Relazione esercizio hanks

L'esercitazione 3 ha visto la definizione di un algoritmo che implementa la teoria di **Patrick Hanks** relativamente alla costruzione del significato di espressioni complesse. Secondo Hanks l'essenza del significato di una frase risiede nel verbo e nella sua valenza. Per comprendere il significato di una frase occorre studiare le diverse manifestazioni semantiche dei dipendenti di un verbo. Un verbo con valenza "n" avrà associato "n" slot, nei quali rientrano tutte le parole che assumono un certo ruolo sintattico associato allo slot, così per un verbo transitivo con valenza 2 studieremo tutti filler per gli slot di "soggetto" e "oggetto diretto". I filler vengono classificati in gruppi semantici generici, dati dai tipi semantici e le diverse combinazioni di questi, per ogni slot, formano i diversi significati della frase in cui occorre il dato verbo.

Per questa esercitazione è stato recuperato un corpus di frasi in cui osservare uno o più verbi sui quali applicare la teoria di patrick hanks. In particolare, sono stati recuperati da sketch engine 3 corpora di almeno 1000 di frasi che trattano alcuni verbi transitivi disponibili in file separati nella cartella "**input**". I verbi transitivi considerati sono: "**hate**", "**love**" e "**create**". In tutte le frasi recuperate i verbi hanno valenza 2. I file sono stati analizzati e molte frasi sono state rimosse sia perchè non si riferivano ai precedenti verbi, sia per questioni di compattezza. Usando lo strumento di sketch engine, esse sono ordinate per GDEX decrescente (Good dictionary examples), che stima la facilità di lettura, in modo da non complicare troppo il task. Le frasi, organizzate in file separati ("hate.txt","love.txt","create.txt"), presentano tutte la stessa struttura incapsulata in tag <s> </s>. Alcuni esempi sono:

- <s> I failed my exam six times and hated school a lot. </s>
- <s> The far right will hate it. </s>
- <s> I don't know why I loved it". </s>

L'entry point del programma è il file "**main.py**". All'atto dell'esecuzione, il main richiede di dare in input il percorso al file contenente le frasi da analizzare. Verrà eseguito un semplice controllo sul tipo di file, ed in seguito verranno analizzate le frasi eseguendo alcune azioni di preprocessing, fra le quali ritroviamo: rimozione del tag <s> e rimozione di spazi bianchi multipli e dei simboli di punteggiatura. Le frasi raccolte sono pronte per essere parsificate. Con la libreria spacy, impostando il default model per la lingua inglese, si è eseguita un'analisi a dipendenze delle frasi in input analizzando i diversi ruoli sintattici.

Ai fini della teoria di Patrick Hanks sono stati rilevati 2 possibili gruppi sintattici riferiti ai 2 argomenti del verbo:

- **POSSIBLE_SUBJ** = [**'subj'**, **'nsubjpass'**, **'nsubj'**]
- **POSSIBLE_OBJ** = [**'pobj'**, **'dobj'**, **'obj'**, **'iobj'**]

A partire da queste 2 classi di argomenti la funzione "**dependency_parsing",** presente nel modulo "**hanks.py**", esegue l'analisi sintattica delle frasi in input.

Da ogni token vengono considerati 3 elementi:

- Token.**dep_**: è l'indicazione della dipendenza sintattica
- Token.**lemma**: è la forma normale del termine considerato usata come filler
- Token.**pos_**: è l'indicazione del Part of Speech assegnato al lemma

In seguito, ottenuti i diversi filler per i 2 slot, si procede verso la disambiguazione, in modo da ricavare il synset più idoneo e dal quale ottenere il tipo semantico. La funzione "**disambiguate_sentence**" si occupa del processo di disambiguazione, invoca l'algoritmo di **lesk** che riceve in input il filler, il contesto, inteso come la frase da cui è tratto il filler, e l'indicazione del **pos tag** recuperato dalla fase di parsificazione. A partire dai synsets dei 2 filler si ricavano i supersensi sfruttando il grafo di wordnet. Il lesk può restituire valori **None**, in questo caso considereremo la coppia "**semantic_type1 – semantic_type2**" solo se entrambi gli elementi non sono nulli. Le coppie vengono salvate in una lista e infine si restituiscono i risultati finali.

