PART-OF-SPEECH TAGGING

Ing. R. Tedesco. PhD, AA 20-21

(mostly from: Speech and Language Processing - Jurafsky and Martin)

Today

- Parts of speech (POS)
- Tagsets
- POS Tagging
  - HMM Tagging
    - Hidden Markov Models
    - Viterbi algorithm
- Tools

Parts of Speech

- 8 (ish) traditional parts of speech
  - Noun, verb, adjective, preposition, adverb, article, interjection, pronoun, conjunction, etc
- Called: parts-of-speech, lexical categories, word classes, morphological classes, lexical tags...
- Lots of debate within linguistics about the number, nature, and universality of these
  - We'll completely ignore this debate.

POS examples

N	noun	*chair, bandwidth, pacing*
V	verb	*study, debate, munch*
ADJ	adjective	*purple, tall, ridiculous*
ADV	adverb	*unfortunately, slowly*
P	preposition	*of, by, to*
PRO	pronoun	*I, me, mine*
DET	determiner	*the, a, that, those*

POS Tagging

- The process of assigning a part-of-speech or lexical class marker to each word in a collection.

WORD	tag
the	**DET**
koala	**N**
put	**V**
the	**DET**
keys	**N**
on	**P**
the	**DET**
table	**N**

Why is POS Tagging Useful?

- First step of a vast number of practical tasks
- Speech synthesis
  - How to pronounce…
  - INsult　　　　inSULT
  - OBject　　　　objECT
  - OVERflow　　　　overFLOW
  - DIScount　　　　disCOUNT
  - CONtent　　　　conTENT
- Parsing
  - Need to know if a word is an N or V before you can parse
- Information extraction
  - Finding names, relations, etc.
- Machine Translation

Open and Closed Classes

- **Closed class**: a small fixed membership
  - Prepositions: of, in, by, …
  - Auxiliaries: may, can, will had, been, …
  - Pronouns: I, you, she, mine, his, them, …
  - In general, function words (short common words which play a role in grammar)
- **Open class**: new ones can be created all the time
  - English has 4: Nouns, Verbs, Adjectives, Adverbs
  - Many languages have these 4, but not all!

Open Class Words

- Nouns
  - Proper nouns (Boulder, Granby, Eli Manning)
    - English capitalizes these.
  - Common nouns (the rest).
  - Count nouns and mass nouns
    - Count: have plurals, get counted: goat/goats, one goat, two goats
    - Mass: don't get counted (snow, salt, communism) (*two snows)
- Adverbs: tend to modify things
  - Unfortunately, John walked home extremely slowly yesterday
  - Directional/locative adverbs (here, home, downhill)
  - Degree adverbs (extremely, very, somewhat)
  - Manner adverbs (slowly, slinkily, delicately)
- Verbs
  - In English, have morphological affixes (eat/eats/eaten)

Closed Class Words

Examples:

- prepositions: *on, under, over, …*
- particles: *up, down, on, off, …*
- determiners: *a, an, the, …*
- pronouns: *she, who, I, ..*
- conjunctions: *and, but, or, …*
- auxiliary verbs: *can, may should, …*
- numerals: *one, two, three, third, …*

Prepositions from CELEX

of	540,085	through	14,964	worth	1,563	pace	12
in	331,235	after	13,670	toward	1,390	nigh	9
for	142,421	between	13,275	plus	750	re	4
to	125,691	under	9,525	till	686	mid	3
with	124,965	per	6,515	amongst	525	o'er	2
on	109,129	among	5,090	via	351	but	0
at	100,169	within	5,030	amid	222	ere	0
by	77,794	towards	4,700	underneath	164	less	0
from	74,843	above	3,056	versus	113	midst	0
about	38,428	near	2,026	amidst	67	o'	0
than	20,210	off	1,695	sans	20	thru	0
over	18,071	past	1,575	circa	14	vice	0

English Particles

aboard	aside	besides	forward(s)	opposite	through
about	astray	between	home	out	throughout
above	away	beyond	in	outside	together
across	back	by	inside	over	under
ahead	before	close	instead	overhead	underneath
alongside	behind	down	near	past	up
apart	below	east, etc.	off	round	within
around	beneath	eastward(s),etc.	on	since	without

Conjunctions

and	514,946	yet	5,040	considering	174	forasmuch as	0
that	134,773	since	4,843	lest	131	however	0
but	96,889	where	3,952	albeit	104	immediately	0
or	76,563	nor	3,078	providing	96	in as far as	0
as	54,608	once	2,826	whereupon	85	in so far as	0
if	53,917	unless	2,205	seeing	63	inasmuch as	0
when	37,975	why	1,333	directly	26	insomuch as	0
because	23,626	now	1,290	ere	12	insomuch that	0
so	12,933	neither	1,120	notwithstanding	3	like	0
before	10,720	whenever	913	according as	0	neither nor	0
though	10,329	whereas	867	as if	0	now that	0
than	9,511	except	864	as long as	0	only	0
while	8,144	till	686	as though	0	provided that	0
after	7,042	provided	594	both and	0	providing that	0
whether	5,978	whilst	351	but that	0	seeing as	0
for	5,935	suppose	281	but then	0	seeing as how	0
although	5,424	cos	188	but then again	0	seeing that	0
until	5,072	supposing	185	either or	0	without	0

POS Tagging
Choosing a Tagset

- To do POS tagging, we need to choose a standard set of tags to work with
- Could pick very coarse tagsets
  - N, V, Adj, Adv.
- More commonly used set is finer grained, the "Penn TreeBank tagset", 45 tags
  - PRP$, WRB, WP$, VBG
- Even more fine-grained tagsets exist

Penn TreeBank POS Tagset

Tag	Description	Example	Tag	Description	Example
CC	coordin. conjunction	*and, but, or*	SYM	symbol	+,%, &
CD	cardinal number	*one, two, three*	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, singular	*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*			

Using the Penn Tagset

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

- Prepositions and subordinating conjunctions marked IN (“although/IN I/PRP..”)
- Except the preposition/complementizer, “to” is just marked “TO”.

POS Tagging: ambiguity

- 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.

