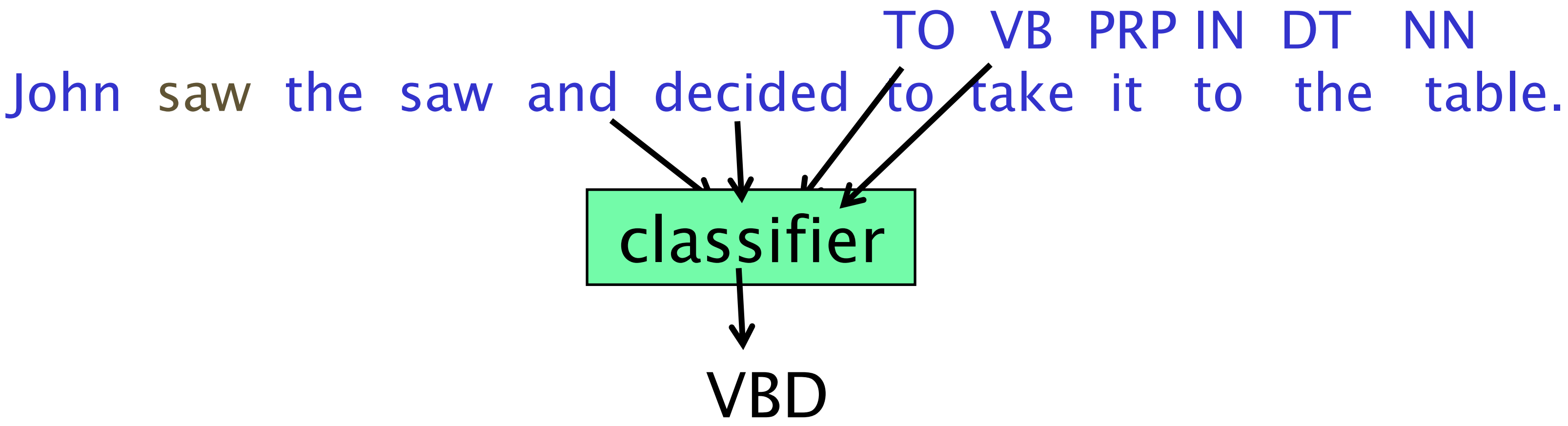
- Disambiguating “to” in this case would be even easier backward.



Slide from Ray Mooney

Backward Classification

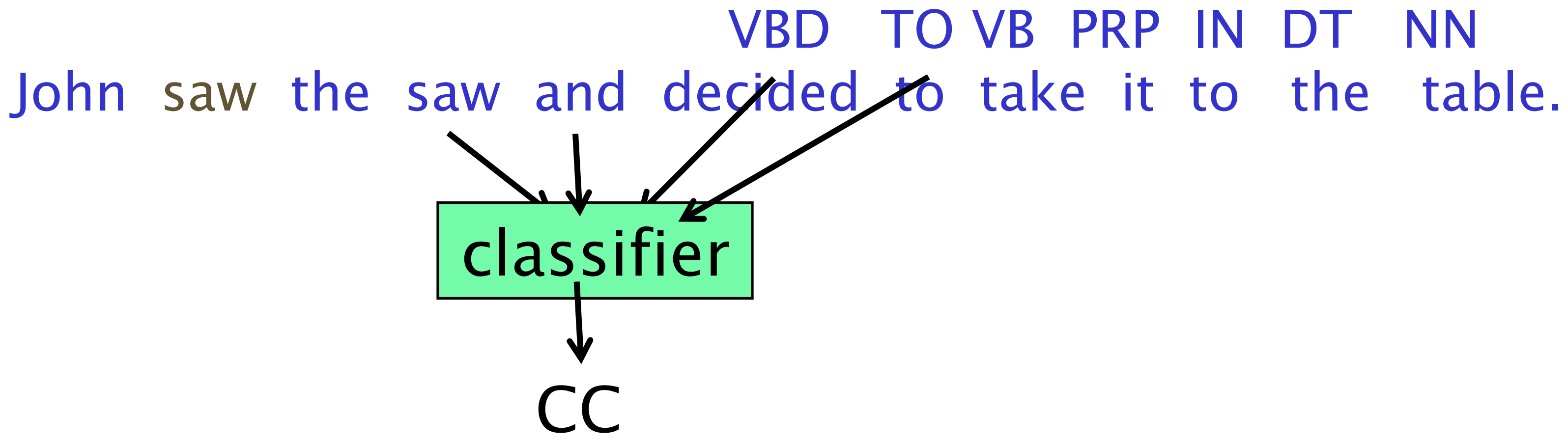
- Disambiguating “to” in this case would be even easier backward.



