Einführung in die Computerlinguistik
Hidden Markov Models (HMMs)

Ivan Habernal

Center for Information and Language Processing

2022-12-19

Slides credits: Hinrich Schütze, Alexander Fraser

Outline

1. StatNLP
2. Basics
3. POS tagging
4. POS setup
5. Probabilistic POS tagging
6. Viterbi

Statistical Natural Language Processing

Definition

Statistical Natural Language Processing (StatNLP) uses methods of supervised, semisupervised and unsupervised learning to address tasks that involve written or spoken (human) language.

What does "statistical" mean?

Adjective for "statistics"

statistics = the practice or science of collecting and analyzing numerical data

Statistical parameter estimation

an important / the most important subfield of machine learning

statistics vs. machine learning

Typical StatNLP applications

- automatic summarization of text
- sentiment analysis (e.g., find all *negative* reviews of the smartphone I want to buy)
- information extraction from text (e.g., find all inhibitors of a particular gene)
- machine translation

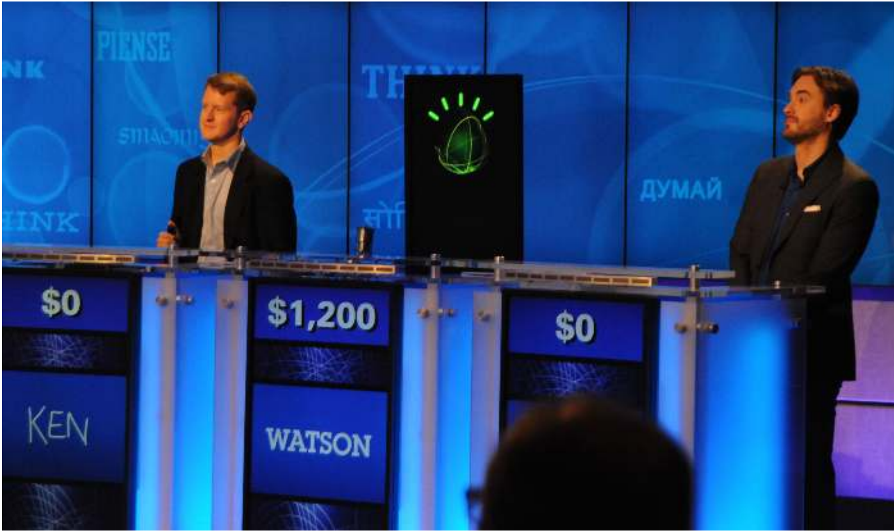
Applications that use some StatNLP

speech recognition
optical character recognition
information retrieval

History of StatNLP (simplified)

- 1940s, early 1950s: language as sequential process, Markov models
- 1950s, 1960s: Chomsky; statistical methods are viewed as inadequate for language.
- 1970s, 1980s: very little academic research on StatNLP, but IBM Watson group does seminal work
- 1990s: IBM Watson paradigm is adopted by computational linguists and becomes dominant approach to natural language processing.
- 2000s: The field splits methodologically into three communities.
    - traditional computational linguistics
    - a large group of researchers that use existing statistical methods
    - a small group of researchers that do active research on machine learning methods

Recent big success story 1



Recent big success story 2

Recent big success story 3

Google Translate – more on this later

max, argmax

max

$\max_x f(x)$
the largest value of $f(x)$

argmax

$\text{argmax}_x f(x)$
that value of x for which $f(x)$ is largest

- $\max_x(-(x-2)^2+5)$
- $\text{argmax}_x(-(x-2)^2+5)$

Positive factor $c > 0$ does not affect argmax

$$\text{argmax}_x f(x) = \text{argmax}_x c \cdot f(x)$$

$$\text{argmax}_x f(x) = \text{argmax}_x 1/c \cdot f(x)$$

$\sum$

$$\sum_{i=m}^{i=n} f(i) = f(m) + f(m+1) + \ldots + f(n-1) + f(n)$$

$\prod$

$$\prod_{i=m}^{i=n} f(i) = f(m) \cdot f(m+1) \cdot \ldots \cdot f(n-1) \cdot f(n)$$

$$\sum_{i=5}^{i=8} i^2 =$$

$$\prod_{i=0}^{i=3} (i+1) =$$

Probability

- What is the probability of rolling a 6 on a fair die? Obviously it is $1/6$.
- We can talk about this in terms of probability.
- Kolmogorov's first two of his three axioms of probability (simplified):
  1. The probability of an event A, which we define as $P(A)$ must be between 0 and 1 (inclusive), i.e., $0 <= P(A) <= 1$.
  2. The sum of the probabilities of outcomes must be 1. (E.g., for rolling a die, $P(1) + P(2) + ... + P(6) = 1$)
- From Axiom 2, it is obvious that $P(A) + P(\overline{A}) = 1$

