SIT330-770: Natural Language Processing

Week 8 - Sequence Labeling

Dr. Mohamed Reda Bouadjenek

School of Information Technology, Faculty of Sci Eng & Built Env

reda.bouadjenek@deakin.edu.au

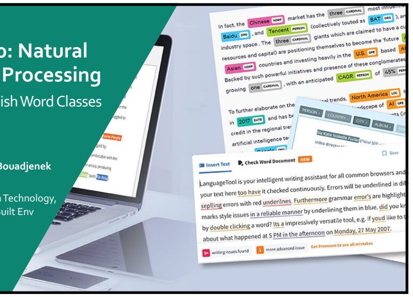
1

SIT330-770: Natural Language Processing

Week 8.1 - English Word Classes

Dr. Mohamed Reda Bouadjenek

School of Information Technology, Faculty of Sci Eng & Built Env



2

Parts of Speech

- From the earliest linguistic traditions (Yaska and Panini 5th C. BCE, Aristotle 4th C. BCE), the idea that words can be classified into grammatical categories
  - part of speech, word classes, POS, POS tags
- 8 parts of speech attributed to Dionysius Thrax of Alexandria (c. 1st C. BCE):
  - noun, verb, pronoun, preposition, adverb, conjunction, participle, article
  - These categories are relevant for NLP today.

3

Two classes of words: Open vs. Closed

- Closed class words
  - Relatively fixed membership
  - Usually **function** words: short, frequent words with grammatical function
    - determiners: *a, an, the*
    - pronouns: *she, he, I*
    - prepositions: *on, under, over, near, by, …*
- Open class words
  - Usually **content** words: Nouns, Verbs, Adjectives, Adverbs
    - Plus interjections: *oh, ouch, uh-huh, yes, hello*
  - New nouns and verbs like *iPhone* or *to fax*

4

Open class ("content") words

- Nouns
  - Proper: *Janet, Italy*
  - Common: *cat, cats, mango*
- Verbs
  - Main: *eat, went*
- Adjectives: *old, green, tasty*
- Adverbs: *slowly, yesterday*
- Numbers: *122,312, one*
- Interjections: *Ow, hello*
- *… more*

Closed class ("function")

- Determiners: *the, some*
- Conjunctions: *and, or*
- Pronouns: *they, its*
- Auxiliary: *can, had*
- Prepositions: *to, with*
- Particles: *off, up*
- *… more*

5

Part-of-Speech Tagging

- Assigning a part-of-speech to each word in a text.
- Words often have more than one POS.
- **book**:
  - VERB: (***Book** that flight*)
  - NOUN: (*Hand me that **book***).

6

"Universal Dependencies" Tagset

	Tag	Description	Example
Open Class	**ADJ**	Adjective: noun modifiers describing properties	*red, young, awesome*
	ADV	Adverb: verb modifiers of time, place, manner	*very, slowly, home, yesterday*
	NOUN	words for persons, places, things, etc.	*algorithm, cat, mango, beauty*
	VERB	words for actions and processes	*draw, provide, go*
	PROPN	Proper noun: name of a person, organization, place, etc..	*Regina, IBM, Colorado*
	INTJ	Interjection: exclamation, greeting, yes/no response, etc.	*oh, um, yes, hello*
Closed Class Words	**ADP**	Adposition (Preposition/Postposition): marks a noun's spacial, temporal, or other relation	*in, on, by under*
	AUX	Auxiliary: helping verb marking tense, aspect, mood, etc.,	*can, may, should, are*
	CCONJ	Coordinating Conjunction: joins two phrases/clauses	*and, or, but*
	DET	Determiner: marks noun phrase properties	*a, an, the, this*
	NUM	Numeral	*one, two, first, second*
	PART	Particle: a preposition-like form used together with a verb	*up, down, on, off, in, out, at, by*
	PRON	Pronoun: a shorthand for referring to an entity or event	*she, who, I, others*
	SCONJ	Subordinating Conjunction: joins a main clause with a subordinate clause such as a sentential complement	*that, which*
Other	**PUNCT**	Punctuation	; , ()
	SYM	Symbols like $ or emoji	$, %
	X	Other	asdf, qwfg