Slide from Ray Mooney

Backward Classification

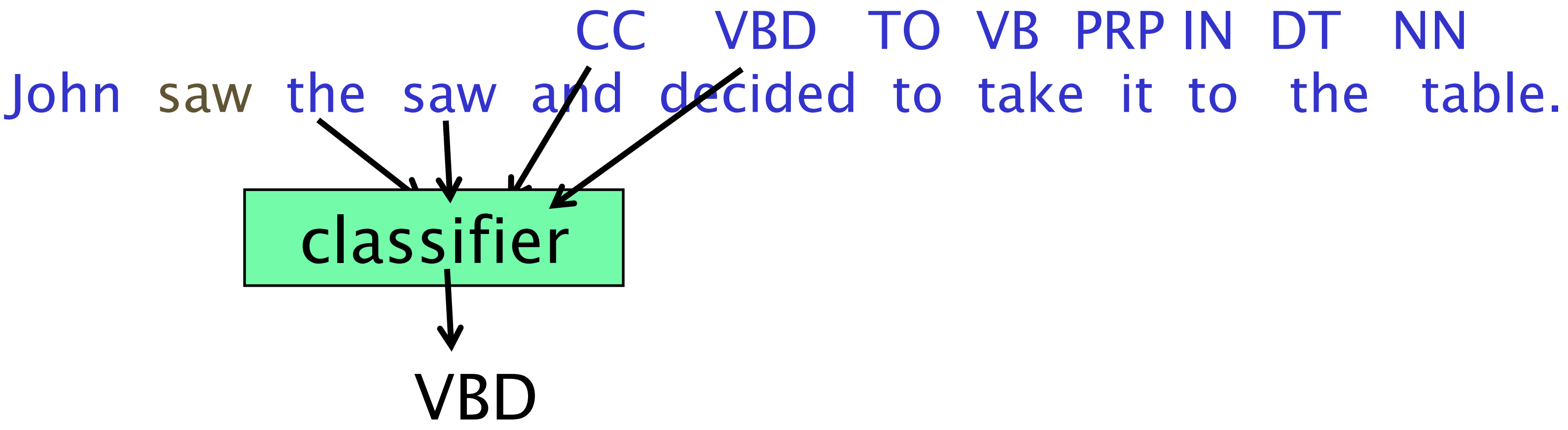
- Disambiguating “to” in this case would be even easier backward.



Slide from Ray Mooney

Backward Classification

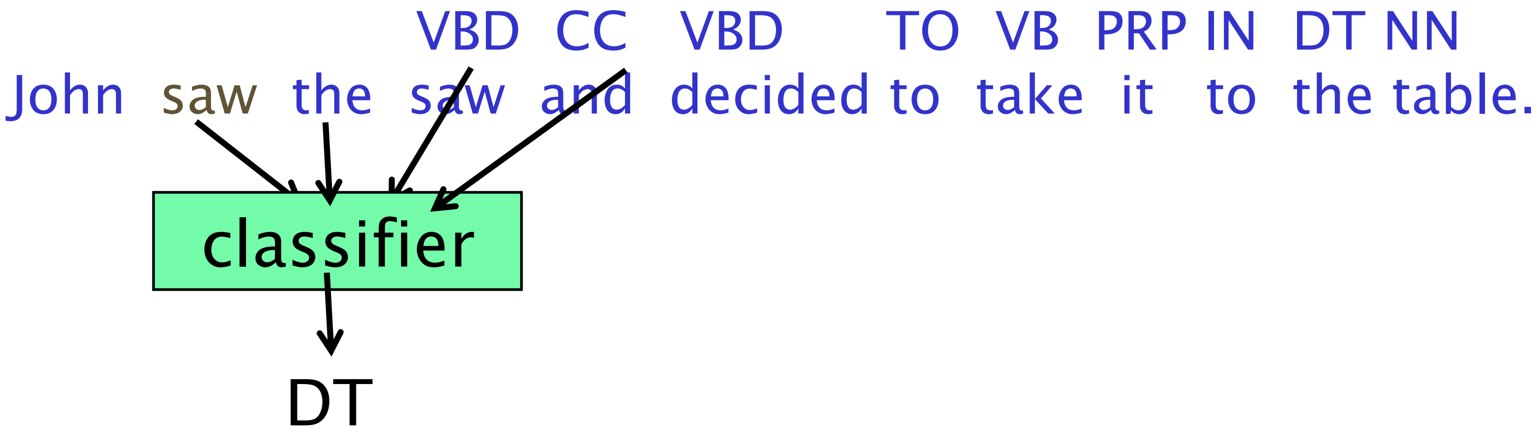
- Disambiguating “to” in this case would be even easier backward.



