Linguistica Computazionale – Progetto finale

A.A. 2022-2023

Obiettivo

Realizzazione di **due programmi** scritti in Python o su Notebook (Jupyter, Colab, …) che utilizzino i moduli di NLTK per analizzare linguisticamente due corpora di testo inglese, confrontarli sulla base di alcuni indici statistici, ed estrarre da essi informazioni.

Creazione dei corpora

I due corpora devono essere creati rispondendo alle seguenti caratteristiche:

1) Devono essere in lingua **inglese**
2) Devono contenere almeno **5000 parole**
3) Ciascuno deve essere rappresentativo di uno specifico **genere testuale** (ad es. testo giornalistico, prosa letteraria, blog, social media, articoli scientifici, ecc.)
4) Devono essere salvati in un file ciascuno di testo semplice con **codifica UTF-8**

Programmi da sviluppare

Ciascuno dei due programmi deve anzitutto **leggere** i file e **analizzarne il contenuto** linguisticamente almeno **fino al Part-of-Speech Tagging**.
Se si sceglie di sviluppare il codice come un **file Python**, il programma deve prendere in input da **riga di comando** i file da analizzare.
Se si sceglie di sviluppare il codice come **Notebook**, è accettabile che il/i file sia/siano specificati all'interno del codice utilizzando un ***path relativo***.

Programma 1

Il codice sviluppato deve prendere in input i **due corpora**, effettuare le operazioni di annotazione linguistica richieste (sentence splitting, tokenizzazione, PoS tagging, lemmatizzazione), e produrre un confronto dei corpora rispetto a:

1) Numero di **frasi** e **token**;
2) **Lunghezza media delle frasi** in token e **lunghezza media dei token**, a eccezione della punteggiatura, in caratteri;
3) Numero di **Hapax** tra i primi 500, 1000, 3000 token, e nell'intero corpus;
4) Dimensione del **vocabolario e ricchezza lessicale** (Type-Token Ratio, TTR), calcolata per **porzioni incrementali di 200 token** (i.e., i primi 200, i primi 400, i primi 600, …);
5) Numero di **lemmi distinti** (i.e., la dimensione del vocabolario dei lemmi).

Programma 2

Il codice sviluppato deve prendere in input **un corpus**, effettuare le operazioni di **annotazione** richieste (sentence splitting, tokenizzazione, PoS tagging), ed estrarre le **seguenti informazioni**:

1) La sequenza **ordinata per frequenza decrescente**, con relativa frequenza, di:
    a. **10 PoS**, **bigrammi di PoS**, e **trigrammi di PoS** più frequenti

NB: risolvere il problema con **una singola funzione** che prenda in input il valore di n

b. **20 Sostantivi, Avverbi, e Aggettivi** più frequenti
NB: risolvere il problema con una **funzione che filtri la lista** di (token, PoS) prendendo in input la lista e la/le PoS su cui filtrarla

2) Estratti i **bigrammi** composti da **Aggettivo e Sostantivo** mostare:
   a. I **20 più frequenti**, con relativa frequenza
   b. I 20 con **probabilità condizionata massima**, e relativo valore di probabilità
   c. I 20 con **forza associativa** (Pointwise Mutual Information, PMI) **massima**, e relativa PMI
3) Considerate le frasi con una **lunghezza compresa tra 10 e 20 token**, in cui almeno la **metà** (considerare la parte intera della divisione per due come valore) dei token occorre **almeno 2 volte nel corpus** (i.e., non è un hapax), si identifichino:
   a. La frase con la **media della distribuzione di frequenza** dei token più **alta**
   b. La frase con la **media della distribuzione di frequenza** dei token più **bassa**
   c. La frase con **probabilità più alta** secondo un **modello di Markov di ordine 2** costruito a partire dal corpus di input
   NB: la media della distribuzione di frequenza dei token è data dalla **somma delle frequenze** (nel corpus) dei token della frase **diviso il numero di token della frase**

4) Estratte le **Entità Nominate** del testo, identificare **per ciascuna classe di NE i 15 elementi più frequenti**, ordinati per frequenza decrescente e con relativa frequenza.

Risultati attesi

Perché il progetto sia considerato idoneo, **devono essere consegnati** all'interno di una cartella compressa:

- I **due corpora**, come file di testo
- I **due programmi/notebook BEN COMMENTATI**
  - Nel caso si scelga di sviluppare programmi in **Python** (file .py), il **risultato** dell'esecuzione deve essere scritto in un **file di testo** (ben formattato) e **consegnato**. Quindi, dovranno essere consegnati tre file di output: uno per il primo programma, e due per il secondo.
  - Nel caso si scelga di sviluppare il codice attraverso un **Notebook**, questo deve essere **consegnato eseguito**. Per farlo, una volta eseguito il codice, è sufficiente esportarlo in formato .ipynb dal menù a tendina. Dovranno essere consegnati quindi 3 notebook: 1 per il primo programma, e 2 copie del notebook per il secondo programma, ciascuna eseguita su un corpus diverso.

Il codice/notebook **deve essere eseguibile** (prestare attenzione ai path assoluti) e replicare i risultati consegnati.
Il progetto **DEVE** essere svolto **INDIVIDUALMENTE**

Consegna

La consegna del progetto deve essere effettuata **almeno una settimana prima della data dell'esame scritto dell'appello a cui volete partecipare**. Il progetto va consegnato per e-

mail a alessandro.bondielli@unipi.it, paolo.pedinotti@phd.unipi.it, e alessandro.lenci@unipi.it.