Extracción de información

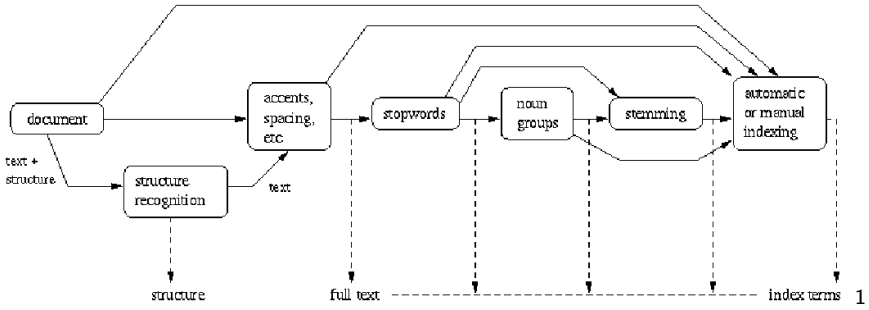
Tecnologías de búsqueda en la web

Marcelo Mendoza

Departamento de Informática
Universidad Técnica Federico Santa María

Preprocesamiento del texto

Preprocesamiento de texto



[1] Ref.: R. Baeza & B. Ribeiro, Modern Information Retrieval, 1999.

Stopwords (inglés)

	Stopwords
A	a, about, again, all, almost, also, although, always, among, an, and, another, any, are, as, at
B	be, because, been, before, being, between, both, but, by
C	can, could
D	did, do, does, done, due, during
E	each, either, enough, especially, etc
F	for, found, from, further
H	had, has, have, having, here, how, however
I	i, if, in, into, is, it, its, itself
J	just
K	kg, km
M	made, mainly, make, may, mg, might, ml, mm, most, mostly, must
N	nearly, neither, no, nor
O	obtained, of, often, on, our, overall
P	perhaps, pmid
Q	quite
R	rather, really, regarding
S	seem, seen, several, should, show, showed, shown, shows, significantly, since, so, some, such
T	than, that, the, their, theirs, them, then, there, therefore, these, they, this, those, through, thus, to
U	upon, use, used, using
V	various, very
W	was, we, were, what, when, which, while, with, within, without, would

[2] http://www.pubmed.gov

Stopwords (español)

a, acá, ahí, ajena, ajenas, ajeno, ajenos, al, algo, alguna, algunas, alguno, algunos, algún, allá, allí, aquel, aquella, aquellas, aquello, aquellos, aquí, cada, cierta, ciertas, cierto, ciertos, como, cómo, con, conmigo, consigo, contigo, cualquier, cualquiera, cualquieras, cuan, cuanta, cuantas, cuánta, cuántas, cuanto, cuantos, cuán, cuánto, cuántos, de, dejar, del, demasiada, demasiadas, demasiado, demasiados, demás, el, ella, ellas, ellos, él, esa, esas, ese, esos, esta, estar, estas, este, estos, hacer, hasta, jamás, junto, juntos, la, las, lo, los, mas, más, me, menos, mía, mientras, mío, misma, mismas, mismo, mismos, mucha, muchas, muchísima, muchísimas, muchísimo, muchísimos, mucho, muchos, muy, nada, ni, ninguna, ningunas, ninguno, ningunos, no, nos, nosotras, nosotros, nuestra, nuestras, nuestro, nuestros, nunca, o, os, otra, otras, otro, otros, para, parecer, poca, pocas, poco, pocos, por, porque, que, qué, quien, quienes, quienesquiera, quienquiera, quién, si, siempre, sí, sín, Sr, Sra, Sres, Sta, suya, suyas, suyo, suyos, tal, tales, tan, tanta, tantas, tanto, tantos, te, tener, ti, toda, todas, todo, todos, tomar, tuya, tuyo, tú, un, una, unas, unos, usted, ustedes, varias, varios, vosotras, vosotros, vuestra, vuestras, vuestro, vuestros, y, yo.

Dos algoritmos de stemming: comparación

Texto de ejemplo: Such an analysis can reveal features that are not easily visible from the variations in the individual genes and can lead to a picture of expression that is more biologically transparent and accessible to interpretation

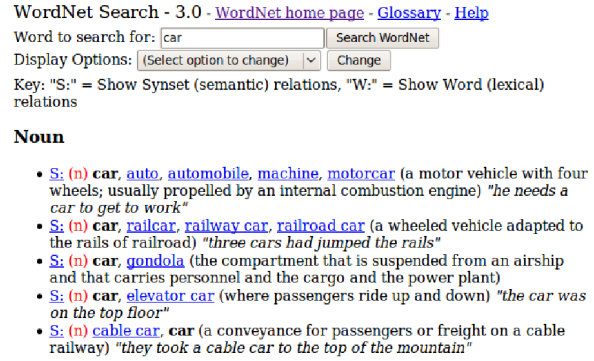
Porter: such an analysi can reveal featur that ar not easili visibl from the variat in the individu gene and can lead to a pictur of express that is more biolog transpar and access to interpret

