Lorenzo Marinelli
Laurea Magistrale in Informatica Umanistica

Seminario di Cultura Digitale

Analisi automatica dei testi: le basi e gli strumenti.

A.A. 2012/2013

INTRODUZIONE

La linguistica computazionale è una materia interdisciplinare che tratta della modellazione statistica o "basata su regole" del linguaggio naturale sfruttando le potenzialità dei computer. Si può fissare la nascita di questa disciplina all'incirca negli anni '50-'60, quando Roberto Busa studia il possibile utilizzo del computer come mezzo di memorizzazione e analisi testuale, producendo il primo corpus elettronico. Negli anni '70 vediamo poi un'applicazione di metodi formali alle precedenti analisi. Il Natural Language Processing (NLP) prende spunto dai metodi deduttivi studiati da Noam Chomsky riguardo la grammatica generativa. La disciplina si sviluppò seguendo una corrente di pensiero che puntava sull'accumulare ingenti quantità di dati testuali da includere in banche dati utilizzabili poi per analisi successive. Queste collezioni prendono il nome di *corpora (*al singolare: *corpus*). Tramite questi dati e l'utilizzo del computer sarà possibile analizzare le varietà linguistiche e interessanti statistiche. Questo approccio ha seguito di pari passo l'evoluzione dei computer e dunque della capacità di immagazzinamento di dati e di velocità di elaborazione.

Questa relazione si propone di esaminare la base dell'analisi (dalla statistica alla semantica) dei testi, riportando esempi di risultati elaborati tramite strumenti open source su un piccolo corpus di carattere letterario in lingua inglese composto da materiale scaricato da internet. Analizzeremo alcune delle fasi fondamentali dell'elaborazione automatica del testo e citeremo alcuni validi lavori di ricerca effettuati nel campo.

LE FASI DELL'ANALISI DEL TESTO

Un testo presentato in un certo formato va innanzitutto ripulito da tutti quei dati che non sono utili alla nostra analisi. Ad esempio, da una pagina web scaricata con intenzione di analizzarne il testo andrebbero eliminate tutte le immagini e relative didascalie, i contenuti ripetuti, i menù di navigazione, il markup html, ecc. A questo punto è necessario analizzare i caratteri del testo per controllare che siano tutti conformi al set utilizzato nel file. Tutto il procedimento di preparazione del testo per una successiva analisi viene chiamato *normalizzazione.* Questa fase è molto importante in quanto un errato svolgimento della stessa potrebbe portare alla restituzione di dati falsati o superflui per i nostri scopi.

Tutto questo non sarebbe comunque possibile senza prima preparare il computer per far sì che i vari elementi che compongono il testo vengano individuati in maniera esatta. La parola, che per noi umani può essere definita in vari modi, per un computer non è infatti di comprensione banale o istantanea. Non è possibile spiegare ad un computer ad esempio che si tratta di un *"Complesso di fonemi, cioè di suoni articolati, o anche singolo fonema (e la relativa trascrizione in segni grafici), mediante i quali l'uomo esprime una nozione generica, che si precisa e determina nel contesto di una frase."*[1]. Partiamo quindi a descrivere quelle che sono le fasi preliminari dell'analisi di un testo da parte di un computer (che seguono l'obbligatoria "pulizia" prima descritta).

La *tokenizzazione* è il primo passo della nostra analisi computazionale e consiste nel dividere le sequenze di caratteri in unità minime di analisi dette *token.* I token comprendono svariate categorie di parti del testo (parole, punteggiatura, numeri, ecc) o possono anche essere delle unità complesse (come le date). Notare quindi che la definizione di token è ben distinta da quella di parola, in quanto questa è soltanto un suo sottinsieme. La tokenizzazione è un passaggio che a seconda della lingua che si analizza può essere un compito estremamente complesso. Nell'italiano ad esempio potremmo accontentarci di considerare token le sequenze di caratteri che non presentano spazi e i vari segni di punteggiatura. Invece, in lingue come il giapponese o il cinese in cui le parole non sono separate da

[1] Definizione presa dal Dizionario Treccani della Lingua Italiana, versione online.

spazi ma l'unione di diversi simboli possono cambiare completamente il significato, il compito è molto più complesso. Ma in generale anche nelle lingue con le parole separate da spazi occorre definire dei criteri precisi in quanto la punteggiatura è spesso ambigua:

N.A.T.O. è un unico token nonostante vi siano quattro punti a dividere le varie lettere; un tokenizzatore banale che considera il punto come conclusione di una frase esaminerebbe N.A.T.O. come 4 frasi distinte formate ognuna da una lettera. Otterrebbe così 8 token invece che uno soltanto, compromettendo poi la correttezza di una qualsiasi futura analisi del testo.