Nivre et al. 2016

7

7

Sample "Tagged" English sentences

- There/PRO were/VERB 70/NUM children/NOUN there/ADV ./PUNC
- Preliminary/ADJ findings/NOUN were/AUX reported/VERB in/ADP today/NOUN 's/PART New/PROPN England/PROPN Journal/PROPN of/ADP Medicine/PROPN

8

8

SIT330-770: Natural Language Processing

Week 8.2 - Part of Speech Tagging

Dr. Mohamed Reda Bouadjenek

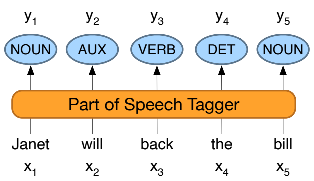
School of Information Technology,
Faculty of Sci Eng & Built Env

9

9

Part-of-Speech Tagging

- Map from sequence $x_1,\ldots,x_n$ of words to $y_1,\ldots,y_n$ of POS tags



10

10

Why Part of Speech Tagging?

- Can be useful for other NLP tasks
  - Parsing: POS tagging can improve syntactic parsing
  - MT: reordering of adjectives and nouns (say from Spanish to English)
  - Sentiment or affective tasks: may want to distinguish adjectives or other POS
  - Text-to-speech (how do we pronounce "lead" or "object"?)
- Or linguistic or language-analytic computational tasks
  - Need to control for POS when studying linguistic change like creation of new words, or meaning shift
  - Or control for POS in measuring meaning similarity or difference

11

11

How difficult is POS tagging in English?

- Roughly 15% of word types are ambiguous
  - Hence 85% of word types are unambiguous
  - *Janet* is always PROPN, *hesitantly* is always ADV
- But those 15% tend to be very common.
- So ~60% of word tokens are ambiguous
- E.g., *back*
  - earnings growth took a *back*/ADJ seat
  - a small building in the *back*/NOUN
  - a clear majority of senators *back*/VERB the bill
  - enable the country to buy *back*/PART debt
  - I was twenty-one *back*/ADV then

12

12

POS tagging performance in English

- How many tags are correct? (Tag accuracy)
  - About 97%
    - Hasn't changed in the last 10+ years
    - HMMs, CRFs, BERT perform similarly .
    - Human accuracy about the same
- But baseline is 92%!
  - Baseline is performance of stupidest possible method
    - "Most frequent class baseline" is an important baseline for many tasks
      - Tag every word with its most frequent tag
      - (and tag unknown words as nouns)
  - Partly easy because
    - Many words are unambiguous

13

13

Sources of information for POS tagging

```
Janet will back the bill
```

AUX/NOUN/VERB? NOUN/VERB?

- Prior probabilities of word/tag
  - "will" is usually an AUX
- Identity of neighboring words
  - "the" means the next word is probably not a verb
- Morphology and wordshape:
  - Prefixes unable: un- → ADJ
  - Suffixes importantly: -ly → ADJ
  - Capitalization Janet: CAP → PROPN

14

14

Standard algorithms for POS tagging

- Supervised Machine Learning Algorithms:
- Hidden Markov Models
- Conditional Random Fields (CRF)/ Maximum Entropy Markov Models (MEMM)
- Neural sequence models (RNNs or Transformers)
- Large Language Models (like BERT), finetuned
- All required a hand-labeled training set, all about equal performance (97% on English)
- All make use of information sources we discussed
- Via human created features: HMMs and CRFs
- Via representation learning: Neural LMs

15

15

SIT330-770: Natural Language Processing

Week 8.3 - Named Entity Recognition (NER)

Dr. Mohamed Reda Bouadjenek

School of Information Technology, Faculty of Sci Eng & Built Env

16

16

Named Entities

- **Named entity**, in its core usage, means anything that can be referred to with a proper name. Most common 4 tags:
  - PER (Person): "Marie Curie"
  - LOC (Location): "New York City"
  - ORG (Organization): "Stanford University"
  - GPE (Geo-Political Entity): "Boulder, Colorado"
- Often multi-word phrases
- But the term is also extended to things that aren't entities:
  - dates, times, prices

17

17

Named Entity tagging

- The task of named entity recognition (NER):
  - find spans of text that constitute proper names
  - tag the type of the entity.

