



UNIVERSITA' DEGLI STUDI DI  
NAPOLI FEDERICO II

Facoltà di Ingegneria  
Corso di Studi in Ingegneria Informatica

tesi di laurea

## **Information extraction da testi non strutturati**

Anno Accademico 2011/2012

relatore

**Ch.mo prof. Vincenzo Moscato**

candidato

**Fabio D'Onofrio**  
**matr. 534001605**

---

*Se in un primo momento l'idea non è assurda, allora non c'è nessuna speranza che si realizzi.*

(Albert Einstein)

---

# Indice

<b>Introduzione</b>	<b>1</b>
<b>1 Trattamento automatico del testo</b>	<b>3</b>
1.1 NLP (Natural Processing Language) . . . . .	3
1.1.1 Definizioni del NLP . . . . .	4
1.1.2 Architettura del NLP . . . . .	5
1.1.2.1 Analisi Lessicale . . . . .	10
1.1.2.1.1 Sentence Splitter . . . . .	11
1.1.2.1.2 Tokenization . . . . .	11
1.1.2.1.3 StopList . . . . .	11
1.1.2.1.4 Stemming . . . . .	12
1.1.2.1.5 Tagging . . . . .	13
1.1.2.2 Analisi Sintattica . . . . .	15
1.1.2.3 Analisi Semantica . . . . .	16
1.2 Introduzione all'Information Extraction . . . . .	17
1.3 Resource Description Framework . . . . .	18
1.3.1 Triplet Extraction . . . . .	21
<b>2 Estrazione di informazione da testi non strutturati</b>	<b>23</b>
2.1 Estrazione automatica delle informazioni . . . . .	23

2.2	Ottenere informazione dai documenti . . . . .	25
2.3	Information Retrieval . . . . .	27
2.4	Information Extraction . . . . .	31
<b>3</b>	<b>Progettazione ed implementazione di un framework per l'Information Extraction da Testo</b>	<b>36</b>
3.1	Struttura del Software . . . . .	36
3.2	Moduli . . . . .	37
3.2.1	MainExtractorTriple . . . . .	38
3.2.2	Parsing . . . . .	38
3.2.3	ExtractingTriples . . . . .	40
3.2.4	Stemming . . . . .	41
3.2.5	Tagging . . . . .	42
3.2.6	NamedEntityRecognizer . . . . .	43
3.2.7	TypedDependencies . . . . .	44
<b>4</b>	<b>Sperimentazione e Sviluppi Futuri</b>	<b>47</b>
4.1	Risultati Sperimentazione . . . . .	47
4.1.1	Valutazione Testo Estratto . . . . .	47
4.1.1.1	Testo Estratto da file txt . . . . .	48
4.1.1.2	Testo Estratto da pagine Web . . . . .	48
4.1.2	Valutazione Generazione Triple RDF . . . . .	49
4.1.2.1	Estrazione di Triple RDF su documenti testuali con frasi semplici . . . . .	49
4.1.2.2	Estrazione di Triple RDF su documenti testuali con frasi complesse . . . . .	49
4.1.2.3	Estrazione di Triple RDF su testo di pagine Web . . . . .	50

4.2	Sviluppi Futuri . . . . .	50
	<b>Ringraziamenti</b>	<b>52</b>
	<b>Bibliografia</b>	<b>54</b>
A	<b>WordNet</b>	<b>57</b>



# Elenco delle figure

1.1.1 Schema a blocchi di un sistema per l'analisi del linguaggio naturale	10
1.1.2 <i>Un sottoinsieme del Penn Treebank Tagset</i> . . . . .	14
2.2.1 Campi di ricerca del testo . . . . .	26
3.1.1 Struttura software del sistema “Web Semantico” . . . . .	36
3.2.1 Class diagram . . . . .	37
3.2.2 Classe MainExtractorTriple . . . . .	38
3.2.3 Classe Parsing . . . . .	40
3.2.4 Classe ExtractingTriples . . . . .	41
3.2.5 Classe Stemming . . . . .	42
3.2.6 Classe Tagging . . . . .	43
3.2.7 Classe NamedEntityRecognizer . . . . .	44
3.2.8 Classe TypedDependencies . . . . .	46
4.1.1 Statistica Estrazione di testo da file .txt . . . . .	48
4.1.2 Statistica Estrazione di testo da Pagine Web . . . . .	48
4.1.3 Statistica Generazione Triple Frasi Semplici . . . . .	49
4.1.4 Statistica Generazione Triple Frasi Complesse . . . . .	50
4.1.5 Statistica Generazione Triple Frasi Semplici . . . . .	50
A.0.1 <i>Esempio WordNet usando la parola “music”</i> . . . . .	60

# Introduzione

Nel 2001 Tim Berners-Lee, alla decima edizione dell'International World Wide Conference di Hong Kong sul futuro del web, coniò per la prima volta il termine “Semantic Web” descrivendolo come *“un'estensione del Web corrente, dove ogni informazione è già autodefinita, con la possibilità che i computer e gli utenti lavorino in cooperazione”*.

Il Web odierno, come è possibile constatare, è un insieme di testi, un mare vasto di documenti che descrivono dei contenuti. Il Web Semantico agisce da “web intelligente”, cioè uno strumento in grado di creare collegamenti logici tra diversi elementi inerenti l'informazione richiesta.

Obiettivo del Web Semantico è realizzare dei metodi per esprimere il contenuto delle pagine Web e rendere disponibile l'informazione ai programmi dei computer. La caratteristica principale del Web Semantico è la definizione di un linguaggio *XML (Extensible Markup Language)* basato su un meccanismo sintattico che consente di definire il significato degli elementi contenuti in un testo. Il secondo componente importante del Web Semantico è lo standard *RDF (Resource Description Framework)* che codifica l'XML in un set di triple mediante un grafo con nodi e rami entrambi etichettati, dove ogni nodo corrisponde a una risorsa ed ogni ramo rappresenta una proprietà. Una frase RDF è definita da una tripla (soggetto-verbo-oggetto) che asserisce che il soggetto della frase ha una proprietà il cui valore è l'oggetto stesso della frase. Un altro elemento fondamentale del Web Semantico è dato dalla definizione di *ontologie*, cioè un set di rappresentazioni formali per de-

scrivere costrutti, le quali forniscono dei modelli concettuali per l'interpretazione dell'informazione contenuta nelle pagine Web.

Il lavoro di ricerca si è concentrato sull'idea di implementare in linguaggio JAVA un framework che si articola in due fasi: la prima fase si propone di estrarre testo da pagine Web oppure da file testuali (.txt) effettuandone il *parsing*, cioè la determinazione della sua struttura grammaticale mediante una data grammatica formale; la seconda fase invece, partendo dal testo estratto nella prima fase, si propone di generare triple RDF effettuando operazioni di *stemming*, cioè riduzione della forma flessa di una parola alla sua forma radice, e *tagging*, il cui termine indica un'operazione di attribuzione di una o più parole chiave, dette tag, che individuano l'argomento di cui si sta trattando.

Nel primo capitolo è presente la definizione e l'architettura del *NLP* (*Natural Language Processing*).

Nel secondo capitolo vengono approfondite le tematiche e le problematiche dovute all'estrazione di testo da fonti testuali non strutturate.

Nel terzo capitolo è definita la struttura del framework sviluppato durante la fase di tirocinio, con un approfondimento sui moduli da cui è composto.

Nel quarto ed ultimo capitolo sono presenti verifiche e statistiche delle prestazioni del framework sviluppato; a fine capitolo inoltre è presente un accenno ad eventuali sviluppi futuri.



# Capitolo 1

## Trattamento automatico del testo

### 1.1 NLP (Natural Processing Language)

L'elaborazione del linguaggio naturale, anche detta NLP (dall'inglese *Natural Language Processing*), è il processo di estrazione di informazioni semantiche da espressioni del linguaggio umano, tramite l'elaborazione da parte di un calcolatore elettronico. Il linguaggio naturale può essere espresso in vari modi: scritto, parlato o a gesti. Dalla sua nascita avvenuta alla fine degli anni '50 e dalla sua configurazione come disciplina autonoma, ha subito una crescita esponenziale in diverse direzioni arrivando ad attingere contributi da ambiti quali la linguistica, che produce i modelli teorici del linguaggio, la psicologia, che fornisce un'analisi dei processi cognitivi, la teoria dell'informazione, che analizza le modalità comunicative, la matematica e la statistica, che forniscono gli strumenti per esprimere tali modelli in modo trattabile dal punto di vista computazionale e naturalmente l'informatica per quanto riguarda lo sviluppo degli algoritmi atti a implementare i modelli teorici dei fenomeni linguistici.

Il primo sistema commerciale di traduzione automatica, il **Systran**, è stato realizzato nel 1960, e nelle decadi successive si è avuto un costante flusso di trasferimento di tecnologia, dall'uso di teorie dei linguaggi formali nella costruzione di compilatori,

al controllo ortografico nei word processor.

Le ragioni per cui le applicazioni software hanno subito una forte impennata sono:

- la rapida crescita tecnologica che ha notevolmente aumentato la velocità dei processori e la capacità di memoria, a cui è corrisposta una drastica diminuzione dei prezzi
- la crescente disponibilità su larga scala di risorse linguistiche on-line.
- la domanda di applicazioni in un mondo in cui i testi elettronici sono cresciuti in volume e dove le comunicazioni elettroniche e la mobilità hanno accresciuto l'importanza della comunicazione multilingua.

### 1.1.1 Definizioni del NLP

Il linguaggio è come *“un sistema per l'espressione dei pensieri, bisogni, etc..., mediante l'uso di suoni parlati o simboli convenzionali”*[1], il linguaggio è un meccanismo di comunicazione il cui tramite è il testo o il discorso.

Il Natural Language Processing è un termine utilizzato in vari modi e in differenti contesti. E' un ramo dell'informatica che studia *“sistemi automatici per l'elaborazione del linguaggio naturale, include lo sviluppo di algoritmi per il parsing, la generazione e l'acquisizione di conoscenza linguistica; l'indagine sulla complessità spaziale e temporale di tali algoritmi; la progettazione di linguaggi formali computazionalmente utili (come grammatiche e formalismi lessicali) per codificare conoscenza linguistica; l'indagine su architetture software appropriate per i vari compiti del NLP e considerazioni sui tipi di conoscenza non linguistica che vengono a contatto con il NLP. È un'area di studio discretamente astratta che non mette particolare impegno nello studio della mente umana, e neppure mette particolare impegno nel produrre artefatti utili.”*[2].

In definitiva NLP è quella parte dell'informatica che si occupa dei sistemi computerizzati per l'elaborazione del linguaggio naturale.

Il Language Engineering è l'applicazione del NLP alla costruzione di sistemi computerizzati che elaborano il linguaggio per qualche compito come la modellazione del linguaggio stesso, o *“... l'uso strumentale della elaborazione del linguaggio, tipicamente come parte di un sistema più grande con qualche obiettivo pratico, per esempio l'accesso ad un database”* [3].

Quindi possiamo dare la seguente definizione: il NLP è *“lo studio dei sistemi informatici per la comprensione e generazione del linguaggio naturale”*.

L'elaborazione automatica del linguaggio naturale ha lo scopo di implementare strumenti informatici per analizzare, comprendere e generare testi che gli uomini possano comprendere in maniera naturale, come se stessero comunicando con un altro interlocutore umano e non un computer. È caratterizzato da due prospettive diverse, che mirano l'una all'analisi del materiale testuale, l'altra alla generazione di testi linguistici:

- Natural Language Analysis (NLA) o Natural Language Understanding (NLU): data una frase ha l'obiettivo di darne una rappresentazione della sua analisi, ossia del processo di comprensione della frase;
- Natural Language Generation (NLG): data una grammatica di una lingua naturale, ha lo scopo di produrne frasi di senso compiuto.

### 1.1.2 Architettura del NLP

Come detto precedentemente, il NLP consiste in un range di metodi e tecniche computazionali, con fondamenti teorici, per analizzare e rappresentare testi in linguaggio naturale ad uno o più livelli di analisi linguistica allo scopo di ottenere elaborazioni di linguaggio “human-like” per un range di task ed applicazioni.