These examples from Dekang Lin

How Hard is POS Tagging? Measuring Ambiguity

		87-tag Original Brown		45-tag Treebank Brown	
Unambiguous (1 tag)		**44,019**		**38,857**	
Ambiguous (2–7 tags)		**5,490**		**8844**	
Details:	2 tags	4,967		6,731	
	3 tags	411		1621	
	4 tags	91		357	
	5 tags	17		90	
	6 tags	2	(*well*, *beat*)	32	
	7 tags	2	(*still*, *down*)	6	(*well*, *set*, *round*, *open*, *fit*, *down*)
	8 tags			4	(*'s*, *half*, *back*, *a*)
	9 tags			3	(*that*, *more*, *in*)

Two Methods for POS Tagging

1. Rule-based tagging
   - We'll ignore this approach (too old…)
2. Stochastic
   - Probabilistic sequence models
     - HMM (Hidden Markov Model) tagging
     - MEMMs (Maximum Entropy Markov Models)
   - We'll present HMM tagging

Hidden Markov Model Tagging

- Using an HMM to do POS tagging is a special case of *Bayesian inference*
  - Foundational work in computational linguistics
  - Bledsoe 1959: OCR
  - Mosteller and Wallace 1964: authorship identification
- It is also related to the "noisy channel" model that's the basis for ASR, OCR and MT

POS Tagging as Sequence Classification

- We are given a sentence (an "observation" or "sequence of observations")
  - *"Secretariat is expected to race tomorrow"*
- What is the best sequence of tags that 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 $w_1 \ldots w_n$

Getting to HMMs

- We want, out of all sequences of n tags $t_1 \ldots t_n$ the single tag sequence such that $P(t_1 \ldots t_n | w_1 \ldots w_n)$ is highest

$$\hat{t}_1^n = \underset{t_1^n}{\operatorname{argmax}} P(t_1^n | w_1^n)$$

- $t_i \in T,\ w_i \in W \quad \forall\, 1 \le i \le n$
- T={NN, JJ, …, start, stop} : *hidden* states → the POS tag set
- W={the, example, …} : *observed* values → the vocabulary
- Input: a sequence of n observed values

Getting to HMMs

- This equation is guaranteed to give us the best tag sequence

$$\hat{t}_1^n = \underset{t_1^n}{\operatorname{argmax}} P(t_1^n | w_1^n)$$

- But how to make it operational? How to compute this value?
- Intuition of Bayesian classification:
  - Use Bayes rule to transform this equation into a set of other probabilities that are easier to compute

Using Bayes Rule

$$P(x|y) = \frac{P(y|x)P(x)}{P(y)}$$

$$\hat{t}_1^n = \underset{t_1^n}{\operatorname{argmax}} \frac{P(w_1^n|t_1^n)P(t_1^n)}{P(w_1^n)}$$

$$\hat{t}_1^n = \underset{t_1^n}{\operatorname{argmax}} \overbrace{P(w_1^n|t_1^n)}^{\text{likelihood}} \; \overbrace{P(t_1^n)}^{\text{prior}}$$

$P(w_1^n)$ independent of t_1^n

Derivations

$$\hat{t}_1^n = \underset{t_1^n}{\operatorname{argmax}} \overbrace{P(w_1^n|t_1^n)}^{\text{likelihood}} \; \overbrace{P(t_1^n)}^{\text{prior}}$$

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

Hp: w_1^n independent

Hp: w_i independent of $t_{i'}$ $\forall\, i' \neq i$

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

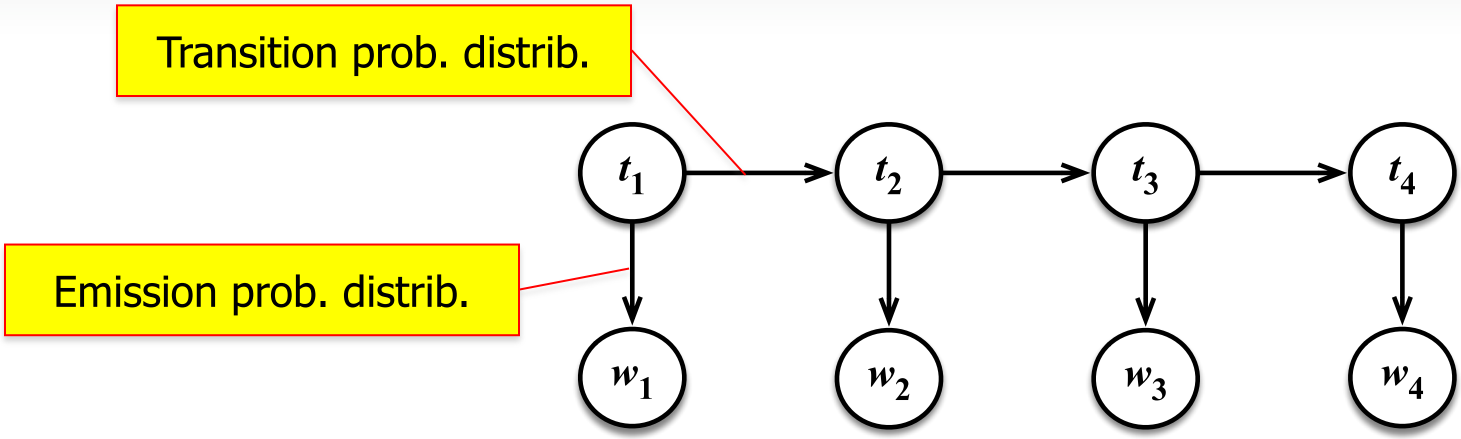
Hp: t_i independent of $t_{i'}$ $\forall\, i' \neq i-1$
(the Markov's assumption)

$$\boxed{\hat{t}_1^n = \underset{t_1^n}{\operatorname{argmax}} P(t_1^n|w_1^n) \approx \underset{t_1^n}{\operatorname{argmax}} \prod_{i=1}^{n} P(w_i|t_i)P(t_i|t_{i-1})}$$