18

18

NER output

Citing high fuel prices, [ORG **United Airlines**] said [TIME **Friday**] it has increased fares by [MONEY **$6**] per round trip on flights to some cities also served by lower-cost carriers. [ORG **American Airlines**], a unit of [ORG **AMR Corp.**], immediately matched the move, spokesman [PER **Tim Wagner**] said. [ORG **United**], a unit of [ORG **UAL Corp.**], said the increase took effect [TIME **Thursday**] and applies to most routes where it competes against discount carriers, such as [LOC **Chicago**] to [LOC **Dallas**] and [LOC **Denver**] to [LOC **San Francisco**].

19

Why NER?

- Sentiment analysis: consumer's sentiment toward a particular company or person?
- Question Answering: answer questions about an entity?
- Information Extraction: Extracting facts about entities from text.

20

Why NER is hard

1) Segmentation
   - In POS tagging, no segmentation problem since each word gets one tag.
   - In NER we have to find and segment the entities!
2) Type ambiguity

[PER Washington] was born into slavery on the farm of James Burroughs.
[ORG Washington] went up 2 games to 1 in the four-game series.
Blair arrived in [LOC Washington] for what may well be his last state visit.
In June, [GPE Washington] passed a primary seatbelt law.

21

BIO Tagging

- How can we turn this structured problem into a sequence problem like POS tagging, with one label per word?
- [PER Jane Villanueva] of [ORG United] , a unit of [ORG United Airlines Holding] , said the fare applies to the [LOC Chicago] route.

22

BIO Tagging

- [PER Jane Villanueva] of [ORG United] , a unit of [ORG United Airlines Holding] , said the fare applies to the [LOC Chicago] route.

Words	BIO Label
Jane	B-PER
Villanueva	I-PER
of	O
United	B-ORG
Airlines	I-ORG
Holding	I-ORG
discussed	O
the	O
Chicago	B-LOC
route	O
.	O

Now we have one tag per token!!!

23

BIO Tagging

B: token that *begins* a span
I: tokens *inside* a span
O: tokens outside of any span

# of tags (where n is #entity types):
1 O tag,
n B tags,
n I tags
total of *2n+1*

Words	BIO Label
Jane	B-PER
Villanueva	I-PER
of	O
United	B-ORG
Airlines	I-ORG
Holding	I-ORG
discussed	O
the	O
Chicago	B-LOC
route	O
.	O

24

BIO Tagging variants: IO and BIOES

- [PER Jane Villanueva] of [ORG United] , a unit of [ORG United Airlines Holding] , said the fare applies to the [LOC Chicago] route.

Words	IO Label	BIO Label	BIOES Label
Jane	I-PER	B-PER	B-PER
Villanueva	I-PER	I-PER	E-PER
of	O	O	O
United	I-ORG	B-ORG	B-ORG
Airlines	I-ORG	I-ORG	I-ORG
Holding	I-ORG	I-ORG	E-ORG
discussed	O	O	O
the	O	O	O
Chicago	I-LOC	B-LOC	S-LOC
route	O	O	O
.	O	O	O

25

25

Standard algorithms for NER

- Supervised Machine Learning given a human-labeled training set of text annotated with tags
  - Hidden Markov Models
  - Conditional Random Fields (CRF)/ Maximum Entropy Markov Models (MEMM)
  - Neural sequence models (RNNs or Transformers)
  - Large Language Models (like BERT), finetuned