Lancaster: such an analys can reve featur that ar not eas vis from th vari in th individu gen and can lead to a pictur of expres that is mor biolog transpar and acces to interpres

Martin Porter (1980)

http://tartarus.org/~martin/

WordNeT

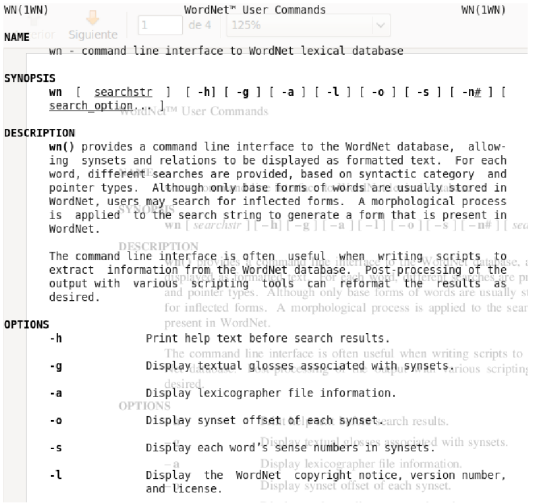

WordNet Search - 3.0 - WordNet home page - Glossary - Help
Word to search for: car Search WordNet
Display Options: (Select option to change) Change
Key: "S:" = Show Synset (semantic) relations, "W:" = Show Word (lexical) relations

Noun

- S: (n) **car**, auto, automobile, machine, motorcar (a motor vehicle with four wheels; usually propelled by an internal combustion engine) *"he needs a car to get to work"*
- S: (n) **car**, railcar, railway car, railroad car (a wheeled vehicle adapted to the rails of railroad) *"three cars had jumped the rails"*
- S: (n) **car**, gondola (the compartment that is suspended from an airship and that carries personnel and the cargo and the power plant)
- S: (n) **car**, elevator car (where passengers ride up and down) *"the car was on the top floor"*
- S: (n) cable car, **car** (a conveyance for passengers or freight on a cable railway) *"they took a cable car to the top of the mountain"*

Ver más en http://www.wordnet.princeton.edu

WordNeT



```
WN(1WN)                    WordNet™ User Commands                    WN(1WN)

NAME
       wn - command line interface to WordNet lexical database

SYNOPSIS
       wn  [  searchstr  ]  [ -h] [ -g ] [ -a ] [ -l ] [ -o ] [ -s ] [ -n# ] [
       search_option... ]

DESCRIPTION
       wn() provides a command line interface to the WordNet database,  allow-
       ing  synsets and relations to be displayed as formatted text.  For each
       word, different searches are provided, based on syntactic category  and
       pointer types.  Although only base forms of words are usually stored in
       WordNet, users may search for inflected forms.  A morphological process
       is  applied  to the search string to generate a form that is present in
       WordNet.

       The command line interface is often  useful  when  writing  scripts  to
       extract  information from the WordNet database.  Post-processing of the
       output with  various  scripting  tools  can  reformat  the  results  as
       desired.

OPTIONS
       -h              Print help text before search results.

       -g              Display textual glosses associated with synsets.

       -a              Display lexicographer file information.

       -o              Display synset offset of each synset.

       -s              Display each word's sense numbers in synsets.

       -l              Display  the  WordNet  copyright notice, version number,
                       and license.
```


Tokenización

- Input:
  amigos, Romans, habitantes. | habia una vez ... Cesar ...
- Output:
  amigo | romano | habitante | cesar ...
- Cada token es candidato a término.
- Cuáles elegimos? Depende del corpus.

Normalización

- Es necesario "normalizar" términos en texto indexado así como los términos de las consultas.
- Ejemplo: Queremos 'matching' entre *U.S.A.* y *USA*
- Implícitamente lo que estamos haciendo es definir clases de equivalencia de términos.
- Alternativa: hacer expansión asimétrica.
  - window → window, windows
  - windows → Windows, windows
  - Windows
- Mas poderosas pero menos eficientes
- Por qué no colocar *window*, *Window*, *windows*, y *Windows* en la misma clase de equivalencias?

Mayúsculas

- Reducir todo a minúsculas
- Excepciones posibles: marcas
- MIT vs. mit
- Fed vs. fed

Stop words

- stop words = palabras usadas comúnmente que tienen un bajo valor descriptivo
- Ejemplos: *e*, *y*, *o*, *en*, *de*, *pero*, *para*, *por*, *el*, *la*, *los las*, *desde*, *hasta*, ...
- Los primeros sistemas IR eliminaban las stop words del vocabulario.
- Para consultas de frases son necesarias las stop words, e.g. "Rey de España"
- Actualmente varios motores de búsqueda indexan stop words.

