Laboratorio 4

November 24, 2022

0.1 Laboratorio 4: Semántica Léxica

0.1.1 1. ¿De qué se trata?

Este laboratorio se centra en la semántica léxica y utiliza NLTK para brindarte experiencia práctica con WordNet y realizar la desambiguación del sentido de las palabras. El ejercicio se basa en tutoriales en el sitio web de NLTK http://www.nltk.org/howto/wordnet.html

0.1.2 2. NLTK enviroment

[1]:
```
import nltk
from nltk.book import *
```

```
*** Introductory Examples for the NLTK Book ***
Loading text1, …, text9 and sent1, …, sent9
Type the name of the text or sentence to view it.
Type: 'texts()' or 'sents()' to list the materials.
text1: Moby Dick by Herman Melville 1851
text2: Sense and Sensibility by Jane Austen 1811
text3: The Book of Genesis
text4: Inaugural Address Corpus
text5: Chat Corpus
text6: Monty Python and the Holy Grail
text7: Wall Street Journal
text8: Personals Corpus
text9: The Man Who Was Thursday by G . K . Chesterton 1908
```

0.1.3 3. Accediendo a WordNet

Podes importar WordNet ejecutando:

[2]:
```
from nltk.corpus import wordnet as wn
```

Como vimos en las clases, WordNet está organizado como synsets, cada synset es un conjunto de palabras sinónimas. Una palabra aparecerá en múltiples synsets si tiene múltiples sentidos(senses), así como si puede aparecer en más de un POS. Por ejemplo: sustantivo y verbo. Podes recuperar todos los synsets asociados con una palabra usando la llamada al método synsets():

[3]:
```
nltk.download('omw-1.4')
```

```
[nltk_data] Downloading package omw-1.4 to
[nltk_data]     C:\Users\Usuario\AppData\Roaming\nltk_data…
[nltk_data]   Package omw-1.4 is already up-to-date!
```

[3]: True

[4]:
```
wn.synsets('dog')
```

[4]:
```
[Synset('dog.n.01'),
 Synset('frump.n.01'),
 Synset('dog.n.03'),
 Synset('cad.n.01'),
 Synset('frank.n.02'),
 Synset('pawl.n.01'),
 Synset('andiron.n.01'),
 Synset('chase.v.01')]
```

Vemos que hay 7 synsets de sustantivos y 1 synset de verbos asociados con la palabra "perro". Puede restringir la recuperación a un POS particular utilizando el argumento pos.

[5]:
```
wn.synsets('dog', pos=wn.VERB)
```

[5]:
```
[Synset('chase.v.01')]
```

Como habrás visto en los ejemplos anteriores, se hace referencia a un conjunto de sistemas en particular usando una palabra, una etiqueta de POS y un indicador de número.

Como hablamos en clase, los synsets se ordenan según la frecuencia. dog.n.01 es el primer sentido (o el más frecuente) de perro. El segundo sentido de perro se puede denominar dog.n.02.

En la salida anterior, el segundo synset se conoce como frump.n.01. Este synset es el mismo que dog.n.02. frump.n.01 solo significa que este synset es también el primer synset para la palabra "frump". Pruebe lo siguiente para comprender mejor:

[6]:
```
wn.synset('dog.n.01')
```

[6]:
```
Synset('dog.n.01')
```

[7]:
```
wn.synset('dog.n.02')
```

[7]:
```
Synset('frump.n.01')
```

[8]:
```
wn.synset('frump.n.01')
```

[8]:
```
Synset('frump.n.01')
```

Ahora, preguntemosle a WordNet definiciones de cada sentido y los sinónimos agrupados en un synset.

[9]:
```
wn.synset('dog.n.01').definition()
```

[9]:
```
'a member of the genus Canis (probably descended from the common wolf) that has
been domesticated by man since prehistoric times; occurs in many breeds'
```

[10]:
```
wn.synset('dog.n.02').definition()
```

[10]:
```
'a dull unattractive unpleasant girl or woman'
```

Los sinónimos de un synset pueden ser obtenidos usando el método lemmas().

