

Tecnologie Semantiche

Final Project

2021/2022

**Team 17**

Immagine che contiene testo, clipart

Descrizione generata automaticamente****

Sommario

[1 Executive Summary 3](#_Toc109389044)

[1.1 Source Code 3](#_Toc109389045)

[1.2 Analisi del Contesto 3](#_Toc109389046)

[2 Soluzione Adottata 4](#_Toc109389047)

[2.1 Riepilogo 4](#_Toc109389048)

[2.2 Word Tokenization 5](#_Toc109389049)

[2.3 Stop Word Removal 5](#_Toc109389052)

[2.4 Part of Speech Tagging 6](#_Toc109389054)

[2.4.1 Token and Tag Correction 6](#_Toc109389055)

[2.5 Lemmatization 7](#_Toc109389056)

[2.6 Semantic Information Extraction 8](#_Toc109389057)

[2.6.1 Noun Phrase Chunking 8](#_Toc109389058)

[2.6.2 Word Meaning Extraction 10](#_Toc109389059)

[2.6.3 Query on Dbpedia 11](#_Toc109389060)

# Executive Summary

Partendo da un file testuale scritto in linguaggio naturale, estrarre le informazioni semantiche rilevanti in modo da produrre delle triple *Soggetto – Predicato - Oggetto*.

## Source Code

Il codice sorgente e il testo utilizzato sono reperibili nel seguente repository GitHub: <https://github.com/GiuseppeGaetano/TecnologieSemantiche.git>.

## Analisi del Contesto

Essendo il testo utilizzato per operare l’estrazione delle informazioni semantiche proveniente dal libro *“Harry Potter and the Prisoner of Azkaban”*, ed essendo tale testo scritto in lingua inglese, sono state costruite delle liste contenenti i nomi e cognomi dei personaggi presenti nel libro e dei nomi dei luoghi descritti ad una lista dei possibili titoli usati nella lingua inglese per una persona[[1]](#footnote-1) in modo da operare delle correzioni in seguito al *POS Tagging*.

# Soluzione Adottata

## Riepilogo

Figura 1: Pipeline Soluzione adottata

Partendo dal testo in linguaggio naturale, è stata operata:

* la tokenizzazione sulle parole, in modo da ottenere una lista in cui ogni elemento rappresenta una parola del testo;
* la rimozione delle *stop word*, in modo da eliminare le informazioni meno rilevanti per focalizzare l’analisi sulle informazioni più importanti;
* il *Part of Speech Tagging*, in modo da associare ad ogni parola un tag che ne identifichi il tipo;
* la lemmatizzazione, in modo da ottenere i lemmi delle parole ottenute essendo questi sufficienti per analizzare il senso della parola[[2]](#footnote-2);

Infine, con le informazioni così ottenute, abbiamo operato le operazioni di:

* *Noun Phrases Chunking*, in modo da comprendere la semantica del testo;
* *Querying su Dbpedia*, in modo da ottenere le informazioni ricavate da Dbpedia sulle varie parole;
* *Word Meaning Extraction*, in modo da ottenere le definizioni di *Wordnet* delle parole analizzate;
* *Triples Extraction*, in modo da ottenere delle triple *Soggetto – Predicato – Oggetto*.

## Word Tokenization

Partendo dal testo in linguaggio naturale, attraverso la libreria nltk, sono stati estratti i token relativi alle varie parole tramite la funzione word\_tokenize(text), che restituisce il testo dato in input come lista di parole (token).



## Stop Word Removal

Partendo dalla lista di token ottenuta al passo precedente, abbiamo operato la rimozione delle così dette *stop word[[3]](#footnote-3)*, ottenute tramite la funzione stopwords.words(‘english’) e andando ad eliminare da tale lista tutti i verbi[[4]](#footnote-4).

Tale operazione è stata effettuata in quanto queste sono disponibili in abbondanza in qualsiasi lingua umana. Rimuovendole quindi, rimuoviamo le informazioni di basso livello dal nostro testo per dare maggiore attenzione alle informazioni importanti, ovvero la rimozione di tali parole non comporta alcuna conseguenza negativa e permette di ridurre definitivamente la dimensione del set di dati e quindi di ridurre il tempo di esecuzione totale a causa del minor numero di token utilizzati.

