Curso de procesamiento del lenguaje natural

César Antonio Aguilar

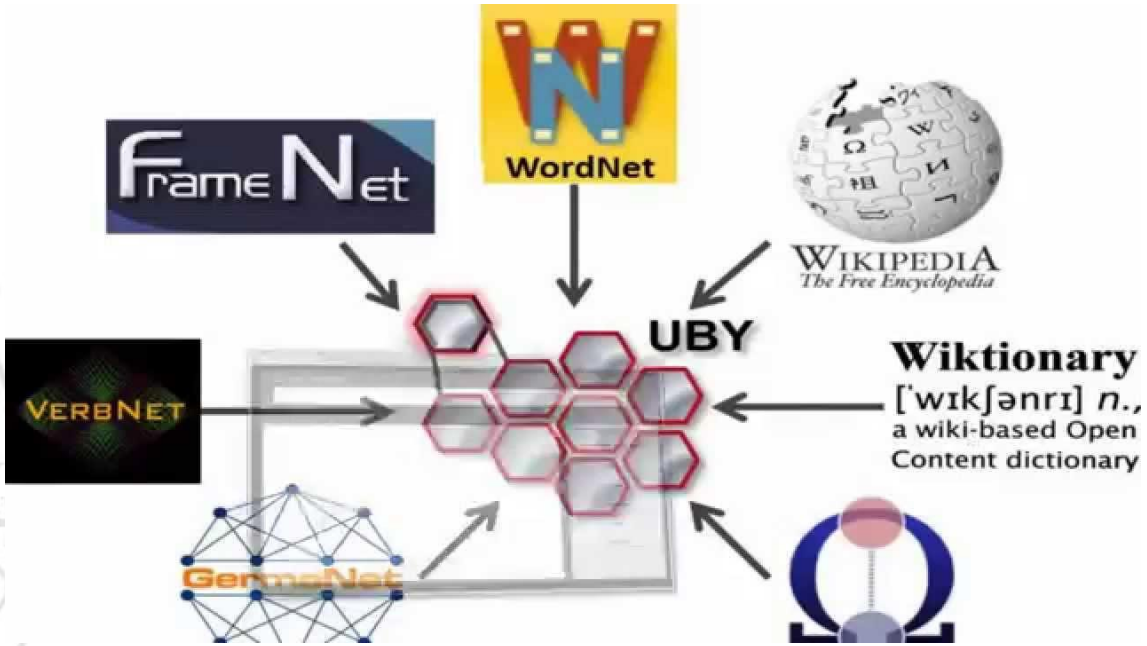
Facultad de Letras

21/11/2018

Cesar.Aguilar72@gmail.com

Explorando WordNet desde NLTK (1)

En la clase pasada, vimos varios tipos de recursos denominados como bases de conocimientos léxico (o BCL), basadas en el modelo propuesto por WordNet.



Gracias a la implementación de WordNet se han desarrollado hoy en día múltiples recursos enfocados en colectar y organizar información semántica, tales como Wikipedia o Dbpedia.

Explorando WordNet desde NLTK (1)

En NLTK contamos con una versión de WordNet como un corpus, el cual podemos emplear usando los siguientes comandos:

Corpus	Compiler	Contents
Brown Corpus	Francis, Kucera	15 genres, 1.15M words, tagged, categorized
CESS Treebanks	CLiC-UB	1M words, tagged and parsed (Catalan, Spanish)
Chat-80 Data Files	Pereira & Warren	World Geographic Database
CMU Pronouncing Dictionary	CMU	127k entries
CoNLL 2000 Chunking Data	CoNLL	270k words, tagged and chunked
CoNLL 2002 Named Entity	CoNLL	700k words, pos- and named-entity-tagged (Dutch, Spanish)
CoNLL 2007 Dependency Treebanks (sel)	CoNLL	150k words, dependency parsed (Basque, Catalan)
Dependency Treebank	Narad	Dependency parsed version of Penn Treebank sample
FrameNet	Fillmore, Baker et al	10k word senses, 170k manually annotated sentences
Floresta Treebank	Diana Santos et al	9k sentences, tagged and parsed (Portuguese)
Gazetteer Lists	Various	Lists of cities and countries
Genesis Corpus	Misc web sources	6 texts, 200k words, 6 languages
Gutenberg (selections)	Hart, Newby, et al	18 texts, 2M words
Inaugural Address Corpus	CSpan	US Presidential Inaugural Addresses (1789-present)
Indian POS-Tagged Corpus	Kumaran et al	60k words, tagged (Bangla, Hindi, Marathi, Telugu)
MacMorpho Corpus	NILC, USP, Brazil	1M words, tagged (Brazilian Portuguese)
Movie Reviews	Pang, Lee	2k movie reviews with sentiment polarity classification
Names Corpus	Kantrowitz, Ross	8k male and female names
NIST 1999 Info Extr (selections)	Garofolo	63k words, newswire and named-entity SGML markup
Nombank	Meyers	115k propositions, 1400 noun frames
NPS Chat Corpus	Forsyth, Martell	10k IM chat posts, POS-tagged and dialogue-act tagged
Open Multilingual WordNet	Bond et al	15 languages, aligned to English WordNet
PP Attachment Corpus	Ratnaparkhi	28k prepositional phrases, tagged as noun or verb modifiers
Proposition Bank	Palmer	113k propositions, 3300 verb frames
Question Classification	Li, Roth	6k questions, categorized
Reuters Corpus	Reuters	1.3M words, 10k news documents, categorized
Roget's Thesaurus	Project Gutenberg	200k words, formatted text
RTE Textual Entailment	Dagan et al	8k sentence pairs, categorized
SEMCOR	Rus, Mihalcea	880k words, part-of-speech and sense tagged
Senseval 2 Corpus	Pedersen	600k words, part-of-speech and sense tagged
SentiWordNet	Esuli, Sebastiani	sentiment scores for 145k WordNet synonym sets
Shakespeare texts (selections)	Bosak	8 books in XML format
State of the Union Corpus	CSPAN	485k words, formatted text
Stopwords Corpus	Porter et al	2,400 stopwords for 11 languages
Swadesh Corpus	Wiktionary	comparative wordlists in 24 languages
Switchboard Corpus (selections)	LDC	36 phonecalls, transcribed, parsed
Univ Decl of Human Rights	United Nations	480k words, 300+ languages
Penn Treebank (selections)	LDC	40k words, tagged and parsed
TIMIT Corpus (selections)	NIST/LDC	audio files and transcripts for 16 speakers
VerbNet 2.1	Palmer et al	5k verbs, hierarchically organized, linked to WordNet
Wordlist Corpus	OpenOffice.org et al	960k words and 20k affixes for 8 languages
WordNet 3.0 (English)	Miller, Fellbaum	145k synonym sets