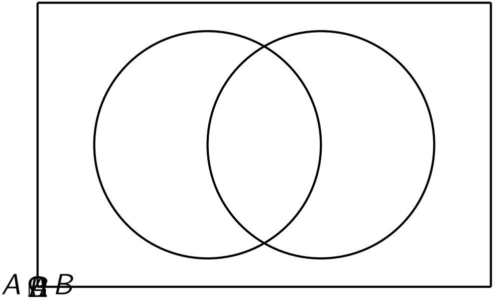
Joint probability

- The joint probability $P(AB)$ is the probability that A and B occur together / at the same time (i.e., jointly).
- We can write $P(AB)$ as $P(A \cap B)$ if A and B are formalized as sets.
- Kolmogrov Axiom 3:
  3. If A and B are mutually exclusive (same as $P(AB) = 0$) then the probability of A or B occurring is $P(A) + P(B)$

Conditional probability

- The conditional probability is the updated probability of an event given some knowledge .
- Definition: $P(A|B) = \frac{P(AB)}{P(B)}$ $(P(B) > 0)$

Venn diagram

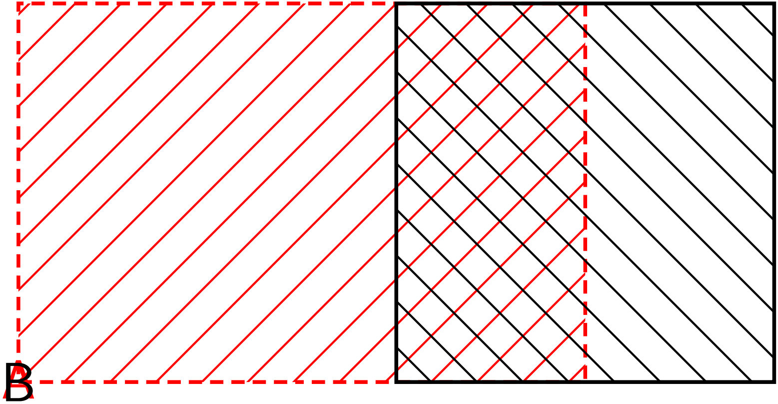


To compute $P(A|B)$: Divide the area of $A \cap B$ by the area of B.

$P(A|B) = P(A \cap B)/P(B)$

$P(B|A) = P(A \cap B)/P(A)$

Übung



Compute $P(A|B) = P(A \cap B)/P(B)$ and $P(B|A) = P(A \cap B)/P(A)$

Chain rule

$$P(X_1 X_2 X_3 \ldots X_n) =$$

$$P(X_1) \cdot P(X_2|X_1) \cdot P(X_3|X_1 X_2) \cdot \ldots \cdot P(X_n|X_1 X_2 \ldots X_{n-1})$$

Bayes' theorem

- $P(B|A) = \frac{P(BA)}{P(A)} = \frac{P(A|B)P(B)}{P(A)}$
- Or: $P(B|A) = \frac{P(A|B)P(B)}{P(A|B)P(B)+P(A|\overline{B})P(\overline{B})}$
- Follows from
  $P(A) = P(AB) + P(A\overline{B}) = P(A|B)P(B) + P(A|\overline{B})P(\overline{B})$

Independence

- Two events A and B are independent iff $P(AB) = P(A)P(B)$
- If I learn that A is true, then that doesn't change my assessment of the probability of B (and vice versa).
- If A and B are independent, then:
  $P(A) = P(A|B)$, $P(B) = P(B|A)$

Testing for independence

- Estimate $P(A)$, $P(B)$, $P(AB)$
- Simplest way of doing this: relative frequency: $P(A) = \frac{\text{count}(A)}{\text{count}(\textit{everything})}$
- Then: Compare $P(A)P(B)$ with $P(AB)$
- Recall: A, B independent iff $P(AB) = P(A)P(B)$
- $P(AB) \gg P(A)P(B)$: This indicates A and B are strongly dependent (and positively correlated).
- $P(AB) \approx P(A)P(B)$: This indicates A and B are independent.
- $P(AB) \ll P(A)P(B)$: This indicates A and B are strongly dependent (and negatively correlated).
- Why $\approx$?

Testing for independence: Example

A = champagne, B = sparkling

Übung

Find either two independent words or two words that occur less often on the same page than expected by chance

Part-of-speech tagging: Definition

- Part-of-speech tagging is the process of disambiguating the syntactic category of a word in context.
- Example: “book” is either a verb or a noun.
- In the context “the book” it can only be a noun.
- In the context “to book a flight” it can only be a verb.
- Part-of-speech tagging assigns to “book” the correct syntactic category in context.

Is part-of-speech tagging hard?

