Introduction to Human Language Technologies
1. Document structure

UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Facultat d'Informàtica de Barcelona

Outline

Outline

Document types

- Documents containing text:
  - Structured documents (e.g., web pages being tables)
  - Semi-structured documents (e.g., web pages containing pieces of plain text, figures and tables)
  - Documents with plain text only(e.g., text files, emails, tweets, oral transcripts)

Accessing to plain text contained in web pages may be relevant.

XML Parsers

- Transform an XML/HTML/XHTML document into a tree of standard objects.
- Provide an interface to manage that tree.
- Textual zones in the document can be extracted from that tree using the interface.

```
<?xml version="1.0"?>
<doc type="novel" title="The green apple">
<chapter id="1">
<p>There are lots of trees in Amsteel Hill. I remember
going there and spend all the morning climbing those
trees, trying to get as many apples as possible.</p>
<p> James always wanted to come with me but he
was too young to get climbing.</p>
...
</doc>
```

Using ElementTree.py

```
for c in root:
    lp=c.findall('p')
    for p in lp:
        print p.text
```

Outline

Goal of tokenization

- Goal: split plain text into *basic units*
- Use: IR tasks, text categorization, sentence splitting, language identification, text normalization ...
- Different *basic units* depending on the task,
  - *Naïve* tokenizations: split by blanks and punctuation marks occurring after alphanum-string.
  - Complex tokenizations: names, clitics, abbreviations, collocations...

Goal of tokenization

- Goal: split plain text into *basic units*
- Use: IR tasks, text categorization, sentence splitting, language identification, text normalization ...
- Different *basic units* depending on the task,
  - *Naïve* tokenizations: split by blanks and punctuation marks occurring after alphanum-string.
  - Complex tokenizations: names, clitics, abbreviations, collocations...

Relevant definitions:

Word N-gram: sequence of words occurring in a text

Collocation: sequence of words that frequently occur together. Ex: "break a leg", "On the one hand"

Examples of tokenization

Blanks	outer punct.	Abbr.	Clitics	Colloc.	text normalized
Of	Of	Of	Of	Of_course	Of_course
course	course	course	course		
I'll	I'll	I'll	I	I	I
			'll	'll	will
go	go	go	go	go	go
to	to	to	to	to	to
U.P.C.	U.P.C	U.P.C	U.P.C	U.P.C	Universitat...
	.	.	.	.	.
	”	”	”	”	”
”Daily,	Daily	Daily	Daily	Daily	Daily
	,	,	,	,	,
Mr.	Mr	Mr.	Mr.	Mr.	Mister
	.				
John	John	John	John	John	John_Smith
Smith...”	Smith	Smith	Smith	Smith	
	...	...	...	...	...
	”	”	”	”	”

Examples of tokenization

Blanks	outer punct.	Abbr.	Clitics	Colloc.	text normalized
Of	Of	Of	Of	Of_course	Of_course
course	course	course	course		
I'll	I'll	I'll	I	I	I
			'll	'll	will
go	go	go	go	go	go
to	to	to	to	to	to
U.P.C.	U.P.C	U.P.C	U.P.C	U.P.C	Universitat...
	.	.	.	.	.
	„	„	„	„	„
"Daily,	Daily	Daily	Daily	Daily	Daily
	,	,	,	,	,
Mr.	Mr	Mr.	Mr.	Mr.	Mister
	.				
John	John	John	John	John	John_Smith
Smith..."	Smith	Smith	Smith	Smith	
	...	...	...	...	...
	„	„	„	„	„

Examples of tokenization

Blanks	outer punct.	Abbr.	Clitics	Colloc.	text normalized
Of	Of	Of	Of	Of_course	Of_course
course	course	course	course		
I'll	I'll	I'll	I	I	I
			'll	'll	will
go	go	go	go	go	go
to	to	to	to	to	to
U.P.C.	U.P.C	U.P.C	U.P.C	U.P.C	Universitat...
	.	.	.	.	.
	,,	,,	,,	,,	,,
"Daily,	Daily	Daily	Daily	Daily	Daily
	,	,	,	,	,
Mr.	Mr	Mr.	Mr.	Mr.	Mister
	.				
John	John	John	John	John	John_Smith
Smith..."	Smith	Smith	Smith	Smith	
	...	...	...	...	...
	,,	,,	,,	,,	,,

Problems: Non-standard text? Chinese? Japanese?