26

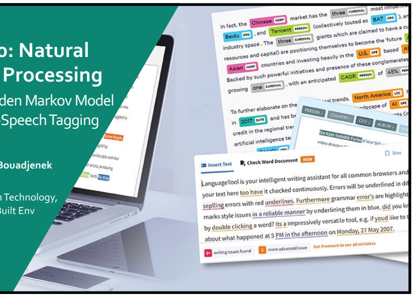
26

SIT330-770: Natural Language Processing

Week 8.4 – Hidden Markov Model (HMM) Part-of-Speech Tagging

Dr. Mohamed Reda Bouadjenek

School of Information Technology, Faculty of Sci Eng & Built Env



27

27

Introduction to Markov Chains

- A Markov chain is a model that describes the probabilities of sequences of random variables or states, where each state can take on values from a predefined set.
- It operates on the principle that future predictions depend solely on the current state, with no consideration of previous states except through the current one.
- Markov chains can represent various scenarios, such as predicting weather patterns or word sequences.

28

28

Markov Chain Representation

$Q = q_1 q_2 \ldots q_N$	a set of N **states**
$A = a_{11} a_{12} \ldots a_{N1} \ldots a_{NN}$	a **transition probability matrix** A, each a_{ij} representing the probability of moving from state i to state j, s.t. $\sum_{j=1}^{n} a_{ij} = 1 \quad \forall i$
$\pi = \pi_1, \pi_2, \ldots, \pi_N$	an **initial probability distribution** over states. π_i is the probability that the Markov chain will start in state i. Some states j may have $\pi_j = 0$, meaning that they cannot be initial states. Also, $\sum_{i=1}^{n} \pi_i = 1$

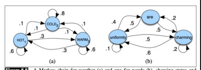


Figure 8.8 A Markov chain for weather (a) and one for words (b), showing states and transitions. A start distribution π is required; setting $\pi = [0.1, 0.7, 0.2]$ for (a) would mean a probability 0.7 of starting in state 2 (cold), probability 0.1 of starting in state 1 (hot), etc.

- Markov Assumption:
  - Formally stated as: $P(q_i=a|q_1 \ldots q_{i-1}) = P(q_i=a|q_{i-1})$ $P(q_i=a|q_1 \ldots q_{i-1})=P(q_i=a|q_{i-1})$ implying that when predicting the future, only the present state matters

29

29

The Hidden Markov Model

- A Markov chain computes probabilities for sequences of observable events.
- But often, the events of interest are hidden.
  - **Example:** Part-of-speech tags in text—hidden because we don't observe them directly.
- **Solution:** Hidden Markov Model (HMM) handles both observed and hidden events.
  - HMMs augment Markov chains

30

30

Probabilistic Sequence Modeling with HMMs

- A Hidden Markov Models (HMM) is a probabilistic sequence model that, given a sequence of units (words, letters, morphemes, sentences, etc.), computes a probability distribution over possible sequences of labels.
  - HMMs determine the likelihood of different label sequences and select the most probable sequence based on the observed data.
  - HMM is based on augmenting the **Markov chain**

31

Input and Assumptions

- Input (O): Sequence of observations (o_1, o_2, ..., o_T) drawn from vocabulary V.
- Assumptions of first-order HMM:
  - Markov Assumption:
    - Probability of state qi depends only on the previous state (q_{i-1}).
  - Output Independence:
    - Probability of observation o_i depends only on the state that produced it (q_i).

32

SIT330-770: Natural Language Processing

Week 8.5 – The components of an HMM tagger

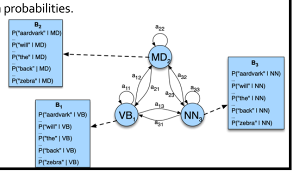
Dr. Mohamed Reda Bouadjenek

School of Information Technology, Faculty of Sci Eng & Built Env

33

Components of an HMM Tagger

- An HMM tagger consists of two main components:
  - Matrix A which represents the tag transition.
  - Matrix B which represents emission probabilities.