- The example of “book” in the phrase “the book” is easy.
- The rule “a word after ‘the’ cannot be a verb” takes care of it.
- Are all cases of part-of-speech tagging this easy?

Hard example

The following sentence is ambiguous wrt POS. Why?

The	representative	put	chairs	on	the	table
AT article	NN noun	VBD verb-d	NNS noun-s	IN prep	AT article	NN noun
AT article	JJ adjective	NN noun	VBZ verb-z	IN prep	AT article	NN noun

In this case, finding the correct parts of speech for the sentence is more difficult.

Why part-of-speech tagging?

- Part-of-speech tagging is used as a preprocessing step .
- It is solvable: Very high accuracy rates can be achieved (95–98% for English).
- It helps with many things you want to do with text, e.g., chunking, information extraction, question answering and parsing.

Part-of-speech tagging of tweets

ikr	smh	he	asked	fir	yo	last
!	G	O	V	P	D	A
name	so	he	can	add	u	on
N	P	O	V	V	O	P
fb	lololol					
^	!					

Tagging is a preprocessing step for many NLP tasks.

Example from: Owoputi et al. (2012). Part-of-Speech Tagging for Twitter: Word Clusters and Other Advances. Tech Report. See `http://www.cs.cmu.edu/~ark/TweetNLP/`

Setup

- We will first look at the Brown corpus tag set.
- Early work on part-of-speech tagging was done on the Brown corpus.
- It's still an important corpus in NLP.

Brown part-of-speech tags

Tag	Part Of Speech
AT	article
BEZ	the word "is"
IN	preposition
JJ	adjective
JJR	comparative adjective
MD	modal
NN	singular or mass noun
NNP	singular proper noun
NNS	plural noun
PERIOD	. : ? !
PN	personal pronoun

Tag	Part Of Speech
RB	adverb
RBR	comparative adverb
TO	the word "to"
VB	verb, base form
VBD	verb, past tense
VBG	verb, present participle, gerund
VBN	verb, past participle
VBZ	verb, 3rd singular present
WDT	wh-determiner: "what", "which", ...

Are these typical syntactic categories?

Tag: "Peter arrived in London on Tuesday"

What information can we use for tagging?

- Let's look again at our example sentence:
  "The representative put chairs on the table."
- What information is available to disambiguate this sentence syntactically?

Hard example

The following sentence is ambiguous wrt POS. Why?

The	representative	put	chairs	on	the	table
AT article	NN noun	VBD verb-d	NNS noun-s	IN prep	AT article	NN noun
AT article	JJ adjective	NN noun	VBZ verb-z	IN prep	AT article	NN noun

In this case, finding the correct parts of speech for the sentence is more difficult.

Two main sources of information

1. The context of the ambiguous word: the words to the left and to the right
   - Example: for a JJ/NN ambiguity in the context "AT _ VBZ", NN is much more likely than JJ.
2. A word's bias for the different parts of speech
   - Example: "put" is much more likely to occur as a VBD than as an NN.

Information sources

- Information source 2: The frequency of the different parts of speech of the ambiguous word
- This source of information lets us do 90% correct tagging of English very easily: Just pick the most frequent tag for each word.
- For most words in English, the distribution of tags is very uneven : there is one very frequent tag and the others are rare.

Notation

w_i	the word at position i in the corpus
t_i	the tag of w_i
w^l	the l^{th} word in the lexicon
t^j	the j^{th} tag in the tag set
$C(w^l)$	the number of occurrences of w^l in the training set
$C(t^j)$	the number of occurrences of t^j in the training set
$C(t^j t^k)$	the number of occurrences of t^j followed by t^k
$C(w^l : t^j)$	the number of occurrences of w^l that are tagged as t^j

Notation: Example

the	representative	put	chairs	on	the	table
w_1	w_2	w_3	w_4	w_5	w_6	w_7
w^5	w^{81}	w^3	w^4	w^1	w^5	w^6
AT	NN	VBD	NNS	IN	AT	NN
article	noun	verb-d	noun-s	prep	article	noun
t_1	t_2	t_3	t_4	t_5	t_6	t_7
t^{16}	t^{12}	t^2	t^9	t^3	t^{16}	t^{12}

$$
\begin{array}{lcl|lcl}
C(w^5) & = & 2 & C(w^4) & = & 1 \\
C(t^{16}) & = & 2 & C(t^2) & = & 1 \\
C(t^{16}t^{12}) & = & 2 & C(t^{12}t^2) & = & 1 \\
C(t^{16}t^2) & = & 0 & C(w^5w^{81}) & = & 1 \\
C(w^5 : t^{16}) & = & 2 & C(w^5 : t^{12}) & = & 0
\end{array}
$$