Risultati

Di seguito vengono riportati i risultati della costruzione del significato con la teoria di Patrick Hanks. Significativo il confronto fra i verbi "hate" e"love", che possiamo immaginare abbiano un significato simile e che operino con le stesse costruzioni semantiche. Le coppie di tipi semantici sono state ordinate per frequenza di modo da evidenziare i comportamenti delle prime 10 più ricorrenti negli slot.

Risultato dell'analisi per il verbo **love:**

```
PS C:\Users\Michele\Desktop\Mei Projectii\EsercitazioneDiCaro\Esercizio 3> python .\main.py .\input\love.txt

total number of sentences:  1000
total semantic types couples:  496

First 10 semantic types per frequency:
Semantic Types:  (('Tops', 'Tops'), 115)
Semantic Types:  (('Tops', 'person'), 42)
Semantic Types:  (('Tops', 'cognition'), 31)
Semantic Types:  (('Tops', 'communication'), 24)
Semantic Types:  (('Tops', 'act'), 22)
Semantic Types:  (('Tops', 'artifact'), 22)
Semantic Types:  (('Tops', 'group'), 20)
Semantic Types:  (('Tops', 'time'), 14)
Semantic Types:  (('Tops', 'state'), 10)
Semantic Types:  (('Tops', 'attribute'), 9)

Number of non duplicated semantic type couples found:  116
```

Risultato dell'analisi per il verbo **hate:**

```
PS C:\Users\Michele\Desktop\Mei Projectii\EsercitazioneDiCaro\Esercizio 3> python .\main.py .\input\hate.txt

total number of sentences:  1000
total semantic types couples:  429

First 10 semantic types per frequency:
Semantic Types:  (('Tops', 'Tops'), 65)
Semantic Types:  (('Tops', 'person'), 39)
Semantic Types:  (('Tops', 'communication'), 23)
Semantic Types:  (('Tops', 'act'), 20)
Semantic Types:  (('Tops', 'group'), 19)
Semantic Types:  (('Tops', 'artifact'), 19)
Semantic Types:  (('person', 'Tops'), 17)
Semantic Types:  (('Tops', 'cognition'), 16)
Semantic Types:  (('Tops', 'state'), 16)
Semantic Types:  (('Tops', 'attribute'), 9)

Number of non duplicated semantic type couples found:  119
```

Salvo le differenze di frequenza, dovute senza ombra di dubbio alla composizione delle frasi nei 2 file, i risultati sono molto simili. La coppia più ricorrente è "Tops", "Tops". Questo supersenso si riferisce a circa 40 argomenti molto generali ad esempio: "entità" "oggetto", ecc. La coppia Tops, Person potrebbe essere associata ad una frase tipo: "Life hate me!".

Risultato dell'analisi per il verbo **create:**

```
PS C:\Users\Michele\Desktop\Mei Projectii\EsercitazioneDiCaro\Esercizio 3> python .\main.py .\input\create.txt

total number of sentences:  1000
total semantic types couples:  371

First 10 semantic types per frequency:
Semantic Types:  (('Tops', 'communication'), 18)
Semantic Types:  (('Tops', 'artifact'), 16)
Semantic Types:  (('artifact', 'artifact'), 12)
Semantic Types:  (('Tops', 'cognition'), 10)
Semantic Types:  (('Tops', 'person'), 10)
Semantic Types:  (('Tops', 'act'), 8)
Semantic Types:  (('act', 'cognition'), 7)
Semantic Types:  (('Tops', 'location'), 6)
Semantic Types:  (('person', 'artifact'), 6)
Semantic Types:  (('Tops', 'state'), 6)

Number of non duplicated semantic type couples found:  157
```

Per il verbo create è più facile trovare una frase tipo, ad esempio: "I created a chair". In questo caso la coppia al primo posto è "Tops" – "communication"