[11]:
```
wn.synset('dog.n.01').lemmas()
```

[11]:
```
[Lemma('dog.n.01.dog'),
 Lemma('dog.n.01.domestic_dog'),
 Lemma('dog.n.01.Canis_familiaris')]
```

Hay tres sinónomos para el synset de 'dog': 'dog', 'domestic dog' y 'Canis familiaris'. Estos aparecen en el ouput de la última instrucción. Ejercicio: Intenta obtener los lemas para el segundo sentido del "dog"

[12]:
```
wn.synset('dog.n.02').lemmas()
```

[12]:
```
[Lemma('frump.n.01.frump'), Lemma('frump.n.01.dog')]
```

Los lemmas son 'frump' y 'dog'

Las otras Part of Speech son NOUN, ADJ y ADV. Un synset se identifica por un conjunto de 3 partes, siguiendo la forma: word.pos.nn

[13]:
```
wn.synset('dog.n.01')
```

[13]:
```
Synset('dog.n.01')
```

[14]:
```
print(wn.synset('dog.n.01').definition())
```

```
a member of the genus Canis (probably descended from the common wolf) that has
been domesticated by man since prehistoric times; occurs in many breeds
```

[15]:
```
len(wn.synset('dog.n.01').examples())
```

[15]:
```
1
```

[16]:
```
print(wn.synset('dog.n.01').examples()[0])
```

```
the dog barked all night
```

[17]:
```
wn.synset('dog.n.01').lemmas()
```

[17]:
```
[Lemma('dog.n.01.dog'),
 Lemma('dog.n.01.domestic_dog'),
 Lemma('dog.n.01.Canis_familiaris')]
```

[18]:
```
[str(lemma.name()) for lemma in
wn.synset('dog.n.01').lemmas()]
```

[18]:
```
['dog', 'domestic_dog', 'Canis_familiaris']
```

[19]:
```
wn.lemma('dog.n.01.dog').synset()
```

[19]:
```
Synset('dog.n.01')
```

0.1.4 4. Relaciones entre palabras en WordNet

Ahora veamos cómo consultar las relaciones entre los synsets de WordNet. Recuerde que el hiperónimo se refiere al superconjunto de una entidad y el hipónimo se refiere a subconjuntos más específicos. Puede consultar hipernimos e hipónimos utilizando los siguientes métodos:

[20]:
```
dog = wn.synset('dog.n.01')
```

[21]:
```
dog.hypernyms()
```

[21]:
```
[Synset('canine.n.02'), Synset('domestic_animal.n.01')]
```

[22]:
```
dog.hyponyms()
```

[22]:
```
[Synset('basenji.n.01'),
 Synset('corgi.n.01'),
 Synset('cur.n.01'),
 Synset('dalmatian.n.02'),
 Synset('great_pyrenees.n.01'),
 Synset('griffon.n.02'),
 Synset('hunting_dog.n.01'),
 Synset('lapdog.n.01'),
 Synset('leonberg.n.01'),
 Synset('mexican_hairless.n.01'),
 Synset('newfoundland.n.01'),
 Synset('pooch.n.01'),
 Synset('poodle.n.01'),
 Synset('pug.n.01'),
 Synset('puppy.n.01'),
 Synset('spitz.n.01'),
 Synset('toy_dog.n.01'),
 Synset('working_dog.n.01')]
```

Como ya vimos en la sección anterior, las formas de palabras individuales en un conjunto se conocen como lemas. Algunas relaciones solo se mantienen entre lemas (es

decir, entre formas de palabras específicas) en lugar de los conjuntos de sinónimos.

Por ejemplo, recupere el primer synset asociado con el adjetivo "good" (bueno, en inglés) e imprima sus lemas.