Il metodo più esplicativo per presentare ciò che in effetti avviene in un sistema di elaborazione di linguaggio naturale è tramite l'approccio dei “livelli di linguaggio”, detto anche modello sincronico che si distingue dal modello sequenziale il quale

ipotizza che i modelli di elaborazione seguano l'un l'altro in modalità strettamente sequenziale; tale modello, invece, risulta più dinamico poiché i livelli interagiscono tra di loro in vario ordine. Di frequente si utilizzano le informazioni, che si ricavano da ciò che è tipicamente inteso come livello superiore di elaborazione, per supporto nell'analisi di livello più basso. Per esempio, la conoscenza pragmatica che il documento che si sta leggendo riguarda la biologia sarà utilizzata quando si riscontra una specifica parola che ha più significati possibili e la parola sarà interpretata come avente un significato che rientra nell'ambito della biologia.

I livelli del Natural Language Processing sono:

- **Fonologico:** questo livello ha a che fare con l'interpretazione del suono della pronuncia delle parole e tra le parole;
- **Morfologico:** questo livello ha a che fare con il fatto che le parole sono composte di morfemi, le più piccole unità di significato. Per esempio, la parola "preregistrazione" può essere morfologicamente analizzata in tre morfemi distinti: il prefisso 'pre', la radice 'registra' ed il suffisso 'zione'. Poiché il significato di ogni morfema rimane invariato nelle parole, una persona può suddividere la parola nei suoi morfemi costituenti al fine di comprendere il suo significato. Allo stesso modo, un sistema NLP può riconoscere il significato da ogni morfema in modo da ottenerne e rappresentarne il significato; per esempio, in inglese, aggiungendo il suffisso 'ed' ad un verbo, si comunica che l'azione del verbo avviene nel passato.
- **Lessicale:** a questo livello le persone, così come i sistemi NLP, interpretano il significato di parole singole. Vari tipi di elaborazione contribuiscono alla comprensione a livello della parola: la prima di queste è l'assegnamento di un singolo tag di parte del discorso ad ogni parola; in tale elaborazione, alle parole che possono fungere da più di una parte del discorso, è assegnato il tag di parte del discorso più probabile in base al contesto in cui si trova la parola;

- **Sintattico:** questo livello si focalizza sull'analisi delle parole in una frase in modo da rivelare la struttura grammaticale della frase stessa. Ciò richiede sia una grammatica sia un parser. L'output di questo livello di elaborazione è una rappresentazione della frase che evidenzia le relazioni di dipendenza strutturale tra le parole. La sintassi porta al significato in molti linguaggi poiché l'ordine e la dipendenza contribuiscono al significato;
- **Semantico:** questo è un livello dal quale molte persone credono sia determinato il significato; comunque, come possiamo vedere nella definizione precedente di livelli, sono tutti i livelli a contribuire al significato. L'elaborazione semantica determina il possibile significato di una frase, concentrandosi sulle interazioni tra i significati delle parole nella frase in esame. Questo livello di elaborazione può includere la disambiguazione semantica di parole con multipli significati, in modo analogo a come è realizzata la disambiguazione di parole che possono fungere da più parti del discorso a livello sintattico. Per esempio, tra i vari significati, 'file' come nome può intendere sia una cartella per memorizzare documenti sia una linea di individui in una coda. Se è richiesta l'informazione dal resto della frase per la disambiguazione, il livello semantico, non lessicale, effettuerà la disambiguazione. Vari metodi possono essere implementati per realizzare la disambiguazione, alcuni dei quali richiedono informazioni sulla frequenza con la quale ogni senso si verifica in un particolare corpus di interesse, altri richiedono la considerazione del contesto locale ed altri ancora utilizzano la conoscenza pragmatica del dominio del documento;
- **Discorso:** mentre la sintassi e la semantica lavorano con unità di lunghezza pari ad una frase, tale livello lavora invece su testi più lunghi, ovvero non interpreta più sentenze (frasi) di testi semplicemente come sentenze concatenate, ognuna delle quali può essere interpretata singolarmente; piuttosto, si

concentra sulle proprietà del testo per intero che porta al significato facendo connessioni tra sentenze componenti. Vari tipi di elaborazione possono avvenire a questo livello, due dei più comuni sono la risoluzione di anafore ed il riconoscimento di struttura del testo/discorso. La risoluzione di anafore consiste nel sostituire le parole come i pronomi, che sono semanticamente poco rilevanti, con l'entità appropriata cui si riferiscono. Il riconoscimento della struttura del testo/discorso determina le funzioni delle sentenze nel testo, che, a turno, si aggiungono alla rappresentazione significativa del testo;

- **Pragmatico:** questo livello ha a che fare con l'utilizzo determinato del linguaggio in situazioni ed utilizza il contesto per la comprensione. Per esempio, le seguenti due sentenze richiedono la risoluzione del termine anaforico 'they' ma tale risoluzione richiede la comprensione pragmatica:

*"The city councilors refused the demonstrators a permit because they feared violence."*

*"The city councilors refused the demonstrators a permit because they advocated revolution."*

I sistemi NLP attuali tendono ad implementare moduli per realizzare principalmente i livelli più bassi di elaborazione, per vari motivi: prima di tutto l'applicazione può non richiedere l'interpretazione a livelli più alti; in secondo luogo, i livelli più bassi sono stati studiati ed implementati più accuratamente; infine, i livelli più bassi hanno a che fare con unità di analisi più piccole, come morfemi, parole e sentenze, che sono governate da regole, a differenza dei livelli più alti di elaborazione del linguaggio, che invece hanno a che fare con testi e che sono solo governati da regolarità.

Qualsiasi applicazione che utilizza testo è candidata per essere un'applicazione di NLP; le applicazioni più frequenti che utilizzano NLP includono le seguenti:



- **Information Retrieval:** è l'insieme delle tecniche utilizzate per il recupero mirato dell'informazione in formato elettronico. Per "informazione" si intendono tutti i documenti, i metadati, i file presenti all'interno di banche dati o nel World Wide Web;
- **Information Extraction:** si focalizza su riconoscimento, tagging ed estrazione, da ampie collezioni di testo, di elementi chiave come persone, locazioni, organizzazioni;
- **Question-Answering:** a differenza dell'Information Retrieval, che fornisce una lista di documenti potenzialmente rilevanti in risposta ad una query dell'utente, question-answering fornisce all'utente solo il testo della risposta o i passaggi che forniscono la risposta;
- **Sintesi:** i livelli più alti di NLP possono consentire un'implementazione che riduce un testo lungo in uno più breve, una rappresentazione narrativa abbreviata e significativa del documento originale;

Molte tecniche di NLP, incluse lo stemming, il part-of-speech tagging, il riconoscimento di parole composte, decomposizione, word sense disambiguation e altre, sono utilizzate nell'Information Retrieval (IR) e non solo.

Molti altri task dell' IR utilizzano tecniche molto simili come il clustering di documenti, il filtering, il rilevamento di link, che possono essere combinate col NLP in maniera simile al document retrieval.

Un sistema per l'analisi di input linguistici ha un'architettura, rappresentata secondo uno schema a blocchi in Figura 1.1.1, che si compone dei seguenti elementi:

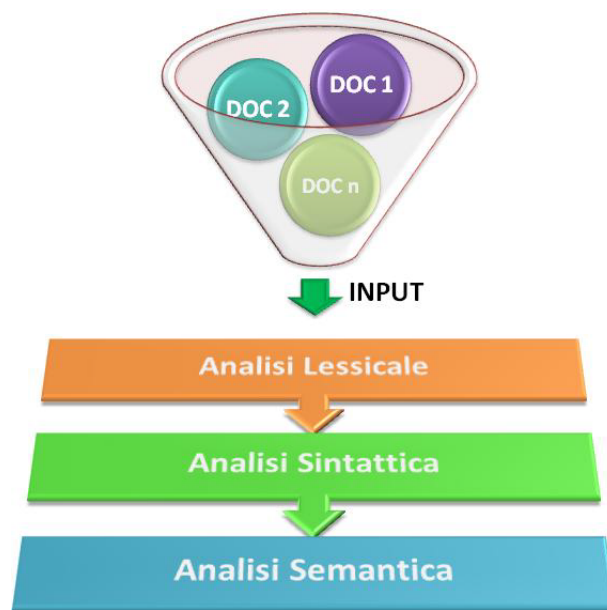


Figura 1.1.1: Schema a blocchi di un sistema per l'analisi del linguaggio naturale

- **Analisi lessicale:** ha il compito di riconoscere gli elementi lessicali e assegnarvi informazioni in merito alla loro categoria grammaticale, resolvendo le ambiguità;
- **Analisi sintattica, o Parser:** ha il compito di assegnare una caratterizzazione sintattica alla frase;
- **Analisi semantica:** ha il compito di eseguire un'analisi semantica del testo in ingresso, generando meaning representations.

Nei seguenti paragrafi si analizzano i singoli moduli, evidenziandone caratteristiche principali e difficoltà legate alla loro implementazione.

#### 1.1.2.1 Analisi Lessicale

Il processo di analisi lessicale si compone di due sottosistemi:

1. **Riconoscitore di forme:** ha il compito di riconoscere le forme atomiche oggetto delle future elaborazioni. Si compone di un **Tokenizer** che compone



la successione di caratteri in ingresso in unità linguistiche, ad esempio parole; e di uno **Stemmer**, che riconosce le possibili forme flesse di una unità linguistica e ne associa la forma radicale e le meta-informazioni di flessione;

2. **Categorizzazione**, o **Tagger**: associa ad ogni unità linguistica una delle possibili classi morfologico-sintattiche.

**1.1.2.1.1 Sentence Splitter** Sentence Splitter è il processo di suddivisione del testo in frasi. Tale modulo, partendo dal documento iniziale, restituisce un insieme di stringhe, dove ogni elemento rappresenta una frase.

Un problema da non sottovalutare è legato al fatto che i simboli di interpunzione forte, quali ‘.’, ‘!’ e ‘?’, sono in generale condizione necessaria ma non sufficiente per dire che la frase è terminata.

Infatti il carattere ‘.’ si può trovare in corrispondenza di abbreviazioni (Mr., O.N.U.), di url ([www.unina.it](http://www.unina.it)), indirizzi e-mail, numeri, etc.

**1.1.2.1.2 Tokenization** In tale fase all’interno di ogni singola frase vengono individuati degli elementi atomici, detti token, in base ai quali è possibile effettuare un’analisi e valutazione della frase stessa. Durante la fase di tokenizzazione, quindi, non solo vengono riconosciuti e valutati tali elementi, ma in alcuni casi vi è la conversione di costrutti negativi come ad esempio ‘**don’t**’ da tre token (‘**don**’, ‘**’**’ e ‘**t**’) a due token (‘**do**’ e ‘**n’t**’).

**1.1.2.1.3 StopList** Una stop-list è un elenco di parole (stop-words) che devono essere eliminate quando vengono incontrate all’interno del documento che si sta analizzando.

Le parole contenute in una stop-list sono, generalmente, termini non significativi per la selettività del documento e/o della frase, ad esempio: articoli, preposizioni, avverbi, congiunzioni, etc.

Inoltre, nella stop-list, possono essere inserite tutte quelle parole che, in base al contesto del fabbisogno informativo dell'utente e dell'applicazione del sistema per il reperimento delle informazioni, vengono ritenute non significative.

L'eliminazione delle stop-words serve ad evitare che due frasi o documenti risultino simili perché contengono le stesse congiunzioni e gli stessi articoli. Eliminando le stop-words si dà maggior peso alle altre parole che, solitamente, hanno un maggiore significato semantico.

Ci sono però molti contro-esempi che mostrano come l'eliminazione delle stop-words sia inefficace e controproducente, ad esempio:

1. *To be or not to be*
2. *New Year celebrations*
3. *On the road again*

*(Le parole in corsivo sono considerate stopwords)*

Adattare la lista delle stop-words al dominio in esame può portare ad un miglioramento significativo dei risultati. Comunque, bisogna star attenti nel valutare le parole da aggiungere alla lista delle stop-words, altrimenti, si corre il rischio di non considerare token particolari.