Notation: Übung

Confidence/NN in/IN the/AT pound/NN is/BEZ widely/RB expected/VBN to/TO take/VB another/AT sharp/JJ dive/NN if/IN trade/NN figures/NNS for/IN September/NNP ,/, due/JJ for/IN release/NN tomorrow/NN ,/, fail/VB to/TO show/VB a/AT substantial/JJ improvement/NN from/IN July/NNP and/CC August/NNP 's/POS near-record/JJ deficits/NNS ./. Chancellor/NNP of/IN the/AT Exchequer/NNP Nigel/NNP Lawson/NNP 's/POS restated/VBN commitment/NN to/TO a/AT firm/JJ monetary/JJ policy/NN has/VBZ helped/VBN to/TO prevent/VB a/AT freefall/NN in/IN sterling/NN over/IN the/AT past/JJ week/NN ./.

Give the values of the following: w_4, t_5, $C(w_8)$, $C(t_9)$, $C(t_1 t_2)$, $C(w_3 : t_3)$

Supervised learning

- Labeled training set: each word is annotated (or marked or tagged) by a linguist, with correct part-of-speech
- Train a statistical model on the training set
  - Result: A set of parameters (= numbers) that were learned from the specific properties of the training set
- Apply statistical model to new text that we want to analyze for some task (information retrieval, machine translation etc)

Tagged training corpus/set: Example

Confidence/NN in/IN the/AT pound/NN is/BEZ widely/RB expected/VBN to/TO take/VB another/AT sharp/JJ dive/NN if/IN trade/NN figures/NNS for/IN September/NNP ,/, due/JJ for/IN release/NN tomorrow/NN ,/, fail/VB to/TO show/VB a/AT substantial/JJ improvement/NN from/IN July/NNP and/CC August/NNP 's/POS near-record/JJ deficits/NNS ./. Chancellor/NNP of/IN the/AT Exchequer/NNP Nigel/NNP Lawson/NNP 's/POS restated/VBN commitment/NN to/TO a/AT firm/JJ monetary/JJ policy/NN has/VBZ helped/VBN to/TO prevent/VB a/AT freefall/NN in/IN sterling/NN over/IN the/AT past/JJ week/NN ./.

Contents of this section

- Parameter estimation: context parameters
- Parameter estimation: bias parameters
- Greedy tagging
- Viterbi tagging

Parameter estimation: Context

- The conditional probabilities $P(t^k|t^j)$ are the context parameters of the model.
- This will be our formalization of the first source of information in tagging: the context.
- Note that this is a very impoverished model of context.
  - Limited horizon, Markov assumption: we assume that our memory is limited to a single preceding tag .
  - Time invariance, stationary: we assume that these conditional probabilities don't change. (e.g., the same at the beginning and at the end of the sentence)

Parameter estimation: Context

- How can we estimate $P(t^k|t^j)$?
- For example: how can we estimate $P(\text{NN}|\text{JJ})$?
- First: maximum likelihood estimate
- Training text: long tagged sequence of words

Tagged training corpus/set: Example

Confidence/NN in/IN the/AT pound/NN is/BEZ widely/RB expected/VBN to/TO take/VB another/AT sharp/JJ dive/NN if/IN trade/NN figures/NNS for/IN September/NNP ,/, due/JJ for/IN release/NN tomorrow/NN ,/, fail/VB to/TO show/VB a/AT substantial/JJ improvement/NN from/IN July/NNP and/CC August/NNP 's/POS near-record/JJ deficits/NNS ./. Chancellor/NNP of/IN the/AT Exchequer/NNP Nigel/NNP Lawson/NNP 's/POS restated/VBN commitment/NN to/TO a/AT firm/JJ monetary/JJ policy/NN has/VBZ helped/VBN to/TO prevent/VB a/AT freefall/NN in/IN sterling/NN over/IN the/AT past/JJ week/NN ./.

Parameter estimation: Context

- How can we estimate $P(t^k|t^j)$?
- For example: how can we estimate $P(\text{NN}|\text{JJ})$?
- ml = maximum likelihood = relative frequency
- $$\hat{P}_{ml}(t^k|t^j) = \frac{\hat{P}_{ml}(t^j t^k)}{\hat{P}_{ml}(t^j)} \approx \frac{\frac{C(t^j t^k)}{C(.)}}{\frac{C(t^j)}{C(.)}} = \frac{C(t^j t^k)}{C(t^j)}$$
- $$\hat{P}_{ml}(\text{NN}|\text{JJ}) = \frac{C(\text{JJ NN})}{C(\text{JJ})}$$

n^{th} order Markov model

In an n^{th} order Markov model,

the tag at time t depends on the n previous tags.

- Order 0: Tag does not depend on previous tags.
- Order 1: Tag depends on immediately preceding tag.
- Order 2: Tag depends on two immediately preceding tags.
- Order 3: Tag depends on three immediately preceding tags.
- ...