Explorando WordNet desde NLTK (3)

¿Qué se puede hacer con este corpus? Básicamente realizar consultas, de un modo similar a lo que podemos hacer con un diccionario electrónico accesible a través de comandos o instrucciones. Veamos:

```
import nltk, re, os

from nltk.corpus import wordnet as wn

wn.synset('car.n.01').lemma_names()
Out[3]: ['car', 'auto', 'automobile', 'machine', 'motorcar']

wn.synsets('motorcar')
Out[4]: [Synset('car.n.01')]
```

Explorando WordNet desde NLTK (4)

Cambiemos el ejemplo. ¿Qué cosa es un *wildcat*?:

```
wn.synsets('wildcat')
Out[5]: [Synset('wildcat_well.n.01'),
 Synset('beast.n.02'),
 Synset('wildcat.n.03'),
 Synset('wildcat.s.01'),
 Synset('unauthorized.s.02'),
 Synset('wildcat.s.03')]
```

Si asociamos *wildcat* con *beast*, ¿cómo podemos concebir al segundo? Algunos ejemplos:

```
wn.synset('beast.n.02').lemma_names()
Out[6]: ['beast', 'wolf', 'savage', 'brute', 'wildcat']
```

Explorando WordNet desde NLTK (5)

¿Cuántos synsets podemos asociar con *cat*?:

```
wn.synsets('cat')
Out[7]:
[Synset('cat.n.01'),
 Synset('guy.n.01'),
 Synset('cat.n.03'),
 Synset('kat.n.01'),
 Synset('cat-o'-nine-tails.n.01'),
 Synset('caterpillar.n.02'),
 Synset('big_cat.n.01'),
 Synset('computerized_tomography.n.01'),
 Synset('cat.v.01'),
 Synset('vomit.v.01')]
```

Explorando WordNet desde NLTK (6)

¿Qué lemas podemos ligar a la palabra *cat*?:

```
wn.synset('cat.n.01').lemma_names()
Out[8]: ['cat', 'true_cat']
```

¿Hay una definición mínima para el lema *cat*?:

```
wn.synset('cat.n.01').definition()
Out[9]: 'feline mammal usually having thick soft fur and
no ability to roar: domestic cats; wildcats'
```

Curiosamente, no tenemos ejemplos que nos ayuden a comprender mejor qué cosa es *cat*:

```
wn.synset('cat.n.01').examples()
Out[10]: []
```

Explorando WordNet desde NLTK (7)

Cambiemos entonces a *car*, y obtenemos el siguiente ejemplo:

```
wn.synset('car.n.01').examples()
Out[11]: ['he needs a car to get to work']
```

Volvamos con *cat*, y veamos cuántos synsets se asocian a todos los lemas posibles:

```
for synset in wn.synsets('cat'):
    print(synset.lemma_names())
```

Y el resultado es:

Explorando WordNet desde NLTK (8)

['cat', 'true_cat']
['guy', 'cat', 'hombre', 'bozo']
['cat']
['kat', 'khat', 'qat', 'quat', 'cat', 'Arabian_tea', 'African_tea']
["cat-o'-nine-tails", 'cat']
['Caterpillar', 'cat']
['big_cat', 'cat']
['computerized_tomography', 'computed_tomography', 'CT', 'computerized_axial_tomography', 'computed_axial_tomography', 'CAT']
['cat']
['vomit', 'vomit_up', 'purge', 'cast', 'sick', 'cat', 'be_sick', 'disgorge', 'regorge', 'retch', 'puke', 'barf', 'spew', 'spue', 'chuck', 'upchuck', 'honk', 'regurgitate', 'throw_up']

Explorando WordNet desde NLTK (9)

Ahora pasemos a ver cómo podemos generar hipónimos asociados a una palabra. Usemos la siguiente instrucción:

```
motorcar = wn.synset('car.n.01')

types_of_motorcar = motorcar.hyponyms()
```

Supongamos que los hipónimos asociados a *car* cuentan con un índice que inicia en *0* (cero). ¿Cuál es el primero?:

```
types_of_motorcar[0]
Out[33]: Synset('ambulance.n.01')
```

Explorando WordNet desde NLTK (9)

Pasemos entonces a identificar todos los hipónimos, y para hacer esto, podemos organizarlos con la siguiente instrucción:

```
sorted(lemma.name() for synset in types_of_motorcar for lemma in synset.lemmas())
```

