Computational Linguistics

Tokenisation

Suhaila Saee & Bali Ranaivo-Malançon

Faculty of Computer Science and Information Technology
Universiti Malaysia Sarawak

August 2014

What is a Word?

- An **orthographic word** is a written sequence of characters
- **Phonological word** or **prosody word** is a unit of speech which behaves as a unit for certain kinds of phonological processes, especially stress or accent

EXAMPLE:
I've in *I've done it* is a single phonological word

What is a Word? (cont)

- A **lexeme** or **lexical item** is an abstract unit of the lexicon; it has a specific sound and a specific meaning
- A **lemma** or **citation form** is a set of lexical forms having the same stem, the same major part of speech, and the same word-sense

EXAMPLE:
LEMMA: *run, runs, ran, running*
LEXEME: *RUN*

What is a Word? (cont)

- A **content word** or an **open-class word** or a **lexical word** is a word, such as a noun, verb, or adjective, and to which an independent meaning can be assigned
- A **function word** or a **grammatical word** is a word with little lexical meaning or has ambiguous meaning
- Shortened forms of a word: e.g. abbreviations (Dept.), logograms ($)

How about Word Processing?

- Processing the word into a required output
- The required outputs are:
  - a token
  - an analysis structure of word
  - a labelled word with its word class

What is a Tokenisation?

A process that dividing a sequence of characters (letters + digits + punctuations + any other symbols) into tokens

- **Tokens** are running words
- **Types** are distinct words
- The tool is called **tokeniser**
- The process is different for space delimited languages and non-segmented languages
  - In **space delimited** languages, words are delimited by whitespaces and punctuations
    - Examples: English, French, Malay (*Bahasa Rumi*)
  - In **non-segmented** languages, tokens do not have explicit boundaries
    - Examples: Arabic, Chinese, Japanese, Thai, Balinese

Why do you need tokenisation?

- Text processing applications such as parsers, stemmers, POS taggers etc. operate on **words** and **sentences**
- Texts in a raw form are sequences of characters without *information* about *word* and *sentence boundaries*
- Text needs to be segmented into *word* and *sentences* before any further processing can be done
- This process is called *tokenisation*

Some issues

1. ***Periods***: not all periods are punctuation. They serve as:
   - markers for abbreviations. For instance, *etc.*, *U.A.E.*
   - ordinal numbers. For instance, German date expression: *1. Oktober 2014*
2. ***Other punctuation***: sentence markers "!" or "?" are unambiguous with company name *Yahoo!*
3. ***Ordinal numbers***: In German, ordinal numbers are written with a trailing period after the number. For instance, *17th* is written *17.*

Common practice in period disambiguation

- Disambiguation of periods: the most difficult problem for the tokenization of alphabetic languages
- The periods indicate:
  1. the end of a sentence
  2. an abbreviation
  3. an ordinal number
  4. an abbreviation at the end of a sentence
  5. an ordinal number at the end of a sentence

Common practice in period disambiguation (cont'd)

The approaches to handle the disambiguation of periods are:

1. Disambiguation Heuristics:
   - Character sequences like "Mrs.", "St." or "Eds." are abbreviations
   - If strings of this form are always classified as sentence-internal abbreviations, the accuracy of the sentence boundary detection is raised
2. Classification Approach:
   - Period disambiguation known *classification problem*
   - The contextual information available for disambiguation is turned into a feature vector
   - A classifier is trained on manually disambiguated training data
   - The classifier learns to assign the correct class (abbreviation, full stop or abbreviation+full stop) based on the feature vector

Common practice in tokenization of ideographic languages

- The tokenization of texts from languages with ideographic writing systems: Chinese, Japanese and Korean is difficult because there are no blanks to indicate the word boundaries
- A Chinese text consists of a sequence of characters which is only interrupted by punctuation

Common practice in tokenization of ideographic languages (cont'd)

Two approaches to solve the problem are:

1. Rule-based systems:
   - identify potential words which are listed in a dictionary
   - Overlapping candidate words represent ambiguities which are resolved by rules
   - The disambiguation rules are developed by hand and make use of linguistic principles, statistical information, and heuristics
   - Unknown words are recognized with another set of rules

Common practice in tokenization of ideographic languages (cont'd)

2. Statistical systems:
   - define tokenization as the task of finding the most probable word sequence which yields the character sequence

Which existing tool is recommended?

- OpenNLP tokenizers tokenise an input character sequence into tokens
- Three tokenizer implementations offers by OpenNLP:
  - Whitespace Tokenizer
  - Simple Tokenizer
  - Learnable Tokenizer
- For more details, go to ▸ OpenNLP tokenizer

Which existing tool is recommended?

- Stanford Natural Language Processing Group provides open source softwares
- The softwares are statistical NLP toolkits for various major computational linguistics problems
- All the program is written in Java
- A tokenizer for English text can be accessed from ▸ Standford tokenizer

Which existing tool is recommended?

- NLTK offers a collection of Python programs, modules, data set and tutorial to support research and development in Natural Language Processing (NLP)
- nltk.tokenize module in NLTK has the same features with other tokenizer tools in the previous slides
- For more details, go to the links below:
  - http://streamhacker.com/
  - http://text-processing.com/
  - http://www.nltk.org/book/
  - https://github.com/japerk/nltk-trainer
- For demo on NLTK tokeniser module: demo

How to evaluate tokenisation?

No standard evaluation for tokenisation as long as the text tokenised into token then, the tokenisation process can be considered done