(analogous for Markov model that emits words instead of tags)

Parameter estimation: Word bias

- What about the second source of information: frequency of different tags for a word?
- We need to estimate: $P(t_i|w_i)$
- Actually: $P(w_i|t_i)$
- Example: $P(\text{book}|\text{NN})$

Parameter estimation: Word bias

- How to estimate $P(\text{book}|\text{NN})$
-
$$\hat{P}_{ml}(w^l|t^j) = \frac{\hat{P}_{ml}(w^l : t^j)}{\hat{P}_{ml}(t^j)} = \frac{\frac{C(w^l:t^j)}{C(.)}}{\frac{C(t^j)}{C(.)}} = \frac{C(w^l : t^j)}{C(t^j)}$$
-
$$\hat{P}_{ml}(\text{book}|\text{NN}) = \frac{C(\text{book} : \text{NN})}{C(\text{NN})}$$

Tagged training corpus/set: Example

Confidence/NN in/IN the/AT pound/NN is/BEZ widely/RB expected/VBN to/TO take/VB another/AT sharp/JJ dive/NN if/IN trade/NN figures/NNS for/IN September/NNP ,/, due/JJ for/IN release/NN tomorrow/NN ,/, fail/VB to/TO show/VB a/AT substantial/JJ improvement/NN from/IN July/NNP and/CC August/NNP 's/POS near-record/JJ deficits/NNS ./. Chancellor/NNP of/IN the/AT Exchequer/NNP Nigel/NNP Lawson/NNP 's/POS restated/VBN commitment/NN to/TO a/AT firm/JJ monetary/JJ policy/NN has/VBZ helped/VBN to/TO prevent/VB a/AT freefall/NN in/IN sterling/NN over/IN the/AT past/JJ week/NN ./.

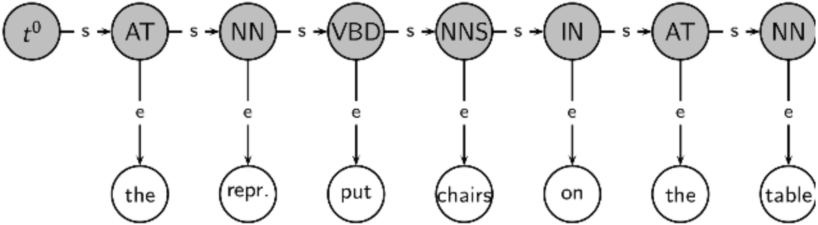
Estimate $P(\text{take}|\text{VB})$ and $P(\text{AT}|\text{IN})$

Parameter estimation: Word bias

- What about the second source of information: frequency of different tags for a word?
- We need to estimate: $P(t_i|w_i)$
- Actually: $P(w_i|t_i)$
- Example: $P(\text{book}|\text{NN})$

$P(w|t)$ versus $P(t|w)$

(s = sequence, e = emission)



- This is a so-called "generative model".
- We assume that the tag sequence generates the words (not vice versa).
- Hence: The tags are given and the words are conditioned on the tags ...
- ...and the correct formalization is $P(w|t)$.

How do we actually do the tagging?

- Context: $P(t_{i+1}|t_i)$
- Word bias: $P(w_i|t_i)$
- Given a sequence of words (a sentence), how do we compute the corresponding (disambiguated) part-of-speech sequence?
- Example:
  - Input: the representative put chairs on the table
  - Output: AT NN VBD NNS IN AT NN
- At decoding time, our task is to recover the tags (= states). This model is called a "Hidden Markov Model" because we don't know the states.
- How can we do this?

"Greedy" tagging

- Suppose we've tagged a sentence up to position i.
- Then simply choose the tag t for the next word w_{i+1} that is most probable .
- At position i, choose tag that maximizes: $P(t_i|t_{i-1})P(w_i|t_i)$
- Let's do this for: "The representative put chairs on the table."
- $P(\text{VBD}|\text{NN})P(\text{put}|\text{VBD})$
- $t_3 = \text{VBD}$ maximizes $P(t_3|\text{NN})P(\text{put}|t_3)$

Problems with greedy tagging

- What can go wrong with greedy tagging?
- Example?
- The representative put costs 20% more today than a month ago.