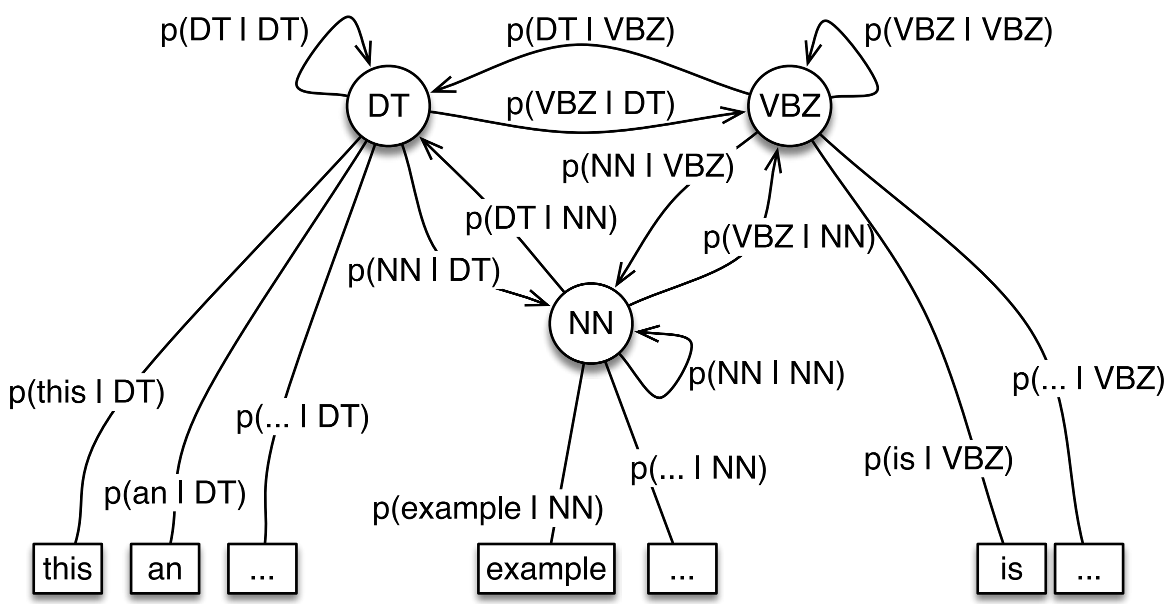
Probability distributions

- $P(t_i \mid t_{i-1})$: *transition* probability distribution
- $P(w_i \mid t_i)$: *emission* probability distribution
- $P(t_1) = P(t_1 \mid t_0) = P(t \mid \text{start})$: *initial* probability distribution
- Hp: those distributions are **time-invariant**
  - The model is described by one transition distribution and one emission distribution
- Use a tagged corpus (MLE) to train distributions
- Pros
  - Works on sequences
  - Small model; fast calculation using Viterbi
  - Each emission probability distribution (in case of multiple observations) is trained independently
- Cons: All those independence and time-invariance assumptions…

Graphical views



Unrolled view



Graph view

Two Kinds of Probabilities

- Tag transition probabilities $P(t_i|t_{i-1})$
  - Determiners (DT) likely to precede adjectives (JJ) and nouns (NN)
    - That/DT flight/NN
    - The/DT yellow/JJ hat/NN
    - So we expect $P(\mathrm{NN}|\mathrm{DT})$ and $P(\mathrm{JJ}|\mathrm{DT})$ to be high
    - But $P(\mathrm{DT}|\mathrm{JJ})$ to be low
  - Computing $P(\mathrm{NN}|\mathrm{DT})$: counting in a labeled corpus:

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

$$P(NN|DT) = \frac{C(DT,NN)}{C(DT)} = \frac{56509}{116454} = 0.49$$

Two Kinds of Probabilities

- Word probabilities $P(w_i|t_i)$
  - VBZ (3sg Pres verb) likely to be "is"
  - Compute $P(is|\mathrm{VBZ})$ by counting in a labeled corpus:

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

$$P(is|VBZ) = \frac{C(VBZ, is)}{C(VBZ)} = \frac{10073}{21627} = 0.47$$

HMM formal definition

- Transition probabilities
  - Transition probability matrix A = $\{a_{ij}\}$ from state i to state j, where $i, j \in W$

    $$a_{ij} = P(q_t = j \mid q_{t-1} = i) \quad 1 \le i, j \le N_w, N_w = |W|$$
- Observation likelihoods
  - Output probability matrix B=$\{b_i(k)\}$; emitting observation o_k, being in state i, where $o_k \in T$

