

Relazione Progetti semantica lessicale

Andrea Cacioli Matricola: 914501

[TOC]

Esercitazione 1 - WSD

Esercitazione sul word sense disambiguation utilizzando **Wordnet**. In questa prima parte esercitazione era richiesta la disambiguazione tra termini a partire dal contesto utilizzando l'algoritmo di Lesk.

Parte 1 - SemCor

A partire dal dataset SemCor, disponibile in NLTK, si é utilizzata **l'estrazione di nomi e verbi** della stessa frase da usare come contesto per l'algoritmo di Lesk.

Cosí facendo si ottiene una disambiguazione corretta dal 30% al 50% delle parole.

Parte 2 - WordSim353

Utilizzando il dataset che contiene associazioni di parole annotati con la similarit  semantica si procede in questo modo:

- Disambiguazione dei due termini usando uno come contesto per l'altro
- Calcolo dei seguenti 3 coefficienti di Similarit  tra synset:
 - Indice di Wu e Palmer
 - Indice di Vicinanza Semantica
 - Indice di Leacock e Chodorow
- Calcolo dell'indice di correlazione di Pearson e Spearman (scipy)

Risultati

```
#WUPALMER Pearson correlation PearsonRResult(statistic=0.31636166816161415,
pvalue=1.5783684150823945e-09) Spearman correlation Signifi-
canceResult(statistic=0.2788094053525254, pvalue=1.2391525466651858e-
07)
```

```
#LEACOCKCHODOROW Pearson correlation PearsonRRe-
sult(statistic=0.2962880264345914, pvalue=1.7576680839565154e-
08) Spearman correlation SignificanceResult(statistic=0.2265823173663636,
pvalue=1.978909912962172e-05)
```

```
#SEMANTIC DISTANCE Pearson correlation PearsonRResult(
  statistic=0.24862051813137037, pvalue=2.6635287435986987e-
  06) Spearman correlation SignificanceResult(
  statistic=0.2265823173663636, pvalue=1.978909912962172e-05)
```

Scelte implementative WSD

- Come **algoritmo di ricerca** é stato utilizzata una ricerca in ampiezza non informata implementata da me. Tale ricerca utilizza come relazioni del grafo le relazioni di iponimia e iperonimia.

Esercitazione 2 - Ngrams

Esercitazione sugli N-grammi utilizzando come corpus una serie di tweet dell'ex presidente degli stati uniti Trump.

A partire da tale corpus si é implementata prima una ricerca (lineare nella lunghezza del corpus, si vedano le scelte implementative) degli N-grammi e degli N-1-grammi. In questo modo si é potuto trovare la probabilità di generazione di una parola date le N parole precedenti nel seguente modo:

$$P(w|prevs) = \frac{P(w, prevs)}{P(prevs)} \simeq \frac{count(prevs, w)}{count(prevs)}$$

Si ottiene cosí una distribuzione di probabilità che é utilizzabile per simulare la generazione di tweet nello stile di Trump

Scelte implementative N-Grams

- In maniera naive avevo iniziato a fare un conteggio per ogni N-upla di parole possibili del dizionario. Tale algoritmo é altamente inefficiente poiché la matrice di transizione é sparsa. Nella seconda versione si calcola direttamente (in una singola “passata”) tutti gli N-grammi e gli N-1-grammi e poi si converte solo successivamente in probabilità.
- Si é implementata una funzione che data una distribuzione di probabilità qualsiasi, sottoforma di dizionario, simula tale variabile aleatoria. Ad Esempio:

```
probability_distribution = {
  "a": .1,
  "b": .5,
  "c": .4,
}
```

```
simulate_random_variable(probability_distribution)
```

restituisce un valore casuale in $\{a, b, c\}$ con probabilità descritta nel dizionario.

- Si utilizzano come delimitatori di frasi i simboli speciali `<s>` e `</s>`.

Preprocessing

È stato fatto poco pre-processing, in particolare si sono rimossi gli Username (“parole che iniziano con @”) e le si è sostituite col token `<user>`. Lo stesso è stato fatto per i link, sostituendoli col token `<link>`.

Utilizzo

Si possono specificare:

- Il numero di parole di una frase da generare
- N per la dimensione degli N-grammi
- Una finestra iniziale (prime parole di una frase da cui continuare a generare)
 - Alternativamente si può richiedere una finestra iniziale casuale

Esempio di Output

Ecco alcune frasi generate dal programma con lunghezza uguale a **50 parole** utilizzando **trigrammi**:

```
<s> <user> is known as a Connecticut politician bragging
that he was a great nominee gas prices and why George
Will is a total joke in political circles. Hence he
writes for Loser Salon. <user> </s> <s> Losers and
haterseven you as low and dumb as you are a total
```

```
<s> Really dumb <user> Begged my people for a ride.
$700 million washed down the drain. </s> <s> <user>
Danny--You're a total loser who will self-destruct just
like corrupt pol Shifty Schiff who fraudulently made up
lie)! </s> <s> <user> ignoring haters and losers a very
happy Easter! </s>
```