Notation (2)

w_i	the word at position i in the corpus
t_i	the tag of w_i
$w_{i,i+m}$	the words occurring at positions i through $i+m$ (alternative notations: $w_i \cdots w_{i+m}$, $w_i, \ldots, w_{i+m}$, $w_{i(i+m)}$)
$t_{i,i+m}$	the tags $t_i \cdots t_{i+m}$ for $w_i \cdots w_{i+m}$
w^l	the l^{th} word in the lexicon
t^j	the j^{th} tag in the tag set
$C(w^l)$	the number of occurrences of w^l in the training set
$C(t^j)$	the number of occurrences of t^j in the training set
$C(t^j t^k)$	the number of occurrences of t^j followed by t^k
$C(w^l : t^j)$	the number of occurrences of w^l that are tagged as t^j
T	number of tags in tag set
W	number of words in the lexicon
n	sentence length

Part-of-speech tagging: Problem statement

- We define our goal thus: Given a sentence, find the most probable sequence of tags for this sentence.
- Formalization of this goal:

$$t_{1,n} = \text{argmax}_{t_{1,n}} P(t_{1,n}|w_{1,n})$$

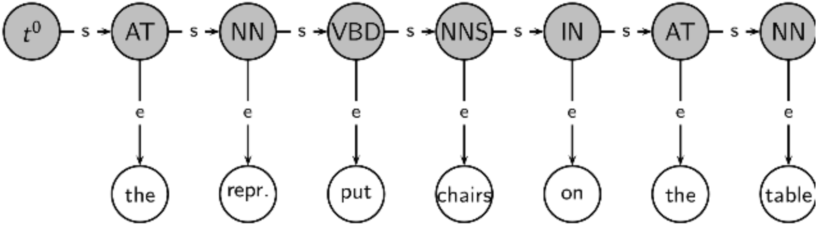
Simplifying the argmax (1)

$$
\begin{aligned}
t_{1,n} &= \text{argmax}_{t_{1,n}} P(t_{1,n}|w_{1,n}) && (1)\\
&= \text{argmax}_{t_{1,n}} P(t_{0,n}|w_{1,n}) && (2)\\
&= \text{argmax}_{t_{1,n}} \frac{P(w_{1,n}|t_{0,n})P(t_{0,n})}{P(w_{1,n})} && (3)\\
&= \text{argmax}_{t_{1,n}} P(w_{1,n}|t_{0,n})P(t_{0,n}) && (4)\\
&= \text{argmax}_{t_{1,n}} [\prod_{i=1}^{n} P(w_i|t_{0,n})]P(t_{0,n}) && (5)
\end{aligned}
$$

2: dummy “start” tag; 3: Bayes; 4: positive factor doesn't affect argmax; 5: assumption: words are independent

$P(w|t)$ versus $P(t|w)$

(s = sequence, e = emission)



- This is a so-called "generative model".
- We assume that the tag sequence generates the words (not vice versa).
- Hence: The tags are given and the words are conditioned on the tags ...
- ...and the correct formalization is $P(w|t)$.

Simplifying the argmax (2)

$$
\begin{aligned}
&= \operatorname{argmax}_{t_{1,n}} [\prod_{i=1}^{n} P(w_i|t_i)] P(t_{0,n}) && (6)\\
&= \operatorname{argmax}_{t_{1,n}} [\prod_{i=1}^{n} P(w_i|t_i)][\prod_{i=1}^{n} P(t_i|t_{0,i-1})] && (7)\\
&= \operatorname{argmax}_{t_{1,n}} [\prod_{i=1}^{n} P(w_i|t_i)][\prod_{i=1}^{n} P(t_i|t_{i-1})] && (8)\\
&= \operatorname{argmax}_{t_{1,n}} \prod_{i=1}^{n} [P(w_i|t_i) P(t_i|t_{i-1})] && (9)
\end{aligned}
$$

7: chain rule; 8: Markov assumption; 9:
$\prod_{i=1}^{n} x_i \prod_{i=1}^{n} y_i = \prod_{i=1}^{n} x_i y_i$

Simplifying the argmax (3)

$$= \operatorname{argmax}_{t_{1,n}} \prod_{i=1}^{n} [P(w_i|t_i)P(t_i|t_{i-1})] \quad (10)$$

$$= \operatorname{argmax}_{t_{1,n}} \sum_{i=1}^{n} [\log P(w_i|t_i) + \log P(t_i|t_{i-1})] \quad (11)$$

11: computation in log space more efficient / convenient

The most probable tag sequence (= tagging)

$$\text{argmax}_{t_{1,n}} \sum_{i=1}^{n} [\log P(w_i|t_i) + \log P(t_i|t_{i-1})]$$

What's the difficulty if you want to tag based on this?

Brute force is very inefficient

$$\text{argmax}_{t_{1,n}} \sum_{i=1}^{n} [\log P(w_i|t_i) + \log P(t_i|t_{i-1})]$$

There are $|T|^n$ different tag sequences. E.g.:
$40^{20} = 109{,}951{,}162{,}777{,}600{,}000{,}000{,}000{,}000{,}000{,}000$
Is there a better way?

