Tokenization and sentence segmentation

course based on Mitkov [2003, Chap.10]

UPPSALA
UNIVERSITET
Marie Dubremetz
marie.dubremetz@lingfil.uu.se

Uppsala, April 2014

Presentation Plan

1. Tokenization
2. Sentence segmentation

Why ?

Why do we need to segmentize a text in NLP ?

Because a text is not a long string of characters :

- Words (that we must deal with lexicons)
- Sentences (that separate unit of sense)
- Paragraphs...

if you are not convinced...

Because, on a screen, when you read that :

Bob eats apples.

Your computer sees that :

01000010011011110110001000100000011001010110000
10111010001110011001000000110000101110000011100
00011011000110010101110011001011 10

Why ?

Why do we need to segmentize a text in NLP ?

Because a text is not a long string of characters :

- Words (that we must deal with lexicons)
- Sentences (that separate unit of sense)
- Paragraphs...

if you are not convinced...

Because, on a screen, when you read that :

Bob eats apples.

Your computer sees that :

01000010011011110110001000100000011001010110000
10111010001110011001000000110000101110000011100
00011011000110010101110011001011 10

Definition

Tokenization ?

Segmentation of a text into linguistic units such as words, punctuations and numbers.

For example : Bob eats apples.
Contains 4 tokens : Bob|eats|apples|.

Let's get things bad

Tokenize : a simple task ?

"11 :45 p.m. Ugh. First day of New Year has been day of horror. Can't quite believe I am once again starting the year in a single bed in my mom's house. [...]When I got to the Alconburys' and rang their entire-tune-of-town-hall-clock-style doorbell I was still in a strange world of my own–nauseous, vile-headed, acidic. "

So...

Should we rely only on spaces ?

Let's get things bad

Tokenize : a simple task ?

"11 :45 p.m. Ugh. First day of New Year has been day of horror. Can't quite believe I am once again starting the year in a single bed in my mom's house. [...]When I got to the Alconburys' and rang their entire-tune-of-town-hall-clock-style doorbell I was still in a strange world of my own–nauseous, vile-headed, acidic. "

So...

Should we rely only on spaces ?

Hyphens

True hyphens :

Lexical hyphen :

- "anti-discriminatory"
- "twenty-two"

Sententially determined hyphenation :

- "their entire-tune-of-town-hall-clock-style doorbell"
- "New York-based"

End-of-line hyphens :

“May I ask you to translate into Swedish the word : "Supercalifra-gilisticexpialidocious" ?”

There is not a unique way to deal with hyphens

- Context dependant : twenty-two vs New York-based
- Task dependent : for instance New York-based cannot be tokenized the same way if we do Part-of-Speech, Information Retrieval, translation...
- Language dependent problem

Specific tokens

The specific tokens :

- C++
- Email addresses (cmpt456@sfu.ca)
- Phone number (778-782-3054)
- City names (San Francisco, New York)
- date 23/05/14, 23-May-2014
- ...

Require the use of processors

Processor

Specialised tokenizer that handle, thanks to rules and list of vocabulary, a specific tokenization or normalisation.

Apostrophe

Mr. O'Neill thinks that the boys' stories about Chile's capital aren't amusing.

The rules for apostrophes are :

- Language specific
- Changing with the context :
  "O'Neill" vs "Chile"+"'s"

Apostrophe

How is it dealt in softwares ?

Here is an example of how some apostrophes are handled in a well known tagger :

```
# character sequences which have to be cut off at the end of a word
my $FClitic;

if (defined($option_english)) {
    # English
    $FClitic = '\'(s|re|ve|d|m|em|ll)|n\'t';
}
```

Figure: Regex for clitics in TreeTagger's tokenizer

Tokenization Conclusion

What you should know

Tokenization is :

- a basic task that almost every NLP task requires
- for Western European languages relatively well handled (and specific problems through other languages : germanic, oriental etc.)

What experience will tell :

- Tokenization is endless task (as many in NLP)
- Copy : lot of software products integrate tokenizers, so get inspired from them
- Performance is directly influenced by this task

Sentence Segmentation Introduction

Sentence Segmentation

Sentence segmentation consists in finding sentence boundaries between words in different sentences. Since most written languages have punctuation marks which occur at sentence boundaries, sentence segmentation is frequently referred to as **sentence boundary detection**, **sentence boundary disambiguation**, or **sentence boundary recognition**. All these terms refer to the same task : determining how a text should be divided into sentences for further processing.