Dunque è necessario far comprendere al computer in che modo individuare questi casi particolari. La tokenizzazione si basa spesso su semplici euristiche:

- Le sequenze di stringhe alfabetiche ininterrotte fanno parte di un unico token;
- I token sono separati fra loro tramite spazi o simboli di punteggiatura;

Come abbiamo visto però queste non bastano per evitare errori. Si introduce quindi la necessità di utilizzare statistiche di testi precedentemente raccolti, *espressioni regolari*[2] o di analizzare le caratteristiche del testo o persino la semantica, quando si parla di linguaggi in scrittura continua[3]. Un testo correttamente[4] tokenizzato è già pronto per alcune analisi statistiche di livello basilare.

TOKEN TYPE

A partire dai token individuati nel testo è possibile esaminarne la ricchezza lessicale. Per fare questo si possono raggruppare i token secondo il loro tipo. Quando includiamo alcuni token in un certo *tipo* intendiamo indicare che fra questi c'è una somiglianza di qualche genere. Possiamo ad esempio considerare token appartenenti ad un certo tipo quelli graficamente identici (escludendo le differenziazioni per effettiva resa grafica tramite differenti stili). In questo caso si può decidere se normalizzare il testo rendendo tutto minuscolo, per non differenziare parole con lo stesso significato solo per la resa grafica diversa[5]. Questa resta comunque una scelta non obbligatoria. Si definisce dunque *vocabolario* di un testo l'insieme di parole tipo che ricorrono all'interno del testo. Per esaminare la ricchezza lessicale del testo si sfrutta la *Token Type Ratio*[6] *(*TTR). Il valore è compreso tra 0 e 1 e più alto è e più il testo è lessicalmente vario. Analizzando un piccolo Corpus contenente una larga parte dei racconti brevi dell'autore H.P. Lovecraft sono stati ricavati i seguenti dati:

Lunghezza Corpus	Vocabolario	TTR
535271	29453	0.0550244642433

Potremmo dire quindi che questo corpus non è lessicalmente vario, ma ciò è certamente dovuto al fatto che stiamo analizzando un grande insieme di racconti e se uno scrittore ha un suo stile e un suo modo di descrivere i fatti nei suoi racconti, lo farà utilizzando spesso le stesse parole. Inoltre va considerato che della lunghezza in token del corpus, circa 150000 token sono congiunzioni che

[2] Sequenze di simboli tramite le quali individuare stringhe di un certo tipo nel testo.

[3] La *scriptio continua* è presente nella storia dei testi scritti sin dall'antichità. Se volessimo analizzare testi latini o greci antichi non ancora trascritti con i dovuti spazi, avremmo problemi simili a quelli che si hanno con il cinese o il giapponese.

[4] La "correttezza" può essere valutata su testi di una certa dimensioni di cui si conoscono già i dati esatti, tramite *precision* e *recall* e altri controlli. Vedi note 16 e 17.

[5] N.B: le due parole pesca (intesa come frutto) e pesca (intesa come attività sportiva o economica) hanno significati differenti ma appartengono allo stesso token type.

[6] Si calcola come il rapporto tra il numero di token type che costituiscono il vocabolario e la grandezza complessiva del corpus.

nella stesura di un testo non possono certamente mancare. Ciò ci serve a capire che questo tipo di analisi va fatta su un singolo racconto o un singolo libro piuttosto che su un'intera raccolta di testi in quanto all'aumentare del testo il vocabolario segue una crescita sempre più lenta e quando andiamo a mettere insieme tutti i testi di un autore è normale scoprire che anche il miglior scrittore ha un vocabolario piuttosto limitato. Analizzando infatti un racconto singolo ("*The Call Of Cthulhu"*) possiamo ricavare questi dati:

Lunghezza Testo	Vocabolario	TTR
13275	3335	0.25

La TTR può essere analizzata anche in altri modi; ad esempio è possibile considerare il rapporto tra il numero di *hapax legomena* (parole che appaiono una sola volta nel testo) e la lunghezza totale di questo. Un valore vicino allo 0 in questo caso può indicare un testo ripetitivo.

Lunghezza Testo	Hapax Legomena	TTR
13275	2118	0,16

Esaminando i tokens e la loro distribuzione nel testo e ordinandoli secondo una lista in ordine decrescente per frequenza, indichiamo la posizione di questi come rango. Dall'esempio sotto riportato possiamo notare, come detto in precedenza, che le parole più frequenti sono congiunzioni, articoli, pronomi e altre utili nella stesura di una frase e nello stabilire relazioni interne tra coordinate e subordinate:

Top 20 delle parole più frequenti di "The Call of Cthulhu" di H.P. Lovecraft

1 - *the* - Frequenza: 744

2 - *of* - Frequenza: 523

3 - *and* - Frequenza: 479

4 - *a* - Frequenza: 235

5 - *in* - Frequenza: 212

6 - *to* - Frequenza: 205

7 - *was* - Frequenza: 170

8 - *had* - Frequenza: 155

9 - *I* - Frequenza: 113

10 - *that* - Frequenza: 109

11 - *which* - Frequenza: 94

12 - *his* - Frequenza: 87

13 - *he* - Frequenza: 84

14 - *it* - Frequenza: 82

15 - *with* - Frequenza: 82

16 - *from* - Frequenza: 79

17 - *for* - Frequenza: 76

18 - *at* - Frequenza: 69

19 - *on* - Frequenza: 69

20 - *by* - Frequenza: 68

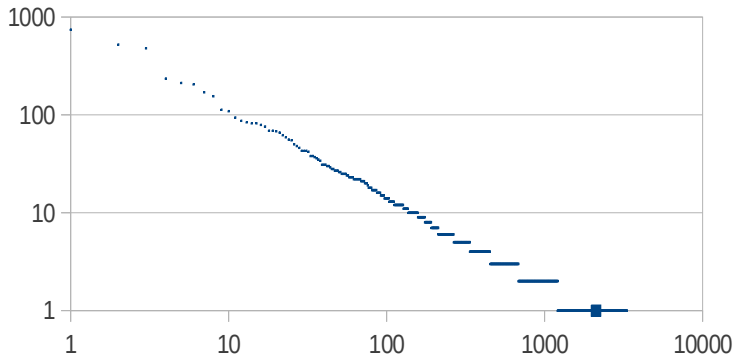


Grafico 1: distribuzione dei tokens in base alla loro frequenza (grafico in scala logaritmica).

Dal **grafico 1** si nota inoltre che sono molti i tokens che appaiono poche volte nel testo mentre sono pochi quelli che appaiono di frequente. Questo particolarità è costante in ogni testo che si può analizzare. Un'altra interessante analisi consiste nel vedere in che modo cresce il vocabolario di un testo. Il **grafico 2** mostra questa crescita nel caso dello stesso racconto di Lovecraft utilizzato sopra. Notiamo un andamento *più o meno* lineare (in realtà infatti non si può affermare che la crescita del vocabolario sia propriamente lineare) che ci informa del fatto che all'aumentare del numero di tokens del testo aumenta anche la grandezza del vocabolario. Questi due dati sono infatti strettamente correlati ed è normale vedere questo fenomeno in qualsiasi tipo di testo. In generale il vocabolario di un testo aumenta sempre al crescere di questo, ma rallenta sempre più in quanto vi sono molte parole che saranno costrette a ripetersi (le parole grammaticali) ed è evidente che c'è un limite alle parole esistenti, ma non c'è alcun limite nel modo di combinarle per dare un senso alle frasi. Un'interessante analisi potrebbe essere fatta in testi di diversi generi per scoprire se la crescita del vocabolario segue uno schema particolare.

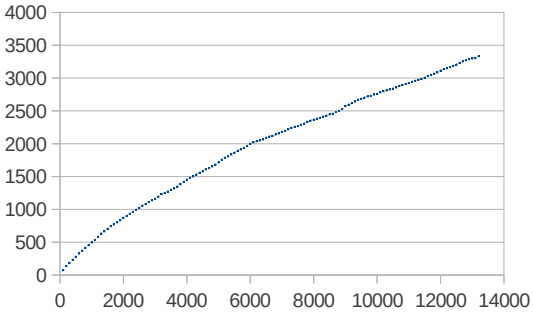


Grafico 2 : crescita del vocabolario in “The Call of Cthulhu”

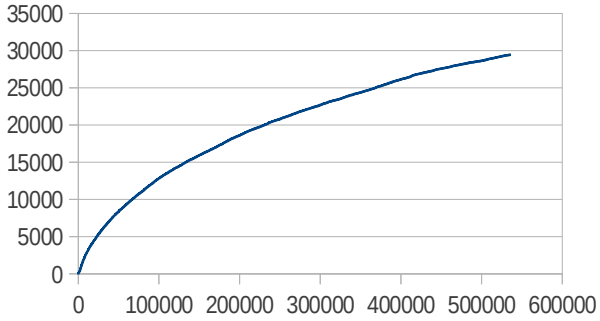


Grafico 3: crescita del vocabolario nel corpus dei racconti di Lovecraft

Si noti in quest'altro grafico (**3**) che mostra la crescita del vocabolario dell'intero corpus dei racconti brevi di Lovecraft. Come detto in precedenza la crescita non è lineare, ma infatti ha un'impennata iniziale per poi crescere sempre più lentamente. Il vocabolario raggiunge infatti metà della sua grandezza già quando siamo a circa un quinto del corpus intero.

CORPORA

Il contenuto di un corpus è determinato da operazioni accurate di selezione basate sul scegliere contenuti in grado di soddisfare poi le ricerche che si è intenzionati a fare su di esso. Quindi la raccolta di testi effettuata per formare un corpus è sempre mirata a creare una serie di dati adatta agli scopi prefissati. La classificazione dei corpora si basa su alcuni punti, quali la generalità, la modalità , la cronologia , la lingua , l'integrità dei testi e infine la codifica digitale dei testi. Quando parliamo di generalità possiamo avere:

- Un corpus specialistico: contiene testi di un certo tipo di linguaggio (giornalistico, sportivo, ecc) e punta allo studio di questo ristretto dominio.
- Un corpus generale: presenta diverse varietà di uno stesso linguaggio e punta a rappresentarne tutte le sfaccettature.