**1.1.2.1.4 Stemming** Lo stemming è un processo che comporta la normalizzazione (o radicalizzazione) delle parole di un documento. Questo metodo permette di ridurre le varianti di una stessa parola ad una radice comune. Nella maggior parte dei casi, infatti, le variazioni morfologiche delle parole hanno interpretazioni semantiche simili e possono essere considerate come equivalenti allo scopo delle applicazioni in esame. L'idea del processo di stemming consiste nel ricercare la radice di una parola, applicando una serie successiva di regole che eliminano gli eventuali suffissi.

Ad esempio:

*Collected, Collecting, Collection, Collections* vengono tutte ridotte alla stessa radice *Collect*.

Negli algoritmi di stemming si possono verificare due tipi di errore:

- **Overstemming:** lo stemmer rende alla stessa radice parole che, in realtà, hanno significati diversi facendo sì che il termine non sia correttamente interpretato;
- **Understemming:** lo stemmer crea diverse radici da parole che, in realtà, hanno la stessa origini.

**1.1.2.1.5 Tagging** Nei sistemi di elaborazione del linguaggio naturale esiste una fase chiamata Part-Of-Speech tagging o POS-tagging (in italiano la traduzione sarebbe “etichettatura delle parti del discorso”). In questa fase si etichettano le parole individuate nella fase di analisi lessicale con il POS corrispondente, senza eseguire una vera e propria analisi sintattica, ma ricorrendo in genere ad informazioni statistiche (ad esempio il TnT tagger [4], basato su trigrammi) oppure a determinate regole.

Le etichette, o tags, sono reperite attraverso tagset, mappe da categorie lessicali in tags, che oltre a contenere i tag per le otto categorie lessicali di base, includono delle specializzazioni o raffinamenti, che distinguono delle sottocategorie, spesso in base al significato della parola. Nella tabella in Figura 1.1.2 è riportato un estratto dal tagset utilizzato nel Penn Treebank corpus [5], il primo corpus etichettato sintatticamente.

Tag	Descrizione	Esempio	Tag	Descrizione	Esempio
CC	congiunzione	<i>and, or</i>	PP	pronome personale	<i>I, you</i>
CD	numero	<i>one</i>	PP\$	pronome possessivo	<i>my, your</i>
DT	articolo	<i>the, a</i>	RB	avverbio	<i>fully</i>
FW	parole straniere	<i>coup</i>	SYM	simbolo	<i>+, &amp;</i>
IN	preposizione	<i>of, by</i>	TO	to finale	<i>to</i>
JJ	aggettivo	<i>white, big</i>	UH	interiezione	<i>oops!</i>
JJR	aggettivo comparativo	<i>bigger</i>	VB	verbo, forma base	<i>eat</i>
JJS	superlativo	<i>biggest</i>	VBD	verbo, passato	<i>ate</i>
MD	verbo modale	<i>can</i>	VBG	verbo, -ing form	<i>eating</i>
NN	nome	<i>cat</i>	VBZ	verbo, 3ª persona pres.	<i>eats</i>
NNS	nome, plurale	<i>cats</i>	WDT	pronome determinativo	<i>which, that</i>
NNP	nome proprio	<i>George</i>	WP	pronome Wh-	<i>what, who</i>
POS	genitivo sassone	<i>'s</i>	WRB	avverbio Wh-	<i>how, where</i>

Figura 1.1.2: Un sottoinsieme del Penn Treebank Tagset

### Un esempio di POS-tagging

Data la frase “**book that flight**” (prenota quel volo), le etichettature possibili, considerando il tagset del Penn Treebank corpus, sono:

1. **book(NN) that(WDT) flight(NN)**
2. **book(VB) that(WDT) flight(NN)**

Per via dell’ambiguità della parola book. In questo caso l’etichettatura corretta è quella che assegna a book il tag (VB).

Gli approcci al POS-tagging sono principalmente due:

- **rule-based:** si usa un dizionario per assegnare a ciascuna parola una lista di possibili tag. Quindi si applicano regole (ad esempio “se una parola ambigua segue un articolo determinativo, allora è un nome”) per eliminare i tag non consistenti.
- **statistico:** si usa un corpus per calcolare la probabilità di un tag in una certa sequenza di parole, scegliendo il tag con probabilità massima.

### 1.1.2.2 Analisi Sintattica

Dato in ingresso una frase ed una grammatica, il compito del parser è determinare se la frase può essere generata dalla grammatica e, in caso affermativo, assegnare alla frase un'adeguata rappresentazione, detto albero di parsing. Un albero di parsing è un grafo aciclico etichettato caratterizzato da: un nodo radice, detto Sentence (S), dei nodi foglia con le parole della frase e dei nodi intermedi, che rappresentano la struttura sintattica assegnata alla frase.

Un processo di parsing può essere visto con un algoritmo di ricerca del corretto albero sintattico per una data frase, all'interno dello spazio di tutti i possibili alberi sintattici generabili a partire delle regole di una grammatica. I parametri che vanno dati al processo di definizione dell'albero sono:

- **le regole grammaticali**, che predicano come da un nodo radice S ci siano solo alcune vie di scomposizione possibili per ottenere i nodi terminali;
- **le parole della frase**, che ricordano come la (s)composizione di S debba terminare.

I due principali approcci al parsing sono:

- **Top-down** o goal-driven approach, cerca il corretto albero applicando le regole grammaticali a partire dal nodo radice S, provando a raggiungere i nodi foglia;
- **Bottom-up** o data-driven approach, si inizia con le parole che compongono la frase di input, da cui si inizia ad applicare le regole grammaticali fino a poter arrivare al nodo radice S.

La strategia top-down non perde tempo esplorando alberi che non portino a S come nodo radice, cosa che invece si verifica con la strategia bottom-up. Il top-down,

però, genera un grande insieme di alberi S-rooted che sono inconsistenti con l'ingresso fornito, dal momento che gli alberi sono generati senza esaminare l'input linguistico. Bottom-up non produce mai alberi inconsistenti con l'input linguistico. Quando in un nodo dell'albero sintattico si applicano delle regole grammaticali, si possono generare un insieme di percorsi alternativi verso uno o più nodi. Tale ramificazione non è espandibile in parallelo, ma va considerato un percorso per volta. Per questo l'esplorazione è fatta secondo due distinte strategie:

- **Depth-first**, la ricerca procede espandendo sempre il primo nodo generato, e operando un backtracking nel caso il percorso non fosse giusto;
- **Breadth-first**, la ricerca procede espandendo prima tutti nodi di un livello, per poi scendere al livello successivo.

Ci sono molti modi di combinare previsioni top-down con dati bottom-up per ottenere ricerche più efficienti. La maggior parte usano un tipo come meccanismo di controllo per la generazione degli alberi, e l'altro come filtro per scartare a priori alberi che certamente non sono corretti.

Anche per il parsing esistono degli approcci statistici, dove si scelgono le regole da espandere in base a probabilità calcolate a partire da un corpus, per arrivare il prima possibile ad un'analisi e restituirla come "più probabile".

### 1.1.2.3 Analisi Semantica

In tale fase si assegna a pezzi di struttura pezzi di significato. La struttura è composta da simboli e relazioni tra simboli che rappresentano stati del "mondo".

Si introduce in questo ambito il concetto di similarità semantica. Essa fornisce una misura sulla somiglianza semantica fra due lemmi. In linea generale, si può affermare che il calcolo della similarità semantica fra due lemmi si basa sulla lunghezza del cammino necessario a percorrere la distanza che li separa dal concetto minimo comune.



## 1.2 Introduzione all'Information Extraction

L'Information Extraction (IE-Estrazione dell'informazione) è un'applicazione della Information Retrieval (IR-Recupero di informazioni), composta da un insieme di tecniche utilizzate per il recupero mirato dell'informazione in formato elettronico.

Lo scopo principale dell'IE è l'estrazione (semi-)automatica di informazione strutturata a partire da informazione non strutturata. Un esempio di IE, può essere quello di trovare, estrarre e memorizzare titolo ed autore da un documento in una base di dati. L'IE eredita alcune delle tecniche dall'IR ma, in realtà, l'obiettivo e gli approcci sono profondamente diversi. L'IE richiede conoscenze interdisciplinari, che vanno dalla progettazione di linguaggi artificiali all'impiego di tecniche avanzate d'intelligenza artificiale all'elaborazione del linguaggio naturale (NLP), etc.

Mentre l'IR ha come obiettivo il recupero di informazioni a partire da chiavi di ricerca, l'IE ha scopi più ambiziosi: cercare di estrarre conoscenza strutturata da testo libero. Nonostante l'impossibilità di rendere il linguaggio naturale esclusivamente formale, l'IE è comunque utilizzato in diverse aree di ricerca, forse giustificato dalla sempre crescente necessità di organizzare, archiviare ed interrogare il più grande serbatoio di informazione non strutturata: il Web.

L'uomo si è sempre preoccupato di memorizzare dati ed informazioni: le tecnologie informatiche rendono questo processo automatizzabile ed applicabile a collezioni di dati di dimensione praticamente infinita. Data questa disponibilità, i problemi che ne sono discesi risiedono principalmente nella necessità di poter effettivamente utilizzare l'informazione in tempi ragionevoli; operazioni come: sommarizzazione, ricerca mirata, estrazione, trasformazione, conversione e aggregazione sono sempre più richieste in un moderno sistema informativo.

Gli obiettivi di chi lavora nel campo dell'IE sono ancora più estesi; in particolar modo sono focalizzati al Web Semantico, che risulta essere il campo più complesso per elaborare dati non strutturati. Non esiste riserva di informazione multi-formato,

de-strutturata, de-localizzata, ridondante e frammentata più del Web; la pubblicazione di risorse nel Web, oggi è alla portata di tutti e, praticamente, non ha regole; la crescita incontrollata e poco regolamentata di documenti, immagini, ipertesti, diari o libri, giustifica pienamente l'applicazione dell'IE: una riorganizzazione non automatizzata sarebbe senz'altro improponibile e inattuabile in tempi ragionevoli.

Un altro problema che si riscontra nel Web, è che le risorse testuali presenti sono profondamente diverse dal testo libero, e questa differenza pone nuovi, e non banali, ostacoli per l'IE. Il processo di estrazione, attualmente, punta al riconoscimento di entità, alla ricerca di co-referenze e all'estrazione di termini. Per entità in un testo libero, di solito si intende un sostantivo, un nome, un'espressione comune e simili: il riconoscimento è spesso utile ai fini di altri (sotto)processi.

Un po' più complicata è invece la ricerca di co-referenze, ovvero quelle frasi che si riferiscono allo stesso oggetto. Anche se l'IE ancora non ha lanciato prodotti finiti sul mercato, esistono già alcuni sistemi, anche commerciali, con le promesse più svariate. Da questo proposito nasce il lavoro presente, nel tentativo di proporre un approccio originale e alternativo all'estrazione di informazioni da documenti reperiti dal Web.

Il processo di estrazione realizzato nel lavoro di tesi si pone come obiettivo l'estrazione da testo libero di triple RDF; vale a dire l'estrazione di sequenze del tipo: < soggetto - predicato - oggetto>.

### 1.3 Resource Description Framework

RDF<sup>1</sup> (Resource Description Framework) è uno standard proposto dal W3C (World Wide Web Consortium) per descrivere le risorse. Una risorsa è qualsiasi cosa fisica o astratta che ha un'identità: una persona, un numero, una pagina web, etc.

<sup>1</sup><http://www.w3.org/RDF/>



Ogni risorsa è identificata da un indirizzo URI. Affinché una risorsa possa essere elaborata da una macchina, deve essere accompagnata da informazioni che la descrivono. Le informazioni sulla risorsa vengono generalmente dette metadati ed hanno la caratteristica di essere comprensibili per la macchina.

I metadati sono dati e come tali possono essere memorizzati in una risorsa (che può quindi contenere informazioni relative a se stessa o ad un'altra risorsa) ed essere descritti da altri metadati.

RDF fornisce un modello per descrivere le risorse, tramite delle proprietà.

Le proprietà sono aspetti specifici, caratteristiche, attributi, relazioni, usate per descrivere una risorsa. Ogni proprietà ha un nome, che è anch'esso un URI, ed ha un significato specifico che definisce i valori ammissibili, i tipi di risorse che può descrivere e le sue relazioni con altre proprietà.