Outline

Goal of sentence splitting

- Goal: Recognition of sentence boundaries in plain text (e.g., '.' '?' '!' '...').
- Language-dependent task
  Ex: German: "Mein 2. Semester kommt bald zu Ende."
  Ex: Traditional chinese?
- Domain-dependent task
  Ex: "It is expressed as (x=1)? T.add('-') : T.add(x)."
- Methods:
  - Hand-crafted rules
  - Machine learning methods
- Input:
  - *Naïve* tokenization that depends on the particular method.
  - For simplicity, we will assume *blanks+outer_puntuation*

    " I'll go to U.P.C. "Daily, Mr. John Smith..." "
    → " I 'll go to U.P.C . " Daily , Mr . John Smith ... " "

Problems of sentence splitting

Main problems:

- Abbreviations and acronyms (most difficult one)

  Ex: "I will meet with Mr. Smith to talk about it."

  Ex: "Lisa run 25 km. She ended up in N.Y."

  How to detect them?
- Ellipsis

  Ex: "There're different methods (A, B, . . .) but . . . "
- Internal quotation

  Ex: " 'Stop!' he shouted."
- Ordinal numbers (German)
- Special cases:

  Ex: " We have some variables. x stands for the weight,"

Hand-crafted rules for sentence splitting

- Specific hand-crafted rules for specific cases
  - Abbreviation classes (Lists of abbreviations)
    (month name, unit-of-measure, title, address name, . . .)

    Ex: TITLE=('Mr', 'Mrs', 'Dr', . . .)
  - Regular expressions for general cases, abbreviations, ellipsis, . . .

    Ex: / ([?!]+) / $\rightarrow$ t $\in$ s_boundary
    Ex: / (\.\.\.) [A-Z]/ $\rightarrow$ t $\in$ s_boundary
    Ex: / ([?!.]) [A-Z]/ $\rightarrow$ t $\in$ s_boundary
    Ex: / ($TITLE) \. / $\rightarrow$ t $\notin$ s_boundary
    Ex: / [A-Z] \. / $\rightarrow$ t $\notin$ s_boundary
- Problem:
  - Highly expensive adaptation to new languages
    (rules and abbreviation classes)

Supervised ML for sentence splitting

- The most frequently used (ME, SVM, Perceptron, . . . -discriminative methods-)
- Requires manually annotated corpora. Commonly,
  $e^+, e^- = ['.','!','?']$ and some preceding and following tokens
- Represents each e as a set of features, which depend on the approach, the language and the domain, although normally they tend to be binary features.
- Problem:
  - Requires very large sets of examples (tens of thousands to hundreds of thousands)

Supervised ML for sentence splitting

- Examples of features used in the state of the art

  tok-1_X: 1srt token before '.' is X

  tok-2_X: 2nd token before '.' is X

  tok+1_X: 1st token after '.' is X

  len_tok-1_X: length of 1st token before '.' is X

  len_tok-2_X: length of 2nd token before '.' is X

  len_tok+1_X: length of 1st token after '.' is X

  [up|lo|cap|num]_tok-1: 1st token before '.' is Upper, Lower, CAP, Numbers

  [up|lo|cap|num]_tok-2: same for 2nd token before '.'

  [up|lo|cap|num]_tok+1: same for 1st token after '.'

  class_tok-1_X: abbreviation class of 1st token before '.' is X

  ...