Y el resultado es:

```
['Model_T', 'S.U.V.', 'SUV', 'Stanley_Steamer', 'ambulance', 'beach_waggon',
'beach_wagon', 'bus', 'cab', 'compact', 'compact_car', 'convertible',
'coupe', 'cruiser', 'electric', 'electric_automobile', 'electric_car',
'estate_car', 'gas_guzzler', 'hack', 'hardtop', 'hatchback', 'heap',
'horseless_carriage', 'hot-rod', 'hot_rod', 'jalopy', 'jeep', 'landrover',
'limo', 'limousine', 'loaner', 'minicar', 'minivan', 'pace_car', 'patrol_car',
'phaeton', 'police_car', 'police_cruiser', 'prowl_car', 'race_car', 'racer',
'racing_car', 'roadster', 'runabout', 'saloon', 'secondhand_car', 'sedan',
'sport_car', 'sport_utility', 'sport_utility_vehicle', 'sports_car', 'squad_car',
'station_waggon', 'station_wagon', 'stock_car', 'subcompact', 'subcompact_car',
'taxi', 'taxicab', 'tourer', 'touring_car', 'two-seater', 'used-car', 'waggon',
'wagon']
```

Explorando WordNet desde NLTK (10)

Hagamos el camino contrario: ¿cuál es el hiperónimo de *motorcar*?:

```
motorcar.hypernyms()
Out[35]: [Synset('motor_vehicle.n.01')]
```

Dado que estamos analizando relaciones semánticas de tipo jerárquicas, supondríamos que podemos desplazarnos desde un nivel inferior (hipónimo) hasta uno superior (hiperónimo).

Si identificamos estos desplazamientos con el nombre *path*, ¿cuántos caminos podemos recorrer hasta llegar al último hiperónimo asociado a *motorcar*? Primero ubiquemos cuántos *paths* tenemos:

```
paths = motorcar.hypernym_paths()

len(paths)
Out[37]: 2
```

Explorando WordNet desde NLTK (11)

Ahora, vamos a recorrerlos cada uno. Usemos esta instrucción:

```
[synset.name() for synset in paths[0]]
```

El resultado es:

```
['entity.n.01', 'physical_entity.n.01', 'object.n.01', 'whole.n.02', 'artifact.n.01',
'instrumentality.n.03', 'container.n.01', 'wheeled_vehicle.n.01',
'self-propelled_vehicle.n.01', 'motor_vehicle.n.01', 'car.n.01']
```

¿Qué pasa cuando hacemos el recorrido del *path 1*? Veamos:

```
[synset.name() for synset in paths[1]]
```

```
['entity.n.01', 'physical_entity.n.01', 'object.n.01', 'whole.n.02', 'artifact.n.01',
'instrumentality.n.03', 'conveyance.n.03', 'vehicle.n.01', 'wheeled_vehicle.n.01',
'self-propelled_vehicle.n.01', 'motor_vehicle.n.01', 'car.n.01']
```

Similitud semántica (1)

Un tema interesante que emerge a partir del uso de WordNet es la posibilidad de determinar qué tan cercanos o alejados los significados propios de un grupo de palabras.

A esto se le conoce usualmente como **similitud semántica**, la cual puede ser medida para determinar si hay cercanía o no.

Veamos un ejemplo usando WordNet. Escriban los siguientes comandos:

```
right = wn.synset('right_whale.n.01')

orca = wn.synset('orca.n.01')

minke = wn.synset('minke_whale.n.01')

tortoise = wn.synset('tortoise.n.01')

novel = wn.synset('novel.n.01')
```

Similitud semántica (2)

¿Qué relación hay en el significado de las palabras anteriores? Veamos:

```
right.lowest_common_hypernyms(minke)
Out[45]: [Synset('baleen_whale.n.01')]

right.lowest_common_hypernyms(orca)
Out[46]: [Synset('whale.n.02')]

right.lowest_common_hypernyms(tortoise)
Out[47]: [Synset('vertebrate.n.01')]

right.lowest_common_hypernyms(novel)
Out[48]: [Synset('entity.n.01')]
```

Similitud semántica (3)

Si nos imaginamos un árbol conceptual (esto es, un grafo que represente una taxonomía), ¿en qué posiciones quedarían las entidades asociadas a nuestras palabras? Veamos:

```
wn.synset('baleen_whale.n.01').min_depth()
Out[49]: 14

wn.synset('whale.n.02').min_depth()
Out[50]: 13

wn.synset('vertebrate.n.01').min_depth()
Out[51]: 8

wn.synset('entity.n.01').min_depth()
Out[52]: 0
```

Similitud semántica (4)

Finalmente, derivemos una medida de similitud entre todas las palabras que tenemos en la lista, usando la instrucción *path_similarity*. Consideremos que nuestro hiperónimo es *right-whale* (en español, se le conoce como **ballena franca**), ¿cuán cercanas o alejadas están de su significado las otras palabras?:

```
right.path_similarity(minke)
Out[53]: 0.25

right.path_similarity(orca)
Out[54]: 0.16666666666666666

right.path_similarity(tortoise)
Out[55]: 0.07692307692307693

right.path_similarity(novel)
Out[56]: 0.043478260869565216
```

Clasificando textos con WordNet (1)

Desde el 2004, un lingüista computacional norteamericano llamado **Ted Pedersen**, de la Universidad de Minnesota, junto con su equipo de colaboradores empezaron a implementar un método para detectar similitudes semánticas entre documentos, a partir del uso de WordNet.

