

# **Introduction to Corpus Resources, Annotation and Access: *Morpho-Syntactic Annotation***

---

Sabine Schulte im Walde  
Universität Stuttgart

*Foundational Course*  
Departament de Traducció i Filologia  
Universitat Pompeu Fabra  
April 16-20, 2007

# Part-of-Speech Tagging



# Tagging (1)

---

<i>John</i>	{PN}
<i>sends</i>	{V}
<i>his</i>	{PRON}
<i>girl-friend</i>	{N}
<i>presents</i>	{N, V}

- Tagging is the task of **labeling each word** in a sequence of words with its appropriate **part-of-speech** (POS).
- Words are often ambiguous with respect to their POS:
  - » *saw* → singular noun
  - » *saw* → past tense of verb see
- Purposes and applications (examples):
  - » pre-processing step for morpho-syntactic and further analyses: lemmata, syntactic structure, etc.
  - » text indexing, e.g. nouns are more useful than verbs
  - » pronunciation in speech processing

# Tagging (2)

---

- Tagging performs a limited syntactic disambiguation.
- Tagging accuracy is high (on a per-word basis): 95-98%; transferring 96% accuracy on a per-word basis to a 20-word sentence basis corresponds to  $0.96^{20} \approx 44\%$ .
- The syntagmatic context helps to disambiguate tags.
- Some POS sequences are common, e.g. **DET ADJ N**.
- Words are associated with dominant POS tags: The distribution of a word's POS tags is extremely uneven.
- Statistical approaches often combine syntagmatic information and lexical information on POS preferences.

# Tagsets

---

- A **tagset** is a **set of part-of-speech tags**.
- The size and choice of the tagsets vary.
- Classical 8 classes (Thrax, 100 BC): noun, verb, article, participle, pronoun, preposition, adverb, conjunction
- Morphologically rich languages (such as German) need more detailed tagsets (such as gender and case).
- Criteria:
  - » **specifiability**: degree to which humans use the tagset uniformly on the same text
  - » **accuracy**: evaluation of output on tagged text
  - » **suitability for intended application**

# Tagset Examples

Penn Treebank Tagset (English) - 37 tags		STTS Tagset (German) - 54 tags	
JJ	adjective, positive	ADJA	adjective, attributive
JJR	adjective, comparative	ADJD	adjective, predicative
JJS	adjective, superlative	NN	common noun
NN	non-plural common noun	NE	proper name
NNS	plural common noun	APPR	preposition
NNP	non-plural proper name	APPRART	preposition incorporating article
NNPS	plural proper name	APPO	postposition
IN	preposition	VVFIN	base verb, finite
VB	base verb	VVIMP	base verb, imperative
VBD	base verb, past tense	VVINFIN	base verb, non-finite
VBG	base verb, gerund or participle I	VVIZU	base verb incorporating <i>zu</i>
VBN	base verb, participle II	VVPP	base verb, participle II
VBP	base verb, non-3rd person	PPOSS	possessive pronoun, substituting
VBZ	base verb, 3rd person	PPOSAT	possessive pronoun, attributive
POS	possessive pronoun	PRF	personal pronoun, reflexive

# Tagging Example: Penn Treebank

---

Under/IN

[ the/DT proposal/NN ]

,/,

[ Delmed/NNP ]

would/MD issue/VB about/IN

[ 123.5/CD million/CD additional/JJ Delmed/NNP common/JJ shares/NNS ]

to/TO

[ Fresenius/NNP ]

at/IN

[ an/DT average/JJ price/NN ]

of/IN about/IN

[ 65/CD cents/NNS ]

[ a/DT share/NN ]

,/, though/IN under/IN

[ no/DT circumstances/NNS ]

more/JJR than/IN

[ 75/CD cents/NNS ]

[ a/DT share/NN ]

./.