34

The A Matrix - Transition Probabilities

- The A matrix encapsulates the tag transition probabilities, $P(t_i|t_{i-1})$, which express how likely a tag follows its predecessor.
  - Example:
    - The modal verb "***will***" commonly precedes the base form of a verb (VB), as in "***will race***", leading to a high transition probability.
  - These probabilities are derived using maximum MLE by counting tag occurrences in a labeled corpus.
- Calculating Transition Probabilities:
  - In the WSJ corpus example, the modal verb tag (MD) is observed 13,124 times.
  - Out of these, MD transitions to a base verb (VB) 10,471 times.
  - Using MLE, we estimate P(VB|MD) = C(MD, VB) / C(MD) = 10,471 / 13,124 ≈ 0.80.

35

The B Matrix - Emission Probabilities

- The B matrix contains emission probabilities, $P(w_i|t_i)$, which quantify the likelihood of a word being tagged with a specific tag.
- Emission Probability Calculation
  - To calculate emission probabilities, we count how often a word occurs with a particular tag in a corpus.
  - For instance, the MD tag associated with the word 'will' occurs 4,046 times in the WSJ corpus.
  - Hence, P(will|MD) is calculated as C(MD, will) / C(MD) = 4,046 / 13,124 ≈ 0.31.

36

Components of HMM

$Q = q_1 q_2 \ldots q_N$	a set of N **states**
$A = a_{11} \ldots a_{ij} \ldots a_{NN}$	a **transition probability matrix** A, each a_{ij} representing the probability of moving from state i to state j, s.t. $\sum_{j=1}^{N} a_{ij} = 1 \quad \forall i$
$B = b_i(o_t)$	a sequence of **observation likelihoods**, also called **emission probabilities**, each expressing the probability of an observation o_t (drawn from a vocabulary $V = v_1, v_2, ..., v_V$) being generated from a state q_i
$\pi = \pi_1, \pi_2, ..., \pi_N$	an **initial probability distribution** over states. π_i is the probability that the Markov chain will start in state i. Some states j may have $\pi_j = 0$, meaning that they cannot be initial states. Also, $\sum_{i=1}^{n} \pi_i = 1$

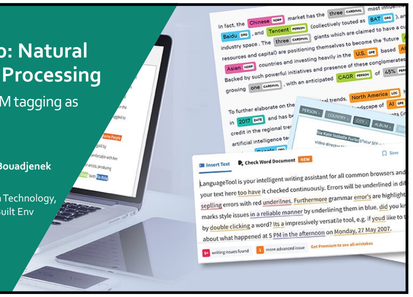
37

SIT330-770: Natural Language Processing

Week 8.6 – HMM tagging as decoding

Dr. Mohamed Reda Bouadjenek

School of Information Technology, Faculty of Sci Eng & Built Env



38

Decoding with Hidden Markov Models

- Decoding is the process of determining the most probable sequence of hidden states (tags) based on observed data.
  - Given a sequence of observations $O = o_1, o_2, \ldots, o_T$, decoding aims to find the most probable sequence of states $Q = q_1 q_2 \ldots q_T$.
  - The input is an HMM $\lambda = (A, B)$, with **A** being the transition probabilities and **B** the emission probabilities.

$$\hat{t}_{1:n} = \underset{t_1 \ldots t_n}{\operatorname{argmax}} P(t_1 \ldots t_n | w_1 \ldots w_n)$$

39

Decoding with Hidden Markov Models (i)

$$\hat{t}_{1:n} = \underset{t_1 \ldots t_n}{\operatorname{argmax}} P(t_1 \ldots t_n | w_1 \ldots w_n)$$

MAP is "maximum a posteriori" = most likely sequence

$$\hat{t}_{1:n} = \underset{t_1 \ldots t_n}{\operatorname{argmax}} \frac{P(w_1 \ldots w_n | t_1 \ldots t_n) P(t_1 \ldots t_n)}{P(w_1 \ldots w_n)}$$