Supervised ML for sentence splitting

Example of annotation and binary features extraction

I 'll go to U.P.C . " Daily , Mr . John Smith ... "

e^+		e^-	
	tok-1_U.P.C		tok-1_Mr
	len_tok-1_3		len_tok-1_2
	CAP_tok-1		up_tok-1
	tok-2_to		tok-2,
	len_tok-2_2		len_tok-2_1
	lo_tok-2		class_tok-1_title
	tok+1_"		tok+1_John
	len_tok+1_1		len_tok+1_4
			up_tok+1

Unsupervised ML for sentence splitting

- Based on corpus statistics
- Easily adaptable to new languages
  - They require large unanotated training corpora
- Mainly focus on abbreviations and ellipsis
- Heuristics and statistics calculated from the training corpus to decide:
  1. Which tokens are abbreviations?
  2. When the final period of the elements is a sentence boundary?
- Example: Punkt [Kiss and Strunk, 2006]

Unsupervised ML for sentence splitting

1. **Punkt: Is token t considered an abbreviation?**

   Measured by considering the following heuristics:

   - $t' = < t, . >$ should be a collocation
   - the length of t should be short
   - t could include periods (acronyms)
   - t is not ordinary word preceeding a period most of the times. (e.g., verbs in Turkish)

Unsupervised ML for sentence splitting

1. **Punkt: Is token t considered an abbreviation?**

   Measured by considering the following heuristics:

   - $t' =< t, . >$ should be a collocation
   - the length of t should be short
   - t could include periods (acronyms)
   - t is not ordinary word preceeding a period most of the times. (e.g., verbs in Turkish)

2. **Punkt: Is the final period of abbreviation $t' =< t, . >$ considered a sentence boundary?**

   Either one of the following heuristics must be true:

   - $t'' = \mathit{following}(t')$ is a frequent sentence (from [1]) starter
   - t'' is uppercase, occurs at least once in lowercase in the training corpus but never in uppercase inside sentences (from [1])

Exercise

Explain why Punkt fails (red) or not (blue) with the following texts:

- " "Good night !", said Laura. "
- " Abbrev . is a common abbreviation of abbreviation. "
- " We are meeting with our mr . You are late! "
- " We are meeting with our Mr . However, we'll finish soon."

Demo sentence splitters:
`http://text-processing.com/demo/tokenize/`

Outline

Goal of language identification

- Can be seen as a particular classification problem.
- Given a document, d, and a set of languages, $L = \{l_1, \ldots, l_k\}$, assign l_i to d.
- Method:
  - $\hat{d} = \text{representation}(d)$
  - $M(\hat{d}) \rightarrow l_i$
- Model M can be learned from training corpus $T = \{T_i\}_{1 \ldots k}$ where $T_i = \{d_x | d_x \text{ written in } l_i\}$:
  - Supervised Machine Learning methods
  - Statistical Language models

Survey: `https://arxiv.org/pdf/1804.08186.pdf`

Language models for language identification

Method with language models:

$M = \{P^{l_i}\}_{l_i \in L}$

$P^{l_i}(\hat{d})$: probability of $\hat{d}$ to belong to l_i

$$l_i = \operatorname*{argmax}_{l \in L}(P^l(\hat{d}))$$

$P^{l_i}(\hat{d}) \approx P^{T_i}(\hat{d})$: probability of $\hat{d}$ observing data from T_i

Language models for language identification

Method with language models:

$M = \{P^{l_i}\}_{l_i \in L}$
$P^{l_i}(\hat{d})$: probability of $\hat{d}$ to belong to l_i

$$l_i = \underset{l \in L}{\operatorname{argmax}}(P^{l}(\hat{d}))$$

$P^{l_i}(\hat{d}) \approx P^{T_i}(\hat{d})$: probability of $\hat{d}$ observing data from T_i

1. Which is the representation $\hat{d}$?
2. How is $P^{T_i}(\hat{d})$ computed?

Language models for language identification

Method with language models:

$M = \{P^{l_i}\}_{l_i \in L}$
$P^{l_i}(\hat{d})$: probability of $\hat{d}$ to belong to l_i

$$l_i = \operatorname*{argmax}_{l \in L}(P^{l}(\hat{d}))$$

$P^{l_i}(\hat{d}) \approx P^{T_i}(\hat{d})$: probability of $\hat{d}$ observing data from T_i

1. Which is the representation $\hat{d}$?
2. How is $P^{T_i}(\hat{d})$ computed?

They depend on the particular type of model.