Ogni proprietà può assumere un valore, che può essere una risorsa (identificata da un URI) oppure un letterale (un valore che ha un tipo reale, come una stringa di caratteri, un numero, una data, etc.).

Usando RDF si possono esprimere delle affermazioni, ma prima è necessario identificare l'oggetto che si vuole descrivere, la specifica proprietà dell'oggetto (o relazione tra oggetti) sulla quale si vuole predicare, e il valore assunto dalla proprietà o l'oggetto con cui viene messa in relazione l'entità sulla quale si sta predicando. Queste tre componenti di una affermazione RDF prendono il nome rispettivamente di soggetto, predicato e oggetto; mentre un'affermazione RDF viene anche detta tripla RDF, perché è composta da tre parti.

Quindi, gli elementi chiave di una tripla RDF sono i seguenti:

- **Soggetto:** in grammatica, è il nome o frase nominale che compie l'azione, e a cui la frase si riferisce. Per esempio, nella frase "l'azienda vende automobili" il soggetto è "l'azienda". In logica matematica, questo è il termine sul quale si asserisce qualcosa. In RDF, è la risorsa che viene descritta dal predicato e

dall'oggetto seguenti. Inoltre, in RDF per evitare ambiguità il soggetto viene espresso mediante un URI, ad esempio *http://www.business.org/ontology#azienda*, per indicare che ci si riferisce ad un preciso concetto universalmente riconosciuto;

- **Predicato:** in grammatica, è la parte della proposizione che indica ciò che si dice del soggetto. Ritornando alla frase “l'azienda vende automobili”, il predicato è “vende”. In logica matematica, un predicato è una funzione che, a partire da un certo numero di individui, restituisce un valore di verità. In RDF, un predicato è una relazione tra il soggetto e l'oggetto. Perciò, in RDF, si preferisce definire un unico URI per il concetto “vende”.

Ad esempio, *http://www.business.org/ontology#vende*;

- **Oggetto:** in grammatica indica la persona o cosa su cui termina l'azione espressa dal verbo. Così nella frase “l'azienda vende automobili”, il complemento oggetto è “automobili”. In logica, un oggetto è sostenuto dal predicato. In RDF, un oggetto è una risorsa a cui il predicato fa riferimento oppure un letterale. Nell'esempio qui discusso, potrebbe esser stato definito un URI per indicare il concetto di automobili, come ad esempio:

*http://www.business.org/ontology#automobili*;

- **Tripla:** in RDF, la combinazione dei tre elementi precedenti (soggetto, predicato e oggetto) come una singola unità viene chiamata tripla o frase (statement, in inglese).

Il modello dei dati RDF è estremamente semplice e permette di codificare qualunque struttura dati in modo sintatticamente neutro: fornisce uno schema logico per esprimere affermazioni, non fornisce un linguaggio.

Ciò che risulta necessario è l'utilizzo di una sintassi concreta, in modo che le descrizioni RDF siano comprensibili ad una macchina; per questo motivo, il W3C ha proposto una sintassi per RDF compatibile con XML, che garantisca l'interoperabilità sintattica: RDF/XML infatti usa la sintassi XML per descrivere modelli RDF. Nell'ambito dell'applicazione realizzata, si è utilizzata la rappresentazione RDF per rappresentare il contenuto dei testi da sommarizzare.

In particolare dato un testo in input all'applicazione su di esso vengono effettuate una serie di manipolazioni atte a estrarre da ogni frase contenuta nel testo una o più triple del tipo < soggetto - predicato - oggetto > (s-p-o).

### 1.3.1 Triplet Extraction

L'obiettivo, quindi, è quello di partire da un documento testuale, pre-processato attraverso una fase di NLP, ed ottenere una sua rappresentazione come insieme di triple s-p-o. Esistono, essenzialmente, due approcci alla Triplet Extraction:

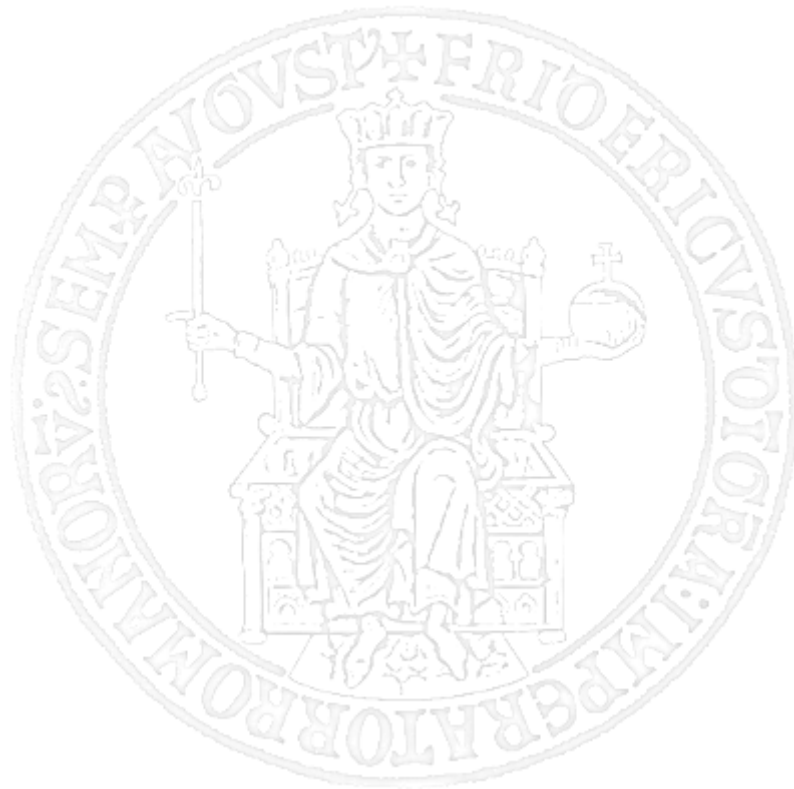
- estrazione di una tripla s-p-o a partire dal parse tree di una frase attraverso l'utilizzo di regole euristiche;
- approccio basato sull'utilizzo di Machine Learning come le Support Vector Machine.

I vantaggi di estrarre informazioni da un testo sottoforma di triple s-p-o sono:

1. rappresentazione compatta delle informazioni contenute in ogni frase componente il testo;
2. più semplice valutazione e manipolazione delle frasi;
3. le triple s-p-o contengono informazioni semantiche.

Per quanto riguarda le applicazioni si possono distinguere essenzialmente due ambiti:

- costruzione di grafi semantici di documenti;
- sommarizzazione di documenti.



## Capitolo 2

# Estrazione di informazione da testi non strutturati

### 2.1 Estrazione automatica delle informazioni

L'ambiente web ormai rappresenta una piattaforma di scambio per i dati insostituibile. Se prima del suo utilizzo in termini massicci, ogni azienda o struttura organizzata aveva una propria collezione di dati cui accedeva con applicazioni specializzate e dedicate, oggi la necessità di scambio e di condivisione delle informazioni tramite il web ha messo ben in evidenza la difficoltà di conciliare reperimento di informazione, utilizzo di banche dati e testo non strutturato. La diffusione della connettività internet ha favorito un uso intensivo della stessa da parte degli utenti accelerando nel contempo la produzione sempre più abbondante di contenuti web. Contestualmente non si è posta sufficiente attenzione alla struttura dei documenti web, pertanto se da un lato i motori di ricerca si sono resi necessari per consultare tali contenuti (e la loro creazione ha decretato il vero e proprio successo del web), dall'altro il loro uso comporta alcuni problemi:

*Alto richiamo, bassa precisione.*

La produzione di informazione ha creato “sovrabbondanza”: in questa situazione,

L'utente che sottopone una richiesta al motore di ricerca, anche se ben formata, si vede restituire un set di documenti che comporta spesso un alto costo in termini di tempo ed energie per selezionare l'informazione ritenuta rilevante e trarre valore da essa. Anche se le principali risorse web rilevanti sono recuperate, non sempre esse appaiono tra le prime visualizzate, e diventa difficile individuarle tra altre migliaia, più o meno rilevanti.

*Richiamo scarso o assente.*

Non sempre si ottengono risposte rilevanti per le nostre richieste, oppure pagine importanti e rilevanti possono non essere recuperate. Nonostante il basso richiamo sia il problema meno frequente con gli attuali motori di ricerca, esso potrebbe comunque verificarsi.

*Risultati molto sensibili al vocabolario.*

Spesso la nostra parola chiave iniziale non ci fornisce i risultati sperati e non perchè non ci siano documenti rilevanti, bensì perchè quest'ultimi, al loro interno, contengono una terminologia differente da quella utilizzata dalla query. Questo crea insoddisfazione perchè query semanticamente simili dovrebbero fornire lo stesso risultato.

I risultati sono singole pagine web.

Se noi abbiamo bisogno di una informazione diffusa su numerosi documenti, siamo costretti a fornire ripetute query per collezionare documenti rilevanti ed estrarre manualmente la parziale informazione presente in ognuno per integrarla con il resto. Contenuti non sempre direttamente accessibili.

I risultati delle ricerche web, quando non sono semplici pagine web, sono accessibili solo se si dispone di altri strumenti software (per esempio: .pdf, .doc, .xls, .ppt, .jpg ). Fortunatamente si assiste alla diffusione di prodotti open source generalmente gratuiti che consentono da parte dell'utente l'utilizzo concreto di tali contenuti. Nel contesto web il termine "information retrieval" usato in abbinamento a "search

engine” può trarre in inganno: “location finder” potrebbe essere un termine più appropriato. Fare “information extraction” su una collezione così vasta e varia come quella dei documenti in rete è un compito particolarmente arduo. Il principale ostacolo per fornire supporto all’utente web, al momento, è rappresentato dal fatto che il significato del contenuto web non è accessibile alla macchina: naturalmente ci sono strumenti che possono recuperare testi, suddividerli in più parti, verificare lo spelling, contare i termini, etc, ma non sono ancora in grado di interpretare le frasi ed estrarre informazione utile per gli utenti, o almeno le capacità in tal senso sono per ora limitate. Questo non significa progettare sistemi software intelligenti in grado di “capire” l’informazione, è sufficiente, per ora, che essi siano capaci di processarla effettivamente rendendola utile (machine-understandable). Inoltre, per ottenere risultati soddisfacenti, è preferibile che l’informazione sia facilmente elaborabile dalla macchina: un testo strutturato in XML oppure opportunamente annotato è sicuramente più “comprensibile” per un elaboratore rispetto ad un testo non strutturato.

## 2.2 Ottenere informazione dai documenti

Per poter parlare di classificazione è necessario innanzitutto poter accedere ai documenti e “guardare” al loro contenuto. La grande maggioranza di documenti disponibili nei computer e nel web è costituita da testo: ne costituiscono esempi i giornali, i report aziendali, le pagine web, gli articoli scientifici, i documenti medici; il testo può presentarsi in vari formati elettronici (**Word, PDF, PS, HTML, XML**), in varie lingue (inglese, italiano, giapponese), in vari caratteri di codifica (ASCII, Unicode). Per un elaboratore, il testo è inteso come un insieme di caratteri alfanumerici e caratteri speciali; per un utente, il testo rappresenta un insieme di dati che, interpretati attraverso la conoscenza e in un preciso contesto, si trasformano in informazione.

L’accesso tradizionale alle informazioni è rappresentato dall’Information Retrieval



(IR) e dall'Information Extraction (IE) e, più recentemente, anche da altri campi di ricerca tra cui Question Answering e Text Understanding.