Bayes Rule

$$\hat{t}_{1:n} = \underset{t_1 \ldots t_n}{\operatorname{argmax}} P(w_1 \ldots w_n | t_1 \ldots t_n) P(t_1 \ldots t_n)$$

Dropping the denominator

40

Decoding with Hidden Markov Models (ii)

$$\hat{t}_{1:n} = \underset{t_1 \ldots t_n}{\operatorname{argmax}} \overbrace{P(w_1 \ldots w_n | t_1 \ldots t_n)}^{\text{"Likelihood"}} \overbrace{P(t_1 \ldots t_n)}^{\text{"Prior"}}$$

- HMM taggers make two further simplifying assumptions.
  - The probability of a word appearing depends only on its own tag and is independent of neighboring words and tags:

$$P(w_1 \ldots w_n | t_1 \ldots t_n) \approx \prod_{i=1}^{n} P(w_i | t_i)$$

  - The second assumption, the bigram assumption, is that the probability of a tag is dependent only on the previous tag, rather than the entire tag sequence;

$$P(t_1 \ldots t_n) \approx \prod_{i=1}^{n} P(t_i | t_{i-1})$$

41

Decoding with Hidden Markov Models (iii)

- Plugging the simplifying assumptions results in the following equation for the most probable tag sequence from a bigram tagger:

$$\hat{t}_{1:n} = \underset{t_1 \ldots t_n}{\operatorname{argmax}} P(t_1 \ldots t_n | w_1 \ldots w_n) \approx \underset{t_1 \ldots t_n}{\operatorname{argmax}} \prod_{i=1}^{n} \overbrace{P(w_i | t_i)}^{\text{emission}} \overbrace{P(t_i | t_{i-1})}^{\text{transition}}$$

- The two parts correspond neatly to the **B** emission probability and **A** transition probability that we defined previously!

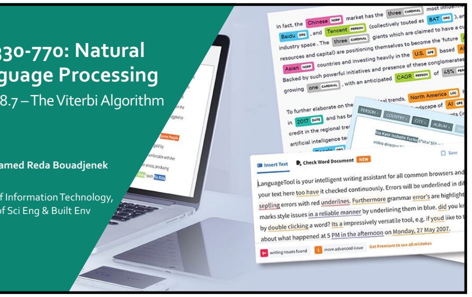
42

SIT330-770: Natural Language Processing

Week 8.7 – The Viterbi Algorithm

Dr. Mohamed Reda Bouadjenek

School of Information Technology, Faculty of Sci Eng & Built Env

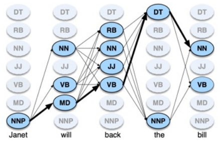


43

The Viterbi Algorithm (i)

- The decoding algorithm for HMMs is the **Viterbi algorithm**
  - As an instance of **dynamic programming**, Viterbi resembles the dynamic programming minimum edit distance algorithm
- The Viterbi algorithm first sets up a probability matrix or lattice:
  - **Columns as observables** (words of a sentence in the same sequence as in sentence)
  - Rows as hidden states (all possible POS Tags are known)

tag the sentence
Janet will back the bill



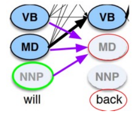
44

The Viterbi Algorithm (ii)

- Each cell of the matrix is represented by **$V_t(j)$** (Viterbi value for t: column, j: row) having the probability that the HMM is in **state j** (present POS Tag) after seeing the **first t observations** (past words for which matrix (cell) values has been calculated) and passing through the most **probable state sequence (previous POS Tag)** $q_1 \ldots q_{t-1}$
- Computed by recursively taking the most probable path that could lead us to this cell

$$v_t(j) = \max_{i=1}^{N} v_{t-1}(i)\, a_{ij}\, b_j(o_t)$$

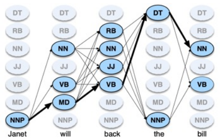
$v_{t-1}(i)$	the **previous Viterbi path probability** from the previous time step
a_{ij}	the **transition probability** from previous state q_i to current state q_j
$b_j(o_t)$	the **state observation likelihood** of the observation symbol o_t given the current state j



