Tokenization and sentence segmentation

course based on Mitkov [2003, Chap.10]

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

Uppsala, April 2016

Presentation Plan

1. **Tokenization**
   - Problems
   - Conclusion
   - Practical Exercise 1
   - Practical Exercise 2
   - Practical Exercise 3
2. **Sentence segmentation**
   - Problems
   - Conclusion
   - Practical Exercise 1
   - Practical Exercise 2
3. **Summary**

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

Table of Contents

1. **Tokenization**
   - Problems
   - Conclusion
   - Practical Exercise 1
   - Practical Exercise 2
   - Practical Exercise 3
2. **Sentence segmentation**
   - Problems
   - Conclusion
   - Practical Exercise 1
   - Practical Exercise 2
3. **Summary**

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|.

QUIZ

Question

Can you tell me how many tokens contains this extract (count everything that is bold):

The wolf said: "Little pig, little pig, let me come in."

1. 11
2. 15
3. 17
4. 19

Table of Contents

1. **Tokenization**
   - Problems
   - Conclusion
   - Practical Exercise 1
   - Practical Exercise 2
   - Practical Exercise 3
2. **Sentence segmentation**
   - Problems
   - Conclusion
   - Practical Exercise 1
   - Practical Exercise 2
3. **Summary**

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

Table of Contents

1. **Tokenization**
   - Problems
   - Conclusion
   - Practical Exercise 1
   - Practical Exercise 2
   - Practical Exercise 3
2. **Sentence segmentation**
   - Problems
   - Conclusion
   - Practical Exercise 1
   - Practical Exercise 2
3. **Summary**

Conclusion on Tokenization

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

Table of Contents

1. **Tokenization**
   - Problems
   - Conclusion
   - Practical Exercise 1
   - Practical Exercise 2
   - Practical Exercise 3
2. **Sentence segmentation**
   - Problems
   - Conclusion
   - Practical Exercise 1
   - Practical Exercise 2
3. **Summary**

Tests on Tokenization

In the following slides I present you some data. Can you tell if this is tokenized or not?

Is this text tokenized?

The wolf said : " Little pig , let me come in . "

Is this text tokenized?

The wolf said : " Little pig , let me come in . "

Note

tokenization separators for machine translation purpose

Is this text tokenized?

The wolf said: "Little pig, let me come in."

Is this text tokenized?

<tok>The</tok> <tok>wolf</tok> <tok>said</tok>
<tok>:</tok> <tok>"</tok> <tok>Little</tok>
<tok>pig</tok> <tok>,</tok> <tok>little</tok>
<tok>pig</tok> <tok>,</tok> <tok>let</tok>
<tok>me</tok> <tok>come</tok> <tok>in</tok>
<tok>.</tok> <tok>"</tok>

Is this text tokenized?

<tok>The</tok> <tok>wolf</tok> <tok>said</tok>
<tok>:</tok> <tok>"</tok> <tok>Little</tok>
<tok>pig</tok> <tok>,</tok> <tok>little</tok>
<tok>pig</tok> <tok>,</tok> <tok>let</tok>
<tok>me</tok> <tok>come</tok> <tok>in</tok>
<tok>.</tok> <tok>"</tok>

Note

tokenization separators for XML document

Is this text tokenized?

The
wolf
said
:
"
Little
pig
,
let
me
come
in
.
"

Is this text tokenized?

The
wolf
said
:
"
Little
pig
,
let
me
come
in
.
"

Note

tokenization separators for tagging purpose

Exercise 1 Conclusion

If someone ask you to tokenize a text (boss, teacher etc.) either ask/find an example or check for what purpose it is made. This will influence the type of separator that you must use.

Table of Contents

1. **Tokenization**
   - Problems
   - Conclusion
   - Practical Exercise 1
   - Practical Exercise 2
   - Practical Exercise 3
2. **Sentence segmentation**
   - Problems
   - Conclusion
   - Practical Exercise 1
   - Practical Exercise 2
3. **Summary**

Methodology for Your Reports

Mr Folks asked his students to choose between two tokenizers and justify why. Three student's answers are below. None of them is perfect. Can you give a positive feedback and tell how to improve each of them?

1. Kim says: Well tokenizer one systematically tokenizes wrong the punctuations (for instance it gives the token 'cheese!' instead of the tokens 'cheese' and '!') whereas tokenizer two only gets wrong on tokenizing some hyphens which is a less recurrent mistake. So I choose tokenizer two
2. Julia says: tokenizer one is the worse because it divides the text into 1150 tokens whereas tokenizer two divides into 1100 and manually checked the text has 1000 tokens. 1100 is closer to 1000 so I choose tokenizer two.
3. Hans answers: "When I run diff command I get *'896v721,42467*
   ⟩ *apple.*
   ⟨ *apple*
   ⟨.
   '

   So I choose tokenizer two

Methodology for Your Reports

To improve your reports, what can you do?