[23]:
```
good = wn.synset('good.a.01')
```

[24]:
```
good.lemmas()
```

[24]:
```
[Lemma('good.a.01.good')]
```

Solo hay una forma de lema o palabra en este sintetizador. Puede recuperar antónimos para una forma de palabra de la siguiente manera:

[25]:
```
g0 = good.lemmas()[0]
```

[26]:
```
g0.antonyms()
```

[26]:
```
[Lemma('bad.a.01.bad')]
```

Exploremos también dos relaciones más: meronym y holonym. Un merónimo de una palabra es una subparte o miembro. Un holónimo es un todo del cual la palabra es parte o miembro. Hay métodos separados en NLTK que recuperan merónimos / holónimos para las relaciones entre partes y miembros. P.ej:

[27]:
```
dog.part_meronyms()
```

[27]:
```
[Synset('flag.n.07')]
```

[28]:
```
dog.member_meronyms()
```

[28]:
```
[]
```

[29]:
```
dog.part_holonyms()
```

[29]:
```
[]
```

[30]:
```
dog.member_holonyms()
```

[30]:
```
[Synset('canis.n.01'), Synset('pack.n.06')]
```

Este resultado dice que flag.n.07 es parte de dog.n.01 y dog.n.01 es miembro de canis.n.01 y pack.n.06.

Ejercicio: Utilizando lo que ha estudiado hasta ahora, imprima las definiciones de flag.n.07, canis.n.01 y pack.n.06 y vea si ve por qué estos synsets están relacionados de esta manera.

Definición de flag.n.07

[31]:
```
wn.synset('flag.n.07').definition()
```

[31]:
```
'a conspicuously marked or shaped tail'
```

Definición de canis.n.01

[32]:
```
wn.synset('canis.n.01').definition()
```

[32]:
```
'type genus of the Canidae: domestic and wild dogs; wolves; jackals'
```

Definición de pack.n.06

[33]:
```
wn.synset('pack.n.06').definition()
```

[33]:
```
'a group of hunting animals'
```

flag.n.07 es la definición "Una cola visiblemente marcada o con forma" por lo que puede ser considerado una subparte de dog.n.01, entonces es su merónimo.

pack.n.06 es la definición "un grupo de animales de caza" y canis.n.01 es " género de los Canidae: perros domésticos y salvajes; Lobos; chacales". En ambos casos, las palabras representan el holónimo de dog.n.01 porque este puede ser una subparte de los mismos.

0.1.5 5. Desambiguación de palabras por sentido (Word Sense Desambiguation)

Como vimos en las clases, la tarea de desambiguación del sentido de las palabras es tomar una palabra en el contexto de la oración y mapearla a uno de los sentidos de la palabra, por ejemplo. asignar a un synset en WordNet. Estudiamos dos enfoques, un sistema de clasificación supervisado donde las palabras de contexto entran como feautres(características) y el otro es el algoritmo de Lesk que utilizó el recurso del diccionario para la desambiguación. Ejercicio: recuerde el algoritmo de Lesk. ¿Cuáles fueron los pasos? Usamos el siguiente ejemplo en la clase, intentemos con el mismo.

Ejercicio: recuerde el algoritmo de Lesk. ¿Cuáles fueron los pasos? Usamos el siguiente ejemplo en la clase, intentemos con el mismo.

"The bank can guarantee deposits will eventually cover future tuition costs because it invests in adjustable- rate mortgage securities."

(El banco puede garantizar que los depósitos eventualmente cubrirán los costos de matrícula futuros porque invierte en valores hipotecarios de tasa ajustable.)

Queremos eliminar la ambigüedad de "bank" en este contexto. Ejercicio: ¿Cuántos sentidos crees que tiene la palabra bank? .

bank puede tener varios significados. Podría ser para referirse a la orilla de un río, como en la oración "we sat at the bank of the river"; para referirse a una institución financiera como en "I need to go to the bank to withdraw money"; entre otros.

Use lo que ha estudiado hasta ahora para recuperar todos los synsets asociados con el "bank" con el POS de sustantivo. ¿Cuántos sentidos sustantivos de "banco" hay en WordNet? .

[34]:
```
wn.synsets('bank', pos=wn.NOUN)
```

[34]:
```
[Synset('bank.n.01'),
 Synset('depository_financial_institution.n.01'),
 Synset('bank.n.03'),
 Synset('bank.n.04'),
 Synset('bank.n.05'),
 Synset('bank.n.06'),
 Synset('bank.n.07'),
 Synset('savings_bank.n.02'),
 Synset('bank.n.09'),
 Synset('bank.n.10')]
```

[35]:
```
len(wn.synsets('bank', pos=wn.NOUN))
```

[35]:
```
10
```

WordNet tiene 10 sentidos sustantivos para la palabra "bank"

¿Qué synset es el sentido correcto para la palabra en el contexto de la oración anterior?