La scelta di non eliminare i verbi presentI nella lista delle stop words è nata dal fatto che operando una rimozione anche di tali parole, risultava impossibile succesivamente fare l’estrazione delle triple *Soggetto – Predicato – Oggetto* essendo praticamente assenti i predicati.



## Part of Speech Tagging

Ottenuta la lista dei token, abbiamo operato il *Part of Speech Tagging (PoS Tagging)* attraverso la funzione pos\_tag(tokenized\_text).

Tale operazione classifica le parole in un testo in corrispondenza di una parte particolare del discorso, a seconda della definizione della parola e del suo contesto, associando ad ognuna di queste un tag che ne identifichi il tipo. Inoltre, tale operazione serve a risolvere in una certa misura le ambiguità lessicali che possono verificarsi quando una parola può assumere diversi significati a seconda del contesto.

### Token and Tag Correction

Dopo aver effettuato il *PoS Tagging*, è stato effettuato un controllo sulla qualità dei token e dei relativi tag associati, e abbiamo notato alcuni problemi:

* Ad alcuni token erano stati associati tag sbagliati[[5]](#footnote-5);
* Alcune sequenze di parole che facevano riferimento alla stessa entità venivano divisi e classificati individualmente[[6]](#footnote-6);
* I titoli associati alle persone venivano divisi dall’entità relativa e classificati individualmente.

Per risolvere tali problemi, sono state definite tre diverse liste:

* ch\_list, contenente nomi e cognomi dei personaggi presenti nel libro;
* places, contenente i nomi dei vari luoghi descritti nel libro;
* ch\_titles, contenente tutti i possibili titoli che possono essere associati ad una persona nella lingua inglese.

Tali liste sono state utilizzate per associare il tag *NNP* a tutti i token presenti nelle prime due liste, e per fare in modo da unire i token relativi ad una stessa entità in un unico token[[7]](#footnote-7).

## Lemmatization

Dopo aver ottenuto la lista dei token a cui sono stati associati il relativo tag, abbiamo operato la lemmatizzazione, procedura utilizzata per ottenere la forma radicale della parola, facendo uso del vocabolario e dell'analisi morfologica di tali token.

Tale operazione è stata effettuata attraverso due funzioni:

* WordNetLemmatizer().lemmatize(word, pos), che restituisce il lemma della parola fornita (word) in base al suo tipo (pos);
* get\_wordnet\_pos(word), che data in input la coppia (word, pos), restituisce la stringa che ne identifica il tipo in base al tag applicato in fase di *Pos Tagging*.

## Semantic Information Extraction

Alla fine della pipeline presentata, composta dalle operazioni di tokenizzazione, rimozione delle stop word, PoS Tagging e lemmatizzazione, abbiamo provveduto ad effettuare l’estrazione delle informazioni semantiche in tre modi diversi:

* *Word Meaning Extraction;*
* *Query* su *Dbpedia;*
* *Noun Phrase Chunking;*
* *Triples Extraction*

### Word Meaning Extraction

Partendo dalla lista dei token ottenuta in uscita alla pipeline precedentemente descritta, abbiamo realizzato tre liste, contenenti rispettivamente tutti i token etichettati come *noun*, come *adjective* e come *verb* e, per ognuni elemento di queste liste è stata effettuata l’operazione di estrazione del significato delle parole attraverso l’utilizzo di wordnet.

WordNet è un grande database lessicale di inglese in cui nomi, verbi, aggettivi e avverbi sono raggruppati in insiemi di sinonimi cognitivi detti *synsets* interconnessi per mezzo di relazioni concettuale-semantica e lessicale, ognuno dei quali esprime un concetto distinto.

Attraverso la funzione wordnet.synsets(subject) sono stati estratti i *synset* associati alla parola fornita in input (subject), e per ognuno di questi sono state estratte informazioni sulla loro definizione oltre a degli esempi grazie alle funzioni syn.definition() e syn.examples().