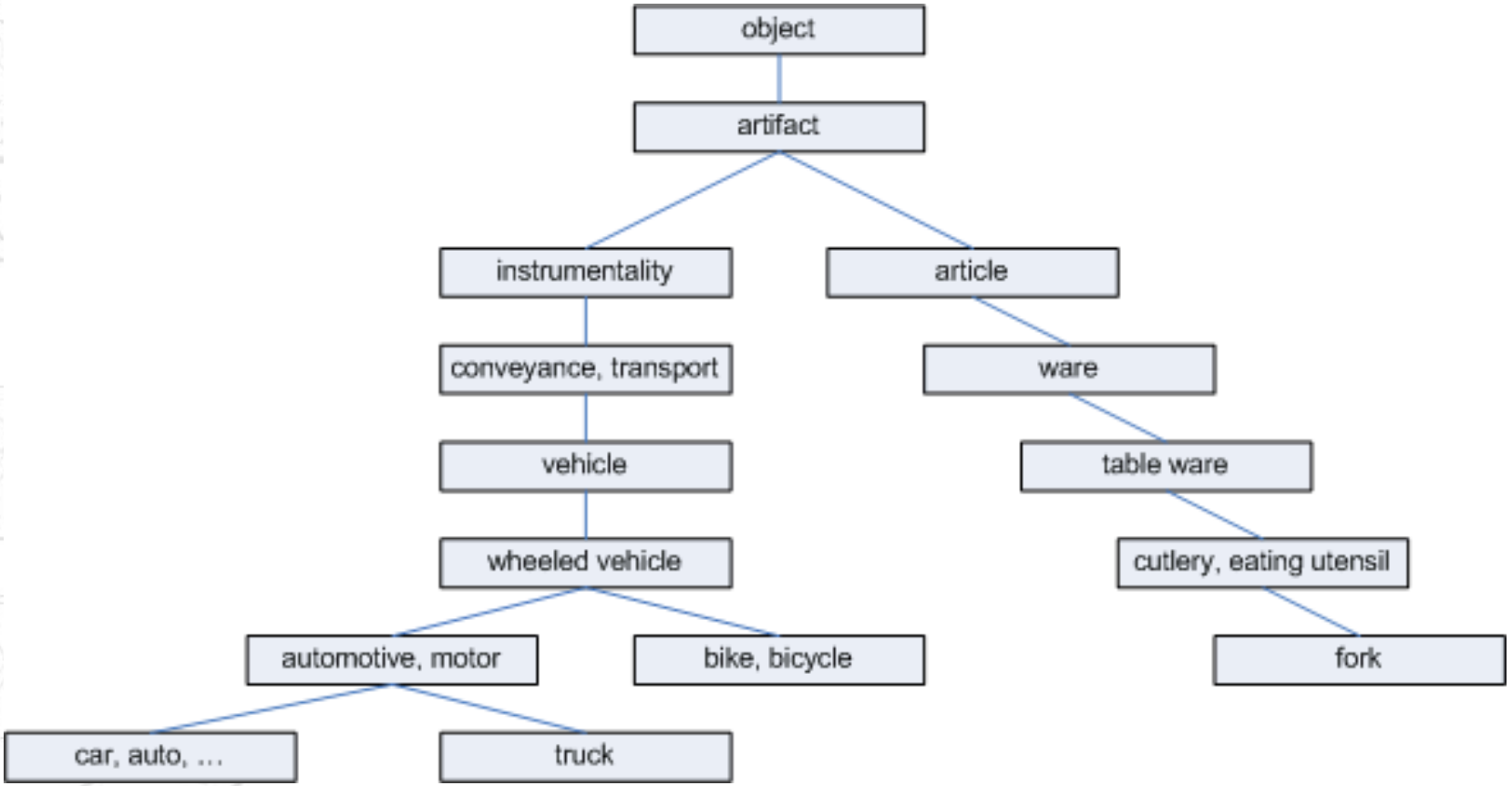
Ted Pedersen

Para más detalles sobre la propuesta de Ted, pueden revisar el siguiente sitio WEB:

www.d.umn.edu/~tpederse/similarity.html

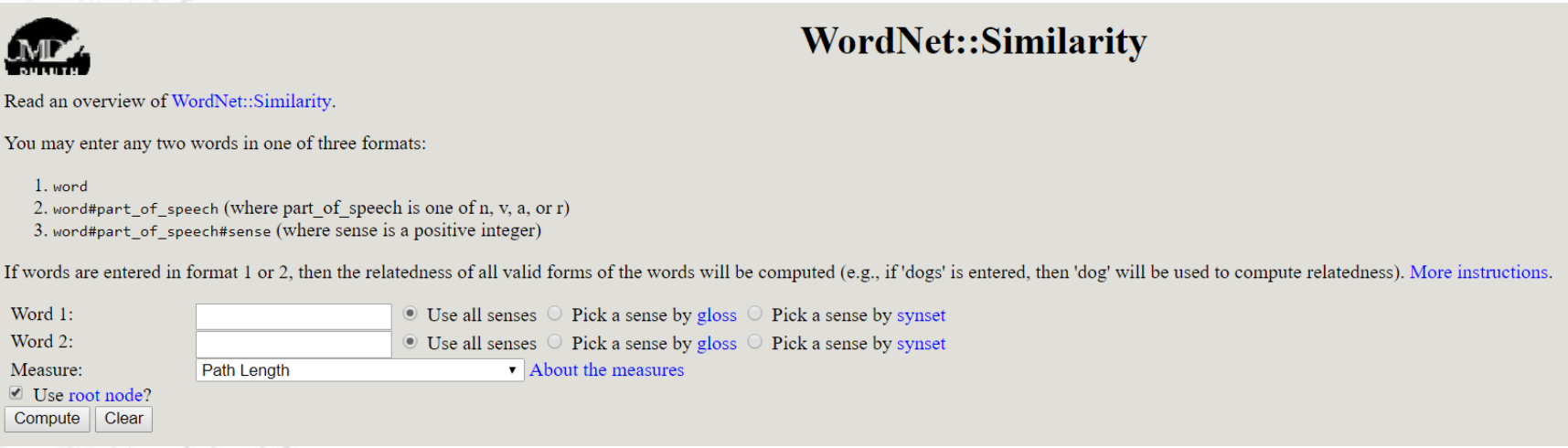
Clasificando textos con WordNet (2)

La similitud semántica con WordNet, como hemos visto, se basa en tratar de inferir qué tan cercano (o alejado) es el significado de dos o más palabras, dependiendo de los *synsets* que describen dicho significado.