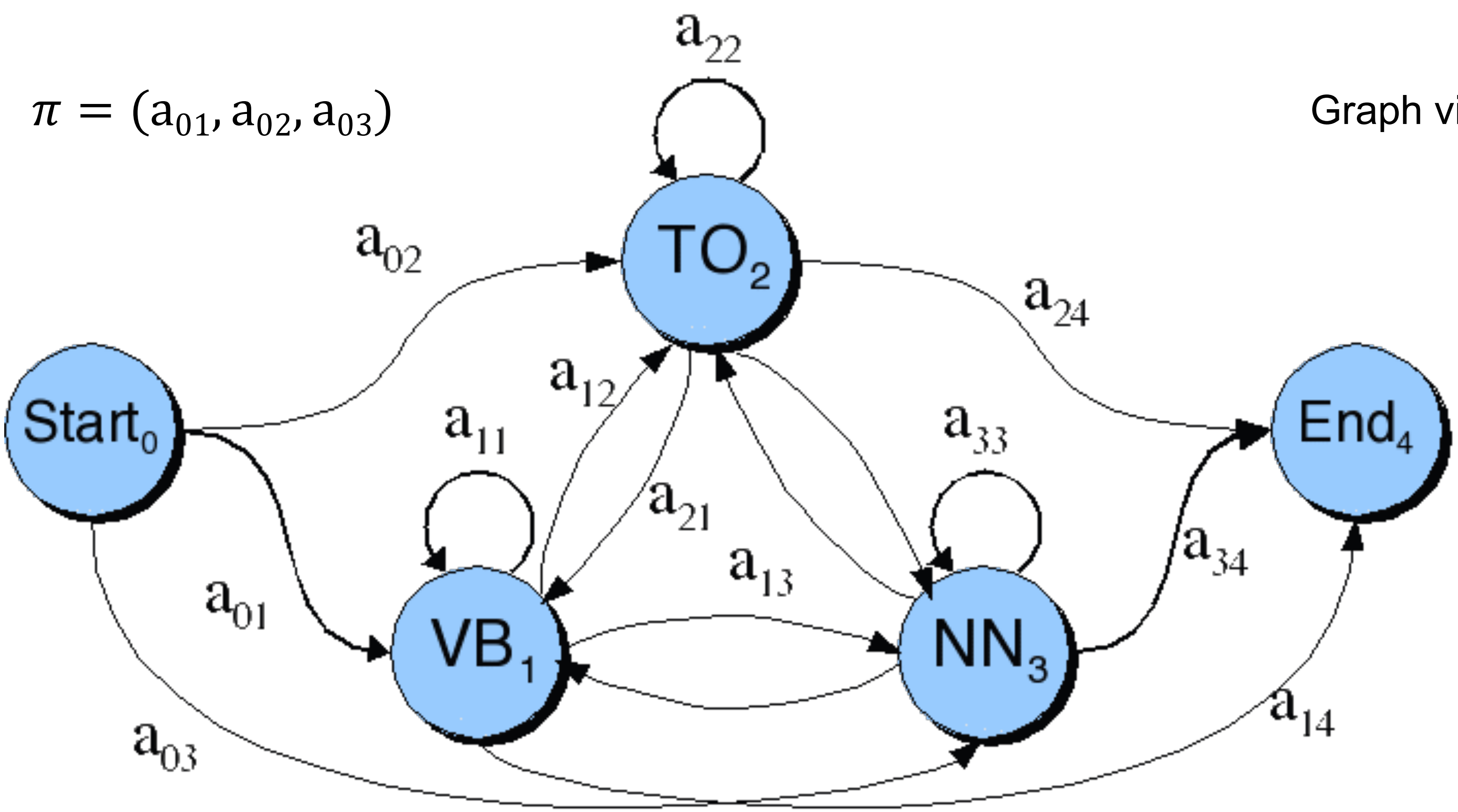
    $$b_i(k) = P(X_t = o_k \mid q_t = i)$$
- Special initial probability vector π

$$\pi_i = P(q_1 = i) \quad 1 \le i \le N_w$$

Transition Probabilities

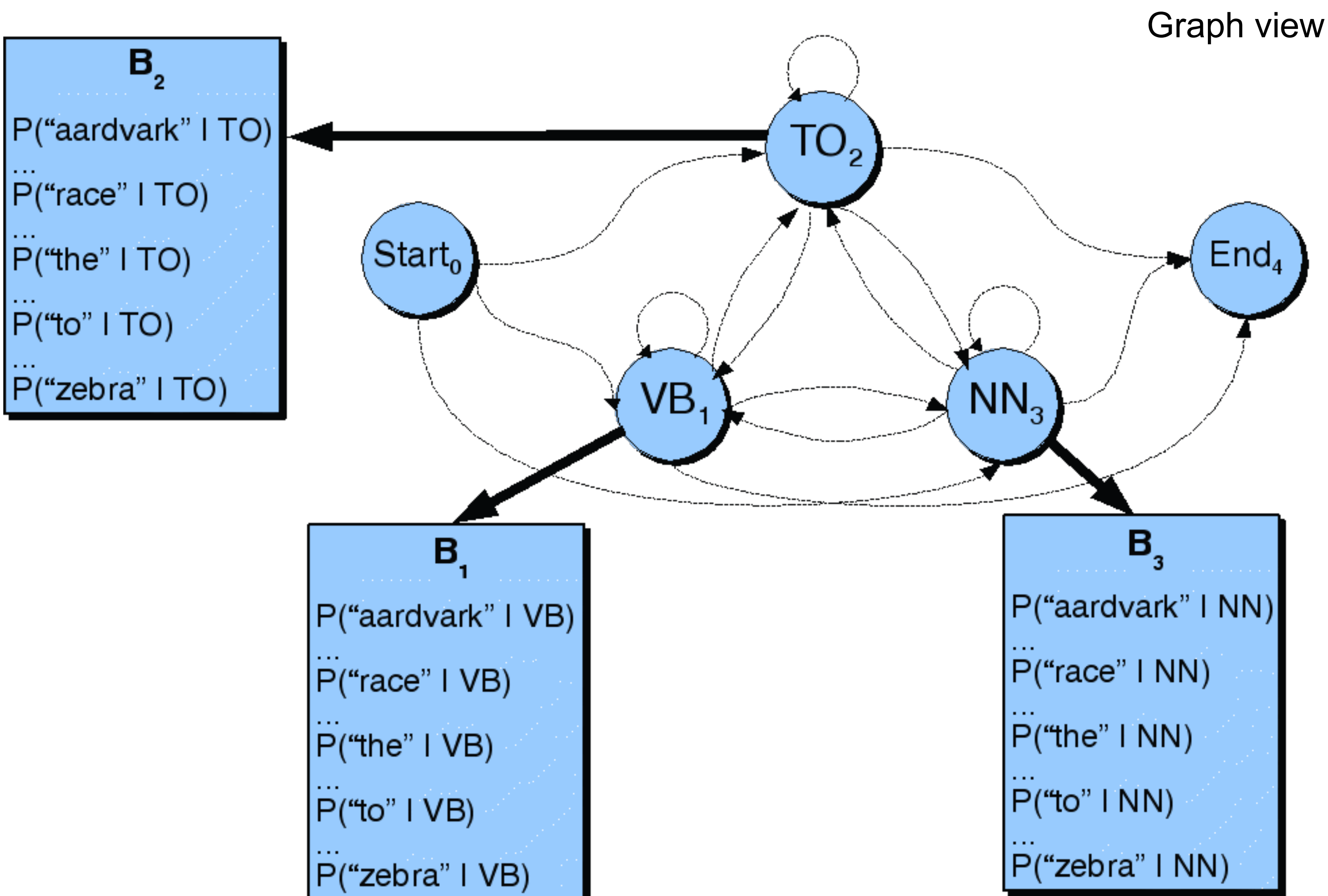
$\pi = (a_{01}, a_{02}, a_{03})$

Graph view



- The special state Start used to represent π
- The special state End is useful…

Observation Likelihoods



Decoding

- Ok, now we have a complete model that can give us what we need. Recall that we need to get