# Tagging with Markov Models

---

- Sequence of tags in a text is regarded a Markov chain.
- Limited horizon: A word's tag only depends on the previous tag:  $p(x_{i+1} = t^j \mid x_1, \dots, x_i) = p(x_{i+1} = t^j \mid x_i)$
- Time invariant: This dependency does not change over time:  $p(x_{i+1} = t^j \mid x_i) = p(x_2 = t^j \mid x_1)$
- **Task:** Find the **most probable tag sequence** for a sequence of words.
- Maximum likelihood estimate of tag  $t^k$  following  $t^j$ :  
$$p(t^k \mid t^j) = f(t^j, t^k) / f(t^j)$$
- Optimal tags for a sentence:  
$$t'_{1,n} = \arg \max p(t_{1,n} \mid w_{1,n}) = \prod p(w_i \mid t_i) p(t_i \mid t_{i-1})$$

# Transformation-based Tagging (TbT)

---

- Tags can be conditioned on words and on more context: TbT exploits a range of lexical and syntactic regularities.
- TbT encodes interdependencies between words and tags by **selecting and sequencing transformations** that transform an imperfect tagging into one with fewer errors.
- Components:
  - » **specification of transformations**
  - » **learning algorithm**
- Procedure:
  1. Each word is tagged with its most frequent tag.
  2. The learning algorithm constructs a ranked list of transformations.

# Transformations

---

- A transformation consists of two parts:
  - » triggering environment
  - » rewrite rule
- Examples:

Triggering Environment	Source Tag	Target Tag
previous tag is TO	NN	VB
one of previous three tags is MD	VBP	VB
next tag is JJ	JJR	RBR
one of previous two words is <i>n't</i>	VBP	VB

# Transformation Ranking

---

- Greedy search for the optimal transformation sequence
- Steps:
  1. Each word is tagged with its most frequent tag.
  2. The transformation that reduces the error rate most is chosen.
  3. Step 2 is repeated iteratively until no transformation is left that reduces the error rate by more than a pre-specified threshold.
- The error rate is measured as the number of mistagged words.

# Tagging: References

---

- Anne Schiller and Simone Teufel (1995): "Guidelines für das Tagging deutscher Textkorpora mit STTS". Manuscript, in German.
- Christopher D. Manning and Hinrich Schütze (1999): "Foundations of Statistical Natural Language Processing", chapter 10. MIT Press.
- Atro Voutilainen (2003): "Part-of-speech tagging". In: Ruslan Mitkov, editor: "The Oxford Handbook of Computational Linguistics", pp. 219-232. Oxford University Press.
- Helmut Schmid (2007?): "Tagging". In: Anke Lüdeling and Merja Kytö, editors: "Corpus Linguistics. An International Handbook." Mouton de Gruyter, Berlin.

# Tagging: Tools

---

- **TnT - Trigram Tagger:** [www.coli.uni-saarland.de/~thorsten/tnt/](http://www.coli.uni-saarland.de/~thorsten/tnt/)  
Thorsten Brants (2000): "TnT - A statistical part-of-speech tagger." In *Proceedings of the 6th Applied Natural Language Processing Conference*.
- **Tree Tagger:** [www.ims.uni-stuttgart.de/projekte/corplex/TreeTagger/](http://www.ims.uni-stuttgart.de/projekte/corplex/TreeTagger/)  
Helmut Schmid (1994): "Probabilistic Part-of-Speech Tagging Using Decision Trees". In *Proceedings of the International Conference on New Methods in Language Processing*.
- **Brill Tagger:** [www.cs.jhu.edu/~brill](http://www.cs.jhu.edu/~brill)  
Eric Brill (1995): "Transformation-based error-driven learning and Natural Language Processing: A case study in part-of-speech tagging". *Computational Linguistics*, 21(4):543-565.

# Morphological Annotation



# Morphology (1)

---

- Morphology is concerned with the **inner structure of words** and the **formation of words** from smaller units.
- The part of the word (morpheme) that carries the central meaning is called the **root**.
- How much and what sort of information is expressed by morphology differs widely between languages. Information that is expressed by syntax in one language is expressed morphologically in another one.
- Applications (examples):
  - » hyphenation
  - » grapheme-to-phoneme conversion in text-to-speech
  - » spelling correction

# Morphology (2)

---

- Basic functions of morphology:
  - » **inflection**: change of word form which does not change the part-of-speech category, such as **conjugation** (*lese, liest, las*, etc. in German)
  - » **derivation**: a new word is produced by adding a morph to the base form, such as verb → adjective:  
*essen* `eat` → *essbar* `eatable`
  - » **compounding**: joining of two or more base forms to form a new word, such as *coffee filter*, *Kaffeefilter*
- **Morphotactics**: A word grammar determines the way how morphs are put together to form words.