Dynamic programming: Viterbi

- Optimal substructure: The optimal solution to the problem contains within it subsolutions , i.e., optimal solutions to subproblems.
- Overlapping subsolutions : The subsolutions overlap. These subsolutions are computed over and over again when computing the global optimal solution in a brute-force algorithm.
- Subproblem in the case of tagging: what is the best path (tag sequence) that gets me to tag t at position j?
- Overlapping subsolutions: The best path that gets me to tag t at position j is needed for computing all T paths at position $j+1$ ...
- ...but I only compute it once!

$P(t_i|t_{i-1})$
Example: $P(\text{VB}|\text{MD}) = 0.7968$

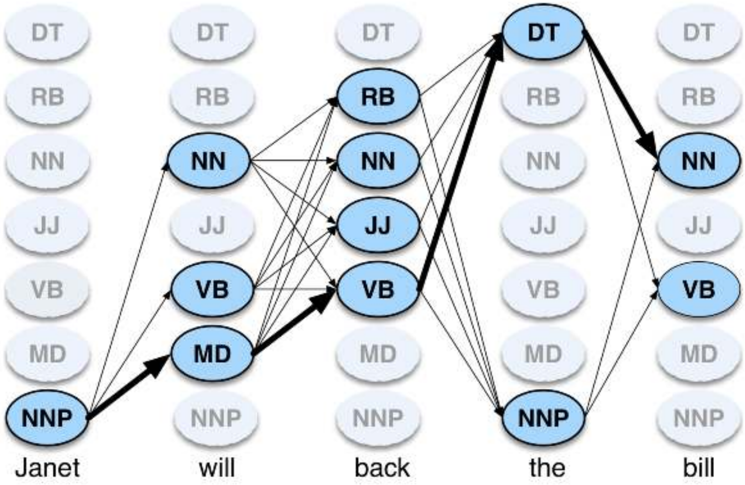
	NNP	**MD**	**VB**	**JJ**	**NN**	**RB**	**DT**
$<s>$	0.2767	0.0006	0.0031	0.0453	0.0449	0.0510	0.2026
NNP	0.3777	0.0110	0.0009	0.0084	0.0584	0.0090	0.0025
MD	0.0008	0.0002	0.7968	0.0005	0.0008	0.1698	0.0041
VB	0.0322	0.0005	0.0050	0.0837	0.0615	0.0514	0.2231
JJ	0.0366	0.0004	0.0001	0.0733	0.4509	0.0036	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

vertical axis: t_{i-1}
horizontal axis: t_i

$P(w|t)$
Example: $P(\textit{the}|\text{DT}) = 0.506099$

	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

Key idea of Viterbi: Lattice



Viterbi

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

create a path probability matrix *viterbi[N,T]*
for each state s **from** 1 **to** N **do** ; initialization step
$viterbi[s,1] \leftarrow \pi_s * b_s(o_1)$
$backpointer[s,1] \leftarrow 0$
for each time step t **from** 2 **to** T **do** ; recursion step
for each state s **from** 1 **to** N **do**
$viterbi[s,t] \leftarrow \max_{s'=1}^{N} viterbi[s',t-1] * a_{s',s} * b_s(o_t)$
$backpointer[s,t] \leftarrow \operatorname{argmax}_{s'=1}^{N} viterbi[s',t-1] * a_{s',s} * b_s(o_t)$
$bestpathprob \leftarrow \max_{s=1}^{N} viterbi[s,T]$; termination step
$bestpathpointer \leftarrow \operatorname{argmax}_{s=1}^{N} viterbi[s,T]$; termination step
bestpath ← the path starting at state *bestpathpointer*, that follows backpointer[] to states back in time
return *bestpath*, *bestpathprob*

$P(t_i|t_{i-1})$
Example: $P(\mathrm{VB}|\mathrm{NN}) = 0.5$

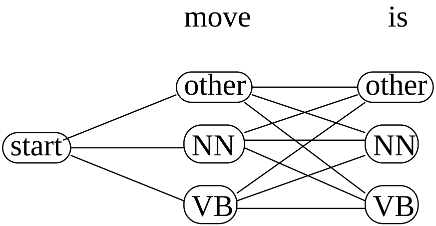
next / prev	other	NN	VB
start	0.3	0.4	0.3
other	0.2	0.2	0.6
NN	0.4	0.1	0.5
VB	0.1	0.8	0.1

vertical axis: t_{i-1}
horizontal axis: t_i

$P(w|t)$
Example: $P(bear|\text{NN}) = 0.45$

	other	NN	VB
bear	0.1	0.45	0.4
is	0.3	0.05	0.05
on	0.3	0.05	0.05
the	0.2	0.05	0.05
move	0.1	0.4	0.45

Lattice



Goal: Compute

$$\arg\max_{t_1,t_2} p(t_1, move, t_2, is) =$$

$$\arg\max_{t_1,t_2} p(t_1|start)p(move|t_1)p(t_2|t_1)p(is|t_2)$$

viterbi = vtrb
backpointer = bptr
lattice = path probability matrix

vtrb and bptr

$\text{vtrb}_j(t_i)$ is the probability of [the most probable path from 0 to j that tags word w_j with tag t_i].