$$\hat{t}_1^n = \underset{t_1^n}{\operatorname{argmax}} P(t_1^n | w_1^n)$$

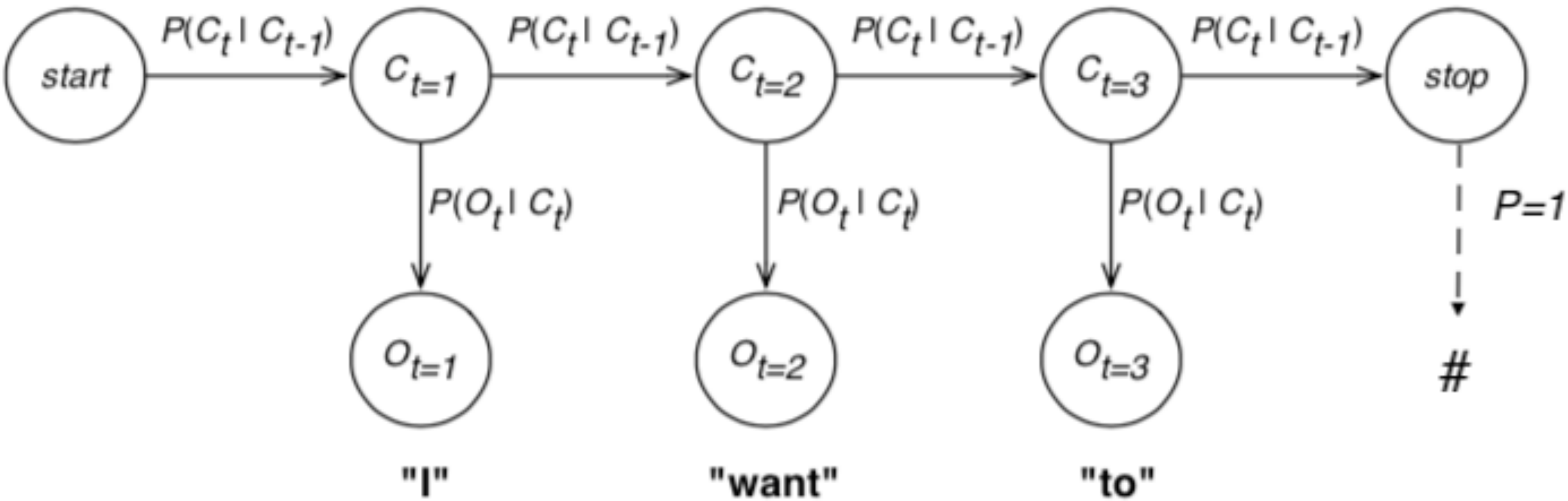
- We could just enumerate all paths given the input and use the model to assign probabilities to each.
  - Not a good idea.
  - Luckily, dynamic programming helps us here

The Viterbi Algorithm

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

create a path probability matrix *viterbi[N+2,T]*
for each state s **from** 1 **to** N **do** ; initialization step
$viterbi[s,1] \leftarrow a_{0,s} * b_s(o_1)$
$backpointer[s,1] \leftarrow 0$
for each time step t **from** 2 **to** T **do** ; recursion step
for each state s **from** 1 **to** N **do**
$$viterbi[s,t] \leftarrow \max_{s'=1}^{N} viterbi[s',t-1] * a_{s',s} * b_s(o_t)$$
$$backpointer[s,t] \leftarrow \operatorname*{argmax}_{s'=1}^{N} viterbi[s',t-1] * a_{s',s}$$
$$viterbi[q_F,T] \leftarrow \max_{s=1}^{N} viterbi[s,T] * a_{s,q_F}$$; termination step
$$backpointer[q_F,T] \leftarrow \operatorname*{argmax}_{s=1}^{N} viterbi[s,T] * a_{s,q_F}$$; termination step
return the backtrace path by following backpointers to states back in time from $backpointer[q_F,T]$

Viterbi Example



Example…

Beware the symbols:
C: hidden state (instead of *t*)
O: observations (instead of *w*)
t: token position

Viterbi Summary

- Assumptions:
  $C_{t=0} = \text{start}$; $C_{t=n} = \text{stop}$; $P(O_{t=n} = \# \mid C_{t=n}) = 1$
  '#' is a fictitious observation → end of sequence
- Create an array
  - With columns corresponding to inputs
  - Rows corresponding to possible states
- Sweep through the array in one pass filling the columns left to right using our transition probs and observations probs
- Dynamic programming key is that we need only store the MAX prob path to each cell, (not all paths)

Supervised & unsupervised learning

- Task: *decoding* a sequence
  - Generate a sequence of hidden symbols, given a sequence of observed symbols
- Training of HMM: tagged corpus (MLE)
  - As we did so far…
- Another task: *recognize* a noisy sequence
  - Input: a sequence of symbols that is supposed to change, due to some kind of random noise (often: Gaussian noise)
  - Output: probability that the HMM could have generated the sequence
- Training for that task: the Baum–Welch algorithm
  - Estimates transition and emission probabilities
  - Training data: the same sequence, added with random noise
- Do that for *n* HMMs: *n* sequences can be recognized

Training & cross-validation

- Sample file
  - Split samples file into training set and test set
- Cross-validation
  - Several methods
- K-fold cross-validation
  - Samples file is randomly partitioned into K subsets
    - For example, 80% training set, 20% test set → K=5
  - A single subset is the test set, K-1 subsets are used as a training set
  - Repeat K times, with each of the K subsets used exactly once as a test set

Training & cross-validation

- Repeated random subsampling validation
  - Split samples file into training set and test set, at random
    - for example, extract 20% of the samples, at random; this is the new test set
    - The remaining samples will be the training set
  - Train and test the model
  - Repeat at will
- In both methods, performance indexes are averaged
- Often, supervised learning algorithms require the user to determine control parameters
  - Use a subset of the training set (the validation set) to adjust such parameters

Model evaluation

- Confusion matrices
- Indexes
  - Precision
  - Recall
  - F-measure
  - Accuracy
- Comparing indexes for different models: the *t*-test

Confusion matrix

PREDICTED CLASSES (e.g., TAGS)