# Computational Morphology

---

- **Task:** Take a string of characters or phonemes as input and deliver an analysis of the underlying morphemes or the morphosyntactic interpretation as output:

*incompatibilities*

in+con+patible+ity+s (morphemes)

incompatibility+Noun+Plural (interpretation)

- Methods:
  - » **lexicon:** full-form or lemmata
  - » **finite-state morphology:** use regular expressions
  - » **machine learning** of morphological structure

# Stemming and Lemmatisation

---

- **Stemming**: a process that strips off affixes and leaves you with the stem:

*cats, catlike, catty* → cat

*rauchst* → rauch

- **Lemmatisation**: one is attempted to find the lemma or lexeme of the inflected form; lemmatisation includes disambiguation at the level of lexemes, depending on the part-of-speech:

*rauchst* → *rauchen*

# Two-Level Morphology (1)

---

- **Two-level morphology** represents a word as a correspondence between two levels:
  - » **lexical level**: abstract morphemes and features, such as **cat+N+plural**
  - » **surface level**: actual spelling of the word: *cats*
- Two-level morphology takes care of **morphophonology**.
- Two-level morphology can be represented using finite-state transducers.
- A **finite-state transducer** is a finite-state automaton that recognises pairs of strings, by mimicing an ordered set of rewrite rules.

# Two-Level Morphology (2)

---

- Two-level morphology is **non-directional** (applicable to analysis and generation).
- Two-level morphology is **language-independent**.
- A set of pairs of lexical and surface characters constitutes possible mappings. Rules function as constraints on the mapping between surface and lexical form of morphs. They are applied in parallel.
- A rule consists of
  - » **substitution**
  - » **left and right context**
  - » **operators**

# Two-Level Morphology (3)

---

- Example rule:  **$+ : e \leftarrow \{s \ x \ z \ [ \{s \ c\} \ h \ ] \} : \_ s$**
- e is inserted between stem and an inflectional affix starting with s (plural, 3rd person, superlative) in English. By default, the morph boundary will map to *null*, but in the given specific context it maps to e.
- Application of this rule:  

#bliss+s#	#fox+s#	#dish+s#	#watch+s#
0blisses0	0foxes0	0dishes0	0watches0

# Automatic Discovery of Morphology

---

- Statistical methods for detecting morphological structure, affixes and suffixes
- Basis:
  - » absolute/relative frequencies of potential morphemes
  - » co-occurrence patterns
  - » length of words and their substrings ...
- Applications:
  - » morphological acquisition
  - » (probabilistic) morphological lexicons (for new languages)
- PASCAL challenge: <http://www.cis.hut.fi/morphochallenge2005/>

# Morphology: References

---

- Kimmo Koskenniemi (1984): “A general computational model for word-form recognition and production”. In *Proceedings of the 10th International Conference on Computational Linguistics*, pp. 178-181.
- Kenneth R. Beesley and Lauri Karttunen (2003): “Finite state morphology”. CSLI Publications.
- Harald Trost (2003): “Morphology”. In: Ruslan Mitkov, editor: “*The Oxford Handbook of Computational Linguistics*”, pp. 25-47. Oxford University Press.
- Marco Baroni (2003): “Distribution-driven morpheme discovery: A computational/experimental study”. In: Geert Booij and Jaap van Marle, editors: “*Yearbook of Morphology 2003*”, pp. 213-248.
- Harald Hammarström (2006): “A naive theory of affixation and an algorithm for extraction”. In *Proceedings of the 8th Meeting of the ACL Special Interest Group in Computational Phonology (SIGPHON)*.

