Coding Text Data

Christophe Rodrigues

Resume

- Character encoding
- Text pre-treatments
- Strings comparison

Basics – character encoding

- We need to represent characters in a binary world.
- The first standard : American Standard Code for Information Interchange (ASCII, 1960)
- 128 codes on 7 bits
- 95 printable characters (enough for english but not for others languages)

 !"#$%&'()*+,-./
0123456789:;<=>?
@ABCDEFGHIJKLMNO
PQRSTUVWXYZ[\]^_
`abcdefghijklmno
pqrstuvwxyz{|}~

ASCII TABLE

Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char
0	0	*[NULL]*	32	20	*[SPACE]*	64	40	**@**	96	60	`
1	1	*[START OF HEADING]*	33	21	**!**	65	41	**A**	97	61	**a**
2	2	*[START OF TEXT]*	34	22	**"**	66	42	**B**	98	62	**b**
3	3	*[END OF TEXT]*	35	23	**#**	67	43	**C**	99	63	**c**
4	4	*[END OF TRANSMISSION]*	36	24	**$**	68	44	**D**	100	64	**d**
5	5	*[ENQUIRY]*	37	25	**%**	69	45	**E**	101	65	**e**
6	6	*[ACKNOWLEDGE]*	38	26	**&**	70	46	**F**	102	66	**f**
7	7	*[BELL]*	39	27	**'**	71	47	**G**	103	67	**g**
8	8	*[BACKSPACE]*	40	28	**(**	72	48	**H**	104	68	**h**
9	9	*[HORIZONTAL TAB]*	41	29	**)**	73	49	**I**	105	69	**i**
10	A	*[LINE FEED]*	42	2A	*****	74	4A	**J**	106	6A	**j**
11	B	*[VERTICAL TAB]*	43	2B	**+**	75	4B	**K**	107	6B	**k**
12	C	*[FORM FEED]*	44	2C	**,**	76	4C	**L**	108	6C	**l**
13	D	*[CARRIAGE RETURN]*	45	2D	**-**	77	4D	**M**	109	6D	**m**
14	E	*[SHIFT OUT]*	46	2E	**.**	78	4E	**N**	110	6E	**n**
15	F	*[SHIFT IN]*	47	2F	**/**	79	4F	**O**	111	6F	**o**
16	10	*[DATA LINK ESCAPE]*	48	30	**0**	80	50	**P**	112	70	**p**
17	11	*[DEVICE CONTROL 1]*	49	31	**1**	81	51	**Q**	113	71	**q**
18	12	*[DEVICE CONTROL 2]*	50	32	**2**	82	52	**R**	114	72	**r**
19	13	*[DEVICE CONTROL 3]*	51	33	**3**	83	53	**S**	115	73	**s**
20	14	*[DEVICE CONTROL 4]*	52	34	**4**	84	54	**T**	116	74	**t**
21	15	*[NEGATIVE ACKNOWLEDGE]*	53	35	**5**	85	55	**U**	117	75	**u**
22	16	*[SYNCHRONOUS IDLE]*	54	36	**6**	86	56	**V**	118	76	**v**
23	17	*[ENG OF TRANS. BLOCK]*	55	37	**7**	87	57	**W**	119	77	**w**
24	18	*[CANCEL]*	56	38	**8**	88	58	**X**	120	78	**x**
25	19	*[END OF MEDIUM]*	57	39	**9**	89	59	**Y**	121	79	**y**
26	1A	*[SUBSTITUTE]*	58	3A	**:**	90	5A	**Z**	122	7A	**z**
27	1B	*[ESCAPE]*	59	3B	**;**	91	5B	**[**	123	7B	**{**
28	1C	*[FILE SEPARATOR]*	60	3C	**<**	92	5C	****	124	7C	**|**
29	1D	*[GROUP SEPARATOR]*	61	3D	**=**	93	5D	**]**	125	7D	**}**
30	1E	*[RECORD SEPARATOR]*	62	3E	**>**	94	5E	**^**	126	7E	**~**
31	1F	*[UNIT SEPARATOR]*	63	3F	**?**	95	5F	**_**	127	7F	*[DEL]*