CORRECT CLASSES (e.g., TAGS)

	IN	JJ	NN	NNP	RB	VDB	VBN
IN	**760**	20	0	0	70	0	0
JJ	20	**4350**	330	210	170	20	270
NN	0	870	**5460**	0	0	0	20
NNP	20	330	410	**3508**	20	0	0
RB	220	200	50	0	**2358**	0	0
VDB	0	30	50	0	0	**1480**	440
VBN	0	280	0	0	0	260	**1650**

Table of confusion for NN

True Positive=5460	False Positive=840
False Negative=890	True Negative=16686

Indexes

- Precision, Recall, and F-measure for a class i:

$$\Pr_i = \frac{TP_i}{TP_i + FP_i} \qquad \mathrm{Re}_i = \frac{TP_i}{TP_i + FN_i} \qquad \mathrm{F}_{\beta,i} = \frac{(1+\beta^2)\cdot \Pr_i \cdot \mathrm{Re}_i}{\beta^2 \cdot \Pr_i + \mathrm{Re}_i}$$

Usually, β=1

- Mean $\Pr$, Re, F:

$$\Pr = \sum_i \frac{\Pr_i}{\#\,classes} \qquad \mathrm{Re} = \sum_i \frac{\mathrm{Re}_i}{\#\,classes} \qquad \mathrm{F}_\beta = \sum_i \frac{\mathrm{F}_{\beta,i}}{\#\,classes}$$

- Weighted mean $\Pr$, Re, F:

$$\Pr = \sum_i \alpha_i \cdot \Pr_i \qquad \mathrm{Re} = \sum_i \alpha_i \cdot \mathrm{Re}_i \qquad \mathrm{F}_\beta = \sum_i \alpha_i \cdot \Pr$$

α_i = (# instances of class i in corpus) / (#samples)

- Accuracy:

$$\mathrm{Ac} = \frac{\sum_i TP_i}{\#\,samples}$$

Confusion matrix of errors

- Values on diagonal → right classification; other values → errors
- Each cell indicates percentage of the overall tagging error

PREDICTED CLASSES (e.g., TAGS) / CORRECT CLASSES (e.g., TAGS)

	IN	JJ	NN	NNP	RB	VDB	VBN
IN	-	0.0046	0	0	0.016	0	0
JJ	0.0046	-	0.076	0.049	0.039	0.0046	0.062
NN	0	0.2	-	0	0	0	0.0046
NNP	0.0046	0.076	0.095	-	0.0046	0	0
RB	0.051	0.046	0.011	0	-	0	0
VDB	0	0.0069	0.011	0	0	-	0.1
VBN	0	0.065	0	0	0	0.06	-

$$\mathrm{Err}(correct\_tag_1, predicted\_tag_2) = \frac{\mathrm{C}(correct\_tag_1, predicted\_tag_2)}{\sum_{i \neq j} \mathrm{C}(correct\_tag_i, predicted\_tag_j)}$$

Example: $$\mathrm{Err}(\mathrm{IN}, \mathrm{JJ}) = \frac{\mathrm{C}(\mathrm{IN}, \mathrm{JJ})}{\sum_{i \neq j} \mathrm{C}(correct\_tag_i, predicted\_tag_j)} = \frac{20}{4310} = 0.0046$$

Paired, two-tailed *t*-test

- Significance of difference variables

$$D = I^{(M2)} - I^{(M1)}$$

Index I to compare, for models $M1$ and $M2$

$$t = \frac{\bar{D}}{S_D / \sqrt{N}}$$

Student's t distribution with N-1 degrees of freedom

$$\bar{D} = \frac{\sum_i^N I_i^{(M2)} - I_i^{(M1)}}{N}$$

Mean (for each model, we have N values for I)

$$S_D = \sqrt{\frac{\sum_i^N \left(D_i - \bar{D}\right)^2}{N-1}}$$

Standard deviation

$$2 \cdot P(t, N)$$

Two-tailed P-value

- if $2 \cdot P(t,N) < 0.01$ (or 0.05), difference is significant
- Used to compare metrics of two systems

Paired, two-tailed *t*-test

http://www.blogforgood.net/2011/09/20/t-table/

#	$I^{(M1)}$	$I^{(M2)}$	D
1	25	35	10
2	43	84	41
3	39	15	-24
4	75	75	0
5	43	68	25
6	15	85	70
7	20	80	60
8	52	50	-2
9	49	58	9
10	50	75	25

$\overline{M}1=41.1$

$\overline{M}2=62.5$

is $M2$ better than $M1$?

$df = N-1=9$

$$\bar{D} = 21.4; \quad S_D = 29.1 \rightarrow t = 2.33$$

Two-tailed test: $2 \cdot P(t,10)=2 \cdot 0.02 < 0.05$ → OK

$M2$ better than $M1$ with $1-0.04$ confidence (96%)