Clasificando textos con WordNet (3)

De hecho, Ted y sus colegas han desarrollado una *demo* que ejemplifica cómo se puede establecer grados de similitud entre palabras, infiriendo lo que ellos denominan *distancia semántica*, esto es: proyectando un árbol taxonómico como el de la figura anterior, ¿qué tan cercano o alejado está el significado de dos o más palabras?



WordNet::Similarity

Read an overview of WordNet::Similarity.

You may enter any two words in one of three formats:

1. `word`
2. `word#part_of_speech` (where part_of_speech is one of n, v, a, or r)
3. `word#part_of_speech#sense` (where sense is a positive integer)

If words are entered in format 1 or 2, then the relatedness of all valid forms of the words will be computed (e.g., if 'dogs' is entered, then 'dog' will be used to compute relatedness). More instructions.

Word 1: Use all senses Pick a sense by gloss Pick a sense by synset

Word 2: Use all senses Pick a sense by gloss Pick a sense by synset

Measure: Path Length About the measures

Use root node?

Compute Clear

http://maraca.d.umn.edu/cgi-bin/similarity/similarity.cgi

Clasificando textos con WordNet (4)

Veamos un ejemplo de cómo funciona esto: supónganse que tienen que hacer una clasificación de manuales que tengan que ver con herramientas eléctricas para una carpintería o mueblería, y tienen dudas en saber si una herramienta llamada **saw** (“sierra”) es lo mismo que una llamada **Sander** (“lijadora”). Esto es:

Saw

Sander

Clasificando textos con WordNet (5)

La idea aquí es determinar si estas herramientas son similares o están relacionadas, es decir:

Similar or Related?

- **Relatedness is more general**
  - General term involving many relationships
    - car-wheel (meronymy)
    - hot-cold (antonymy)
    - pencil-paper (functional)
    - penguin-Antarctica (association)
- **Semantic similarity**
  - More specific term involving likeness
    - bank-trust company (synonymy)
- **Hammer and nail are related but they really aren't similar**
- **All similar concepts are related, but not all related concepts are similar**

Clasificando textos con WordNet (6)

Retomando lo que nos enseñó Ana Bertha en sus dos primeras presentaciones, hay dos caminos para resolver esta cuestión:

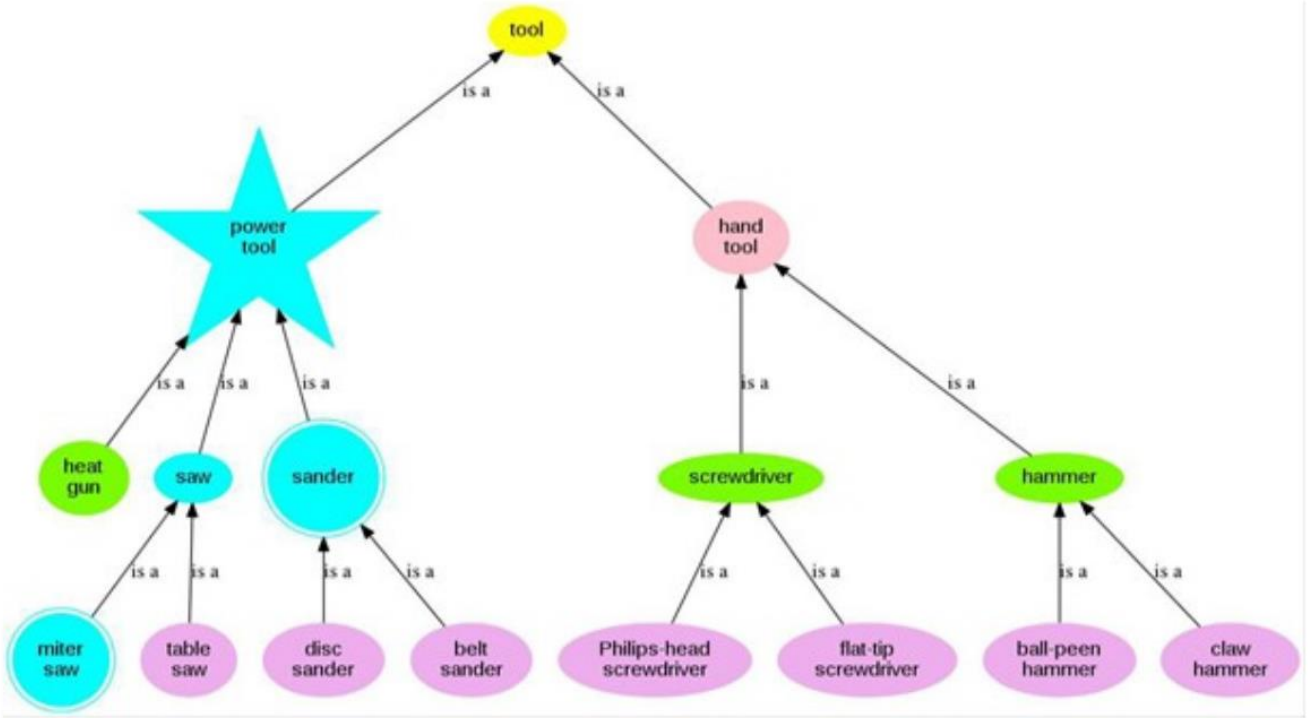
- **Knowledge-based (WordNet)**
  - Path Based
    - Rada et al. 1989 (path)
  - Path + Depth
    - Wu & Palmer, 1994 (wup)
    - Leacock & Chodorow, 1998 (lch)
  - Path + Information Content
    - Resnik, 1995 (res)
    - Jiang & Conrath, 1997 (jcn)
    - Lin, 1998 (lin)
- **Corpus-based**
  - LSA (Landauer et al. 1998)
  - Topicl Modeling (David Blei et al)

