Computational Linguistics

Part-of-Speech Tagging

Suhaila Saee & Bali Ranaivo-Malançon

Faculty of Computer Science and Information Technology
Universiti Malaysia Sarawak

August 2014

Part Of Speech (POS)

- aka **word class**, **grammatical category**, or **lexical category**
- A category of words based on their grammatical function

POS Tagging

The automatic process of **choosing** the **best POS tag** for each word in a text, and **guessing** the tags of **unknown words**.

- The name of the tool is **POS tagger**

Example:
`Tagged sentence:`
Must/md-hl Berlin/np-hl remain/vb-hl divided/vbn-hl ?/.-hl ?/.-hl

Tagset

A list of word classes

- Different from traditional grammar word classes (only 8 classes)
- **Size**
  - **Large** tagset: fine distinction, low accuracy
  - **Small** tagset: high accuracy
  - **Very small** tagset: less information
- **Consistency**
  - Words with the same meaning and function should be tagged with the same tag

Example of Penn Treebank Tagset

1.	CC	Coordinating conjunction
2.	CD	Cardinal number
3.	DT	Determiner
4.	EX	Existential *there*
5.	FW	Foreign word
6.	IN	Preposition or subordinating conjunction
7.	JJ	Adjective
8.	JJR	Adjective, comparative
9.	JJS	Adjective, superlative
10.	LS	List item marker

(SOURCE: MARCUS ET. AL., 1993)

One Word, Multiple Tags

Example:

The word *back* can be:

- `Adjective` as in "The back door"
- `Common noun` as in "On my back"
- `Adverb` as in "Win the voters back"
- `Verb` as in "Promised to back the bill"

Multiple Words

- One lexical word can be made with multiple tokens
- Some tagsets allow multiple words to be treated as a single word by adding numbers to each tag

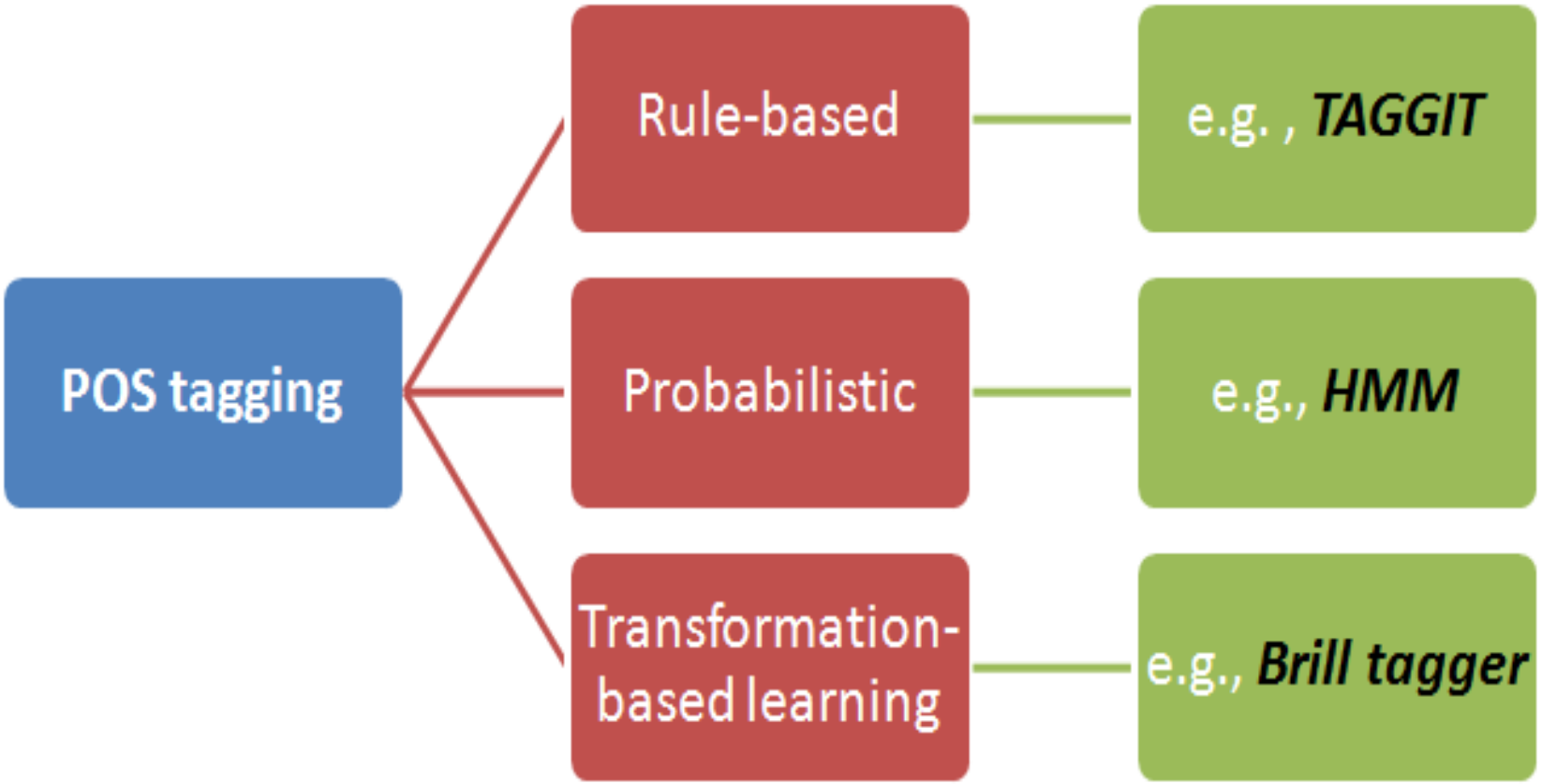
Example:
In *C7* tagset:
in/II31 spite/II32 of/II33