# Morphology: Tools

---

- **Porter Stemmer:** <http://www.tartarus.org/martin/PorterStemmer/>  
Martin F. Porter (1980): “An algorithm for suffix stripping”. *Program*, 14(3):130-137.
- **Snowball:** <http://snowball.tartarus.org/>  
System for stemming algorithms
- **PC-Kimmo:** <http://www.sil.org/pckimmo/>  
Morphological parser based on two-level morphology
- **Stuttgart Finite-State Transducer Tools (SFST):**  
[www.ims.uni-stuttgart.de/projekte/gramotron/SOFTWARE/SFST.html](http://www.ims.uni-stuttgart.de/projekte/gramotron/SOFTWARE/SFST.html)  
Toolbox for the implementation of morphological analysers

# Word Distributions



# Word Distributions

---

## Word distributions:

frequencies of words and combinations of words in corpora

1. Types and Tokens
2. *n*-grams
3. Concordances
4. Collocations

# Types and Tokens

---

- **Token**: total number of word instances in a corpus  
→ corpus size

*Peter<sub>1</sub> 's<sub>2</sub> father<sub>3</sub> is<sub>4</sub> a<sub>5</sub> cook<sub>6</sub> .<sub>7</sub>*

*Peter<sub>8</sub> 's<sub>9</sub> mother<sub>10</sub> is<sub>11</sub> also<sub>12</sub> a<sub>13</sub> cook<sub>14</sub> .<sub>15</sub>*

- **Types**: number of distinct words in a corpus  
→ vocabulary size

*Peter<sub>1</sub> 's<sub>2</sub> father<sub>3</sub> is<sub>4</sub> a<sub>5</sub> cook<sub>6</sub> .<sub>7</sub>*

*Peter<sub>7</sub> 's<sub>7</sub> mother<sub>8</sub> is<sub>8</sub> also<sub>9</sub> a<sub>9</sub> cook<sub>9</sub> .<sub>9</sub>*

# Concordance

---

- **Concordance**: a word with its immediate context
- **KWIC**: key word in context; concordance lines
- Usage:
  - » analysing key words
  - » analysing word frequencies
  - » comparing different uses of the same word (context words and structure)
  - » finding and analysing collocations

# Concordance Examples: *CQP*

- Query: `[lemma="take" & pos="VB.*"]`

A Christmas Carol, Chapter 1

... nothing more remarkable in his **taking** a stroll at night , ...

A Christmas Carol, Chapter 1

... ' said the gentleman , **taking** up a pen , ` ...

A Christmas Carol, Chapter 1

... play at blindman's-buff . Scrooge **took** his melancholy dinner in his ...

A Christmas Carol, Chapter 1

... up that staircase , and **taken** it broadwise , with the ...

A Christmas Carol, Chapter 1

... secured against surprise , he **took** off his cravat ; put ...

A Christmas Carol, Chapter 1

... down before the fire to **take** his gruel . It was ...

A Christmas Carol, Chapter 1

... himself in a condition to **take** a chair ; and felt ...

A Christmas Carol, Chapter 1

... horror , when the phantom **taking** off the bandage round its ...

A Christmas Carol, Chapter 1

... . ` Could n't I **take** ` em all at once ...

A Christmas Carol, Chapter 1

... these words , the spectre **took** its wrapper from the table ...

# Concordance Examples: CQP

- Query: [lemma="take" & pos="VB.\*"]

## A Christmas Carol, Chapter 1

... nothing/NN more/RBR remarkable/JJ in/IN his/PP\$ **taking/VBG** a/DT stroll/VBP at/IN night/NN ,/, ...

## A Christmas Carol, Chapter 1

... '/' said/VBD the/DT gentleman/NN ,/, **taking/VBG** up/RP a/DT pen/NN ,/, `/'` ...

## A Christmas Carol, Chapter 1

... play/VB at/IN blindman's-buff/NN ./SENT Scrooge/NN **took/VBD** his/PP\$ melancholy/JJ dinner/NN in/IN his/PP\$ ...

## A Christmas Carol, Chapter 1

... up/IN that/DT staircase/NN ,/, and/CC **taken/VBN** it/PP broadwise/RB ,/, with/IN the/DT ...

## A Christmas Carol, Chapter 1

... secured/VBN against/IN surprise/NN ,/, he/PP **took/VBD** off/RP his/PP\$ cravat/NN ;/: put/VBN ...

## A Christmas Carol, Chapter 1

... down/RP before/IN the/DT fire/NN to/TO **take/VB** his/PP\$ gruel/NN ./SENT It/PP was/VBD ...