Clasificando textos con WordNet (7)

Si usamos WordNet, entonces nuestro cálculo de distancia semántica consiste en determinar **qué tan cercanos o alejados son los nodos que clasifican (o categorizan) el significado de *saw* y *sander***, esto es:

We count nodes (links)

$$radar(miter\ saw, sander) = 0.25$$



Clasificando textos con WordNet (8)

¿Hay algún otro par de nodos que mantengan una relación equivalente a la de *saw* y *sander*? Sí, veamos el caso de *hammer* y *powe tools*.

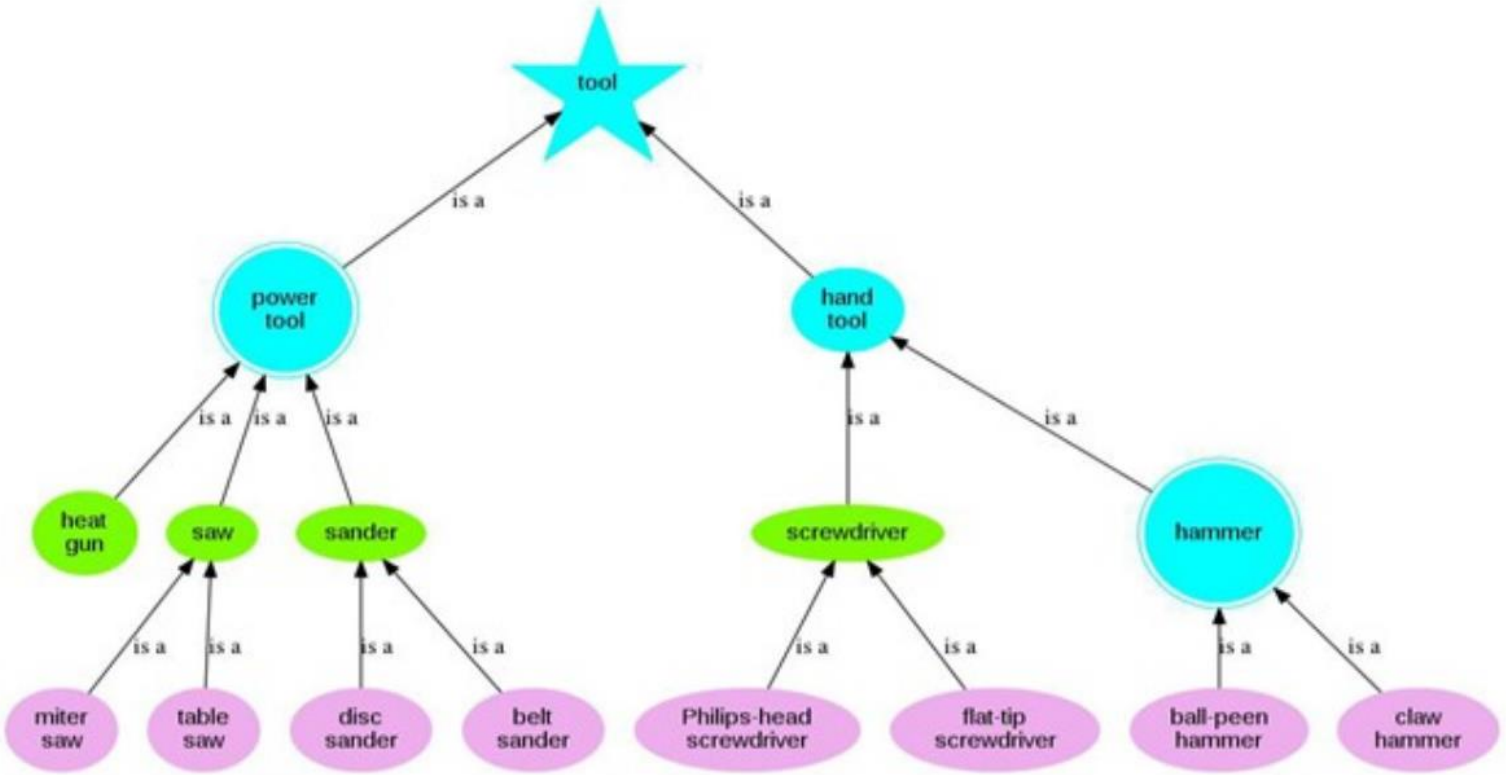
Hammers and power tools

Clasificando textos con WordNet (8)

Veamos el árbol taxonómico:

We count nodes (links)

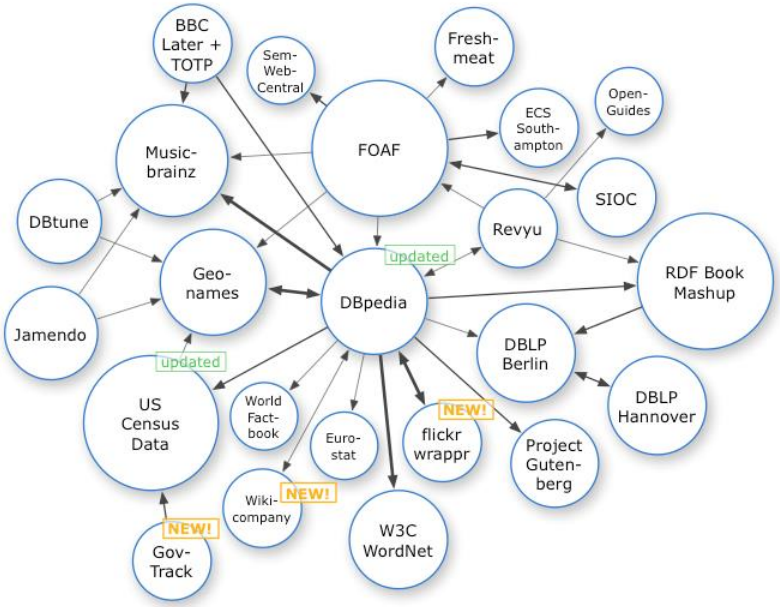
$$Radar(hammer, power\ tool) = 0.25$$



Infiriendo sentidos automáticamente (1)