Multipart Words

- A word can be composed with multiple units
- Some tagged corpora split certain words

Example:
In *Penn Treebank*:
would/MD n't/RB
children/NNS 's/POS

POS Tagging Methods



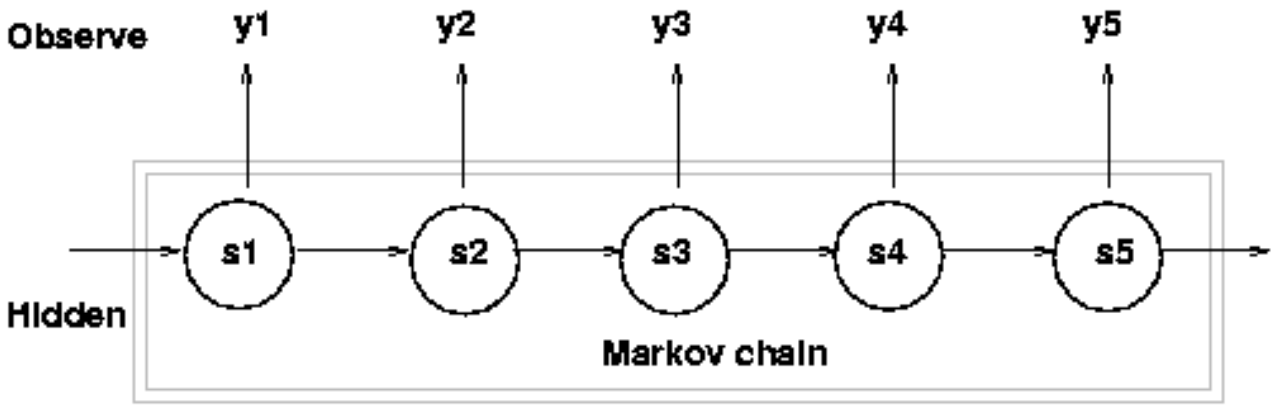
TAGGIT (Green and Rubin, 1971)

- To tag the *Brown* corpus
- 2 steps:
  1. **Initial tag selection** - Identify all possible tags for a token
  2. **Tag disambiguation** - Choose the most appropriate tag
- Uses 86 basic tags
- Accuracy: 77%

Markov Models

- Alternatives for laborious and time-consuming manual tagging
- Extract linguistic knowledge automatically from large corpora
- A Markov model is a finite state automaton (collection of states connected by transitions)
- Each state has 2 probability distribution:
  1. Probability of emitting a symbol
  2. Probability of moving to a particular state (probabilistic transition function)
- From one state, the Markov model emits a symbol and then moves to another state
- The simplest Markov model is the *Markov chain* ▸ Markov chain

Markov Models (cont'd)



(SOURCE: JIA LI, 2006)

Hidden Markov Model (HMM) Tagging

- HMM: a mathematical model of a stochastic process ($\Rightarrow$ Stochastic tagging)
- HMM tagger can be trained from **untagged corpus**
- But, a **lexicon** describing the possible tags for tokens is required
- OBSERVED: tokens in the sentence
- HIDDEN: POS tags
- DECODING: to uncover the most likely state sequence (sequence of POS tags) that could have generated the observation sequence (sequence of tokens)

Transformation-based Learning (TBL)

- Transforms one state to another using transformation rules
- GOAL: To find the suitable tag for each token
- INPUT: corpus
- OUTPUT: An ordered sequence of transformations
- Components:
  1. Initial annotator
  2. Transformations
  3. Objective function

Brill Tagger

- Brill's tagging is an instance of TBL
- Like rule-based taggers, it is **based on rules** which determine when an ambiguous word should have a given tag
- Like stochastic taggers, it **has a machine–learning component**: the rules are automatically induced from a previously tagged training corpus

Brill Tagger Algorithm

❶ Initialisation

- **Known words**: assigning the most frequent tag associated to a form of the word
- **Unknown words**:
  - Proper noun if capitalised; otherwise, simple noun
  - Learning or guessing rules with lexical rules on the same basis as contextual rules