- Give a global overview of the results (%, numbers)
- Exemplify enough to make your point.
- Give a consistent information presentation to the reader (ex. *Write all examples in italic, put data into tables for comparisons etc.*)
- Don't waste your time: target your answer to the question.
- Make sure the information you give has a meaning and is useful out of its context.

Table of Contents

1. **Tokenization**
   - Problems
   - Conclusion
   - Practical Exercise 1
   - Practical Exercise 2
   - Practical Exercise 3
2. **Sentence segmentation**
   - Problems
   - Conclusion
   - Practical Exercise 1
   - Practical Exercise 2
3. **Summary**

Task Defining Method

Advice

You cannot write a program for a task you don't understand yourself!
The following exercise is going to give you some guidance on what to do before writing your program.

Task Defining Method

Mr. Smith now asks you to tokenize a text and compare to a gold standard.

- Take a simple example, non-ambiguous: ***I eat an apple.***
- Create the gold standard tokenization of this example.

Task Defining Method

Mr. Smith now asks you to tokenize a text and compare to a gold standard.

- Take a simple example, non-ambiguous: ***I eat an apple.***
- Create the gold standard tokenization of this example.

Task Defining Method

Now, Mr.Smith gives you examples of (dum) electronic tokenization.

a *I|eat|an|apple.*
b *I|eat|a|n|apple.*
c *I|eat|a|n|a|p|p|l|e.*
d *I|eat an apple.*

- All of them are wrong, but your common sense tells you that some are worse that others.
- Manually count and calculate data that are useful to describe this situation (make meaningful comparisons with percentages and absolute numbers!)

Note

For the lab you will have to practice that but on a big scale. Since you cannot count by hand you will have to find the commands that help you generating those data!

Task Defining Method

Now, Mr.Smith gives you examples of (dum) electronic tokenization.

a *I|eat|an|apple.*
b *I|eat|a|n|apple.*
c *I|eat|a|n|a|p|p|l|e.*
d *I|eat an apple.*

- All of them are wrong, but your common sense tells you that some are worse that others.
- Manually count and calculate data that are useful to describe this situation (make meaningful comparisons with percentages and absolute numbers!)

Note

For the lab you will have to practice that but on a big scale. Since you cannot count by hand you will have to find the commands that help you generating those data!

Task Defining Method

Note

What we did was computing **Recall** and **Precision**
Recall: number of tokens that are right / number of tokens to be produced according to the gold standard. (e.g., *Recall = 2/5 for example 'c'*)
Precision: number of tokens that are right / number of tokens that were automatically generated. (e.g., *Precision = 2/9 for example 'c'*)

How to Compute Recall Precision

Gold standard: I|eat|an|apple|.

1. *I|eat|a|n|a|p|p|l|e.*
2. *I|eat an apple.*

1. Recall=2/5=40% Precision=2/(2+7)=2/9=22%
2. Recall=1/5=20% Precision=1/(1+1)=1/2=50%

Numbers are colored according to what they correspond to in the examples.
And if you want to go beyond: search for the 'F-measure' also called 'F1 Score'

Task Defining Method

Some Methodological Help

1. Take a pen, a paper make the most simple example(s) (e.g., *I eat an apple*)
2. Define what you have to do on it (e.g., *Tokenise*)
3. Make by hand your answer for this example(s) (e.g., *I|eat|an|apple|.*)
4. Now that it is clear, make the program that do the task.
5. Test your program on your example, before applying to the real data
6. If the output is correct, you can apply safely to your data.

Note

In labs steps 1, 2, 3 can always be done in group. (And often steps 4, 5, 6 as well, but not always, so make it sure with the teacher).

Table of Contents

1. Tokenization
   - Problems
   - Conclusion
   - Practical Exercise 1
   - Practical Exercise 2
   - Practical Exercise 3
2. **Sentence segmentation**
   - Problems
   - Conclusion
   - Practical Exercise 1
   - Practical Exercise 2
3. Summary

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?

Table of Contents

1. Tokenization
   - Problems
   - Conclusion
   - Practical Exercise 1
   - Practical Exercise 2
   - Practical Exercise 3
2. **Sentence segmentation**
   - Problems
   - Conclusion
   - Practical Exercise 1
   - Practical Exercise 2
3. Summary

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.mywebsite.com, 94.23.225.162:3128, etc.)

Solutions

Type of sentence boundary disambiguation (SBD)

Some machine learning system exist.