Stemming

- Definición de stemming: Proceso heurístico que corta la derivación de las palabras para encontrar la raíz (es un tipo de lematización).
- Es dependiente del lenguaje
- Infleccional y derivacional
- Ejemplo de derivacional: *automata*, *automatico*, *automatizado* se reduce a *automata*

Algoritmo de Porter

- Algoritmo de stemming más comúnmente usado en Inglés
- Los resultados sugieren de que es al menos tan bueno como otros algoritmos de stemming
- Convenciones + 5 fases de reducción
- Las fases son aplicadas secuencialmente
- Cada fase consiste de un conjunto de reglas.
  - Regla de ejemplo: Eliminar la derivación *ement* si el largo del prefijo es mayor que 1
  - replacement → replac
  - cement → cement
- Convención de ejemplo: Si hay varias reglas que se pueden aplicar en un mismo caso, use aquella que se aplica a un sufijo más largo.

Algoritmo de Porter: Unas pocas reglas

Regla		
SSES	→	SS
IES	→	I
SS	→	SS
S	→	

Ejemplo		
caresses	→	caress
ponies	→	poni
caress	→	caress
cats	→	cat

Mejora el stemming la efectividad de IR?

- En general, stemming mejora la efectividad en algunas consultas, y la desmejora en otras.
- Mientras más regular es la gramática, mejor.
- En castellano es difícil (gramática muy irregular).

Lematización

- Reducir formas inflecciones a su raíz
- Ejemplo: *am*, *are*, *is* $\rightarrow$ *be*
- Ejemplo: *autos*, *auto*, *automoviles* $\rightarrow$ *auto*
- Ejemplo: *Los autos de los jóvenes son de colores* $\rightarrow$ *auto joven es color*
- Para ello, usa *stemming*, pero con una ingeniosa variante: sólo reduce a la raíz si la raíz a su vez pertenece a un corpus.

Lematización

Stemming + Corpus checking!

Extracción de información en NLTK

Ejemplos simples en NLTK

Procesamiento básico Web:

```
1 > import nltk
2 > from urllib import urlopen
3 > url = "http://www.gutenberg.org/files/2554/2554.txt"
4 > raw = urlopen(url).read()
```

Tokenización y creación del objeto texto:

```
1 > tokens = nltk.word_tokenize(raw)
2 > text = nltk.Text(tokens)
```

Ahora podemos hacer NLP sobre el texto:

```
1 > text.collocations()
2 > ...
```

Ejemplos simples en NLTK

Procesamiento de HTML:

```
> url = "http://nltk.org"
> html = urlopen(url).read()
> raw = nltk.clean_html(html)
```

Repetimos el pipe anterior:

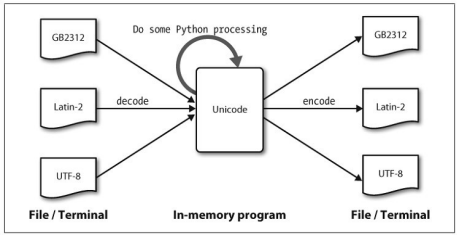
```
> tokens = nltk.word_tokenize(raw)
> text = nltk.Text(tokens)
> text.collocations()
```

Construir el vocabulario (minúsculas y sorted set):

```
> words = [w.lower() for w in text]
> vocab = sorted(set(words))
```


Unicode en NLTK

Leer con decode, procesar en Unicode, print con encode (render glyphs).



Procesamiento de Unicode (Spanish!):

```
> url = "http://www.inf.utfsm.cl"
> html = urlopen(url).read()
> raw = nltk.clean_html(hmtl)
> decoded = raw.decode('utf8')
> print decoded.encode('latin2')
```


Vocabulario en NLTK

Stemmers:

```
> porter = nltk.PorterStemmer()
> lancaster = nltk.LancasterStemmer()
> [porter.stem(t) for t in tokens]
> [lancaster.stem(t) for t in tokens]
```

Lematizador (stemmer + corpus checking):

```
> wnl = nltk.WordNetLemmatizer()
> [wnl.lemmatize(t) for t in tokens]
```


Sentencias en NLTK

Segmentador para texto raw en inglés:

```
> url = "http://www.gutenberg.org/files/2554/2554.txt"
> raw = urlopen(url).read()
> sent_tokenizer = nltk.data.load('tokenizers/punkt/english.pickle')
> sents = sent_tokenizer.tokenize(raw)
```

Entrega una lista de sentencias:

```
> len(sents)
> print sents[1].encode(latin2)
```