$\text{bptr}_j(t_i)$ is the tag of w_{j-1} on [the most probable path from 0 to j that tags word w_j with tag t_i].

Initialization: $\text{vtrb}_0(\textit{start}) = 1$

Viterbi probabilities for the tags of the first word

$\text{vtrb}_1(\text{other}) = \text{vtrb}_0(\text{start})\ p(\text{other}|\text{start})\ p(\text{move}|\text{other}) = 1.0 * 0.3 * 0.1 = 0.03$
$\text{vtrb}_1(\text{NN}) = \text{vtrb}_0(\text{start})\ p(\text{NN}|\text{start})\ p(\text{move}|\text{NN}) = 1.0 * 0.4 * 0.4 = 0.16$
$\text{vtrb}_1(\text{VB}) = \text{vtrb}_0(\text{start})\ p(\text{VB}|\text{start})\ p(\text{move}|\text{VB}) = 1.0 * 0.3 * 0.45 = 0.135$

Viterbi probabilities for the tags of the second word (1)

$\text{vtrb}_2(\text{other}) = \max($
  $\text{vtrb}_1(\text{other})\ \text{p}(\text{other}|\text{other})\ \text{p}(\text{is}|\text{other}) = 0.03 * 0.2 * 0.3 = 0.0018,$
  $\text{vtrb}_1(\text{NN})\ \text{p}(\text{other}|\text{NN})\ \text{p}(\text{is}|\text{other}) = 0.16 * 0.4 * 0.3 = 0.0192,$
  $\text{vtrb}_1(\text{VB})\ \text{p}(\text{other}|\text{VB})\ \text{p}(\text{is}|\text{other}) = 0.135 * 0.1 * 0.3 = 0.00405$
$) = 0.0192$
  $\text{bptr}_2(\text{other}) = \text{NN}$

Viterbi probabilities for the tags of the second word (2)

$\text{vtrb}_2(\text{NN}) = \max($
$\quad \text{vtrb}_1(\text{other})\ p(\text{NN}|\text{other})\ p(\text{is}|\text{NN}) = 0.03 * 0.2 * 0.05 = 0.0003,$
$\quad \text{vtrb}_1(\text{NN})\ p(\text{NN}|\text{NN})\ p(\text{is}|\text{NN}) = 0.16 * 0.1 * 0.05 = 0.0008,$
$\quad \text{vtrb}_1(\text{VB})\ p(\text{NN}|\text{VB})\ p(\text{is}|\text{NN}) = 0.135 * 0.8 * 0.05 = 0.0054$
$) = 0.0054$
$\quad \text{bptr}_2(\text{NN}) = \text{VB}$

Viterbi probabilities for the tags of the second word (3)

$\text{vtrb}_2(\text{VB}) = \max($
$\quad \text{vtrb}_1(\text{other})\ p(\text{VB}|\text{other})\ p(\text{is}|\text{VB}) = 0.03 * 0.6 * 0.05 = 0.0009,$
$\quad \text{vtrb}_1(\text{NN})\ p(\text{VB}|\text{NN})\ p(\text{is}|\text{VB}) = 0.16 * 0.5 * 0.05 = 0.004,$
$\quad \text{vtrb}_1(\text{VB})\ p(\text{VB}|\text{VB})\ p(\text{is}|\text{VB}) = 0.135 * 0.1 * 0.05 = 0.000675$
$) = 0.004$
$\quad \text{bptr}_2(\text{VB}) = \text{NN}$

Read out path

Probability of the most likely path: $0.0192 = \max_t \text{vtrb}_2(t)$
Last tag of the most likely path: other $= \arg\max_t \text{vtrb}_2(t)$
First tag of the most likely path: NN $= \text{bptr}_2(\text{other})$
Result:
NN other $= \arg\max_{t_1 t_2} p(t_1, move, t_2, is)$

Besonders klausurrelevant

- Part-of-speech tagging, informal definition
- Part-of-speech tagging, formal definition

$$\text{argmax}_{t_{1,n}} \sum_{i=1}^{n} [\log P(w_i|t_i) + \log P(t_i|t_{i-1})]$$

- Brown part-of-speech tags
- Parameter estimation: Context

$$\hat{P}(t^k|t^j) = \frac{C(t^j t^k)}{C(t^j)}$$

- Parameter estimation: Word bias

$$\hat{P}(w^l|t^j) = \frac{C(w^l : t^j)}{C(t^j)}$$

- Order of a Markov model
- Viterbi