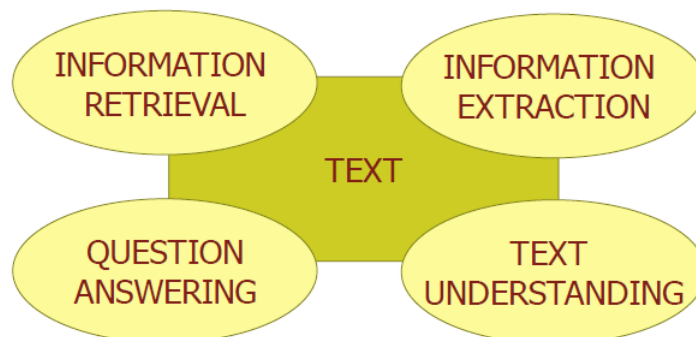


Figura 2.2.1: Campi di ricerca del testo

Queste aree, per le quali non esiste una netta separazione, rappresentano alcuni modi con cui possiamo ottenere informazione dai dati testuali. A seconda degli scopi, un documento di testo può essere considerato come una semplice sequenza di simboli codificati (per un computer), oppure come una sequenza di parole con un possibile significato (IR), o ancora come una sequenza di frasi con significato, possibilmente rilevanti per un argomento (IE). La sfida ulteriore è cogliere il senso generale di un documento per poi rispondere a domande sullo stesso (TU) oppure saper elaborare domande poste dall'utente e riuscire a formulare risposte contenenti l'informazione desiderata senza costringere l'utente a setacciare un lungo elenco di documenti (QA).

Ottenere informazioni è un processo complesso, sia perchè i dati spesso si presentano non strutturati, sia perchè le collezioni di documenti diventano sempre più estese (information overload), senza contare che in alcuni casi l'informazione è duplicata o inesatta. La velocità con cui le collezioni si ingrandiscono, l'eterogeneità e la complessità che le caratterizzano, nonché le richieste sempre più esigenti dell'utente sembrano sminuire, se non vanificare, i costanti miglioramenti introdotti per l'accesso ai documenti[6]. Per avere un'idea delle problematiche che nascono dal



processo di elaborazione dei documenti, si fornisce una panoramica dei primi due approcci sopra citati: l'IR perchè vanta una lunga esperienza in materia di accesso e gestione dei documenti, l'IE perchè, oltre a basarsi su numerosi risultati ottenuti dall'IR, sarà l'approccio utilizzato in questo studio.

## 2.3 Information Retrieval

Una disciplina scientifica con vasta esperienza in materia di accesso e gestione dei documenti è l'Information Retrieval (IR) che nel 1968 è stata così definita:

*“L'Information Retrieval è un campo relativo alla struttura, analisi, organizzazione, memorizzazione, ricerca e recupero di informazione[7]”*

Una definizione alternativa formulata esattamente quarant'anni dopo parla di:

*“ricerca di materiale di natura non strutturata presente all'interno di una vasta collezione che soddisfa un bisogno informativo[8]”.*

Adattando la seconda definizione al nostro caso, potremmo descrivere il nostro studio come una ricerca mirata di referti scritti in testo non strutturato e presenti in un repository, ai fini di una loro classificazione; tuttavia, nonostante gli enormi sviluppi compiuti nel periodo trascorso tra le due definizioni per lo studio del problema e delle soluzioni più efficaci, anche la prima definizione è ancora oggi appropriata e accurata. Il termine “informazione”, infatti, è molto generale e ben si adatta alle esigenze odierne che comprendono, oltre al reperimento di informazione su documenti di testo, anche quello inerente ai contenuti multimediali come immagini, video e audio (per esempio speech e musica). Focalizzando la nostra attenzione sui documenti di testo, ci accorgiamo subito che, pur avendo qualche accenno di struttura come titolo, autore data o abstract, si presentano con un'informazione non strutturata difficile da descrivere e reperire con un algoritmo, contrariamente a quanto accade ai dati contenuti nei record di un database, rigorosamente strutturati in tabelle e campi, e consultabili con apposite query.

Con la diffusione di internet e la grande disponibilità di dati, il processo di reperimento dell'informazione è diventato la forma dominante di accesso ai dati stessi, ma sebbene si assista a numerosi sforzi per rendere l'informazione facilmente elaborabile dalle macchine (pensiamo per esempio alle pagine web scritte in XML), molta informazione si presenta ancora non-strutturata, quindi non chiara, non semanticamente evidente e non “facilmente maneggiabile” per un computer, come lo è invece l'informazione memorizzata in un database.

Se pensiamo ad una pagina web possiamo eventualmente parlare di informazione semi-strutturata, in quanto sono presenti marcatori per identificare il titolo e qualche altro elemento del corpo, marcatori che ci consentono di trarre “qualche” informazione sul contenuto della pagina web stessa. Tuttavia, tale processo risulta spesso molto impreciso e inaffidabile, soprattutto quando l'autore della pagina si è preoccupato più dell'aspetto estetico della pagina che non della “strutturazione” dei contenuti.

Dando una rapida occhiata alla storia, notiamo la cronologia degli studi condotti per gestire e recuperare documenti:

- **1960-70's** : esplorazione iniziale per piccole collezioni (abstract scientifici, leggi e documenti commerciali), sviluppo del modello booleano di base e del Vector-Space Model
- **1980's**: database documentali di enormi dimensioni, alcuni gestiti da imprese (MEDLINE – Medical Literature Analysis and Retrieval System Online<sup>1</sup>)
- **1990's**: ricerca di documenti attraverso Internet (Lycos, Yahoo, Altavista) e categorizzazione automatica di documenti
- **2000's**: Link analysis (Google), Question Answering, IR nei multimedia (immagini, video, audio), text summarization (scelta di frasi dal documento originale da presentare)

---

<sup>1</sup><http://www.nlm.nih.gov/pubs/factsheets/medline.html>

Tipicamente il reperimento dell'informazione comporta le seguenti fasi:

- definire i propri bisogni informativi
- interrogare la collezione
- memorizzare i risultati
- raffinare la soluzione ridefinendo i requisiti informativi, navigando attraverso i dati trovati, elaborando e combinando i dati di diverse ricerche

L'interrogazione dell'utente, di solito, è costituita da una stringa di testo che viene elaborata su una collezione di documenti in linguaggio naturale. Un sistema di IR restituirà un elenco di documenti in ordine decrescente di rilevanza. Ma come valutare quest'ultima?

L'attinenza o rilevanza di un documento ad una query è soggettiva e dipende da:

- appartenenza ad un campo semantico (es: "pesca" intesa come frutto o attività sportiva?)
- puntualità (es: se relativa a cronaca, previsioni del tempo, quotazioni in borsa)
- autorità (provenienza sicura) (es: specifiche di un prodotto sul sito del produttore)
- vicinanza agli obiettivi dell'utente, all'utilizzo previsto (es: mera consultazione o per scopi di ricerca)

La ricerca normalmente avviene tramite keyword costituite da una o più parole: esse costituiscono la nozione più semplice di attinenza intesa come occorrenza letterale nel testo. Tuttavia, dato che il criterio di selezione è basato sulla frequenza delle keyword nel documento, e non sempre si tiene conto del loro ordine (bag of words), i risultati possono essere non soddisfacenti. I problemi più comuni in questo caso

sono quelli legati al silenzio, ossia il non reperimento di documenti che contengono sinonimi della keyword (es: basket-pallacanestro) oppure al rumore, cioè il reperimento di documenti che contengono la keyword ma con significati e contesti differenti (es: Apple: azienda o frutto?).

Negli anni si è assistito allo sviluppo di un IR più sofisticato, rendendo i sistemi più sensibili al significato delle parole (*pesca/frutto*, *pesca/sport*), considerando l'ordinamento delle parole nell'interrogazione (*computer science* – *science and computer*), tenendo conto di eventuali informazioni sul feedback dell'utente, valutando l'autorità/affidabilità delle fonti, eseguendo operazioni sui testi come l'indicizzazione, lo stemming e la lemmatizzazione, operando l'espansione delle query mediante thesaurus<sup>2</sup> e categorizzando automaticamente i documenti.

Un'applicazione pratica delle tecniche di IR su collezioni di testi a larga scala è il motore di ricerca. Il più noto è sicuramente quello web (di cui accenniamo alcune problematiche nel seguito) capace di catturare terabytes di dati e fornire risposte, in frazioni di secondo, a milioni di query provenienti da tutto il mondo. Esistono però altre applicazioni con funzionalità di ricerca:

- *Enterprise search engine* opera all'interno di un'organizzazione e cerca di indicizzare dati e documenti presenti su più sorgenti: esso raccoglie informazioni sui vari file indipendentemente dall'estensione, da dove sono memorizzati, da come sono stati creati e dall'applicazione cui sono correlati. Consulta l'intranet aziendale, i database, le e-mail e, più in generale, tutti i dati disponibili, restituendo una lista di risorse, raccolte dalle varie sorgenti, in ordine di rilevanza (Oracle Secure Enterprise Search o Vivisimo) specificando come sono state create, dove sono memorizzate o a quale applicazione sono associate.
- *Desktop search* cerca i contenuti nei file di un utente, restituendo informazioni relative a cronologie web, archivi di e-mail, documenti di testo e file audio,

<sup>2</sup>*Thesaurus*: insieme delle parole chiave che danno accesso a un elenco di sinonimi

video, immagini (Autonomy o Recoll)

- *Blog search engine*, motore di ricerca specializzato nei blog. (Technorati o Amatomu) o Bookmarks search engine specializzato nei preferiti (delicious)

L'elenco inerente alle tipologie dei motori di ricerca potrebbe continuare a lungo ma non rientra negli scopi di questa tesi approfondire tale argomento. Quello che invece si vuole sottolineare è come i numerosi progetti e applicativi sviluppati nell'ambito dei motori di ricerca costituiscano la prova degli sforzi finora compiuti nel campo dell'Information Retrieval, sforzi che hanno già portato a risultati notevoli ma che mirano a soluzioni ancor più ottimali, adeguandosi ai cambiamenti e al mutevole bisogno informativo.

## 2.4 Information Extraction

Per "information extraction" si intende la creazione di una rappresentazione strutturata dell'informazione rilevante presente in un documento machine-readable non strutturato. Se applicata a documenti testuali, l'Information Extraction è dunque una tecnologia che punta ad estrarre elementi salienti relativi ad un particolare contesto, come entità o relazioni.

Esso si distingue dall'IR in quanto l'enfasi in IR è trovare documenti che già contengono la risposta alla domanda formulata dell'utente: data una collezione di documenti, il sistema di IR che riceve in input una query (set di parole chiave) seleziona un sottoinsieme i documenti che ritiene rilevanti per la query. L'utente poi navigherà la lista di documenti e cercherà l'informazione che più gli interessa. Il sistema di IE, invece, data una selezione di documenti, cerca invece di estrarre in modo strutturato l'informazione rilevante secondo le esigenze fornite in input.

Tradizionalmente esistono tre generazioni di sistemi di IE:

- *Hand-Built Systems - Knowledge Engineering [1980s- ]* in cui le regole sono scritte a mano, è richiesto l'intervento di esperti del sistema e del dominio di applicazione e sono necessari ripetuti tentativi per affinare i risultati
- *Automatic, Trainable Rule-Extraction Systems [1990s- ]* si utilizzano learner per l'apprendimento automatico delle regole ma tale soluzione richiede corpora molto ampi
- *Statistical Models [dal 1997 in poi]* uso del machine learning per l'individuazione delle feature inerenti a tipologie di entità; si tratta spesso di supervised learning, ma può essere parzialmente unsupervised

L'IE è usata in numerose applicazioni, in particolare nel text data mining: il data mining estrae sapere o conoscenza a partire da grandi quantità di dati, attraverso metodi automatici o semi-automatici. Il text data mining è una forma particolare di data mining dove i dati sono costituiti da testi scritti in linguaggio naturale, quindi da documenti "destrutturati". Il text data mining unisce la tecnologia della lingua con gli algoritmi del data mining<sup>3</sup>.

IR e IE sono quindi tecnologie complementari che condividono lo stesso obiettivo finale: l'estrazione di informazione implicita contenuta in un insieme di documenti. Nei motori di ricerca, l'uso primario di IE è l'identificazione di attributi (features) da utilizzare per migliorare il ranking. Con eccessivo ottimismo, si pensa che le tecniche di IE possano trasformare la ricerca su testo in un problema di interrogazione su database, estraendo tutte le informazioni importanti dal testo e memorizzandole in forma strutturata, ma per ora si tratta di obiettivi ancora lontani. Esempi di attributi da identificare possono essere: titolo del documento, testo in grassetto, sostantivi. Se il documento è scritto in XML, alcuni di questi dati sono facilmente individuabili tramite i markup; altri dati, invece, richiedono un processo aggiuntivo prima di annotare il testo con la relativa feature (esempio: sentence). Tuttavia, se

<sup>3</sup>Il data mining cerca patterns all'interno di grossi dataset ricavando informazione inizialmente sconosciuta; se fosse nota, sarebbe reperibile con opportune query



si considerano documenti acquisiti mediante tecniche di riconoscimento ottico dei caratteri (OCR), essi non hanno alcun markup e, quindi, anche semplici elementi strutturali come il titolo devono essere individuati e annotati.