Di seguito è riportato un esempio di output dell’operazione sopra descritta:



Figura 2: Output del Word Meaning Extraction da Wordnet

### Query on Dbpedia

Partendo dalle liste di *noun*, *adjective* e *verb* descritte nel paragrafo precedentre, abbiamo effettuato delle query su Dbpedia.

DBpedia è un progetto nato nel 2007 con lo scopo di estrarre informazioni strutturate da Wikipedia e pubblicarle sul Web come Linked Open Data in formato RDF e mette a disposizione delle API per interrogare l’ontologia attraverso SPARQL, linguaggio di interrogazione per dati rappresentati tramite RDF.

Per ogni elemento delle tre liste, viene effettuata la seguente query[[8]](#footnote-8):



Figura 3: Query utilizzata per interrogare Dbpedia

Tale query seleziona la label con cui è etichettato il subject, il commento e l’immagine *thumbnail* in modo da produrre in output le informazioni associate al token fornito in ingresso.

Di seguito è riportato un esempio di output dell’operazione sopra descritta:



Figura 4: Output della query effettuata su Dbpedia

### Noun Phrase Chunking

Come abbiamo visto, è possibile scomporre una frase in token e quindi eseguire il *PoS Tagging* per identificare le parti del discorso a cui appartengono quelle parole e quindi identificarne il tipo, ma queste semplici operazioni non forniscono abbastanza informazioni significative sulla frase, ed è per questo che è stata effettuata una operazione di chunking.

Il chunking è il processo di scomposizione di un testo in frasi come *Noun Phrases*, *Verb Phrases*, *Adjective Phrases*, *Adverb Phrases* e *Preposition Phrases*, ed è comunemente usato per estrarre le *Noun Phrases* (NP) dalla frase[[9]](#footnote-9).

Tale processo, essenziale per comprendere la semantica del testo e nel recupero delle informazioni semantiche, è stato effettuato definendo prima una regola *RegExp* per il parsing dei vari token (grammar), ovvero:

“NP: {<RB.\*>\*<DT>?<RB.\*>\*<JJ.\*>\*<NN.\*> |<RB.\*>\*<JJ.\*>\*<PRP><RB.\*>\*}”

“VP: {<VB.\*>?<RB.\*>\*<VB.\*>\*}”

Tale regola prevede l’accorpamento in un'unica *Noun Phrase* di tutte le sequenze di token che abbiano tali caratteristiche:

* Possono iniziare o meno con uno o più[[10]](#footnote-10) avverbi (<RB.\*>);
* Possono comprendere o meno[[11]](#footnote-11) un articolo (<DT>);
* Possono iniziare o meno con uno o più avverbi (<RB.\*>);
* Possono comprendere o meno degli aggettivi (<JJ.\*>);
* Devono contenere un *Noun* (<NN.\*>);
* O in alternativa deve contenere un pronome preceduto o meno da uno o più aggettivi, uno o più avverbi e può concludere con uno o più avverbi[[12]](#footnote-12) (|<RB.\*>\*<JJ.\*>\*<PRP><RB.\*>\*).

Tale regola prevede l’accorpamento in un'unica *Verb Phrase* di tutte le sequenze di token che abbiano tali caratteristiche:

* Possono iniziare o meno con un verbo (<VB.\*>);
* Possono contenere o meno degli avverbi (<RB.\*>);
* Possono terminare o meno con dei verbi (<VB.\*>).

Definita la regola, è stato effettuato il chuncking attraverso la funzione RegexpParser(grammar).parse(pos\_tagged\_text), ottenendo una struttura ad albero delle varie *Noun Phrases* e *Verb Phrases* trovate.

### Triples Extraction

Per effettuare l’estrazione delle triple dal testo fornito, siamo partiti dalle *Noun Phrases* e dalle *Verb Phrases* per definire le triple come sequenze *NP – VP – NP*.