## A Christmas Carol, Chapter 1

... himself/PP in/IN a/DT condition/NN to/TO **take/VB** a/DT chair/NN ;/: and/CC felt/VBD ...

## A Christmas Carol, Chapter 1

... horror/NN ,/, when/WRB the/DT phantom/JJ **taking/VBG** off/RP the/DT bandage/NN round/VB its/PP\$ ...

## A Christmas Carol, Chapter 1

... ./SENT `/'` Could/MD n't/RB I/PP **take/VBP** `/'` em/NN all/RB at/IN once/RB ...

## A Christmas Carol, Chapter 1

... these/DT words/NNS ,/, the/DT spectre/NN **took/VBD** its/PP\$ wrapper/NN from/IN the/DT table/NN ...

# Concordance Examples: CQP

- Query: [lemma="take" & pos="VB.\*"]

## A Christmas Carol, Chapter 1

... nothing/NN more/RBR remarkable/JJ in/IN his/PP\$ take/VBG a/DT stroll/VBP at/IN night/NN ,/, ...

## A Christmas Carol, Chapter 1

... 'I' say/VBD the/DT gentleman/NN ,/, take/VBG up/RP a/DT pen/NN ,/, 'I' ...

## A Christmas Carol, Chapter 1

... play/VB at/IN blindman's-buff/NN ./SENT Scrooge/NN take/VBD his/PP\$ melancholy/JJ dinner/NN in/IN his/PP\$ ...

## A Christmas Carol, Chapter 1

... up/IN that/DT staircase/NN ,/, and/CC take/VBN it/PP broadwise/RB ,/, with/IN the/DT ...

## A Christmas Carol, Chapter 1

... secure/VBN against/IN surprise/NN ,/, he/PP take/VBD off/RP his/PP\$ cravat/NN ;/: put/VBN ...

## A Christmas Carol, Chapter 1

... down/RP before/IN the/DT fire/NN to/TO take/VB his/PP\$ gruel/NN ./SENT it/PP be/VBD ...

## A Christmas Carol, Chapter 1

... himself/PP in/IN a/DT condition/NN to/TO take/VB a/DT chair/NN ;/: and/CC feel/VBD ...

## A Christmas Carol, Chapter 1

... horror/NN ,/, when/WRB the/DT phantom/JJ take/VBG off/RP the/DT bandage/NN round/VB its/PP\$ ...

## A Christmas Carol, Chapter 1

... ./SENT 'I' Could/MD not/RB I/PP take/VBP 'I' em/NN all/RB at/IN once/RB ...

## A Christmas Carol, Chapter 1

... these/DT word/NNS ,/, the/DT spectre/NN take/VBD its/PP\$ wrapper/NN from/IN the/DT table/NN ...

# Concordance Examples: *CQP*

- Query: `[lemma="take" & pos="VB.*"] [lemma="up" & pos="RP"] [{"1,3} [pos="NN"]];`

## A Christmas Carol, Chapter 1

... ' said the gentleman , **taking up a pen** , ' it is more ...

## David Copperfield, Chapter 1

... of a room , to **take up the less space** . He walked as softly ...

## David Copperfield, Chapter 4

... an impressive look , and **took up his book** . This was a good ...

## David Copperfield, Chapter 4

... he rose and said , **taking up the cane** : ' Why , Jane ...

## David Copperfield, Chapter 5

... , ' he said , **taking up a table-spoon** , ' is my favourite ...

## David Copperfield, Chapter 14

... earnestly at me , and **taking up his pen** to note it down , ...

## David Copperfield, Chapter 17

... , it will not be **taken up** . **The result** is destruction . The bolt ...

## David Copperfield, Chapter 30

... ' said Mr. Omer , **taking up his glass** , ' because it 's ...

## David Copperfield, Chapter 32

... his birth , - to **take up in a moment** with a miserable girl , ...

## David Copperfield, Chapter 35

... from the time of my **taking up my residence** in Mr. Wickfield 's house ...