Sebbene le feature siano spesso di tipo generale, gli sforzi in IE si concentrano per ottenerne di più complesse, con uno specifico contenuto semantico, come entità e relazioni. La *named entity* è una parola o sequenza di parole utilizzata per riferirsi a qualcosa di particolarmente significativo in un determinato contesto o per una specifica applicazione. Gli esempi più comuni sono i nomi di persona, compagnie, organizzazioni, luoghi, date, quantità, valute monetarie. Data la natura più specifica di queste feature, il processo di identificazione e di marcatura è noto come *semantic annotation* e generalmente gli approcci più usati come identificatori di *named entity* sono basati su regole che usano uno o più lexicons (liste di parole o gruppi di parole). Se la lista è sufficientemente precisa ed estesa, gran parte dell'estrazione può essere fatta semplicemente tramite lookup; combinando regole e patterns, poi, si possono creare nuove entità (per esempio il pattern *via* <word>, <number> può identificare una via).

L'IE si compone generalmente di una serie di fasi (task):

- **Named Entity Task (NE):** marcare nel testo le stringhe che rappresentano entità
- **Template Element Task (TE):** estrarre l'informazione rilevante associata alle entità
- **Template Relation Task (TR):** estrarre l'informazione sulle relazioni tra gli elementi del Template ottenuti con NE e TE.
- **Scenario Template Task (ST):** come TR, ma la scelta del template (slot da riempire) è determinata dal riconoscimento di elementi rilevanti per un certo argomento.

- **Coreference Task (CO):** catturare l'informazione sulle coreferenze legate agli elementi marcati con TE e NE.

Attualmente gli approcci dell'IE sono due:

### 1. *Knowledge Engineering*

- sistema basato su regole
- esperti specializzati nel linguaggio definiscono “a mano” le grammatiche del sistema (i “domain patterns”)
- si basa su intuizioni umane
- il perfezionamento del sistema viene fatto “a mano”
- richiede solo una piccola quantità di training data
- lo sviluppo potrebbe richiedere molto tempo
- potrebbe essere difficile apportare eventuali cambiamenti

In questo scenario, l'intervento umano dell'esperto consente di creare un sistema funzionante con buone prestazioni; tuttavia definire manualmente le regole può andar bene in applicazioni che non hanno grosse pretese di accuratezza, mentre là dove è richiesto un alto grado di precisione, tale modalità potrebbe richiedere tempi molto lunghi, è particolarmente difficile e crea facilmente inconvenienti quando il set di regole diventa molto ampio, senza contare che se il contesto cambia, c'è da ricostruire tutto il sistema.

### 2. *Learning Systems*

- il sistema usa metodi statistici (machine learning)
- gli sviluppatori non sono necessariamente esperti di Language Engineering



- apprende regole da dei corpora di addestramento
- richiede una grossa quantità di training data con annotazioni
- apprende regole dall'interazione con l'utente
- eventuali cambiamenti potrebbero richiedere la riannotazione dell'intero training corpus
- gli annotatori sono a "buon prezzo"

Questo secondo approccio offre il vantaggio di un'alta portabilità tra domini diversi e non richiedendo la presenza di specialisti in Language Engineering, presenta costi contenuti. Il fattore negativo è dato dalla grande quantità di dati richiesta per il training: più abbondante è quest'ultimo, maggiore sarà il grado di precisione raggiunto.



## Capitolo 3

# Progettazione ed implementazione di un framework per l'Information Extraction da Testo

### 3.1 Struttura del Software

Il framework sviluppato durante l'attività di tirocinio è un sistema software sviluppato interamente in Java. Nella figura 3.1.1 viene riportata la struttura del progetto:

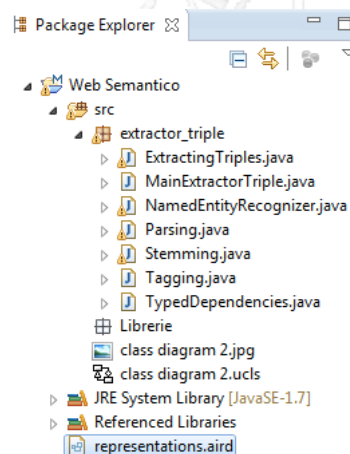


Figura 3.1.1: Struttura software del sistema “Web Semantico”

Il framework è costituito da un unico package(**extractor\_triple**) contenente i seguenti moduli:

- **ExtractingTriples.java**
- **MainExtractorTriple.java**
- **NamedEntityRecognizer.java**
- **Parsing.java**
- **Stemming.java**
- **Tagging.java**
- **TypedDependencies.java**

## 3.2 Moduli

Il framework realizzato è stato implementato secondo il seguente Class Diagram:

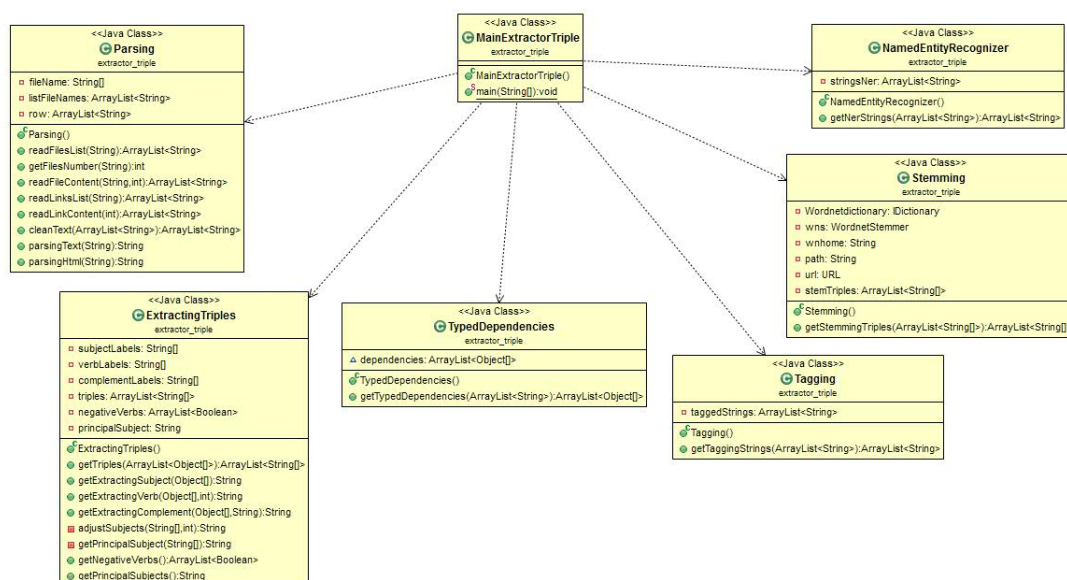


Figura 3.2.1: Class diagram

### 3.2.1 MainExtractorTriple

Tale modulo rappresenta il main del framework. Quando l'applicazione viene mandata in esecuzione, mediante il richiamo ad opportuni moduli, vengono effettuate in sequenza le seguenti operazioni:

- estrazione di testo da file memorizzati in locale o da link internet memorizzati in remoto (richiamando il modulo *Parsing*).
- “pulitura” del testo estratto (richiamando il modulo *Parsing*)
- estrazione delle triple RDF (richiamando il modulo *ExtractingTriples*)
- stemming (richiamando il modulo *Stemming*)
- tagging (richiamando il modulo *Tagging*)
- NER (richiamando il modulo *NamedEntityRecognizer*)
- calcolo delle dipendenze (richiamando il modulo *TypedDependencies*)

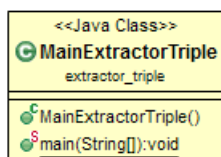


Figura 3.2.2: Classe MainExtractorTriple

### 3.2.2 Parsing

Tale modulo si occupa dell'estrazione del testo utile dalle pagine Web oppure da file di testo selezionati e si suddivide nelle seguenti macrofasi:

- **Estrazione del testo utile:** da uno o più file di testo oppure da una o più pagine web. In particolare:

-*Estrazione di testo da file txt:* il testo viene ricavato da uno o più file .txt presenti in un determinato path.

-*Estrazione di testo da pagina web:* il testo viene ricavato da uno o più link presenti in un determinato file di testo a sua volta presente in un determinato path.

Per effettuare l'estrazione del testo in questo secondo caso, è stata utilizzata libreria “**JSoup**”.

La preferenza di JSoup rispetto ad altre librerie (JTidy, NekoHTML, HTMLCleaner, etc.) è dovuta alle seguenti osservazioni:

- si tratta di una libreria open source;
- permette un utilizzo molto flessibile, in quanto è possibile effettuare parsing HTML da un URL, da un file o da una stringa;
- permette la manipolazione degli elementi HTML, degli attributi e del testo.

Successivamente all'utilizzo della libreria JSoup è stata effettuata un'operazione di pulitura per eliminare i tag HTML e prelevare solo il contenuto dei paragrafi contenenti testo utile.

- **Pulitura del testo:** a seguito dell'estrazione di testo utile è necessaria un'operazione di pulitura del testo, atta, ad esempio, a convertire caratteri specifici della codifica HTML in caratteri compatibili con codifiche standard che meglio si prestano alle elaborazioni testuali. In particolare abbiamo scelto, per una maggior facilità di elaborazione, di eliminare all'interno delle frasi eventuali **segni di interpunzione** (es. ! ? ; :), eventuali **caratteri speciali** (es. @ \$ £), eventuali **forme contratte** (es. *n't* diventa *not*) e di gestire eventuali **conversioni che JSoup non supporta** (es. *&Acirc &copy* diventa *copyright*).

La classe che realizza la funzionalità di tale modulo è la seguente:

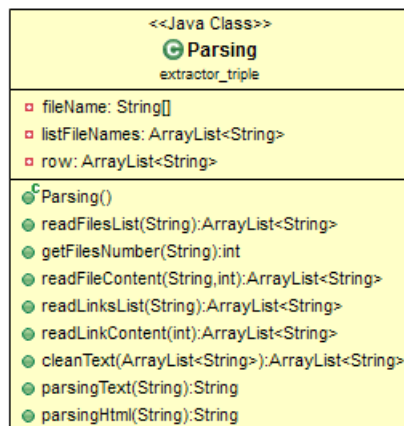


Figura 3.2.3: Classe Parsing

### 3.2.3 ExtractingTriples

Tale modulo si occupa dell'estrazione della tripla RDF (soggetto-predicato-oggetto) più significativa da ogni frase contenuta nel testo.

Mediante l'utilizzo della **libreria Parser di Stanford**<sup>1</sup> vengono associati al soggetto, predicato e oggetto delle etichette atte all'identificazione di ogni tipo.

In particolare la libreria Parser di Stanford associa:

- ai soggetti contenuti in una stringa le etichette {"*csubj*", "*csubjpass*", "*nsubj*", "*nsubjpass*", "*subj*", "*xsubj*", "*nn*", "*npadvmod*", "*root*"}
- ai predicati contenuti in una stringa le etichette {"*cop*", "*infmod*", "*partmod*", "*purpcl*", "*rcmod*", "*root*"}
- agli oggetti contenuti in una stringa le etichette {"*acompl*", "*complm*", "*dobj*", "*iobj*", "*obj*", "*pcomp*", "*pobj*", "*xcomp*", "*ccomp*", "*root*", "*nn*", "*nsubj*"}.

<sup>1</sup><http://nlp.stanford.edu/software/lex-parser.shtml>

In base alle etichette associate, viene determinata la tripla RDF più significativa contenuta nella frase.