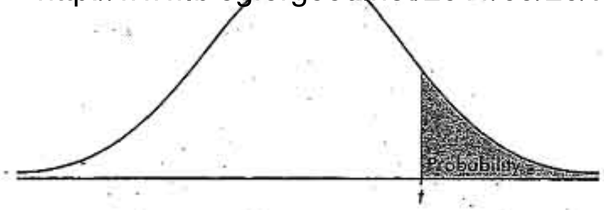


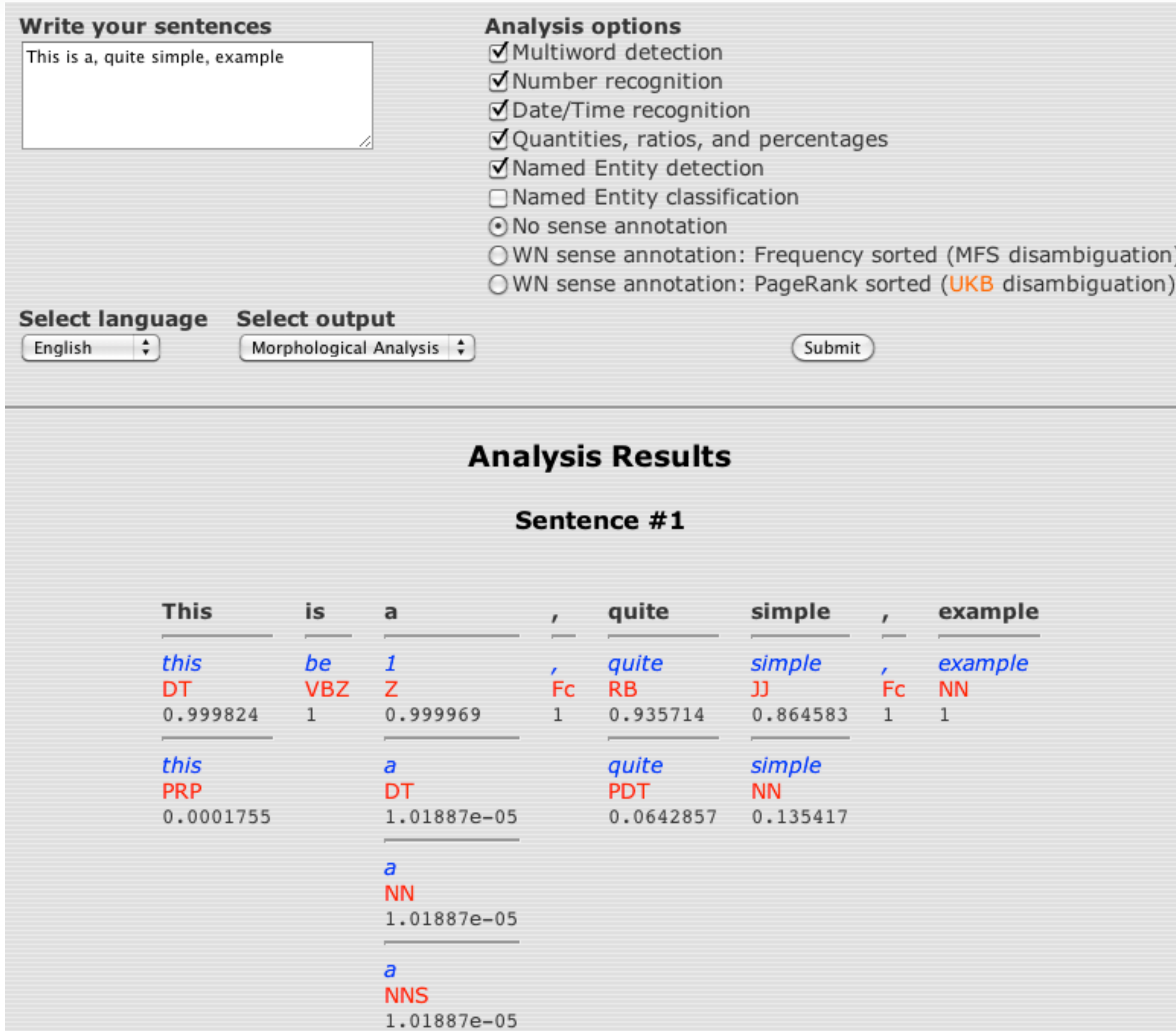
TABLE B: *t*-DISTRIBUTION CRITICAL VALUES

	Tail probability p											
df	.25	.20	.15	.10	.05	.025	.02	[illegible]	.005	.0025	.001	.0005
1	1.000	1.376	1.963	3.078	6.314	12.71	15.89	31.82	[illegible]	127.3	318.3	636.6
2	.816	1.061	1.386	1.886	2.920	4.303	4.849	6.965	[illegible]	14.09	22.33	31.60
3	.765	.978	1.250	1.638	2.353	3.182	3.482	4.541	[illegible]	7.453	10.21	12.92
4	.741	.941	1.190	1.533	2.132	2.776	2.999	3.747	[illegible]	5.598	7.173	8.610
5	.727	.920	1.156	1.476	2.015	2.571	2.757	3.365	[illegible]	4.773	5.893	6.869
6	.718	.906	1.134	1.440	1.943	2.447	2.612	3.143	[illegible]	4.317	5.208	5.959
7	.711	.896	1.119	1.415	1.895	2.365	2.517	2.998	[illegible]	4.029	4.785	5.408
8	.706	.889	1.108	1.397	1.860	2.306	2.449	2.896	[illegible]	3.833	4.501	5.041
9	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	2.398	[illegible]	3.250	3.690	4.297	4.781
10	.700	.879	1.093	1.372	1.812	2.228	2.359	2.764	3.169	3.581	4.144	4.587
11	.697	.876	1.088	1.363	1.796	2.201	[illegible]	2.718	3.106	3.497	4.025	4.437
12	.695	.873	1.083	1.356	1.782	2.179	[illegible]	2.681	3.055	3.428	3.930	4.318
13	.694	.870	1.079	1.350	1.771	2.160	[illegible]	2.650	3.012	3.372	3.852	4.221
14	.692	.868	1.076	1.345	1.761	2.145	[illegible]	2.624	2.977	3.326	3.787	4.140
15	.691	.866	1.074	1.341	1.753	2.131	[illegible]	2.602	2.947	3.286	3.733	4.073
16	.690	.865	1.071	1.337	1.746	2.120	[illegible]	2.583	2.921	3.252	3.686	4.015
17	.689	.863	1.069	1.333	1.740	2.110	[illegible]	2.567	2.898	3.222	3.646	3.965
18	.688	.862	1.067	1.330	1.734	2.101	[illegible]	2.552	2.878	3.197	3.611	3.922
19	.688	.861	1.066	1.328	1.729	2.093	[illegible]	2.539	2.861	3.174	3.579	3.883
20	.687	.860	1.064	1.325	1.725	2.086	[illegible]	2.528	2.845	3.153	3.552	3.850
21	.686	.859	1.063	1.323	1.721	2.080	[illegible]	2.518	2.831	3.135	3.527	3.819
22	.686	.858	1.061	1.321	1.717	2.074	[illegible]	2.508	2.819	3.119	3.505	3.792
23	.685	.858	1.060	1.319	1.714	2.069	[illegible]	2.500	2.807	3.104	3.485	3.768
24	.685	.857	1.059	1.318	1.711	2.064	[illegible]	2.492	2.797	3.091	3.467	3.745
25	.684	.856	1.058	1.316	1.708	2.060	[illegible]	2.485	2.787	3.078	3.450	3.725
26	.684	.856	1.058	1.315	1.706	2.056	[illegible]	2.479	2.779	3.067	3.435	3.707
27	.684	.855	1.057	1.314	1.703	2.052	[illegible]	2.473	2.771	3.057	3.421	3.690
28	.683	.855	1.056	1.313	1.701	2.048	[illegible]	2.467	2.763	3.047	3.408	3.674
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	2.462	2.756	3.038	3.396	3.659
30	.683	.854	1.055	1.310	1.697	2.042	2.147	2.457	2.750	3.030	3.385	3.646
40	.681	.851	1.050	1.303	1.684	2.021	2.123	2.423	2.704	2.971	3.307	3.551
50	.679	.849	1.047	1.299	1.676	2.009	2.109	2.403	2.678	2.937	3.261	3.496
60	.679	.848	1.045	1.296	1.671	2.000	2.099	2.390	2.660	2.915	3.232	3.460
80	.678	.846	1.043	1.292	1.664	1.990	2.088	2.374	2.639	2.887	3.195	3.416
100	.677	.845	1.042	1.290	1.660	1.984	2.081	2.364	2.626	2.871	3.174	3.390
1000	.675	.842	1.037	1.282	1.646	1.962	2.056	2.330	2.581	2.813	3.098	3.300
∞	.674	.841	1.036	1.282	1.645	1.960	2.054	2.326	2.576	2.807	3.091	3.291
	50%	60%	70%	80%	90%	95%	96%	98%	99%	99.5%	99.8%	99.9%
	Confidence level C											