Most frequently used: unigram language models

Unigram language models for language identification

1. **Which is the representation $\hat{d}$?**

   $\hat{d} = e_1, \ldots, e_s$ being the occurrences of unigrams:

   - Words (after *Naïve* tokenization) or
   - Characters n-grams (tokenization is not required)
     - n fixed (the most frequently used) or
     - n variable (improves accuracy, lower efficiency)

Unigram language models for language identification

1. **Which is the representation $\hat{d}$?**

   $\hat{d} = e_1, \ldots, e_s$ being the occurrences of unigrams:

   - Words (after *Naïve* tokenization) or
   - Characters n-grams (tokenization is not required)
     - n fixed (the most frequently used) or
     - n variable (improves accuracy, lower efficiency)

2. **How is $P^{T_i}(\hat{d})$ computed?**

   Each e_j is independent from the rest

$$P^T(\hat{d}) = P^T(e_1, \ldots, e_s) = \prod_{j=1}^{s} P^T(e_j)$$

$$\log P^T(\hat{d}) = \sum_{j=1}^{s} \log P^T(e_j)$$

   Possible estimators of $P^T(e_j)$:

   - Maximum Likelihood Estimator (MLE)
   - Smoothing techniques.

Unigram language models for language identification

Maximum Likelihood Estimator

$$P^T(e_j) \approx P^T_{MLE}(e_j) = \frac{c_T(e_j)}{N_T}$$

$c_T(x)$: #observed occurrences of x in training corpus T

N_T: #observed occurrences of elements in training corpus T

Unigram language models for language identification

Maximum Likelihood Estimator

$$P^T(e_j) \approx P^T_{MLE}(e_j) = \frac{c_T(e_j)}{N_T}$$

$c_T(x)$: #observed occurrences of x in training corpus T

N_T: #observed occurrences of elements in training corpus T

- Problem: data sparseness. Unseen e_j causes the model to fail. MLE is unsuitable for NLP.

Unigram language models for language identification

Maximum Likelihood Estimator

$$P^T(e_j) \approx P^T_{MLE}(e_j) = \frac{c_T(e_j)}{N_T}$$

$c_T(x)$: #observed occurrences of x in training corpus T

N_T: #observed occurrences of elements in training corpus T

- Example:

  $P^{[en]}$('The doctor tell us about his quadriplegia')?

  $c_{[en]}('quadriplegia') = 0 \Longrightarrow P^{[en]}_{MLE}('quadriplegia') = 0$
  $\Longrightarrow P^{[en]}$('The doctor tell us about his quadriplegia')$= 0$!!

Unigram language models for language identification

Smoothing Techniques:

Keep some probability mass for e_j unseen in T_i

E.g., Lidstone's Law (LID)

$$P^T(e_j) \approx P^T_{LID}(e_j) = \frac{c_T(e_j) + \lambda}{N_T + \lambda B} \qquad usually, \lambda = 0,5$$

B: #bins (potentially observable unigrams)

Exercise

Supose we have a Language Identifier for English and Catalan, based on unigram language models with words and the following statistics

w_i	a	he	mail	sent	to	mordorian
English language model [en]						
$c_{[en]}(w_i)$	17.000	10.000	3.900	850	25.000	0
$N_{[en]}$=1.300.000	$B_{[en]}$=22.600					
Catalan Language model [ca]						
$c_{[ca]}(w_i)$	21.000	11.900	420	910	750	0
$N_{[ca]}$=1.100.000	$B_{[ca]}$=36.800					

- Compute $P^{[en]}$ and $P^{[ca]}$ using MLE and LID for the following texts:
  - "he"
  - "he sent a"
  - "he sent a mail"
  - "he sent a mail to a mordorian"
- What language is identified by each estimator for each of the previous texts?
- Explain the effects of the text size