Ad esempio: data una generica frase *Balotelli, 23 years-old, forward of Manchester City, scores amazing goal* viene estratta la tripla (*Balotelli, scores, goal*)

Questo modulo presenta ulteriori due funzionalità:

- **estrazione dell'informazione forma affermativa/negativa:** (ad esempio alla frase *Balotelli scores goal* viene associata la forma affermativa, mentre alla frase *Balotelli doesn't score goal* la forma negativa)
- **estrazione del soggetto principale del testo,** estraendo il soggetto che compare più volte all'interno delle triple.

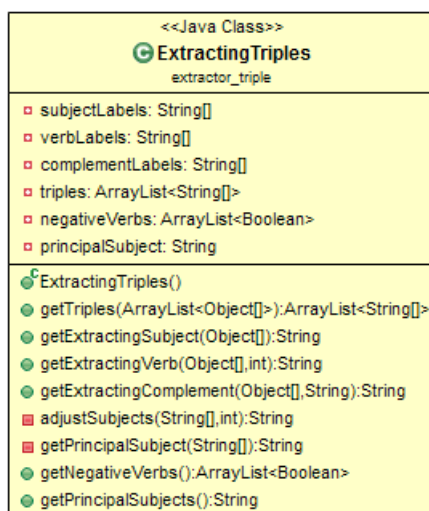


Figura 3.2.4: Classe ExtractingTriples

### 3.2.4 Stemming

Tale modulo si occupa di effettuare lo stemming di triple RDF del tipo (soggetto-predicato-oggetto). Esso si interfaccia col database di **Wordnet-2.1**. In particolare, per ogni tripla estratta:



- Soggetti e oggetti in forma plurale vengono trasformati in forma singolare.
- I verbi vengono sostituiti con la coniugazione all'infinito.
- Se soggetti, predicati o oggetti non sono presenti nel database Wordnet, non vengono modificati.

Ad esempio: data una generica frase *The clients will use our tools* ricaviamo la tripla RDF (*clients-use-tools*), da cui la tripla stemmata (*client-use-tool*).

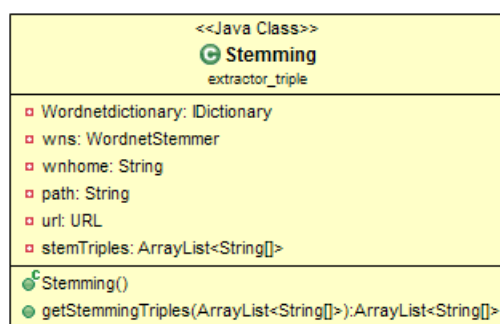


Figura 3.2.5: Classe Stemming

### 3.2.5 Tagging

Tale modulo si occupa di effettuare il tagging delle stringhe di un testo. In particolare:

- i sostantivi singolari vengono marcati col tag **NN** (Noun, singular or mass); i sostantivi plurali col tag **NNS** (Noun, plural).
- i pronomi personali vengono marcati col tag **PRP** (Personal pronoun); i pronomi possessivi col tag **PRP\$** (Possessive pronoun).
- gli avverbi vengono marcati col tag **RB** (Adverb).
- i verbi in forma base vengono marcati col tag **VB** (Verb, base form); i verbi in forma passata col tag **VBD** (Verb, past tense).

- gli aggettivi vengono marcati col tag **JJ** (Adjective).

L'elenco completo di tutti i tag utilizzati è presente al link

[www.computing.dcu.ie/~acahill/tagset.html](http://www.computing.dcu.ie/~acahill/tagset.html)

Ad esempio: data una generica frase

*Domenicali added that Ferrari was looking to the longterm with changes being made to ensure there is no repeat of the issues that have been faced this season*

ricaviamo la frase taggata:

*Domenicali\_NNP added\_VBD that\_IN Ferrari\_NNP was\_VBD looking\_VBG to\_TO the\_DT longterm\_JJ with\_IN changes\_NNS being\_VBG made\_VBN to\_TO ensure\_VB there\_EX is\_VBZ no\_DT repeat\_NN of\_IN the\_DT issues\_NNS that\_WDT have\_VBP been\_VBN faced\_VBN this\_DT season\_NN*




<<Java Class>>	
 <b>Tagging</b>	
extractor_triple	
▣ taggedStrings: ArrayList<String>	
 Tagging()	
 getTaggingStrings(ArrayList<String>): ArrayList<String>	

Figura 3.2.6: Classe Tagging

### 3.2.6 NamedEntityRecognizer

Tale modulo si occupa di effettuare il NER sulle parole contenute in una stringa.

L'operazione di NER viene effettuata utilizzando la **libreria NER di Stanford**<sup>2</sup>.

Ad esempio, data una generica frase:

<sup>2</sup><http://nlp.stanford.edu/software/CRF-NER.shtml>

*Domenicali added that Ferrari was looking to the longterm with changes being made to ensure there is no repeat of the issues that have been faced this season*

ricaviamo :

*Domenicali/PERSON added/O that/O Ferrari/ORGANIZATION was/O looking/O to/O the/O longterm/O with/O changes/O being/O made/O to/O ensure/O there/O is/O no/O repeat/O of/O the/O issues/O that/O have/O been/O faced/O this/O season/O*




<<Java Class>>	
 <b>NamedEntityRecognizer</b>	
extractor_triple	
□	stringsNer: ArrayList<String>
	NamedEntityRecognizer()
	getNerStrings(ArrayList<String>): ArrayList<String>

Figura 3.2.7: Classe NamedEntityRecognizer

### 3.2.7 TypedDependencies

Tale modulo si occupa di effettuare il calcolo delle dipendenze tra le parole contenute in una stringa. Il calcolo delle dipendenze viene effettuato utilizzando la **libreria Parser di Stanford**.

Ad esempio, data una generica frase:

*Domenicali added that Ferrari was looking to the longterm with changes being made to ensure there is no repeat of the issues that have been faced this season*

ricaviamo le DIPENDENZE STRINGA :

*nn(Domenicali-2, ?-1)*

*nsubj(added-3, Domenicali-2)*

*root(ROOT-0, added-3)*  
*complm(looking-7, that-4)*  
*nsubj(looking-7, Ferrari-5)*  
*aux(looking-7, was-6)*  
*ccomp(added-3, looking-7)*  
*prep(looking-7, to-8)*  
*det(longterm-10, the-9)*  
*pobj(to-8, longterm-10)*  
*prep(looking-7, with-11)*  
*pobj(with-11, changes-12)*  
*auxpass(made-14, being-13)*  
*partmod(changes-12, made-14)*  
*aux(ensure-16, to-15)*  
*xcomp(made-14, ensure-16)*  
*expl(is-18, there-17)*  
*ccomp(ensure-16, is-18)*  
*det(repeat-20, no-19)*  
*nsubj(is-18, repeat-20)*  
*prep(repeat-20, of-21)*  
*det(issues-23, the-22)*  
*pobj(of-21, issues-23)*  
*nsubjpass(faced-27, that-24)*  
*aux(faced-27, have-25)*  
*auxpass(faced-27, been-26)*  
*rcmod(issues-23, faced-27)*  
*det(season-29, this-28)*

*dobj(faced-27, season-29)*

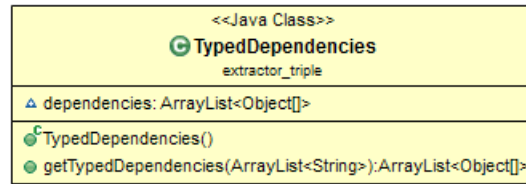


Figura 3.2.8: Classe TypedDependencies



## Capitolo 4

# Sperimentazione e Sviluppi Futuri

### 4.1 Risultati Sperimentazione

La sperimentazione di un qualsiasi tipo di sistema si pone come obiettivo quello di verificare il corretto funzionamento e la qualità in rapporto ai requisiti definiti in fase di analisi e progettazione.

Scopo di questo capitolo è descrivere il processo di sperimentazione del framework sviluppato.

È opportuno sottolineare la difficoltà di ottenere un giudizio assoluto, data l'impossibilità di stabilire con certezza unanime un modello ideale ed universale di estrattore testo e generazione triple. Premesso ciò, si è cercato di effettuare una valutazione di tipo oggettivo, nei limiti imposti dalla natura del problema. Sono condotte due differenti valutazioni del framework, una riguardante la **valutazione del testo “utile” estratto**, l'altra riguardante la **valutazione delle triple RDF generate**.

#### 4.1.1 Valutazione Testo Estratto

L'analisi sulla valutazione del testo estratto è stato suddiviso in due sottocategorie, secondo le funzionalità presenti nel framework.

#### 4.1.1.1 Testo Estratto da file txt

Su un campione di 30 file di testo processati, con la generazione di circa 2000 frasi contenenti un numero di token maggiori di cinque, il framework presenta il seguente parametro statistico:

Precisione Estrazione frasi pulite con numero token >5	
File Testuali	98%

Figura 4.1.1: Statistica Estrazione di testo da file .txt

Come da statistica presente in figura 4.1.1 , si può notare che il testo è estratto in maniera quasi perfetta per estrazioni effettuate su file testuali. La bontà dell'estrazione è ovviamente molto alta in quanto non vi sono presenti particolari codifiche, particolari operazione di pulizia del testo da effettuare e soprattutto particolari elementi di rumore all'interno di un file testuale. Piccole imprecisioni vengono rilevate per la presenza di particolarissimi caratteri speciali non in lingua inglese.

#### 4.1.1.2 Testo Estratto da pagine Web

Su un campione di 20 link processati, con la generazione di circa 2000 frasi contenenti un numero di token maggiori di cinque, il framework presenta il seguente parametro statistico:

Precisione Estrazione frasi pulite con numero token >5	
Pagine Web	76%

Figura 4.1.2: Statistica Estrazione di testo da Pagine Web

Come da statistica presente in figura 4.1.2, possiamo notare che il valore percentuale qualificante la bontà dell'estrazione del testo, è diminuito nel caso di estrazione di testo effettuata da pagine Web. Questo dipende dal fatto che le sorgenti (pagine Web) non sono pagine formate da puro testo, ma si tratta di pagine HTML che, oltre al testo utile, contengono numerosi elementi di rumore, quali codice di markup,



script, link di navigazione, pubblicità, commenti, immagini, etc. Tutte queste informazioni aggiuntive, necessarie alla navigazione delle pagine, risultano un ostacolo agli strumenti di recupero ed estrazione dei dati; nonostante questo, le operazioni di pulizia del testo effettuata tramite operazioni di conversione ci portano ad ottenere un risultato più che accettabile.