# Cooccurrence

---

- “You shall know a word by the company it keeps.”  
(Firth, 1957)
- Words are not combined randomly into phrases and sentences.
- The particular ways in which they go together are a rich and important source of information both about language and about the world we live in.
- Word **associations** are (to some extent) built upon word cooccurrences.
- Issues:
  - » **linear/structural combination** of cooccurrences
  - » **semantic compositionality** of cooccurrences

# *n*-grams

---

- ***n*-gram**: sequence of *n* words
  - » **bigram**: sequence of 2 words
  - » **trigram**: sequence of 3 words ...
- *n*-grams are the basis for language models, to predict the next word given the previous word(s):

$$p(w_n) = p(w_n | w_1, \dots, w_{n-1})$$

- Restrictions on *n*:
  - » number of parameters and sparse data
  - » full-form words vs. lemmata
  - » corpus transfer
- Application (examples): spelling correction, machine translation, speech recognition

# Collocations

---

Lexical items in a specified grammatical relationship

- **Non-compositional** and **idiomatic combination**

Example: *kick the bucket*

- **Habitual co-occurrence**: two or more words that correspond to some conventional way of saying things

Example: direct objects of *eat*: *chocolate, soup, bread, ...*

# Collocation Induction

---

1. Cooccurrence frequencies of bigrams/trigrams/etc.  
 $f(\text{lex}_1, \text{lex}_2)$  for any two lexical items  $\text{lex}_1$  and  $\text{lex}_2$
2. Co-occurrence frequencies in specific relationship:  
 $f(\text{lex}_1, \text{rel}, \text{lex}_2)$  for any two lexical items  $\text{lex}_1$  and  $\text{lex}_2$   
co-occurring within a specific relationship  $\text{rel}$
3. Collocation strength by association measure (examples):
  - » Mutual information (Church and Hanks, 1990)  
$$MI(\text{lex}_1, \text{lex}_2) = \log_2 p(\text{lex}_1, \text{lex}_2) / p(\text{lex}_1) * p(\text{lex}_2)$$
  - » Likelihood ratio (Dunning, 1993)  
$$LLH(\text{lex}_1, \text{rel}, \text{lex}_2) = 2 \sum_{ij} (O_{ij} * \log(O_{ij}/E_{ij}))$$

# Contingency Table for Cooccurrence

	$\text{lex}_2$	$\neg\text{lex}_2$
$\text{lex}_1$	$O_{11} = f(\text{lex}_1, \text{rel}, \text{lex}_2)$ $E_{11} = (O_{11} + O_{12}) * (O_{11} + O_{21}) / N$	$O_{12} = f(\text{lex}_1, \text{rel}, \neg\text{lex}_2)$ $E_{12} = (O_{11} + O_{12}) * (O_{12} + O_{22}) / N$
$\neg\text{lex}_1$	$O_{21} = f(\neg\text{lex}_1, \text{rel}, \text{lex}_2)$ $E_{21} = (O_{11} + O_{21}) * (O_{21} + O_{22}) / N$	$O_{11} = f(\neg\text{lex}_1, \text{rel}, \neg\text{lex}_2)$ $E_{11} = (O_{12} + O_{22}) * (O_{21} + O_{22}) / N$

# Collocation Examples

---

- Basis: **BNC parses**, lexicalised statistical CFG (Carroll and Rooth, 1998; Schulte im Walde, 1998)
- Examples, based on frequencies for relationships:

<i>lex<sub>1</sub></i>	<i>lex<sub>2</sub></i>	relationship
<i>feel</i>	<i>better</i>	adverb
<i>feel</i>	<i>anger, desire, need</i>	direct object
<i>give</i>	<i>up</i>	particle
<i>give</i>	<i>advice, birth, impression, rise, smile</i>	direct object
<i>take</i>	<i>long</i>	adverb with <i>to</i> <sub>INF</sub>
<i>take</i>	<i>advantage, breath, hour, lead, part</i>	direct object

