

## COMP348

# Tokenisation and Sentence Segmentation

## Today's agenda

- The Goal
- Challenges in Tokenisation and Text Segmentation
- Tokenisation
- Sentence Segmentation
  
- Reading: see Natural Language Toolkit

## The Goal

- Want to break down the input into manageable chunks such as words
  - Needs to be done as a first step for use in other applications (e.g. for search, translation)
  - The first application we'll be looking at is text classification (e.g. spam filtering), starting week 4 (and first assignment)
  - These "manageable chunks" are what will be the building blocks of approaches to decide whether text fits into one category or another (e.g. spam vs non-spam)
- Same process as identifying separate units in programming languages, but harder

## Language Dependence

- Several types of writing symbols:
  - Alphabetic: individual symbols represent sounds
  - Syllabic: individual symbols represent syllables
  - Logographic: individual symbols represent words/concepts
- The writing system of a language typically employs symbols of several types:
  - English is mainly alphabetic, but it also uses logographic symbols: 0, 1, 2, \$, &, etc.
- Check out <http://www.omniglot.com/>

# Language Dependence

- Example: Japanese

Alphabetic

Syllabic

Logographic

Document: Done

# Language Dependence

- Different languages may use different conventions to mark word or sentence boundaries:

	Word Boundary	No Word Boundary
Sentence Boundary	Amharic (Ethiopia)	Chinese, Japanese
No Sentence Boundary		Thai

- Word and sentence boundaries are not completely unambiguous in English

# Character Encoding

- Different languages may use different character sets:
  - Latin-1 (ISO 8859-1) for English, Spanish, ...
  - TIS620 for Thai
- Sometimes a “romanisation” of a language involves adding special characters to account for accents: u”ber, ueber, de’ja`, de1ja2
- Some character sets are multiple-byte systems: Chinese, Japanese
- Some languages even have alternative character sets:
  - GB or Big-5 for Chinese
- Sometimes different character sets are used in the same text:
  - Korean or Japanese newswire texts

# Definition of Word and Sentence

- There is no absolute definition of the ‘correct’ word or sentence segmentation:
  - Alternatives for word segmentation:
    - “I’m”
    - “I” - “’m” - “m”
    - “I” - “am”
  - “Governor’s”
  - “Governor” - “’s”
- Alternatives for sentence segmentation:
  - “she arrived, John said.”
  - “she arrived,” - “John said.”

# Corpus Dependence

---

- Punctuation depends on text genre
  - newswire text, closed-captioning data, technical reports
  - differing conventions
- Misspellings; erratic punctuation and spacing
  - email text
  - SMS
- People do not follow the conventions
- Many corpora have additional errors from optical character recognition (OCR) or handwriting recognition
- An algorithm that performs very well in a corpus may not be successful on another corpus

# Today's agenda

---

- Challenges in Tokenisation and Text Segmentation
- **Tokenisation**
- Sentence Segmentation

# Tokenisation

---

- Tokenisation differs greatly between:
  - Space-delimited languages: some word boundaries are delimited by spaces
  - Unsegmented languages: there are no word boundaries
- Tokenisation depends on the writing system
- Tokenisation also depends on the morphology of the words:
  - isolating: words do not divide into smaller units ("morphemes") - Chinese
  - agglutinating: there are clear boundaries between the morphemes - Japanese
  - inflectional: the boundaries between morphemes are not clear - Latin

# Tokenisation in Space-Delimited Languages

---

- Tokenisation ambiguity:
  - The period can be part of the token ("\$.9", "Corp."), or not (end of sentence), or both (abbreviation at end of sentence)
  - The apostrophe can mark a genitive ("John's") or contraction ("can't")
- Tokenisation decisions:
  - "76 cents a share", "a 76-cents-a-share dividend"
  - "\$4 million", "4 million dollars", "\$4,000,000"


# Tokenisation in Space-Delimited Languages

---

- Token: “A sequence of characters preceded and followed by space”
- But:
  - Punctuation marks are usually attached to the word:
    - At the end: “similarly,” “tokens.” “markup)”
    - At the beginning: “(such”
  - Some special character sequences must be considered during tokenisation:
    - mark-up characters (SGML, XML, etc.)

# Tokenisation in Space-Delimited Languages

---

- Abbreviations:
  - The period is part of the token (“Dr.”, “St.”, “Corp.”)
  - If end of sentence, the period is also a sentence delimiter  
“I went to High St. It was a very busy street.”  
 Interdependence with sentence segmentation
- Typical approach:
  - Keep a list of abbreviations  
But this wouldn't account for new or unknown abbreviations

# Tokenisation in Space-Delimited Languages

---

- Quotation marks:
  - may open a passage
  - may close a passage
  - may indicate an apostrophe
  - may indicate an accent in some languages (u”ber, rivie`re)
- Apostrophes:
  - may mark a genitive (though some languages do not use apostrophes for a genitive: German, English emails)
  - may mark a contraction (though some languages do not use apostrophes for a contraction: Spanish)

# Tokenisation in Space-Delimited Languages

---

- Multi-part Words
  - In agglutinating languages, several meaningful tokens may appear together
  - It is quite common to find compound tokens in many languages:  
Example: German
    - N–N            Lebensversicherung
    - Adv–N        Nichtraucher
    - Prep–N       Nachkriegszeit

# Tokenisation in Space-Delimited Languages

---

- Hyphens
  - English uses hyphens to create:
    - single-token words “end-of-line”
    - multi-token words “Boston-based”
  - Hyphen usage varies between dialects of English
  - Some languages use hyphens to create grammatical structures that need to be expanded during tokenisation
    - “va-t-il”, “c’est-à-dire”, “celui-ci”
  - Hyphens are commonly used to break a word at the end of line
    - but a hyphenated word could use its hyphen to break it up