Un corpus può avere diverso modalità:

- Corpus di scritto
- Corpus di parlato
- Corpus misto
- Corpus audio
- Corpus audio-video

Molto interessante è anche la cronologia dei corpora in quanto permette di fare analisi testuali sul modo di scrivere o di parlare di una certa epoca e magari in un corpus che presenta testi di una certa regione d'Italia distribuiti in un certo periodo di tempo può aiutare a comprendere l'evoluzione della lingua.

- Corpus sincronico: raccoglie testi che rappresentano una certa fase di un linguaggio.
- Corpus diacronico: utile a descrivere il mutamento di una certa lingua, raccoglie testi di diverse epoche.

Anche la lingua gioca un ruolo importante nella classificazione dei corpora e nel loro utilizzo. Infatti è possibile avere corpora con documenti di una sola lingua se ci si vuole concentrare su questa o corpora bilingue (o plurilingue) per effettuare operazioni di comparazione delle traduzioni e magari per creare sistemi di apprendimento automatico per traduzioni automatiche (problema molto complesso e ancora non risolto in maniera ottimale).

Infine un corpus può essere comprensivo di testi non totalmente integri e può essere annotato in qualche modo e per qualche scopo. Ad esempio potrebbe essere comprensivo di codice di markup XML con l'obiettivo di sottolineare la struttura linguistica del testo (morfologia, sintassi, ecc).

Quando si decide di creare un corpus lo si fa sempre con uno scopo preciso ed è dunque obbligatorio porre attenzione su tutti i punti sopra citati, cercando di inserire da subito le informazioni necessarie per la nostra ricerca. Un'ultima particolarità molto importante è rappresentata dalla dimensione dei corpora: nei primi anni di ricerche si parlava di corpora con milioni di parole al loro interno, mentre ormai con l'aumento delle dimensioni dei supporti fisici siamo sull'ordine dei miliardi di parole. Infatti un corpus di piccole dimensioni potrebbe non risultare rappresentativo per un certo tipo di linguaggio, se la nostra intenzione è un'analisi approfondita della lingua. Se però la nostra intenzione è quella di analizzare tutti i testi scritti da un solo autore per comprendere il suo modo di scrivere, potrebbe capitare di avere un corpus di piccole dimensioni completamente rappresentativo.

Sono disponibili, alcuni anche per utilizzo gratuito, molti corpora di un gran quantitativo di lingue. Un esempio è il Brown Corpus, corpus di riferimento bilanciato sincronico della lingua inglese americana creato nel 1961 e pubblicato nel 1964. è stato accorpato nell'ICAME[7], un corpus formato da tanti corpora, creato per scopi di ricerca linguistica. Un esempio italiano è dato dal CORIS (Corpus di Italiano Scritto) , progetto partito all'università di Bologna nel 1998, contenente 130 milioni di parole dell'italiano contemporaneo. *La Repubblica* ha formato un Corpus consultabile

[7] http://icame.uib.no/

online[8] contente il linguaggio giornalistico italiano degli articoli apparsi nello stesso quotidiano a partire dal 1985 fino al 2000.

PROBABILITA' NEL LINGUAGGIO

La definizione classica di probabilità è la seguente: *la probabilità di un evento è il rapporto tra il numero dei casi favorevoli all'evento e il numero dei casi possibili, purché questi ultimi siano tutti equiprobabili.* Dato un corpus e preparato questo all'analisi linguistica tramite computer con i processi descritti in precedenza sarà possibile esaminare il testo con uno scopo di calcolo delle probabilità.

Se nel caso di un dado a *x* facce la probabilità di ottenere una certa faccia è data dal rapporto tra il numero di casi favorevoli, ovvero 1, e i casi possibili, ovvero *x*, avremo che in un testo la probabilità di trovare una certa parola sarà data dal rapporto tra la frequenza della parola nel testo e la lunghezza del corpus. Nel caso di *"The Call of Cthulhu"* ad esempio la probabilità di trovare *the* nel testo sarà pari a :

$$P(\text{"}the\text{"}) = \frac{frequenza(\text{"the"})}{|C|} = \frac{744}{13275} = 0{,}05 = 5\%$$

Nel caso di un corpus ipotetico contenente tutti i documenti scritti in italiano nella storia, se calcolassimo la probabilità che appaia la parola "il" otterremmo la probabilità di questa di apparire nel linguaggio italiano. Ovviamente questa probabilità così calcolata sarebbe solo una sovrastima in quanto i dati estratti da un corpus non saranno mai sufficienti per studiare una stima accurata di queste probabilità nel linguaggio. Questa stima viene anche chiamata *Maximum Likelihood Estimation* (MLE). Studiare queste probabilità non è comunque così interessante in quanto si tratta soltanto di definire in quale quantità una parola è presente all'interno di un testo. Potrebbe essere invece più interessante studiare le probabilità *condizionate.*