❷ Learning phase

- Iteratively compute the error score of each candidate rule (difference between the number of errors before and after applying the rule)
- Select the best (higher score) rule
- Add it to the rule set and apply it to the text
- Repeat until no rule has a score above a given threshold, until applying new rules leaves the text in the same state

(Modified from *Wikipedia*: "Brill tagger")

Other Tagging Methods

- Neural networks tagging (Schmid, 1994)
- Constraint relaxation and nearest neighbour tagging (Daelemans et al., 1996)
- Decision trees tagging (Schmid, 1997)
- Maximum entropy tagging (Ratnaparkhi, 1996)

Accuracy

- The most used performance measure for POS taggers
- Degree of correctness

$$Accuracy = \frac{correctly\ tagged\ tokens}{total\ tokens}$$

- Tagger's result is compared to an annotated reference corpus (= **Gold Standard**)
- Correct tagging requires
  - Tagger and reference corpus must use the same word segmentation convention
  - Tagger and reference corpus must use the same tagset
- Most current tagging algorithms have an accuracy of around 96-97% for simple tagsets like the *Penn Treebank* tagset
- These accuracies are for words and punctuation
- Accuracy for words only would be lower

What May Influence the Accuracy Measure?

- **Amount of training data** - More ⇒ Better
- **Size and quality of the tagset** - Large tagset ⇒ More potential ambiguity, tagging becomes harder
- **Difference between training corpus and dictionary on the one hand and the corpus of application on the other** - "If training and application text are drawn from the same source (e.g., the same time period of a particular newspaper), then accuracy will be high"
- **Occurrence of unknown words in the test data** - Many unknown words ⇒ poor performance

Other metrics

- t_i = the set of tags assigned to the i^{th} word w_i by a tagger
- r_i = the set of tags assigned to same word in the reference corpus

$$Recall_{w_i} = \frac{|t_i \cap r_i|}{|r_i|}$$

$$Precision_{w_i} = \frac{|t_i \cap r_i|}{|t_i|}$$

$$F - measure_{w_i} = \frac{1}{\frac{\alpha}{Precision} + \frac{(1-\alpha)}{Recall}}$$

Performance Analysis

- We should interpret an accuracy score relative to a lower-bound **baseline** and an upper-bound **ceiling**
- The choice of baseline is somewhat arbitrary, but it usually corresponds to minimal knowledge about the domain
- For POS tagging, the baseline can be a unigram tagger
- **Human ceiling** - How often do human annotators agree on the same tag?

Error Analysis

- Generate a **confusion matrix** (for development data):
  How often was *tag i* mistagged as *tag j*:

	IN	JJ	NN	NNP	RB	VBD	VBN
IN	-	.2			.7		
JJ	.2	-	**3.3**	2.1	1.7	.2	**2.7**
NN		**8.7**	-				.2
NNP	.2	**3.3**	**4.1**	-	.2		
RB	**2.2**	2.0	.5		-		
VBD		.3	.5			-	**4.4**
VBN		**2.8**				**2.6**	-

- The row labels indicate correct tags
- The column labels indicate the hypothesized tags by the tagger
- Each cell indicates percentage of the overall tagging error
  - e.g., 8.7% of the total errors were caused by mistagging JJ as NN
- Errors that are causing problems:

Error Analysis (cont'd)

- Noun (NN) vs ProperNoun (NNP) vs Adj (JJ)
- Preterite (VBD) vs Participle (VBN) vs Adjective (JJ)

Tool: Brill tagger

- As mentioned in the previous slides, Brill tagger is an open-source program
- For further information, click to ▸ Brill tagger
- Online demo of Brill tagger is available ▸ demo

Tool: Stanford POS Tagger

- The tagger is implemented in Java
- GNU under licensed
- For further information on the tagger, go to Standford tagger

References

Mitchell P. Marcus, Santorini B., & Mary Ann Marcinkiewicz. Building a Large Annotated Corpus of English: The Penn Treebank. Journal of Computational Linguistics. 19(2):313–330, 1993.

J.Li. (2006). Hidden Markov Model. Retrieved from HMM

Schmid H. (1994). Part-Of-Speech Tagging With Neural Networks. In *Proceedings of 15th International Conference on Computational Linguistics*. COLING'94.

Schmid H. (1997). *Probabilistic Part-of-Speech Tagging Using Decision Trees.* New Methods in Language Processing. London: Studies in Computational Linguistics. UCL Press.

Daelmans, W., Zavrel, J., Berck, P. & Gillis, S. (1996). MTB: A Memory-based Part-of-Speech Generator. In *Proceedings of 4th Workshop on Very Large Corpora*, Conpenhagen.

Ratnaparkhi, A. (1996). A Maximum Entropy Model for Part-Of-Speech Tagging.