Example from: B. Croft, D. Metzler, and T. Strohman, Search Engines: Information Retrieval in Practice

FreeLing

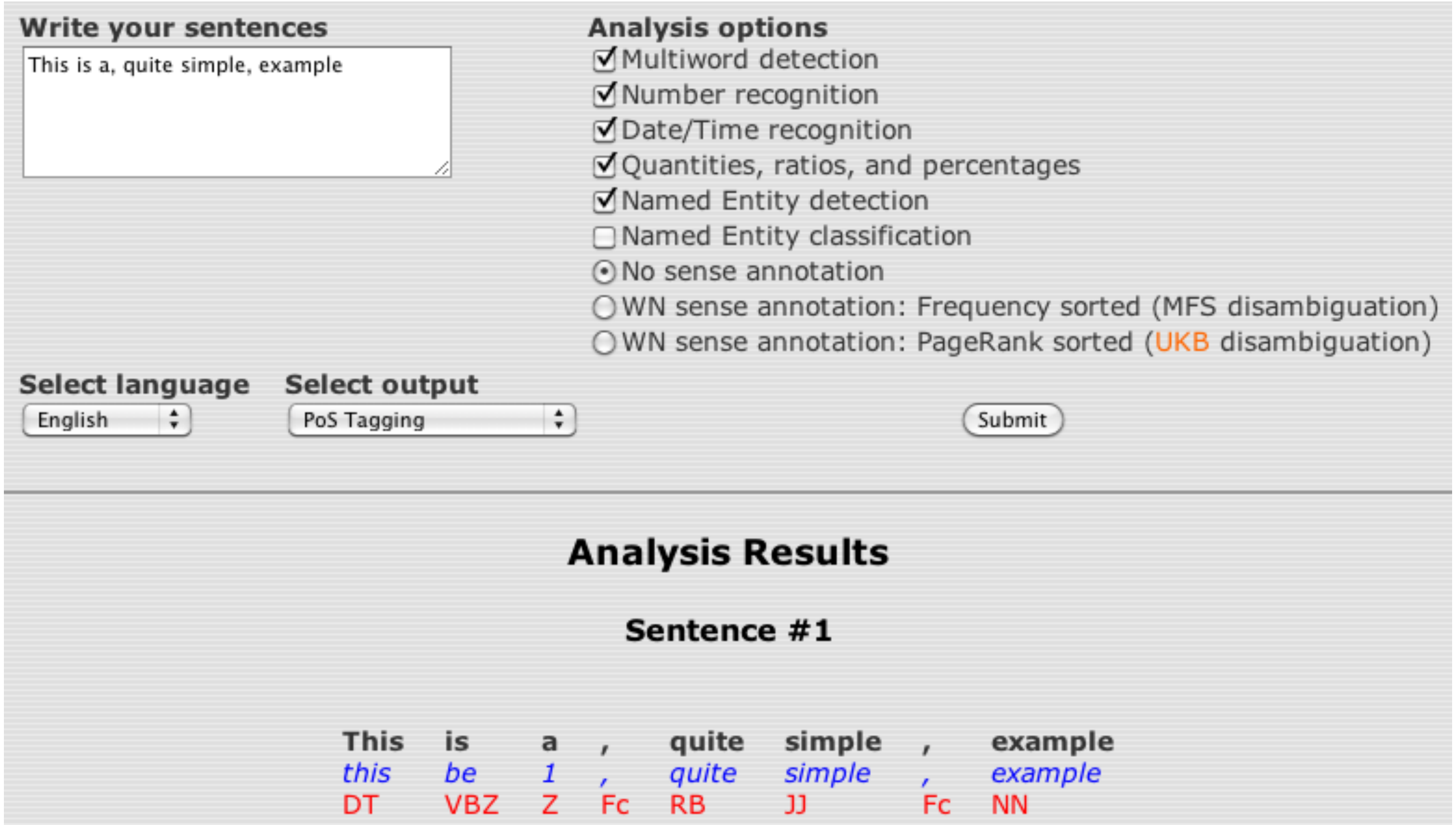
- As a morphologic analyzer



demo

FreeLing

- POS tagging
- HMM



demo

Stanford POS tagger

- Stanford POS Tagger
- Entropy Maximization
  - Uses a CMM (basically a MEMM)
  - A particular Maximum Entropy (MaxEnr) model
  - MaxEnt models are discriminative models
- Java based

- demo

References

- Stanford
  - http://nlp.stanford.edu/software/index.shtml
- FreeLing (POS, parser, morpho analyzer, …)
  - http://nlp.lsi.upc.edu/freeling/