[36]:
```
wn.synset('depository_financial_institution.n.01').definition()
```

[36]:
```
'a financial institution that accepts deposits and channels the money into
lending activities'
```

[37]:
```
wn.synset('bank.n.09').definition()
```

[37]:
```
'a building in which the business of banking transacted'
```

El synset correcto es depository_financial_institution.n.01 o bank.n.09

Ahora veamos qué nos da el algoritmo Lesk.

[38]:
```
from nltk.wsd import lesk
```

[39]:
```
from nltk import word_tokenize
```

[40]:
```
S = "The bank can guarantee deposits will eventually cover future tuition costs␣
 ↪because it invests in adjustable-rate mortgage securities"
```

[41]:
```
S_tok = word_tokenize(S)
```

[42]:
```
print(lesk(S_tok, 'bank', 'n'))
```

```
Synset('bank.n.05')
```

Ejercicio: ¿Qué synset fue producido por Lesk?

[43]:
```
wn.synset('bank.n.05').definition()
```

[43]:
```
'a supply or stock held in reserve for future use (especially in emergencies)'
```

El synset producido por Lesk fue bank.n.05, que significa “un suministro o existencias mantenidas en reserva para uso futuro (especialmente en emergencias)”

¿Entiendes por qué tienes esta predicción? Veamos cómo NLTK está implementando Lesk. Puede encontrar el código fuente aquí http://www.nltk.org/_modules/nltk/wsd.html. Hagamos algunos diagnósticos.

[44]:
```
l = word_tokenize((wn.synset('bank.n.05').definition()))
```

[45]:
```
m = word_tokenize((wn.synset('bank.n.02').definition()))
```

[46]:
```
k = set(S_tok)
```

[47]:
```
k.intersection(l)
```

[47]:
```
{'future', 'in'}
```

[48]:
```
k.intersection(m)
```

[48]:
```
{'deposits'}
```

[49]:
```
S_tok
```

[49]:
```
['The',
 'bank',
 'can',
 'guarantee',
 'deposits',
 'will',
 'eventually',
 'cover',
 'future',
 'tuition',
 'costs',
 'because',
 'it',
 'invests',
 'in',
 'adjustable-rate',
 'mortgage',
 'securities']
```

¿Ves ahora por qué tienes esta predicción? ¿Qué cambios, de haber alguno, sugeriría para la correspondencia implementada por NLTK?

Tiene esa predicción porque la función hace la intersección entre los tokens de la frase pasada y los tokens de la definición de cada synset. A partir de esto, elige la intersección de mayor longitud. En el ejemplo anterior, la intersección con bank.n.05 fue mayor a la de bank.n.02, por eso se equivocó al predecir. Una posible solución sería pasar al método lesk el parámetro synsets para limitar la búsqueda.

[50]:
```
synsets_aux = wn.synsets('bank', pos=wn.NOUN)[0:3]
```

[51]:
```
print(lesk(S_tok, 'bank', 'n', synsets = synsets_aux))
```

```
Synset('depository_financial_institution.n.01')
```

Ejercicio: para cada ejemplo, a continuación descubra el sentido correcto de WordNet según su criterio. Puede usar la búsqueda en línea de WordNet para navegar a través de los sentidos. http://wordnetweb.princeton.edu/perl/webwn.