Puntuación en NLTK

A nivel de sentencia, podemos eliminar puntuación.
Usaremos expresiones regulares:

```
> sent = sents[4]
> puncts = ',;.'
> for sym in puncts:
...     sent = sent.replace(sym,' ')
...
> print sent
```

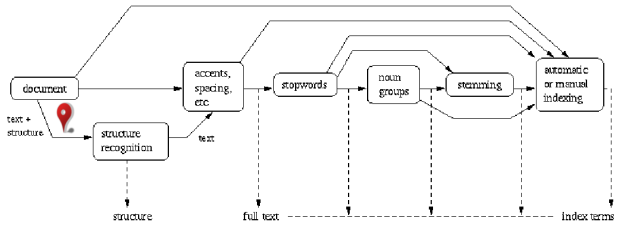

RegExp Tokenizer en NLTK

Podemos mejorar el tokenizer de NLTK, agregando expresiones regulares que queremos detectar como unigramas:

```
> pattern = r'''(?x)
...     ([A-Z]\.)+        # abreviaciones (U.S.A.)
...   | \w+(-\w+)*        # palabras con guiones
...   | \$?\d+(\.\d+)     # precios
...   | \.\.\.            # elipsis
...   | [][.,;"'?():-_]   # tokenizadores
... '''
>>> nltk.regexp_tokenize(text,pattern)
```

Es muy parecido a word_tokenize, mejora la detección de precios.

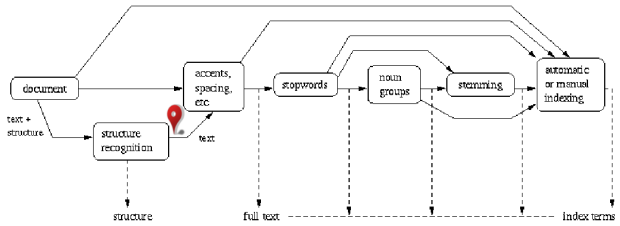
Poniendo todo junto en NLTK - Pipeline (proposal)



Web fetching:

```
> import nltk
> from urllib import urlopen
> url = "http://nltk.org"
> html = urlopen(url).read()
> raw = nltk.clean_html(html)
```

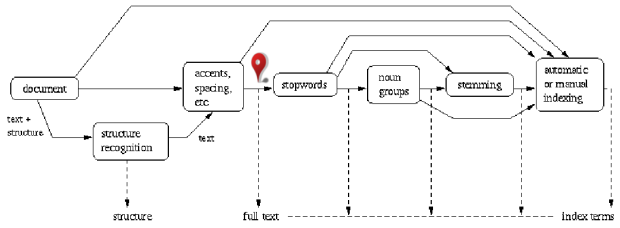

Poniendo todo junto en NLTK - Pipeline (proposal)



Sentence splitting:

```
> sent_tokenizer = nltk.data.load('tokenizers/punkt/english.pickle')
> sents = sent_tokenizer.tokenize(raw)
```

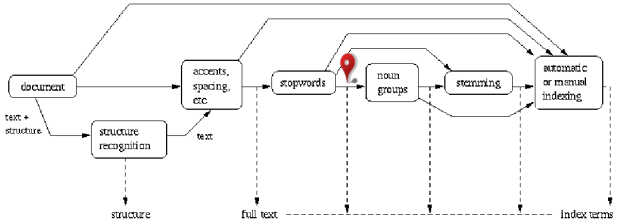
Poniendo todo junto en NLTK - Pipeline (proposal)



Tokenización de sentencias:

```
> for i in range(len(sents)):
...     for sym in puncts:
...         sents[i] = sents[i].replace(sym,' ')
...     sents[i].strip()
...     sents[i].nltk.word_tokenize(sents[i])
...
```

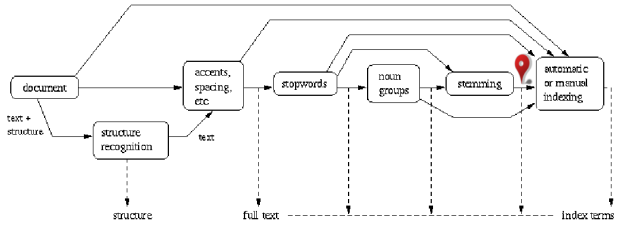

Poniendo todo junto en NLTK - Pipeline (proposal)



Stopwords y lowercase:

```
> from nltk.corpus import stopwords
> stop = stopwords.words('english')
> for i in range(len(sents)):
... sents[i] = [token.lower() for token in sents[i] if token not in stop]
...
```


Poniendo todo junto en NLTK - Pipeline (proposal)



Lematización:

```
> wnl = nltk.WordNetLemmatizer()
> for i in range(len(sents)):
... sents[i] = [wnl.lemmatize(token) for token in sents[i]]
...
```