Con *probabilità condizionata* si indica la probabilità che un evento si verifichi dato il verificarsi di un altro evento. Nel caso dei testi un esempio potrebbe essere cercare "la probabilità che un aggettivo segua un articolo". È fondamentale comprendere che questo tipo di calcoli tengono conto del fatto che queste probabilità sono *dipendenti* fra di loro, infatti non stiamo analizzando un insieme di stringhe casuali, ma un testo scritto con un significato preciso dato dalla sequenza scelta delle parole. Calcoliamo questa probabilità in "The Call of Cthulhu":

$$P(Agg\,|Art) = \left(\frac{frequenza(Art)}{|C|}\right) * \left(\frac{frequenza(Art,Agg)}{frequenza(Art)}\right) = \left(\frac{1362}{13275}\right) * \left(\frac{332}{1362}\right) = 0{,}02 = 2\%$$

Questo genere di calcoli può essere molto utile se applicato ad esempio a programmi di suggerimento automatico. Partendo da un corpus della lingua italiana e elaborando una tabella contenente tutte le coppie di token (bigrammi) e la loro frequenze e dunque la relativa probabilità di apparire in un testo (come sopra) potremmo valutare la possibilità di suggerire ad un utente la parola successiva a quella che sta scrivendo. La stessa cosa può essere fatta con gruppi di più parole (suggerire ad esempio le successive 5 o 6 parole) o con i singoli caratteri (suggerire la successiva lettera da scrivere).

Per calcolare probabilità su sequenze di stringhe più lunghe è necessario ricorrere a modelli più complessi. Si utilizzano le *catene di Markov* che permettono di stimare le probabilità di una frase di

[8] http://dev.sslmit.unibo.it

apparire in un testo. Stabilito l'ordine che si vuole utilizzare, si calcola la probabilità di una frase come il prodotto delle probabilità congiunte tra i vari bigrammi che compongono la frase (nel caso di ordine 1).

$$P(A_1, \dots, A_n) = P(A_1) * P(A_2|A_1) * P(A_3|A_2) * \dots * P(A_n|A_{n-1})$$

Applicando una catena di Markov di ordine 1 sul corpus di Lovecraft abbiamo calcolato la frase più probabile nel testo. Il risultato è:

Ph'nglui mglw'nafh Cthulhu R'lyeh wgah'nagl fhtagn.

La frase è scritta in un linguaggio inventato dall'autore e si tratta di una litania ritualistica. Non è sorprendente il fatto che questa frase risulti molto probabile nonostante la sua stranezza in quanto la sequenza di parole è effettivamente formata da bigrammi con una forte dipendenza reciproca. Ovviamente salendo con l'ordine delle catene di Markov aumenta la precisione dei calcoli.

Questi meccanismi possono basarsi anche su altre caratteristiche desumibili da un testo, come la Mutual Information, una quantità che misura la dipendenza di due variabili (nel nostro caso di due token). Calcolando la Local Mutual Information possiamo definire, sfruttando anche la frequenza di un certo bigramma in un testo (differentemente dalla semplice MI), in che modo due parole sono "legate". Nel testo utilizzato per le analisi troviamo ad esempio un valore di 461.39 per il bigramma "old man". Ciò indica che le due parole sono fortemente dipendenti fra loro. La LMI si calcola in questo modo:

$$LMI(u, v) = f(< u, v >) * log_2 \frac{p(u, v)}{p(u) * p(v)}$$

Il calcolo della MI è necessario per la ricerca delle *collocazioni* di un testo, ovvero le combinazioni di 2 o più parole caratterizzate da un elevato grado di associazione reciproca, determinata dalla tendenza a ricorrere una accanto all'altra, ovvero a co-occorrere (Sinclair, 1991).

L'utilizzo del calcolo della probabilità sui testi è sfruttabile per un gran numero di altre applicazioni. Un esempio importante è quello dei software di traduzione automatica. Per permettere ad un computer di effettuare questo tipo di elaborazioni è comunque necessario un lungo procedimento di training e di raccolta dati. Basti pensare al meccanismo di suggerimento automatico di Google, che genera le sue previsioni basandosi su enormi quantità di dati ricavati dalle ricerche precedentemente effettuate da tutti gli utenti che utilizzano il motore di ricerca. Considerando che parliamo di circa 500 milioni di utenti[9] è facile intuire che si necessità di un hardware ad alta funzionalità e di algoritmi ottimizzati per l'elaborazione di tutti questi dati.

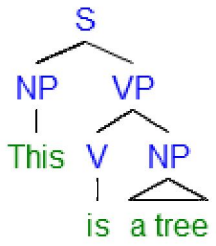
ANALIZZARE IL TESTO

Per effettuare l'analisi di un testo abbiamo già detto come sia necessario suddividere in tokens il nostro documento. Tramite questi tokens sarà poi possibile effettuare ulteriori operazioni, diverse da quelle statistiche. Parliamo di:

- Analisi morfologica: ai tokens sono associate tutte le possibili letture morfologiche utilizzando un dizionario delle forme (lemma, tratti, ecc.)

[9] Dati relativi al 2012, resi noti da Google.