Slide from Ray Mooney

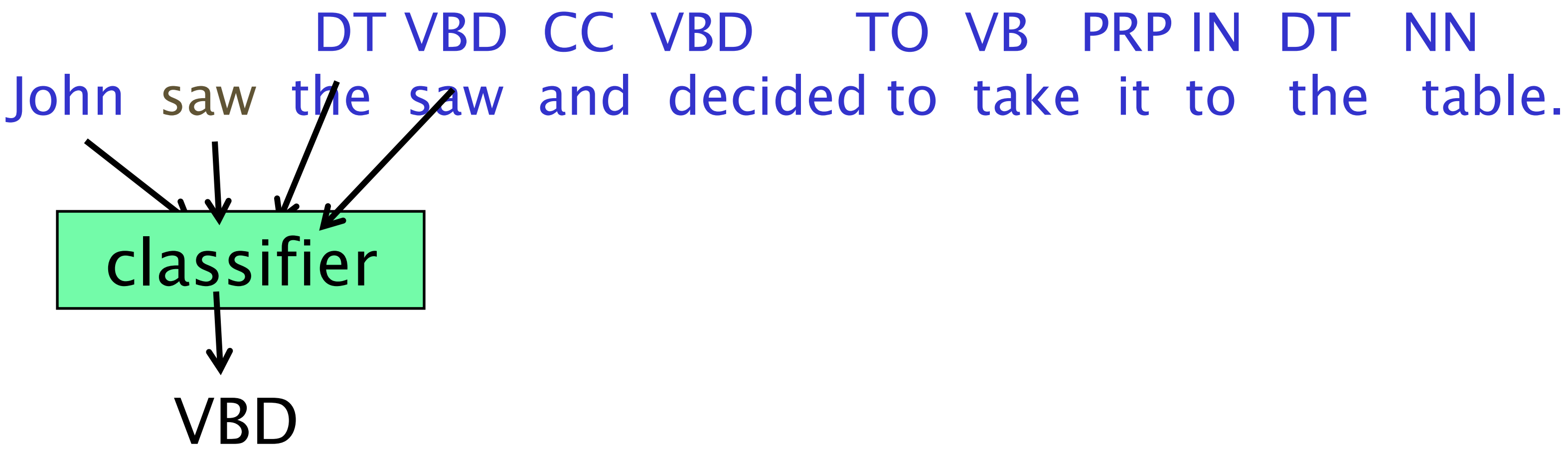
Backward Classification

- Disambiguating “to” in this case would be even easier backward.



Backward Classification

- Disambiguating “to” in this case would be even easier backward.



Slide from Ray Mooney

Backward Classification

- Disambiguating “to” in this case would be even easier backward.

VBD DT VBD CC VBD TO VB PRP IN DT NN

John saw the saw and decided to take it to the table.