45

The Viterbi Algorithm (iii)

- Each cell of the matrix is represented by **$V_t(j)$** (Viterbi value for t: column, j: row) having the probability that the HMM is in **state j** (present POS Tag) after seeing the **first t observations** (past words for which matrix (cell) values has been calculated) and passing through the most **probable state sequence (previous POS Tag)** $q_1 \ldots q_{t-1}$



A sketch of the matrix for Janet will back the bill, showing the possible tags (q_i) for each word and highlighting the path corresponding to the correct tag sequence through the hidden states

States (parts of speech) which have a zero probability of generating a particular word according to the B matrix (such as the probability that a determiner DT will be realized as Janet) are greyed out

46

Working Example (i)

- Janet will back the bill ⟶ Janet/NNP will/MD back/VB the/DT bill/NN

The A transition probabilities $P(t_i|t_{i-1})$ computed from the WSJ corpus without smoothing

	NNP	MD	VB	JJ	NN	RB	DT
$<s>$	0.2767	0.0006	0.0031	0.0453	0.0449	0.0510	0.2026
NNP	0.3777	0.0110	0.0009	0.0084	0.0584	0.0090	0.0025
MD	0.0008	0.0002	0.7968	0.0005	0.0008	0.1698	0.0041
VB	0.0322	0.0005	0.0050	0.0837	0.0615	0.0514	0.2231
JJ	0.0366	0.0004	0.0001	0.0733	0.4509	0.0036	0.0036
NN	0.0096	0.0176	0.0014	0.0086	0.1216	0.0177	0.0068
RB	0.0068	0.0102	0.1011	0.1012	0.0120	0.0728	0.0479
DT	0.1147	0.0021	0.0002	0.2157	0.4744	0.0102	0.0017

Observation likelihoods B computed from the WSJ corpus without smoothing, simplified slightly

	Janet	will	back	the	bill
NNP	0.000032	0	0	0.000048	0
MD	0	0.308431	0	0	0
VB	0	0.000028	0.000672	0	0.000028
JJ	0	0	0.000340	0	0
NN	0	0.000200	0.000223	0	0.002337
RB	0	0	0.010446	0	0
DT	0	0	0	0.506099	0

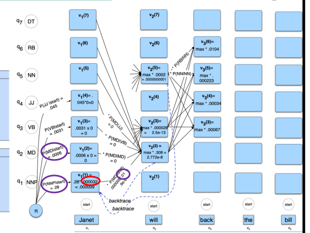
47

Working Example (ii)

$$v_t(j) = \max_{i=1}^{N} v_{t-1}(i)\, a_{ij}\, b_j(o_t)$$

	NNP	MD	VB	JJ	NN	RB	DT
$<s>$	0.2767	0.0006	0.0031	0.0453	0.0449	0.0510	0.2026
NNP	0.3777	0.0110	0.0009	0.0084	0.0584	0.0090	0.0025
MD	0.0008	0.0002	0.7968	0.0005	0.0008	0.1698	0.0041
VB	0.0322	0.0005	0.0050	0.0837	0.0615	0.0514	0.2231
JJ	0.0366	0.0004	0.0001	0.0733	0.4509	0.0036	0.0036
NN	0.0096	0.0176	0.0014	0.0086	0.1216	0.0177	0.0068
RB	0.0068	0.0102	0.1011	0.1012	0.0120	0.0728	0.0479
DT	0.1147	0.0021	0.0002	0.2157	0.4744	0.0102	0.0017

	Janet	will	back	the	bill
NNP	0.000032	0	0	0.000048	0
MD	0	0.308431	0	0	0
VB	0	0.000028	0.000672	0	0.000028
JJ	0	0	0.000340	0	0
NN	0	0.000200	0.000223	0	0.002337
RB	0	0	0.010446	0	0
DT	0	0	0	0.506099	0



48