- Tagging delle parti del discorso: operazione complessa nella quale a partire dall'analisi morfologia si andranno a disambiguare gli eventuali risultati contrastanti (es: “danno” potrebbe essere sia un nome che un verbo e l'analizzatore morfologico li restituirà entrambi come validi). Questa fase dell'analisi di un testo è delicata e per riuscire nel compito si avvale dell'utilizzo di *feature* (carattestiche) dei tokens. Queste feature si distinguono tra locali, contestuali e globali. Molto interessanti e utili sono le feature contestuali, che analizzano i tokens precedenti o successivi a quello che si intende identificare, così da comprendere il contesto nel quale si trova e determinare correttamente la parte del discorso rappresentata.
- Analisi delle dipendenze: si occupa di identificare soggetto, predicato, oggetto, ecc nelle frasi. Anche questo è un compito estremamente complesso e delicato.
- Analisi Semantica del testo, atta alla comprensione del significato vero e proprio delle parole e delle frasi che esse compongono.



Sfruttando i risultati del POS Tagging è possibile calcolare la *densità lessicale* di un testo, un dato molto interessante che avrà un valore compreso fra 0 e 1, indicando, se più vicino al massimo valore, una certa ricchezza di parole lessicali[10] all'interno del testo rispetto alle parole grammaticali. Un testo è denso se ricco di parole piene e spesso un testo più denso è di più difficile comprensione, per quanto ne riguarda la complessità, specialmente se si tratta di un testo specialistico. Per fare un esempio numerico, valutiamo il fatto che la densità lessicale del parlare quotidiano si aggira intorno al valore di 0,3 o 0,4. I testi matematici hanno una invece densità lessicale di 0,7 (*M. Dodman 2006*). Le analisi effettuate sul Corpus dei racconti di Lovecraft ha restituito una densità lessicale di 0.50. Ciò indica che l'intero corpus contiene un linguaggio medio-alto, come ci si poteva aspettare, ma essendo di carattere letterario rivolto al pubblico (Lovecraft pubblicava in riviste quali *Weird Tales*[11]) non raggiunge i livelli dei testi didattici. Effettuando l'analisi sui singoli racconti ci vengono restituiti risultati che variano da un minimo di 0,34 a un massimo di 0,54. La media comunque si attesta su 0,50 in quanto in realtà il caso del racconto con valore minimo (*A Reminiscence Of Dr. Samuel Johnson*, 1917) è un caso isolato.

Il livello più elevato e complesso della linguistica computazionale è probabilmente rappresentato dalla semantica computazionale e tutto ciò che essa comprende. Questo ramo punta alla creazione di modelli formali e computazionali per la rappresentazione del significato e metodi per l’acquisizione e l’estrazione di informazione semantica dai testi. Diversi sono gli ambiti di applicazione che vanno dalla creazione di risorse lessicali, all’analisi semantico-lessicale, all’estrazione di contenuto semantico dai testi. Il significato di una parola per la nostra mente è rappresentato dal concetto che questa esprime, mentre sarebbe più facile parlare di significato come definizione all’interno di un vocabolario. Per un computer è necessaria la creazione di modelli in grado di dare una rappresentazione valida del significato lessicale di una parola. Si distinguono in questo senso i *modelli relazionali*[12] e i *modelli a frame*[13].Nel primo caso il significato delle parole è determinato dalle sue relazioni[14] con il significato di altre parole. Nel secondo caso il significato di una parola è definito tramite i contesti nel quale la si può trovare (con ovvia presentazione di più contesti se parliamo di lemmi che presentano ambiguità).

[10] Denominazione impropria di quegli elementi del linguaggio, detti anche parole piene (come i sostantivi, gli aggettivi, i verbi, gli avverbi) che hanno un proprio contenuto semantico autonomo. (Treccani)
[11] Rivista pulp statunitense di racconti horror e fantascientifici, stampata a partire dal 1923.
[12] Esempio: WordNet
[13] Esempio: FrameNet
[14] Sinonimia, iperonimia, meronimia, ecc...