Unicode Transformation format UTF-8

Variable width character encoding

- Between 1 and 4 octets → 1,112,064 possible codes
- Nowadays UTF-8 an extension of ASCII is the most used on internet :
- In 2012 , UTF-8: 65 % and ASCII: 15 %
- In 2017 , UTF-8 : 90,5 %

Searching for a word in a text

- Searching the word « AAAL » in string:

  « AAAA AAAM AAAN AAAO AAAL »
- Time Complexity demand on string size since we need to read each characters one by one !
- Indexation : comparing integers easier than comparing sequences of characters
- With a prefix tree, time complexity is reduce to the word size (useful to implement a dictionnary).
- A -> A -> A -> A

  -> M

  -> N

  -> O

  -> L
- and for DNA submatching sequences ?

Regular expressions

- Useful to find more complex strings

  ex : all words ended by L

  * : last character is repeated 0,1 or more times

  ex : ab* covers : ab, abb, abbb, abbbb…

  a(bc)* covers : a, abc, abcbc, abcbcbc...

  ?: last character appeared 0 or 1 time.

  ex :plurals ? Covers plural and plurals

Regular expressions

[a-z] any letter between a and z

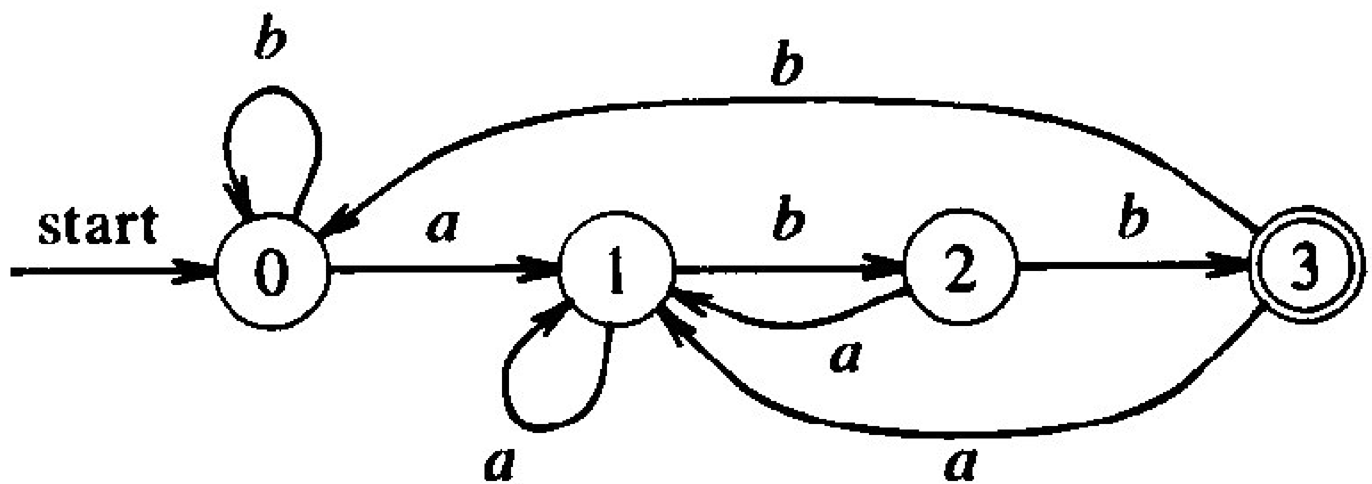
[0-9] any number between 0 and 9

| for alternatives

relat(ional|e) covers relational and relate

Regular expressions

Deterministic finite automaton can easily represent easily complex regular expressions :