Why sentence segmentation ?

Elementary task

- useful task for : syntactical parsing, information extraction, machine translation, text alignment
- but not so simple :

The basic algorithm [. ? !][()"]+[A-Z]
(which means “select every punctuation followed by a space and an upper-case.”) provokes 6.5% errors on the Brown corpus and WSJ

Why ?

Why sentence segmentation ?

Elementary task

- useful task for : syntactical parsing, information extraction, machine translation, text alignment
- but not so simple :

The basic algorithm [. ? !][()"]+[A-Z]
(which means “select every punctuation followed by a space and an upper-case.”) provokes 6.5% errors on the Brown corpus and WSJ

Why ?

Where is the problem ?

Why is it not so simple ?

Mostly because the point is a ambiguous sign

- Decimal ex. 6.22
- Abreviations (your result will directly be influenced by the number of them)
- Named entity : Web sites, ip address (www.lesshadoks.com, 94.23.225.162 :3128, etc.)

Solutions

Type of sentence boundary disambiguation (SBD)

Some machine learning system exist.

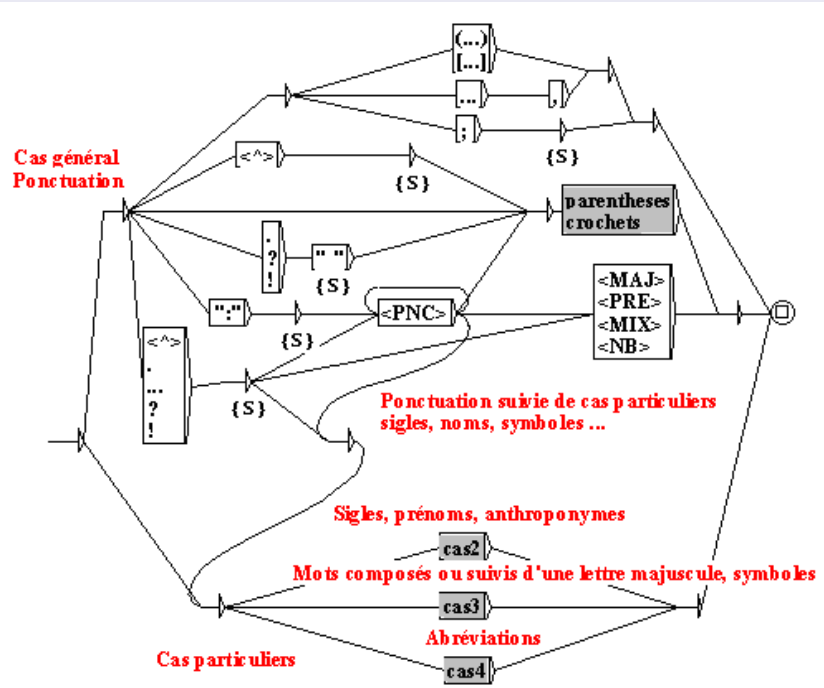
- Unsupervised they can be trained on any language and genre.
- Supervised they require annotation of a training set. We will concentrate on rule based methods

Rule based systems

Which rules ?

First you need to determine some rules that take the local context of the punctuation into account :

- regular expressions [. ? !][()"]+[A-Z]
- or finite states transducers
  http://www-igm.univ-mlv.fr/~unitex/



Rule based systems

Which rules ?

If it is not sufficient you need :

- A list of frequent starting words (The, He, However ...)
- A list of abbreviations (Mr. , O.N.U. ect.)

To finish

There is no universal SDB

As well as for tokens there is not a universal way to deal with sentences :

- Language specific
- Genre specific

References

Ruslan Mitkov. *The Oxford Handbook of Computational Linguistics*, volume 0 of *Oxford Handbooks in Linguistics*. Oxford University Press, 2003. URL http://books.google.com/books?hl=en&lr=&id=0aClhre-vW4C&pgis=1.