STRUMENTI PER L'ANALISI

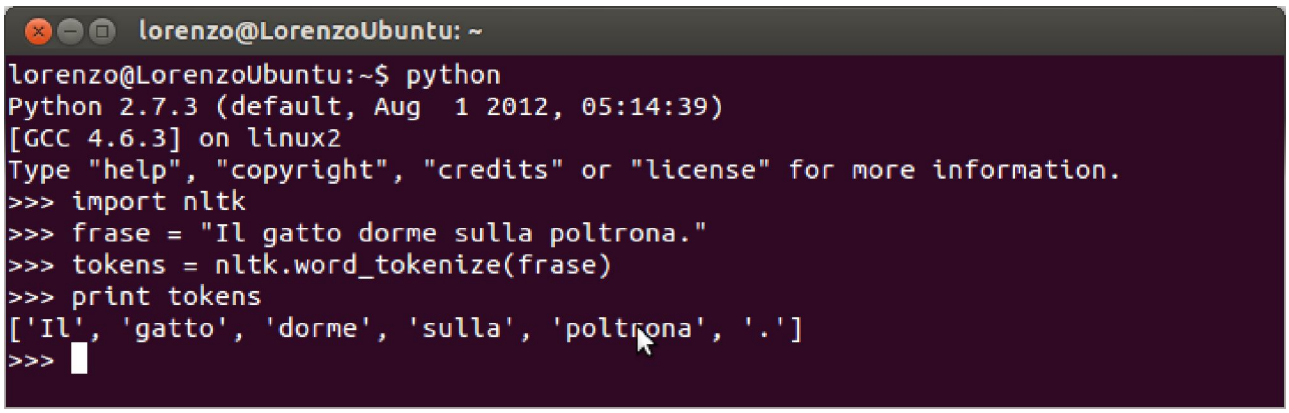


Figura 1 – Esempio di tokenizzazione tramite libreria NLTK su terminale Ubuntu

Per le semplici analisi presentate come esempi durante i precedenti paragrafi sono state utilizzate le librerie di NLTK (Natural Language Toolkit) per il linguaggio di programmazione Python. Queste librerie contengono un gran numero di funzioni utili alla classificazione, tokenizzazione, stemming[15], POS tagging, parsing e analisi semantica. Si tratta di librerie Open Source e ben documentate, utilizzabili in qualsiasi sistema operativo. Con una minima conoscenza della sintassi del Python è possibile creare software per l'analisi del testo, sfruttando gli strumenti di base forniti da NLTK.

Nell'esempio della figura 1 possiamo vedere come ottenere i tokens di una frase con un semplice comando. NLTK basa il suo funzionamento su training corpus dai quali ha appreso in che modo identificare i token di una frase, non cadendo negli errori dati dalle ambiguità che si possono riscontare (come si è detto in precedenza). In realtà anche quella che sembra un compito semplice come la tokenizzazione non è privo di errori. Accade spesso infatti di trovare token che non sono stati riconosciuti come tali, ma comunque in quantità minime. NLTK contiene inoltre alcune basi per il Named Entity Recognition Task, ovvero per il riconoscimento e la catalogazione delle entità nominate. È possibile però riscontrare nello strumento numerosi errori come la confusione da tra nomi di luogo o di persona. Nel racconto "*Azathoth*" (1922) sempre di Lovecraft che consta di soli 515 token, su 5 entità riconosciute come tali, nessuna è corretta: si tratta infatti di aggettivi o sostantivi catalogati come luoghi. Un progetto dell'università di Stanford ha creato invece una versione scritta in Java per questo task ottenendo buoni risultati. Applicando ad esempio lo stesso strumento sullo stesso racconto otterremo ben altri risultati. Il progetto citato si propone anche di riconoscere, oltre a nomi di persona e di luogo, anche dati *temporali*, *di organizzazioni*, *di denaro* e *date*. Proviamo a comparare i strumenti su un racconto più lungo, riprendendo come esempio "*The Call of Cthulhu*", studiandone i risultati e valutandone *precision*[16] *(*misura di esattezza*)* e *recall*[17] *(*misura di completezza*)*. Utilizzeremo i due strumenti limitandoci alla ricerca dei nomi di persona e di luogo. I successivi valori ci aiuteranno a comprendere la differenza del potenziale degli strumenti:

	Stanford NER	**NLTK**
Precision	242 / 247 = 0,98	214 / 278 = 0,77
Recall	242 / 251 = 0,96	214 / 251 = 0,85

[15] Processo di riduzione della forma flessa di una parola alla sua forma radice, detta tema.

[16] PRECISION = $risultati\ corretti / risultati\ corretti + risultati\ errati$

[17] RECALL = $risultati\ corretti / risultati\ corretti + risultati\ mancati$

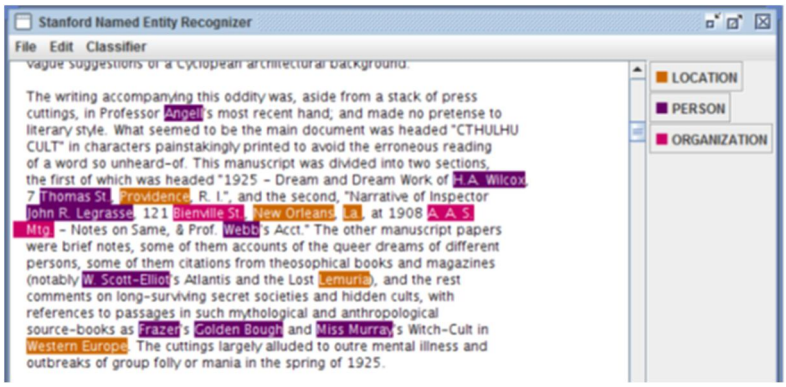


Figura 2 : Stanford Named Entity Recognizer, esempio di applicazione e risultato.