abb, aaabaabbabb are covered by the automaton

Language identification

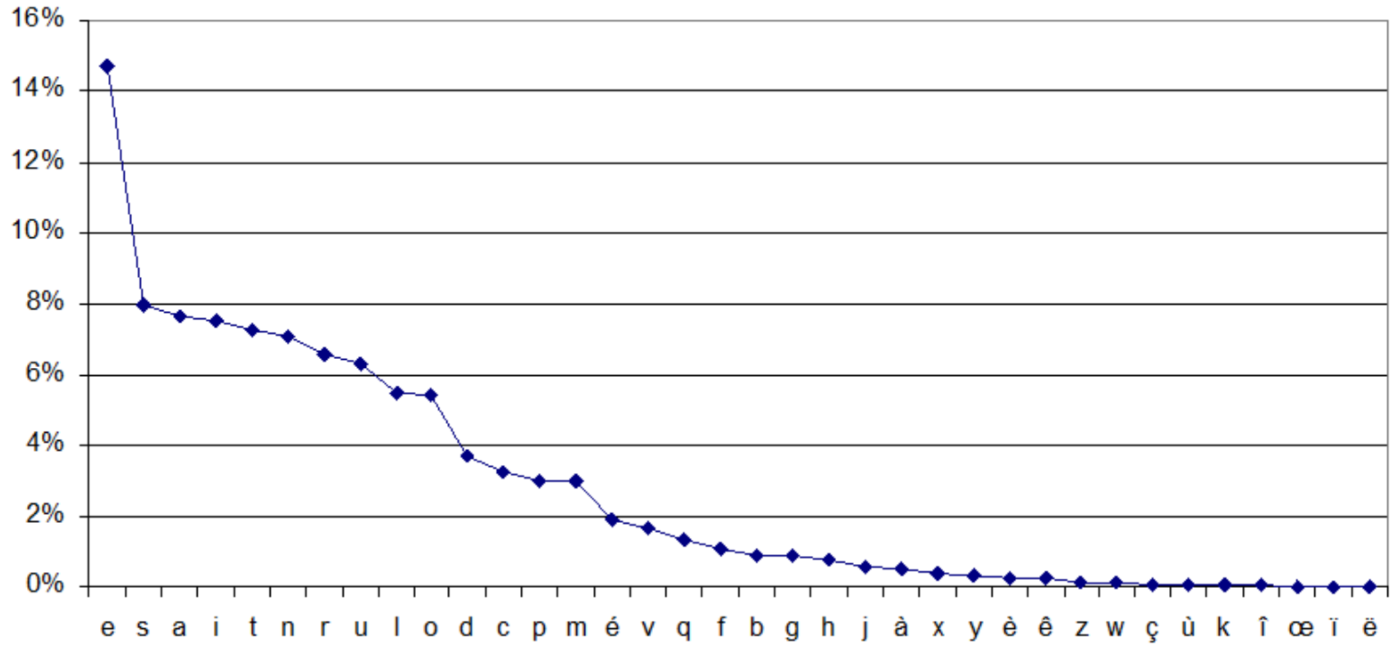
- Easy task based on frequency of letters or group of letters
- N-gram = subsenquence of n items
- The items can be phonemes, syllables, letters, words or base pairs

  example for letters :