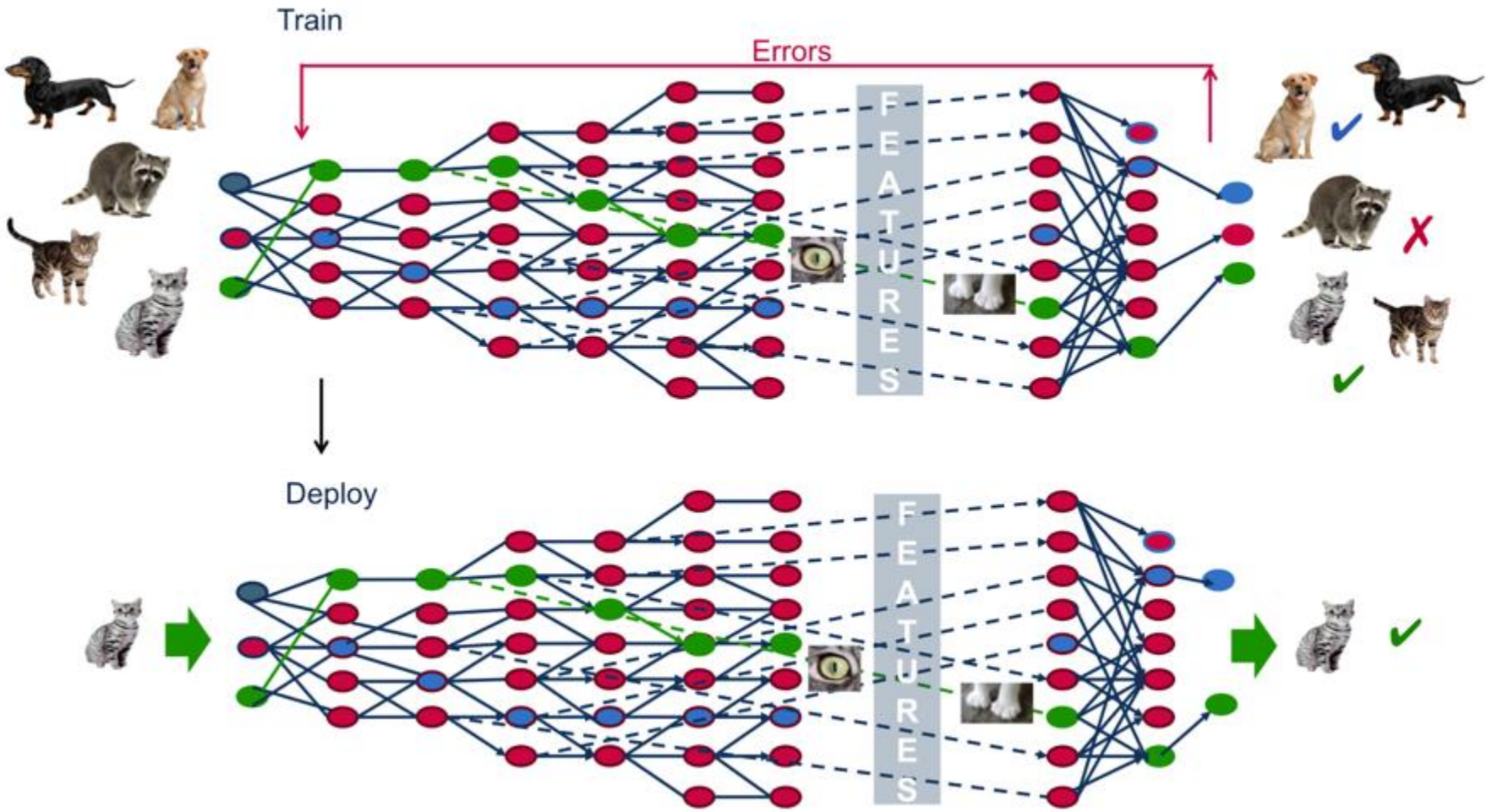
El reconocimiento de similitudes semánticas basadas en WordNet ha dado lugar al desarrollo de recursos con información conceptual mucho más amplios, tales como **Wikipedia**. **Dbpedia** o **Linked Data**, por mencionar algunos ejemplos.