Definendo le funzioni get\_nl\_text\_from\_np(np) e get\_nl\_text\_from\_vp(vp), abbiamo ricavato le triple in linguaggio naturale andando ad eliminare tutti i costrutti inseriti in fase di chunking come le parentesi, i *PoS tag* e i nomi delle *Noun Phrases* e *Verb Phrases* ricavate.

Tali triple poi sono state salvate in un file .csv utile a future analisi e sviluppi.

#### **Note**

Per la *Triples Extraction* sono state effettuate diverse prove con e senza *Stop Word Removal* in quanto il contenuto informativo eliminato da tale operazione, seppur piccolo e di minore rilevanza, andava ad inficiare negativamente sulla qualità delle *Noun Phrases* e delle *Verb Phrases* e di conseguenza sulle triple estratte.

## Coreference resolution and NER Tagging

Infine, è stata applicata la *Coreference Resolution* e il *NER Tagging* al testo in linguaggio naturale da cui sono state estratte le informazioni semantiche.

### Coreference Resolution

La *Coreference Resolution* consiste nel trovare tutte le espressioni che si riferiscono alla stessa entità in un testo. È un passaggio importante per molte attività di *NLP* di livello superiore che coinvolgono la comprensione del linguaggio naturale, come la sintesi dei documenti, la risposta alle domande e l'estrazione di informazioni.

Tale operazione è stata effettuata tramite la libreria sapCy e attraverso le funzioni:

* merge\_phrases(doc) e merge\_punct(doc) per il merging dei token in chunk comprendenti sia i segni di punteggiatura e sia i vari token:



Figura 5: Esempio di output delle funzioni di merging delle frasi e dei segni di punteggiatura

* displacy.render(doc, jupyter=True) per la *Coreference Resolution*:



Figura 6: Esempio di output di Coreference Resolution

### NER Tagging

Il *Named Entity Recognition* (*NER*), noto anche come *Named Entity Identification*, *Enity Chunking* e *Entity Extraction*, consiste nell’individuare e classificare le entità nominative menzionate in un testo non strutturato in categorie predefinite come nomi di persone, organizzazioni, località, codici medici, espressioni temporali, quantità, valori monetari, percentuali, ecc.

Tale operazione è stata effettuata tramite la libreria sapCy e attraverso la funzione:

* displacy.render(doc, style=”ent” jupyter=True) per la *Named Entity Recognition (NER) Tagging*:



Figura 7: Esempio di output di NER Tagging

1. Es: Miss, Mr, Mrs, ecc. [↑](#footnote-ref-1)
2. Es: un verbo nelle sue varie forme (singolare, plurale, ecc) ha lo stesso contenuto informativo della sua forma base. [↑](#footnote-ref-2)
3. Es: then, once, here, there, when, where, why, etc. [↑](#footnote-ref-3)
4. Per eliminare solo i verbi è stato eseguito il *PoS Tagging* sull’intera lista, aggiungendo alla lista di parole da eliminare tutte quelle che non avessero il tag “ *VB.\** ”. [↑](#footnote-ref-4)
5. Alcuni nomi e cognomi di personaggi del libro venivano etichettati diversamente da *NNP* (in alcuni casi persino come verbi). [↑](#footnote-ref-5)
6. Es: Harry Potter diviso come due token, “Harry” e “Potter”. [↑](#footnote-ref-6)
7. Es: “Professor” e “Snape” in “Professor Snape”, “Ron” e “Weasley” in “Ron Weasley”, etc. [↑](#footnote-ref-7)
8. {subject} si riferisce ad un token di una delle liste [↑](#footnote-ref-8)
9. Va notato che il *PoS Tagging* è il prerequisito fondamentale per il processo di chunking per evitare che i blocchi si sovrappongono l'uno all'altro. [↑](#footnote-ref-9)
10. Si noti la presenza del \* che indica 0 o più occorrenze dell’identificativo precedente. [↑](#footnote-ref-10)
11. Si noti la presenza del ? che indica la presenza o meno dell’identificativo precedente. [↑](#footnote-ref-11)
12. Si noti la presenza del ? che indica un OR tra le due condizioni. [↑](#footnote-ref-12)