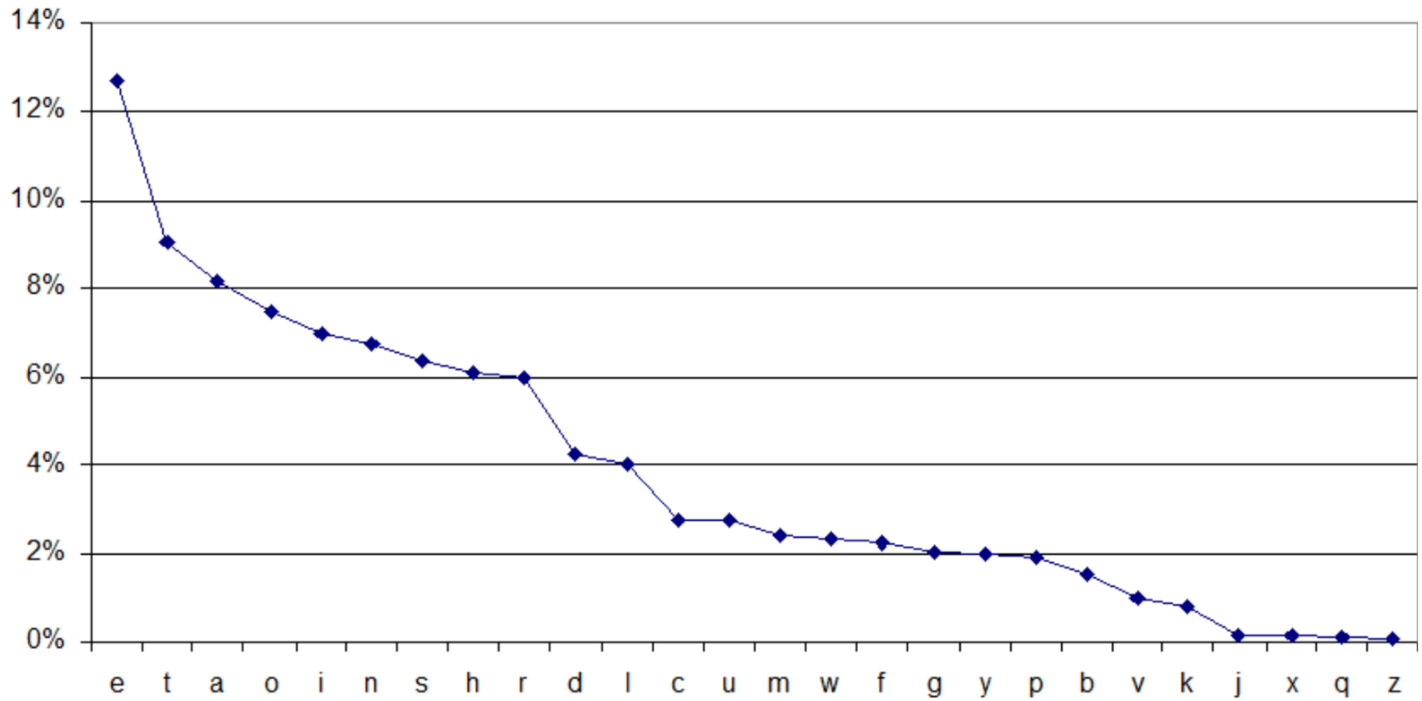
  - « a » is a unigram
  - « ab » is a bigram