A diferencia de WordNet, estos recursos son auténticas ontologías que ayudan a clasificar miles de conceptos, tanto generales como especializados, a partir de la identificación de relaciones léxicas, p. e.:



Infiriendo sentidos automáticamente (1)

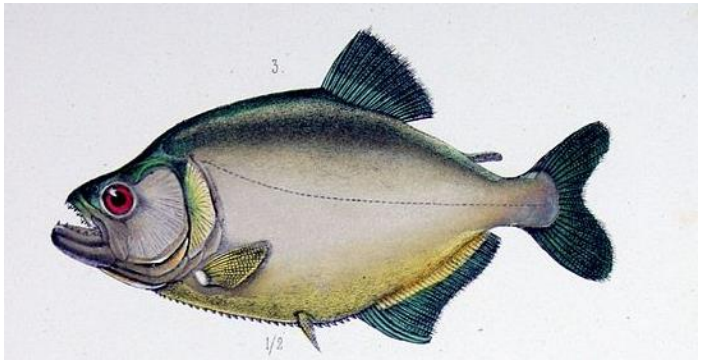
El uso de esta clase de recursos ha dado lugar a una nueva manera de interpretar el aprendizaje automático por medio de redes neuronales, aplicando un enfoque conocido como **aprendizaje profundo** (ing. *Deep Learning*):



Infiriendo sentidos automáticamente (2)

Sin embargo, todavía es necesario explorar y mejorar varias de estas relaciones de similitud entre conceptos. Por poner un caso, estos recursos todavía no toman en cuenta distinciones tales como significado referencial *versus* variantes de sentido, p. e.:

≠



Supongan que analizamos las opiniones políticas generadas en Facebook y Twiteer sobre Sebastián Piñera, y vemos que en un 86% de tales comentarios lo llaman **Piraña**. Pregunta: ¿hay una similitud semántica entre ambos?

Infiriendo sentidos automáticamente (3)

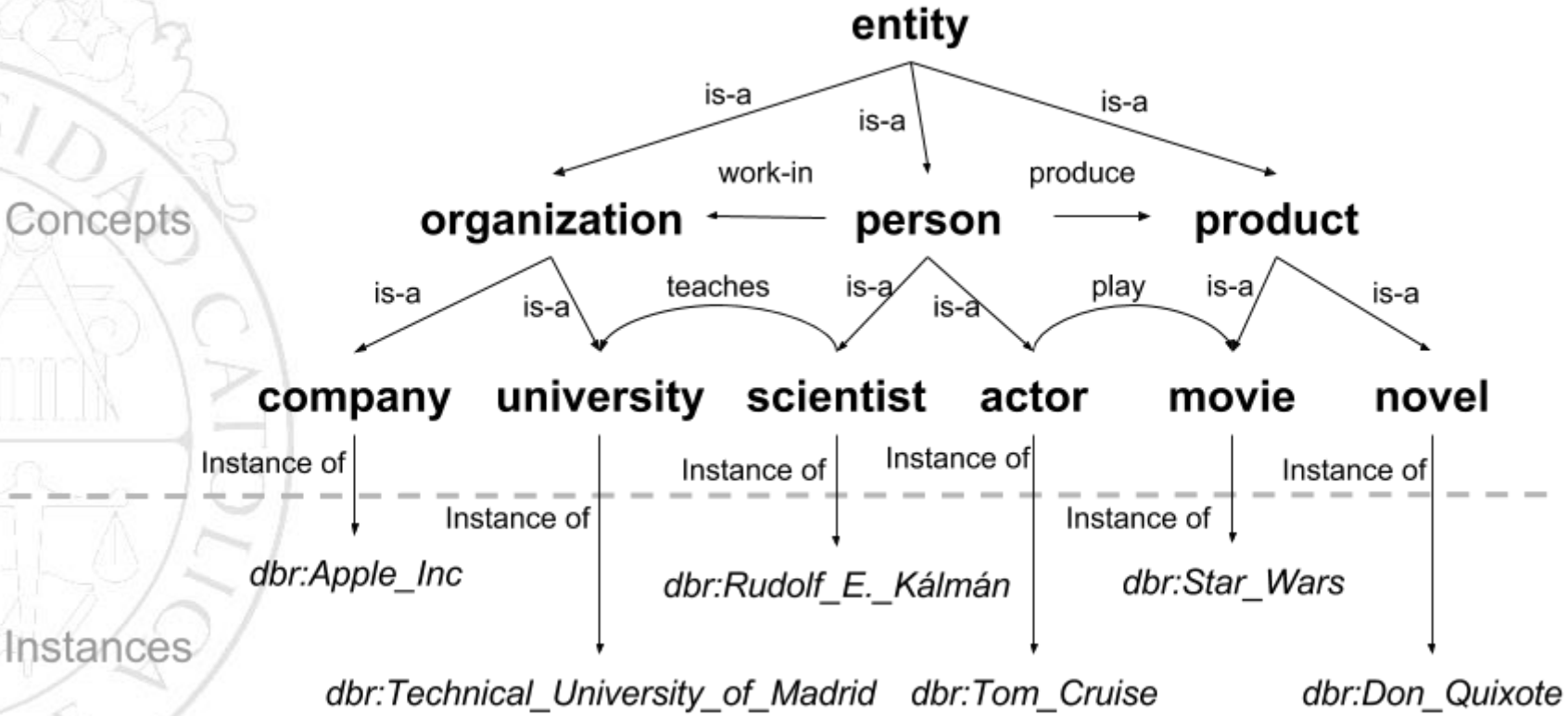
Ahora, esto puede crear problemas serios respecto a qué clase de información semántica puede tomar como pertinente o no un sistema inteligente.

Un caso muy concreto: supongan que el MIT cuenta con un sistema de selección automático para identificar candidatos que pueden recibir una beca para hacer un diplomado especializado.

Y por azares del destino, una estudiante con un alto promedio no recibe tal apoyo, porque su nombre es *Jadiya Hafda*, y aunque nació en Bagdad, lleva diez años residiendo con su familia en Boston.

Infiriendo sentidos automáticamente (4)

En conclusión, el reto entonces es implementar sistemas que puedan hacer distinciones de significados pertinentes, lo que les permita optimizar sus procesos de toma de decisiones para resolver cualquier tarea.



Gracias por su atención

Blog del curso:
http://cesaraguilar.weebly.com/curso-de-procesamiento-del-lenguaje-natural.html