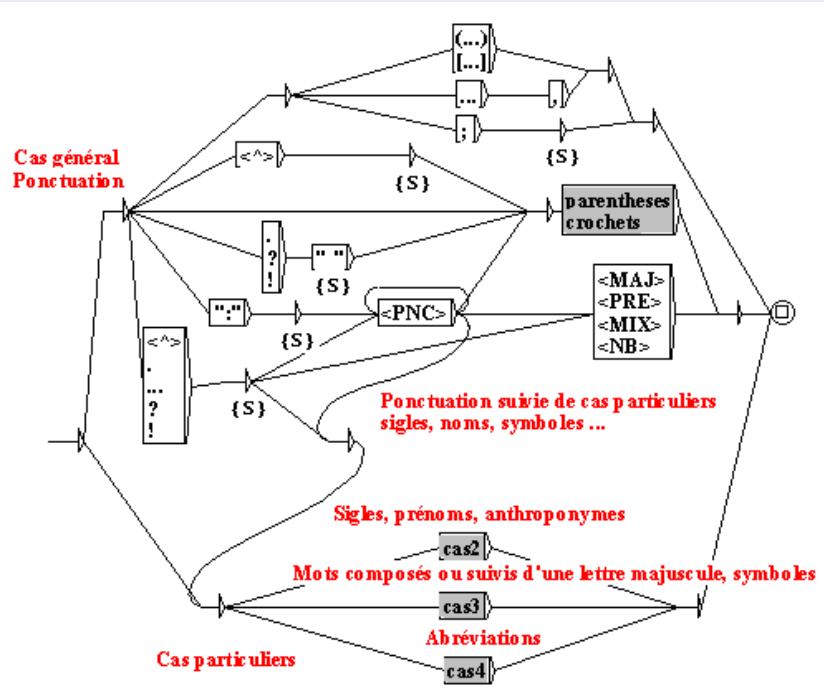
- Unsupervised they can be trained on any language and genre.
- Supervised they require annotation of a training set. We will concentrate on rule based methods

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. , Av., a.m., A.S.A.P*)

Table of Contents

1. Tokenization
   - Problems
   - Conclusion
   - Practical Exercise 1
   - Practical Exercise 2
   - Practical Exercise 3
2. **Sentence segmentation**
   - Problems
   - Conclusion
   - Practical Exercise 1
   - Practical Exercise 2
3. Summary

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

Table of Contents

1. Tokenization
   - Problems
   - Conclusion
   - Practical Exercise 1
   - Practical Exercise 2
   - Practical Exercise 3
2. **Sentence segmentation**
   - Problems
   - Conclusion
   - Practical Exercise 1
   - Practical Exercise 2
3. Summary

Is this Sentence Split?

Welcome! Can I help you? I'm fine.

Is this Sentence Split?

Welcome!
Can I help you?
I'm fine.

Is this Sentence Split?

Welcome
!
Can
I
help
you
?
I
'm
fine
.

Is this Sentence Split?

Welcome
!

Can
I
help
you
?

I
'm
fine
.

Is this Sentence Split?

<sent><tok>Welcome</tok><tok> !
</tok></sent><sent><tok> Can </tok> <tok> I
</tok><tok> help </tok><tok> you
</tok><tok>?</tok></sent>

Exercise Conclusion

A text can be sentence split and tokenized, but the separators must be different.
http://morphadorner.northwestern.edu/sentencesplitter/example/

Table of Contents

1. Tokenization
   - Problems
   - Conclusion
   - Practical Exercise 1
   - Practical Exercise 2
   - Practical Exercise 3
2. **Sentence segmentation**
   - Problems
   - Conclusion
   - Practical Exercise 1
   - Practical Exercise 2
3. Summary

Evaluate a Sentence Splitter

By groups take a computer and go to this site:
http://morphadorner.northwestern.edu/sentencesplitter/example/
(or google 'morphadorner sentence splitter online')

Are you able to find a text that induces errors in the sentence splitting?

Table of Contents

1. Tokenization
   - Problems
   - Conclusion
   - Practical Exercise 1
   - Practical Exercise 2
   - Practical Exercise 3
2. Sentence segmentation
   - Problems
   - Conclusion
   - Practical Exercise 1
   - Practical Exercise 2
3. **Summary**

Learned Notions

Today we saw:

- Two NLP preprocessing:
  - Tokenization
  - Sentence segmentation
- They vary with genre, NLP task to perform, and, of course, language
- Some ambiguities are common and need to be handle such as hyphenation and dots
- Research is done today on the impact of those preprocessing task on overall performance of later processing (Parsing for instance).Foster [2010]

References

Jennifer Foster. "cba to check the spelling": Investigating Parser Performance on Discussion Forum Posts. In *Proceedings of the NAACL-HLT*, pages 381–384. Association for Computational Linguistics, 2010. URL http://aclweb.org/anthology/N10-1060.

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.