I risultati restituiscono un dato indicante il fatto che S-NER è più completo e preciso rispetto allo strumento di NLTK. La precisione e la completezza assoluta non viene raggiunta in ogni caso ma S-NER ci si avvicina molto a dispetto di NLTK, che tende spesso a sbagliare indicando come nomi di persona parole qualsiasi quali sostantivi o aggettivi (ciò che si potrebbe notare è che queste hanno sempre l'iniziale maiuscola e che forse viene dato troppo peso alla stessa nella ricerca delle entità nominate in questo caso).

PROGETTI

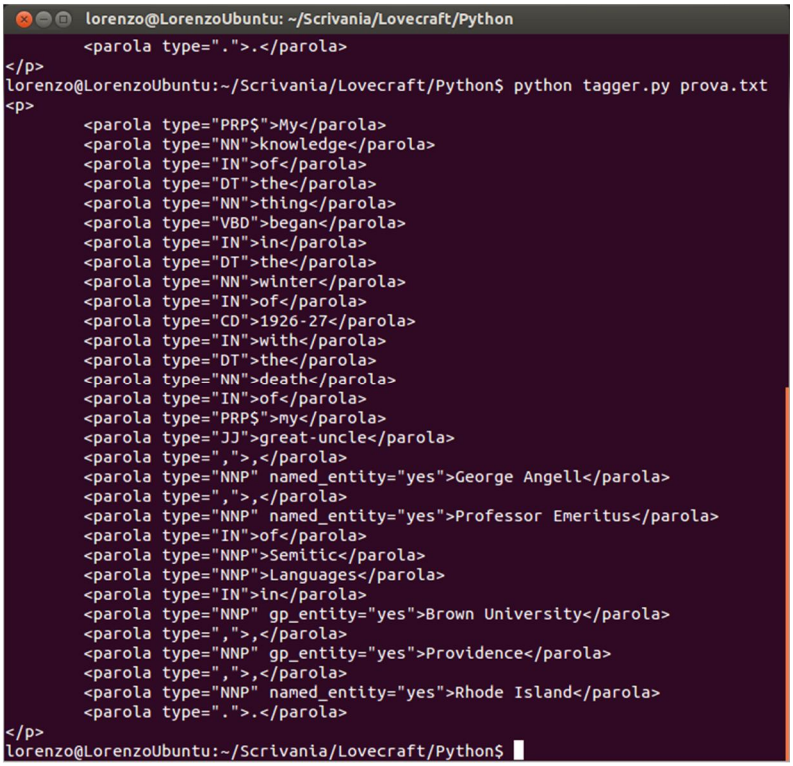


Figura 3 – Esempio di codifica automatica XML-TEI

Un interessante progetto potrebbe consistere nella creazione di un software in grado di analizzare un testo secondo criteri preselezionati ed effettuarne una codifica XML/TEI. Il programma potrebbe aiutare nel creare una base per una successiva codifica. Tramite i risultati di POS Tagging e Named Entity Recognition verrebbero generati dei tag che tengano conto dei dati ottenuti, generando così una codifica automatica. La creazione di un simile strumento dovrà ovviamente tenere conto di tutte le ambiguità possibili riscontrabili in un normale testo e dovrà comprendere tutte quelle fasi di cui abbiamo trattato in precedenza. Lo strumento sarebbe utile per velocizzare la codifica XML/TEI di un testo, lasciando che il computer individui quelle che sono le informazioni che si possono trarre direttamente dal testo e lasciare all'utente la possibilità di modificare in seguito la codifica aggiungendo le informazioni che desidera. Il software dovrebbe ovviamente comprendere la possibilità di scegliere un certo numero di impostazioni che si adattano alle basi dell'XML/TEI. Un esempio: gli elementi di un testo codificato hanno spesso un identificatore, che è scelto da colui che effettua la codifica; si potrebbe quindi decidere se lasciare che il computer li generi da solo o se lasciare degli spazi vuoi appositamente per far sì che ci pensi poi l'utente a definirli.

BIBLIOGRAFIA

- *Bird, S., Klein, E., & E. Loper, Natural Language Processing with Python (Analyzing Text with the Natural Language Toolkit), O'Reilly Media, 2009*
- *Christopher D. Manning and Hinrich Schütze. 1999. Foundations of Statistical Natural Language Processing. Cambridge, MA: MIT Press.*
- *D. Jurafsky & J.H. Martin, Speech and Language Processing, Prentice Hall, 2000*
- *Lenci, S. Montemagni, V. Pirrelli, Testo e computer. Elementi di linguistica computazionale, Carocci, 2005*
- *Sapere linguistico e sapere matematico, in Sbaragli S. (a cura di) La matematica e la sua didattica, Carocci Faber Roma 2006*
- *Sinclair, J. 1991. Corpus, concordance, collocation: Describing English language. Oxford: Oxford University Press.*
- *TEI P5: Guidelines for Electronic Text Encoding and Interchange, edited by Lou Burnard and Syd Bauman, 2013*

I testi di *H.P. Lovecraft* su cui sono state effettuate le analisi per fornire eventuali esempi sono stati scaricati in formato *.txt* da: http://www.psy-q.ch/lovecraft/html/