Unigrams

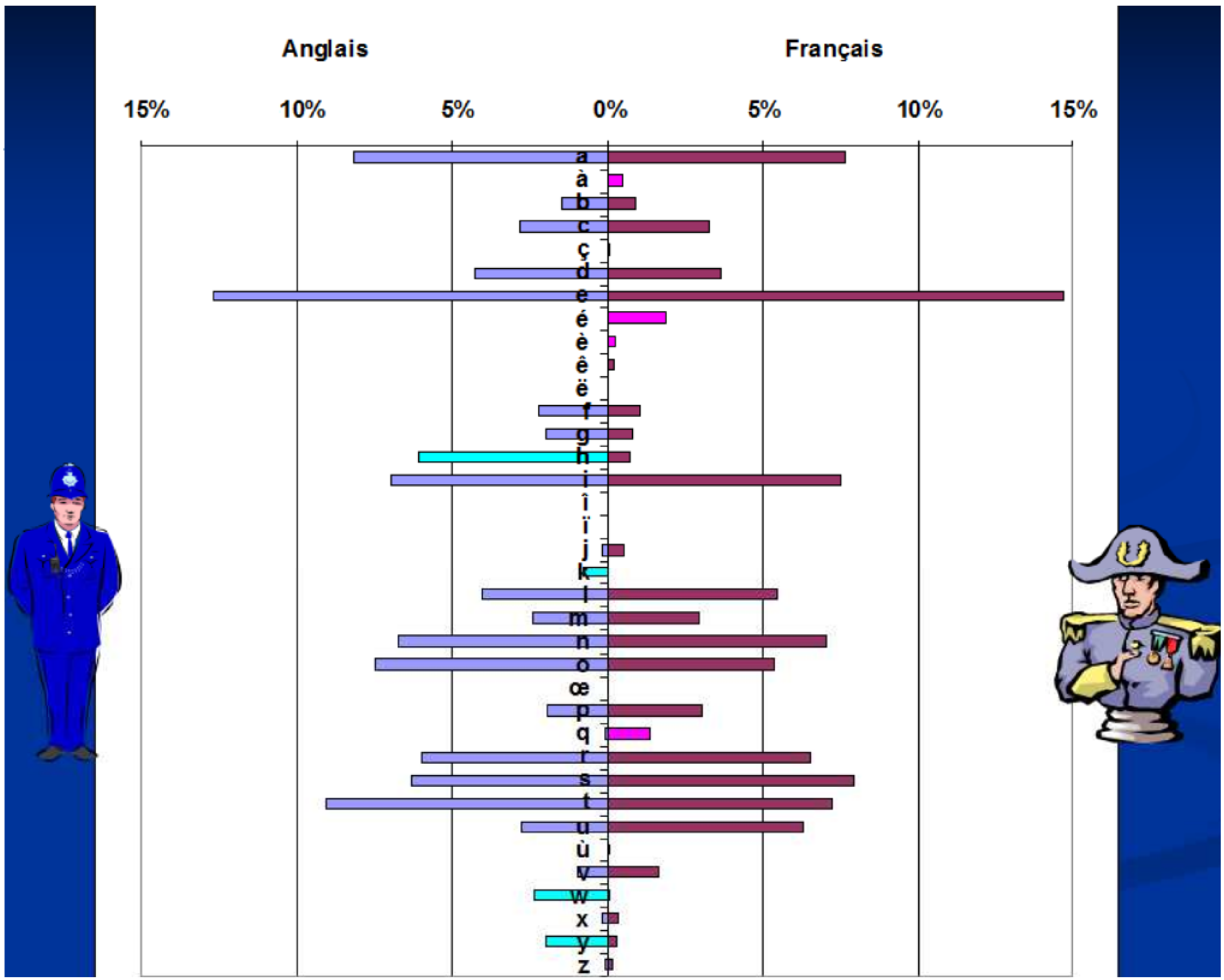
French character frequency



English character frequency



Unigram distribution



Bigrams

Français	Anglais	Allemand	Italien	Espagnol	Portugais
on	th	en	di	de	de
es	on	er	on	en	es
de	an	ch	ri	er	to
te	he	ei	er	on	da
nt	er	un	al	ci	os
re	nd	de	to	es	re
en	in	nd	ta	re	en
le	ti	ge	ne	os	er
it	al	re	in	io	te
er	re	in	re	la	ra
et	io	ie	it	ra	nt
ti	en	te	io	na	em
ou	ri	ng	de	ec	do
io	of	he	li	al	di
la	or	ne	en	ad	it
oi	at	ht	ni	da	al
ne	it	ic	tt	to	ad
me	to	be	la	nt	co
ro	ed	it	ll	ie	ei
ns	nt	sc	el	el	as