A. I went to the bank to deposit some money.

El sentido correcto de 'bank' sería depository_financial_institution.n.01, que significa "una institución financiera que acepta depósitos y canaliza el dinero hacia actividades de préstamo"

[52]:
```
S1 = 'I went to the bank to deposit some money.'
```

[53]:
```
S1_tok = word_tokenize(S1)
```

[54]:
```
print(f'{lesk(S1_tok, "bank", "n")} ')
```

```
Synset('savings_bank.n.02')
```

lesk considero a savings_bank.n.02 como el sentido correcto, que significa "un recipiente (generalmente con una ranura en la parte superior) para guardar dinero en casa"

B. She created a big mess of the birthday cake.

El sentido correcto de 'mess' podría ser mess.n.01 que significa "un estado de confusión o desorden"

[55]:
```
S2 = 'She created a big mess of the birthday cake.'
```

[56]:
```
S2_tok = word_tokenize(S2)
```

[57]:
```
print(f'{lesk(S2_tok, "mess", "n")} ')
```

```
Synset('mess.n.04')
```

lesk consideró a mess.n.04 como el sentido correcto, que significa " una comida consumida en un comedor por personal de servicio"

C. In the interest of your safety, please wear your seatbelt.

El sentido correcto de 'interest' podría ser interest.n.02, que significa "una razón para querer que algo se haga"

[58]:
```
S3 = "In the interest of your safety, please wear your seatbelt."
```

[59]:
```
S3_tok = word_tokenize(S3)
```

[60]:
```
print(f'{lesk(S3_tok, "interest", "n")} ')
```

```
Synset('interest.n.04')
```

lesk consideró a interest.n.04 como el sentido correcto, que significa " un cargo fijo por pedir prestado dinero; generalmente un porcentaje de la cantidad prestada"

D. I drank some ice cold water.

Un sentido correcto de 'ice' podría ser ice.v.02, que significa "hacer que se convierta en hielo o helado"

[61]:
```
S4= 'I drank some ice cold water.'
```

[62]:
```
S4_tok = word_tokenize(S4)
```

[63]:
```
print(f'{lesk(S4_tok, "ice")} ')
```

```
Synset('ice_rink.n.01')
```

lesk consideró a ice_rink.n.01 como el sentido correcto, que significa "una pista con piso de hielo para hockey sobre hielo o patinaje sobre hielo"

Luego use el algoritmo de Lesk para desambiguar las palabras. ¿Cuál es la precisión de la implementación de Lesk de NLTK en estas oraciones?

En estas oraciones, lesk no pude retornar el sentido correcto para ninguna de las palabras

0.1.6 6. Un poco más allá

Si ya estás cómodo con el contenido anterior, podés desafiarte con este ejercicio. Este ejercicio tiene un "final abierto". Ejercicio: un lemma de una palabra agrupa las inflexiones de una palabra. Es decir, 'walked', 'walking', 'walks', 'walk' son todas inflexiones o variantes morfológicas de la palabra walk ("caminar"). NLTK tiene capacidad para lematizar palabras.

[64]:
```
from nltk.stem import WordNetLemmatizer
```

[65]:
```
wnLemmatizer = WordNetLemmatizer()
```

[66]:
```
wnLemmatizer.lemmatize('dogs', 'n')
```

[66]:
```
'dog'
```

[67]:
```
wnLemmatizer.lemmatize('walking', 'v')
```

[67]:
```
'walk'
```

Aquí informamos al lematizador si la palabra dada es un sustantivo o un verbo. La configuración predeterminada para el método es un sustantivo, es decir. si corre sin argumentos, el lema será correcto para los sustantivos pero quizás no para los verbos.

¿Podrías mejorar el algoritmo de Lesk con un lematizador? Si es así, ¿podrías escribir una función de match, que tome dos cadenas y devuelva las palabras coincidentes entre ellas? Queremos que las coincidencias(matches) sean útiles para un algoritmo como Lesk. ¿Quizás también necesites usar un etiquetador POS? Podes usar un etiquetador POS incorporado en WordNet.

Escribí algunos casos de prueba para tu algoritmo. >print(match(‘ice’, ‘icing’)) [Synset(‘ice.v.02’), Synset(‘frosting.n.01’), Synset(‘frost.v.01’), Synset(‘ice.v.03’)]

print(match(‘involvement’, ‘interest’)) [Synset(‘interest.n.01’)]

print(match(‘hot’, ‘spicy’)) [Synset(‘hot.s.09’)]

print(match(‘duck’, ‘evade’)) [Synset(‘hedge.v.01’)]