classifier

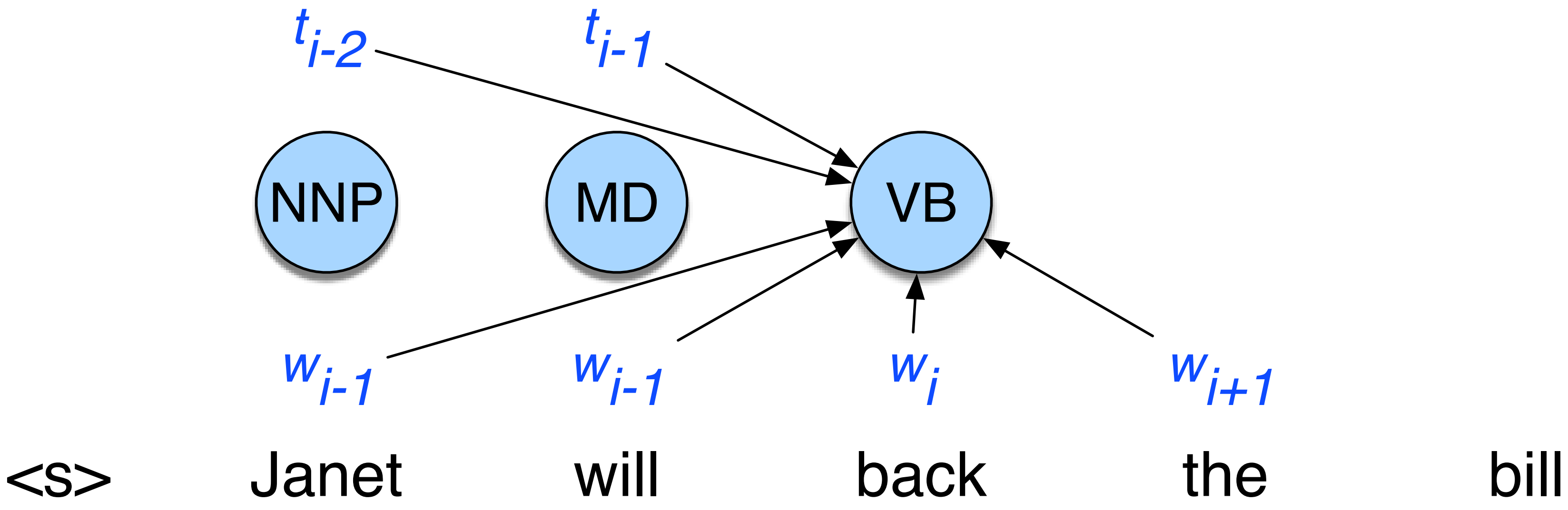
NNP

The Maximum Entropy Markov Model (MEMM)

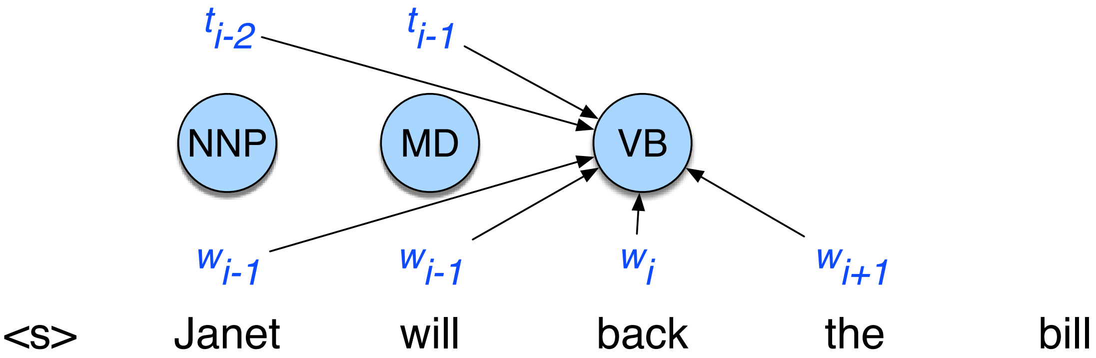
- A sequence version of the logistic regression (also called maximum entropy) classifier.
- Find the best series of tags:

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

The Maximum Entropy Markov Model (MEMM)



Features for the classifier at each tag



$$t_i = \text{VB and } w_{i-2} = \text{Janet}$$
$$t_i = \text{VB and } w_{i-1} = \text{will}$$
$$t_i = \text{VB and } w_i = \text{back}$$
$$t_i = \text{VB and } w_{i+1} = \text{the}$$
$$t_i = \text{VB and } w_{i+2} = \text{bill}$$
$$t_i = \text{VB and } t_{i-1} = \text{MD}$$
$$t_i = \text{VB and } t_{i-1} = \text{MD and } t_{i-2} = \text{NNP}$$
$$t_i = \text{VB and } w_i = \text{back and } w_{i+1} = \text{the}$$

More features

w_i contains a particular prefix (from all prefixes of length ≤ 4)
w_i contains a particular suffix (from all suffixes of length ≤ 4)
w_i contains a number
w_i contains an upper-case letter
w_i contains a hyphen
w_i is all upper case
w_i's word shape
w_i's short word shape
w_i is upper case and has a digit and a dash (like *CFC-12*)
w_i is upper case and followed within 3 words by Co., Inc., etc.

MEMM computes the best tag sequence

$$
\begin{aligned}
\hat{T} &= \operatorname*{argmax}_T P(T|W) \\
&= \operatorname*{argmax}_T \prod_i P(t_i|w_{i-l}^{i+l}, t_{i-k}^{i-1}) \\
&= \operatorname*{argmax}_T \prod_i \frac{\exp\left(\sum_i w_i f_i(t_i, w_{i-l}^{i+l}, t_{i-k}^{i-1})\right)}{\sum_{t' \in \text{tagset}} \exp\left(\sum_i w_i f_i(t', w_{i-l}^{i+l}, t_{i-k}^{i-1})\right)}
\end{aligned}
$$

MEMM Decoding

- Simplest algorithm:

function GREEDY MEMM DECODING(words W, model P) **returns** tag sequence T

for $i = 1$ **to** *length(W)*

$$\hat{t}_i = \underset{t' \in T}{\operatorname{argmax}} \; P(t' \mid w_{i-l}^{i+l}, t_{i-k}^{i-1})$$

- What we use in practice: The **Viterbi** algorithm
- A version of the same dynamic programming algorithm we used to compute minimum edit distance.

The Stanford Tagger

- Is a bidirectional version of the MEMM called a cyclic dependency network

- Stanford tagger:
  - http://nlp.stanford.edu/software/tagger.shtml