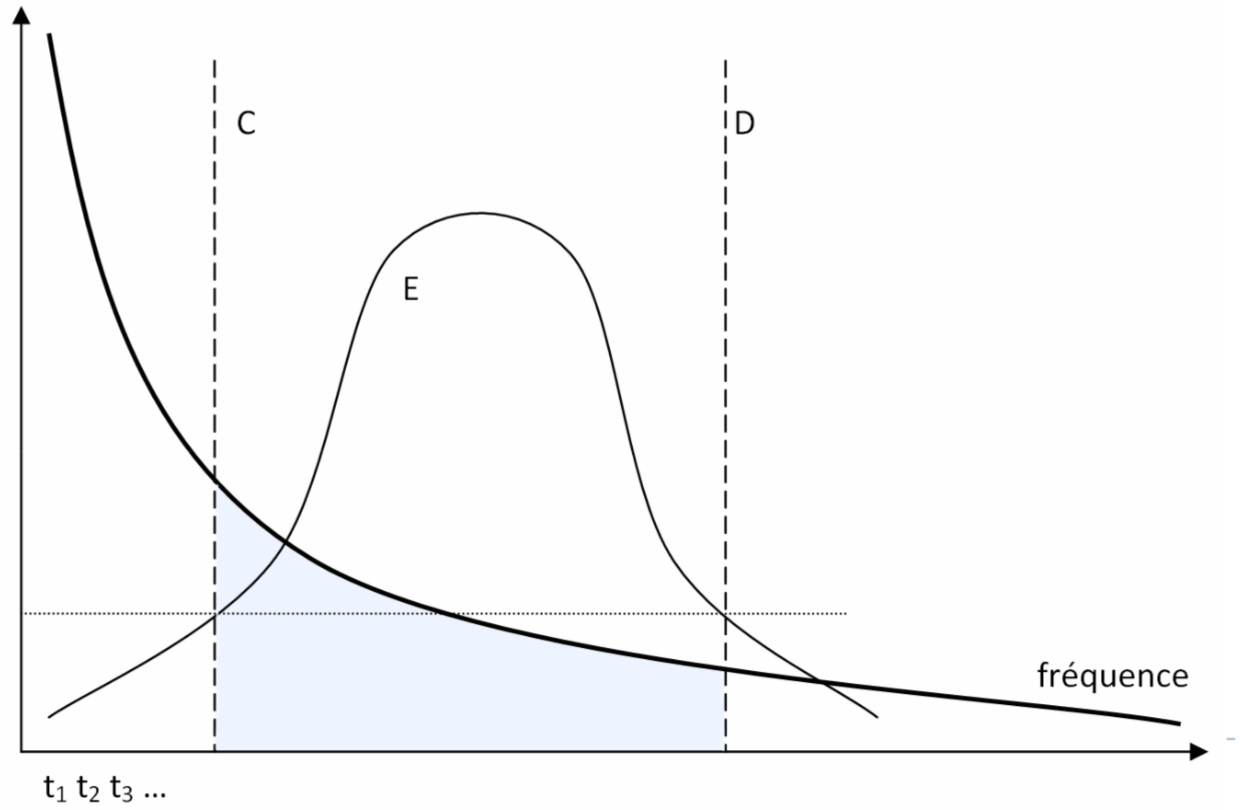
Trigrams

Français	Anglais	Allemand	Italien	Espagnol	Portugais
ion	the	der	ion	ion	ent
tio	and	und	zio	cio	ito
ent	ion	ein	ell	rec	eit
oit	tio	ung	one	ere	dir
ati	ati	cht	lla	der	ire
roi	igh	ich	rit	ien	rei
dro	ght	sch	itt	cho	ção
men	rig	che	del	ent	ade
tou	ent	ech	iri	ech	dad
con	ver	die	dir	aci	men
res	one	rec	ess	ona	nte
que	all	ine	ent	nte	dos
les	eve	eit	azi	con	ess
des	ery	gen	tto	ene	con
eme	his	ver	ere	tod	tod

For words : Zipf's law

- the frequency of any word is inversely proportional to its rank in the frequency table.
- In Ulysse from James Joyce :
  - the most frequent word occurs 8000 times
  - The tenth, 800 times
  - The hundredth, 80 times
  - The tousandth, 8 times.

consequences



Most frequents and rare words are less informatives

Text normalization

Some definitions :

- Corpus (plural corpora) : a computer-readable collection of text or speech.
- Lemma : a set of lexical forms having the same stem, the same major part-of-speech, and the same word sense.
- Types : number of distinct words in a corpus.
- Tokens : total number of running words.

Text normalization

1. Segmenting/tokenizing words from running text
2. Normalizing word formats
3. Segmenting sentences in running text.

Tokenization

Segmenting running text into words.

Splitting on white spaces is insufficient(and maybe incorrect).

Example :

They aren’t listening.

splitted in :

They | aren’t | listening

or :

They | aren | t | listening

or :

They | are | n’t | listening

Tokenization

- Based on regular expressions and sometimes with external ressources to detect numeric values (ex : phone number, price) but also names (ex : New York).
- An error on a tokenization on a sms corpora showed that the number 3 was much more used by womens than mans. Why? Any idea?

Lemmatization

Is determining that two words have the same root, despite their surface differences.

Example :

- am, are and is have the shared lemma be.
- Plurals.

Morphological analysis can be done by stemming with rewriting rules :

Example :

- ATIONAL → ATE (relational → relate)
- ING → _ (doing → do)
- SSES → SS (grasses → grass)

Sentence segmentation

- Mainly based on ponctuation.
- But insufficient with abbreviations.

  Example : Mr.
- Can been done with a dictionnary and machine learning.

Strings comparison

- Useful in spelling correction
- Example : kitten to sitten
- Method minimum edit distance :

  the minimum number of insertion, deletion or substitution on string1 to obtain string2.

  Distance between kitten and sitten is 1 (substitution of k by s)

```
I N T E * N T I O N
| | | | | | | | | |
* E X E C U T I O N
d s s   i s
```