# Collocation Examples: *Sketch Engine*

take BNC freq = 173412

<b>object</b>	<b>106749</b>	<b>5.5</b>	<b>subject</b>	<b>37279</b>	<b>3.7</b>	<b>modifier</b>	<b>18780</b>	<b>1.9</b>	<b>and/or</b>	<b>1440</b>	<b>0.1</b>
place	<u>11603</u>	60.7	change	<u>422</u>	20.65	aback	<u>266</u>	65.01	give	<u>153</u>	25.28
advantage	<u>2084</u>	48.3	event	<u>225</u>	19.0	longer	<u>434</u>	61.02	come	<u>88</u>	21.96
step	<u>2287</u>	47.24	discussion	<u>145</u>	18.91	seriously	<u>578</u>	57.7	smile	<u>20</u>	21.36
precedence	<u>215</u>	41.81	journey	<u>89</u>	17.91	away	<u>1192</u>	53.9	pause	<u>12</u>	19.87
care	<u>2389</u>	41.18	profitboss	<u>16</u>	17.52	on board	<u>157</u>	50.92	go	<u>77</u>	18.55
precaution	<u>308</u>	40.17	ambulance	<u>38</u>	17.1	long	<u>391</u>	48.41	hesitate	<u>9</u>	18.24
sip	<u>180</u>	39.71	government	<u>396</u>	16.47	just	<u>983</u>	40.95	relax	<u>10</u>	18.01
action	<u>2368</u>	38.4	meeting	<u>187</u>	16.2	then	<u>879</u>	40.88	arrest	<u>14</u>	17.71
piss	<u>124</u>	38.3	ceremony	<u>42</u>	15.73	only	<u>830</u>	39.94	nod	<u>12</u>	17.53
toll	<u>306</u>	38.16	patient	<u>161</u>	15.05	kindly	<u>87</u>	39.35	sigh	<u>8</u>	16.48
look	<u>1158</u>	37.73	surprise	<u>49</u>	14.98	also	<u>956</u>	37.27	try	<u>31</u>	16.35
breath	<u>733</u>	37.06	student	<u>153</u>	14.77	up to	<u>137</u>	36.54	stoop	<u>5</u>	16.33

# Multiword Expressions (MWEs)

---

- Multiword expressions: any phrase that is not entirely predictable on the basis of standard rules and lexica
- **Anomalous collocations:** *by and large, on show*
- **Formulae:** *alive and well, as good as gold*
- **Metaphors:** *pack one's bags, kick the bucket*
- **Collocations:** *rancid butter, too ... to ...*

# Qualities of Multiword Expressions

---

- **conventionalisation**: accepted over time
- **fixedness**: rigid, preferred lexical realisation
- **non-compositionality**: semantically opaque
- **syntactic irregularity**: irregular structures
- **non-identifiability**: not predictable from surface form
- **situatedness**: associated with a fixed pragmatic point
- **figuration**: metaphor, metonymy, etc.
- **proverbiality**: explain recurrent situation of interest
- **informality**: associated with informal registers
- **affect**: encodes evaluation

# Applications of Lexical Statistics

---

- Lexicography
- Parsing
- Language models in speech recognition
- Disambiguation cues for ambiguous words  
→ machine translation
- Information retrieval
- Spelling correction ...

# Word Distributions: References (1)

---

- Kenneth W. Church and Patrick Hanks (1990): "Word association norms, mutual information, and lexicography". *Computational Linguistics*, 16(1):22-29.
- Ted Dunning (1993): "Accurate methods for the statistics of surprise and coincidence". *Computational Linguistics*, 19(1):61-74.
- Stefan Evert (2004): "The statistics of word cooccurrences: word pairs and collocations". PhD thesis, Institut für Maschinelle Sprachverarbeitung, Universität Stuttgart.

# Word Distributions: References (2)

---

- Oliver Christ, Bruno M. Schulze, Anja Hofmann, Esther König (1999): “The IMS corpus workbench: Corpus Query Processor”. Technical report, Institut für Maschinelle Sprachverarbeitung, Universität Stuttgart.
- Adam Kilgarriff, Pavel Rychly, Pavel Smrz, and David Tugwell (2004): “The Sketch Engine”. In *Proceedings of the 11th EURALEX International Congress*. Lorient, France.

Collocations and multiword expressions online:

<http://www.collocations.de/>

<http://www.ims.uni-stuttgart.de/projekte/CQPDemos/cqpdemo.html>

<http://www.sketchengine.co.uk/>

<http://mwe.stanford.edu/>