# Tokenisation in Space-Delimited Languages

---

- Multi-word Expressions
  - “in spite of” is equivalent to “despite”
  - Numerical Expressions
  - Dates
  - Other expressions:
    - “76 cents a share”, “\$3-a-share” – should they have the same number of tokens?
- Context may play a role in defining the tokenisation criteria:
  - “I saw **no one**”
  - “**No one** man can do it alone”

# Exercises

---

- What are the tokens in the following texts?
  - Pierre Vinken , 61 years old , will join the board as a nonexecutive director Nov. 29 . Mr. Vinken is chairman of Elsevier N.V. , the Dutch publishing group .
  - Head-driven generation methods combine both top-down search and bottom-up combination, in an ideal way. <REF>Shieber et al. 1990</REF> proposed to define the `head' constituent h of phrase with category x on semantic grounds: the semantic representations of h and x are identical.

# Tokenisation in Unsegmented Languages

---

- Tokenisation is by nature more difficult in unsegmented languages
- The approach depends on the writing system and orthography of the language
- A more informed approach is needed:
  - Keep a word list (but what about unknown words?)
- There is not always agreement about the “correct” segmentation

# Tokenisation in Unsegmented Languages

---

- **Simplest solution:**
    - Consider each character a distinct word
      - But some languages (like Japanese) use syllabic symbols to express inflections
  - **Greedy algorithm:**
    - Keep a word list, and try to match the longest word in the list
    - Unmatched symbols represent new words (two variants)
      - every unmatched symbol represents an unknown word
      - a sequence of unmatched symbols represents an unknown word
- thetabledownthere*
- theta bled own there
  - the table down there (reverse maximum matching)

# Tokenisation in Unsegmented Languages

---

- **Chinese:** Each character represents both a single lexical morpheme (unit of semantic information) as well as a syllable
  - **Statistical Approaches:**  
Determine which characters are most likely to form words
  - **Lexical Approaches:**  
Use manually encoded features about the language:
    - syntactic and semantic information
    - common phrasal structures
    - morphological rules
  - **Hybrid Approaches:**  
Combine information from statistical and lexical sources

# Tokenisation in Unsegmented Languages

---

- **Japanese:**
  - The writing system combines alphabetic (Romaji), syllabic (Hiragana and Katakana), and logographic (Kanji) symbols, plus arabic and punctuation symbols
  - Multiple character sets help tokenisation:
    - a boundary between character sets is a good candidate of token boundary
    - BUT a Japanese word may be a combination of character sets (kanji+kana)

# Today's agenda

---

- Challenges in Tokenisation and Text Segmentation
- Tokenisation
- Sentence Segmentation



# Contextual Factors for Sentence Segmentation

---

- **Case Distinctions:** Languages with consistent use of upper-case and lower-case letters can use them to assist segmentation
- **Part of Speech:** Even a list of the *possible* parts of speech of the words in the context helps
- **Word Length**
- **Lexical Endings:** Can be used to filter words that are not likely to be abbreviations
- **Prefixes:** Can determine the part of speech of the words surrounding the punctuation mark
- **Abbreviation Classes:** Some abbreviations (such as titles) are not likely to occur at the end of sentence

# Segmentation Approaches

---

- **Rule-Based Approaches:**
  - The traditional approach uses regular expressions plus word lists and exception lists
  - Hand-crafted rules for the particular corpus
  - The results depend on the consistency in the corpus
- **Trainable Algorithms:**
  - Use of statistical analysis of annotated corpora
  - More robust and portable than rule-based approaches
- **Hybrid Approaches:**
  - Use rules for the regular cases, and statistical information for the difficult cases
  - Accuracy is higher than pure rule-based or pure trainable algorithms (up to 99.5%)

# Segmentation Tools

---

- Segmentation is necessary, but not very glamorous
- Not too many off-the-shelf tools around
- NLTK (Natural Language Toolkit) is a set of Python modules for DIY segmentation (and other things)
  - <http://nltk.sourceforge.net/index.html>

# NLTK

---

- This is the Natural Language Toolkit, a Python package
- The currently supported version is a basic one called NLTK Lite
- It contains a lot of text-processing modules (for things you'll learn about later in the unit) already written
- It also includes some basic tokenisers

# NLTK

---

- Here's an example:

```
>>> from nltk.tokenize import WordTokenizer
>>> tokenizer = WordTokenizer()
>>> tokenizer.tokenize("In the world's first animated
reality series")
['In', 'the', 'world', 's', 'first', 'animated', 'reality',
'series']
```

- It's basically just a front-end for string and r.e. functions
  - It will come in more useful later

# NLTK

---

- Here's another tokenizer:

```
>>> text = "Would you be more comfortable if I broke your
arm in 3 places then wrapped it in $100 bills?"
>>> pattern = r'''(?x)
... \w+ # sequences of 'word' characters
... | \${?}\d+(\.\d+)? # currency amounts, e.g. $12.50
... | ([A-Z]\.)+ # abbreviations, e.g. U.S.A.
... | [^\w\s]+ # sequences of punctuation
... '''
>>> nltk.tokenize.regexp_tokenize(text, pattern)

['Would', 'you', 'be', 'more', 'comfortable', 'if', 'I',
'broke', 'your', 'arm', 'in', '3', 'places', 'then',
'wrapped', 'it', 'in', '$100', 'bills', '?']
```

# NLTK

---

- Tokenisation is built in to the read() function for nltk\_lite.corpora

# What's Next

---

- Morphological Analysis