## 4.1.2 Valutazione Generazione Triple RDF

L'analisi sulla valutazione della generazione di TRIPLE RDF è stato suddivisa in tre sottocategorie:

### 4.1.2.1 Estrazione di Triple RDF su documenti testuali con frasi semplici

La valutazione dell'estrazione delle triple RDF su documenti testuali composti da brevi e singole frasi aventi un significato ben definito presenta i seguenti parametri statistici :

FILE	SOGGETTI	PREDICATI	OGGETTI	FORMA	TRIPLE
<a href="#">Frase in forma congiuntivo.txt</a>	100%	70%	70%	100%	50%
<a href="#">Frase in forma di durata.txt</a>	100%	100%	75%	100%	75%
<a href="#">Frase in forma futuro.txt</a>	95%	80%	65%	100%	45%
<a href="#">Frase in forma in ing del verbo.txt</a>	94%	94%	81%	100%	75%
<a href="#">Frase in forma interrogativa.txt</a>	88%	69%	69%	100%	56%
<a href="#">Frase in forma negativa.txt</a>	100%	94%	88%	88%	76%
<a href="#">Frase in forma passato.txt</a>	100%	94%	67%	100%	67%
<a href="#">Frase in forma passiva.txt</a>	92%	92%	92%	100%	83%
<a href="#">Frase in forma periodo ipotetico.txt</a>	100%	55%	82%	91%	36%
<a href="#">Frase in forma presente.txt</a>	100%	94%	88%	100%	82%
<b>TOTALE</b>	<b>96%</b>	<b>85%</b>	<b>75%</b>	<b>99%</b>	<b>63%</b>

Figura 4.1.3: Statistica Generazione Triple Frasi Semplici

### 4.1.2.2 Estrazione di Triple RDF su documenti testuali con frasi complesse

La valutazione dell'estrazione delle triple RDF su documenti testuali composti da frasi più complesse (formate da più periodi) aventi un significato ben definito presenta i seguenti parametri statistici :

FILE	SOGGETTI	PREDICATI	OGGETTI	FORMA	TRIPLE
Frase in forma congiuntivo.txt	88%	80%	70%	88%	75%
Frase in forma di durata.txt	95%	94%	75%	84%	75%
Frase in forma futuro.txt	88%	80%	65%	91%	45%
Frase in forma in ing del verbo.txt	88%	80%	67%	85%	75%
Frase in forma interrogativa.txt	88%	69%	69%	83%	56%
Frase in forma negativa.txt	92%	94%	88%	88%	67%
Frase in forma passato.txt	71%	94%	67%	91%	67%
Frase in forma passiva.txt	75%	69%	92%	83%	83%
Frase in forma periodo ipotetico.txt	71%	55%	82%	91%	36%
Frase in forma presente.txt	92%	69%	67%	82%	67%
<b>TOTALE</b>	<b>88%</b>	<b>80%</b>	<b>75%</b>	<b>88%</b>	<b>67%</b>

Figura 4.1.4: Statistica Generazione Triple Frasi Complesse

#### 4.1.2.3 Estrazione di Triple RDF su testo di pagine Web

La valutazione dell'estrazione delle triple su testo presente in pagine Web presenta i seguenti parametri statistici:

FILE	SOGGETTI	PREDICATI	OGGETTI	FORMA	TRIPLE
Frase in forma congiuntivo.txt	57%	53%	46%	55%	53%
Frase in forma di durata.txt	64%	56%	49%	57%	63%
Frase in forma futuro.txt	52%	58%	52%	58%	45%
Frase in forma in ing del verbo.txt	69%	44%	53%	45%	53%
Frase in forma interrogativa.txt	53%	54%	51%	42%	56%
Frase in forma negativa.txt	49%	55%	47%	55%	55%
Frase in forma passato.txt	62%	61%	56%	55%	53%
Frase in forma passiva.txt	65%	53%	56%	53%	51%
Frase in forma periodo ipotetico.txt	52%	55%	52%	45%	53%
Frase in forma presente.txt	51%	53%	51%	55%	47%
<b>TOTALE</b>	<b>59%</b>	<b>55%</b>	<b>51%</b>	<b>53%</b>	<b>52%</b>

Figura 4.1.5: Statistica Generazione Triple Frasi Semplici

## 4.2 Sviluppi Futuri

Nonostante gli aspetti positivi messi in luce nel paragrafo precedente, il framework realizzato può essere ulteriormente migliorato, soprattutto per la fase della generazione delle triple RDF. Per prima cosa si potrebbe sviluppare, per la generazione di triple RDF, un approccio basato sulla tecnica della **Word Sense Disambiguation** la quale coinvolge l'associazione di un termine in un testo con il significato che meglio di altri può essere attribuito a quella parola. Tale associazione può essere ottenuta eseguendo due passi: prima recuperando tutti i significati di tutte le parole necessarie alla comprensione del testo, e successivamente trovando il mezzo con cui assegnare ad ogni parola il significato più appropriato.

Una ulteriore studio che permetterebbe il miglioramento della bontà della generazione di triple RDF, è lo studio della **Sentiment Analysis**, ovvero l'analisi di porzioni di testo come sintagmi ( es. *"con cura"*, *"mancanza di rispetto"*), espressioni multiparola ( es. *"fare acqua"*, *"dalla padella alla brace"*, *"parlare alle spalle"*), modi di dire ( es. *"toccare il cielo con un dito"*) e indicatori testuali come la punteggiatura e l'ortografia con successiva classificazione dei documenti secondo una polarità positiva, negativa o mixed.



# Ringraziamenti

Ed eccoci giunti alla parte più difficile di tutta la tesi (ad esclusione della burocrazia): i ringraziamenti.

Seduto alla mia scrivania, occupato negli ultimi sviluppi della tesi, ancora non mi rendo conto di essere arrivato alla fine di questo percorso che ha coinvolto tutto me stesso negli ultimi anni; guardandomi indietro mi rendo conto che tante cose nella mia vita e dentro di me sono cambiate: il mio modo di pensare, di affrontare le cose, il mio modo di relazionarmi agli altri.....forse sono cresciuto, ma spero di rimanere quella persona umile e che si mette in discussione, che sono sempre stata.

Il mio primo pensiero, ovviamente, va ai miei genitori, senza i quali non sarei mai potuto arrivare a questo punto; non parlo solo del sostegno economico, che sicuramente è stato indispensabile, ma di quell'aiuto tacito o esplicito che tante volte è venuto dal loro cuore: mi riferisco a tutte le occasioni in cui mia madre, celando in silenzio l'ansia, mi ha incoraggiato, vedendomi preso dai libri o preoccupato per un programma troppo lungo che proprio non voleva entrarmi in testa; ai discorsi di mio padre, quando, convinto che non stessi ascoltando, parlava di me orgoglioso, dimenticandosi del mio caratterino non facile.

E insieme a mamma e papà un grazie sincero e quanto più sentito vada a tutta la mia famiglia (nonni, zii, cugini) che non hanno fatto mai mancare il loro sostegno.

Un grazie commosso va a nonna Maria e a nonno Guido che, anche da lassù, sono convinto che sono contenti tanto quanto me.

Non possono mancare nei ringraziamenti gli amici; a partire dal collega (quasi Ing.)

Paolo, grazie ai suoi appunti sempre perfetti (che continuerà a passarmi per la magistrale :D) e le spiegazioni dell'ultim'ora mi ha semplificato non poco la vita universitaria; al mio amico Franco, con cui abbiamo intrapreso il difficile percorso della magistrale, che ho assillato e assillo tutt'ora con i miei innumerevoli dubbi (sempre fuggiti a dire il vero); al mio amico Fabio (ribattezzato Special One dopo la "grande impresa") con il quale ho condiviso il lavoro di tirocinio e che negli ultimi giorni compensava il mio nervosismo pre-laurea con la sua calma olimpica.

Scusate se mi sono dimenticato di qualcuno...vi ringrazierò di persona.

Grazie a tutti.

Fabio





# Bibliografia

- [1] M. Makins, editor. Collins English Dictionary, 3rd Edition. Harper Collins, 1991.
- [2] G. Gazdar. Paradigm merger in natural language processing. In R. Milner and I. Wand, editors, Computing Tomorrow: Future Research Directions in Computer Science, pages 88–109. Cambridge University Press, 1996.
- [3] H. Thompson. Natural language processing: a critical analysis of the structure of the field, with some implications for parsing. In K. SparckJones and Y. Wilks, editors, Automatic Natural Language Parsing. Ellis Horwood, 1985.
- [4] T.Brants, TnT - A Statistical Part-Of-Speech Tagger. [www.coli.unisd.de/thorsten/tnt](http://www.coli.unisd.de/thorsten/tnt).
- [5] M.Marcus, B.Santorini, M.A. Marcinkiewicz, Building a Large Annotated Corpus of English: the Penn Treebank. Computational Linguistics, 1993.
- [6] Marco Ernandes - “Text processing – Basi di dati multimediali” - 2005 [www.slidefinder.net/t/text\\_processing\\_information\\_extraction/4511739](http://www.slidefinder.net/t/text_processing_information_extraction/4511739)
- [7] Gerald Salton - Automatic Information Organization and Retrieval -McGraw-Hill Inc. 1968
- [8] C. Manning, P. Raghavan, H. Schutze - Introduction to Information Retrieval”- Cambridge 2008





# Appendice A

## WordNet

WordNet è un database lessicale basato sulle teorie psicolinguistiche della memoria lessicale umana.

Realizzato manualmente da una équipe di psicologi e linguisti guidata dal prof. George A. Miller presso il laboratorio di Scienze Cognitive all'Università di Princeton, WordNet è disponibile gratuitamente al sito della stessa università; inoltre la licenza d'uso ne permette l'utilizzo gratuito anche a fini commerciali ed al di fuori della ricerca, purché siano citati gli autori ed il sito ufficiale del progetto. Poiché numerosi esperimenti svolti in psicolinguistica hanno dimostrato che molte proprietà del lessico mentale possono essere sfruttate nell'ambito della lessicografia, il gruppo di Princeton ha deciso di combinare le due discipline per costruire un lessico che fosse anche un modello della memoria umana. I dizionari tradizionali sono organizzati mediante un ordinamento alfabetico per rendere più semplice e rapida l'individuazione da parte del lettore del termine cercato. Mediante questo approccio però, vengono accostati termini che rappresentano concetti totalmente diversi, inoltre per ognuno di essi vengono raccolti tutti i significati insieme, sebbene non abbiano niente in comune. Infine un dizionario tradizionale è ridondante per quanto riguarda i sinonimi, poiché lo stesso concetto viene ripetuto più volte.

WordNet non è organizzato alfabeticamente ma su base concettuale: in esso nomi,

verbi, aggettivi ed avverbi sono organizzati in quattro categorie sintattiche, ognuna delle quali è suddivisa in insiemi di sinonimi (**synset**), che a loro volta rappresentano un concetto lessicale condiviso da tutti i termini ad esso associati. Un termine ovviamente può possedere più di un significato ed essere quindi presente in molti di questi insiemi ed anche in più di una categoria sintattica.

Altro importante fattore che contraddistingue WordNet da un semplice dizionario di vocaboli è che questi insiemi di sinonimi sono collegati tra loro e strutturati in una rete tramite una serie di relazioni che riflettono i principi in base a cui tali concetti sono organizzati nella mente, e che permettono di creare all'interno della categoria sintattica gerarchie di significato.

È interessante osservare che WordNet si basa su un insieme ristretto di semplici ipotesi psicolinguistiche, tuttavia il fatto che tali ipotesi siano state testate sull'intero lessico della lingua inglese le rende più robuste e plausibili. Attualmente è considerato la più importante risorsa disponibile per i ricercatori nei campi della linguistica computazionale, dell'analisi testuale, e di altre aree associate.

Introduciamo un insieme di termini significativi propri della terminologia di WordNet:

- **Categoria sintattica:** sono le grandi categorie in cui sono suddivisi i termini (ed anche i file in cui sono contenuti) di WordNet. Le categorie sintattiche trattate sono quattro: nomi, verbi, aggettivi ed avverbi.
- **Lemma:** è la parola, il termine a cui viene associato uno o più significati. A volte un lemma è costituito da due o più parole ed in tal caso i singoli termini sono uniti dal carattere underscore.
- **Synset:** un synset rappresenta il significato che viene associato ad un insieme di lemmi appartenenti alla stessa categoria sintattica. In pratica è corretta l'affermazione che ad un synset appartengono un certo numero di lemmi. Un

synset, infatti, può essere rappresentato, oltre che tramite il suo Gloss, per mezzo dell'insieme dei suoi lemmi.

- **Gloss:** un Gloss è una descrizione a parole di uno specifico significato; ogni synset oltre a contenere un insieme di sinonimi possiede anche un gloss.
- **Relazione semantica:** si tratta di una relazione presente fra due sinse appartenenti alla stessa categoria sintattica.
- **Relazione lessicale:** è una relazione tra due lemmi appartenenti a due synsets distinti, sempre relativamente alla stessa categoria sintattica.

I synset possono quindi essere considerati come designatori non ambigui dei significati delle parole: la loro funzione non è di definire i concetti ma semplicemente di indicare che esistono. Tuttavia nel corso del lavoro di sviluppo di WordNet è emerso che in alcuni casi i synset non sono sufficientemente informativi in quanto alcuni concetti sono espressi da una sola parola, la quale non può così essere disambiguata nemmeno da parte di persone che abbiano già acquisito tutti i suoi significati. Per questo motivo, e per rispondere alle richieste di maggior informazione provenienti dall'ambito della linguistica computazionale, a partire dalla versione di WordNet 1.6, ad ogni synset è stata aggiunta un gloss, cioè una breve definizione del concetto accompagnata da un esempio, che non svolge tuttavia nessun ruolo dal punto di vista psicolinguistico. In Figura A.0.1, è riportato un esempio di come WordNet mostra tali informazioni.

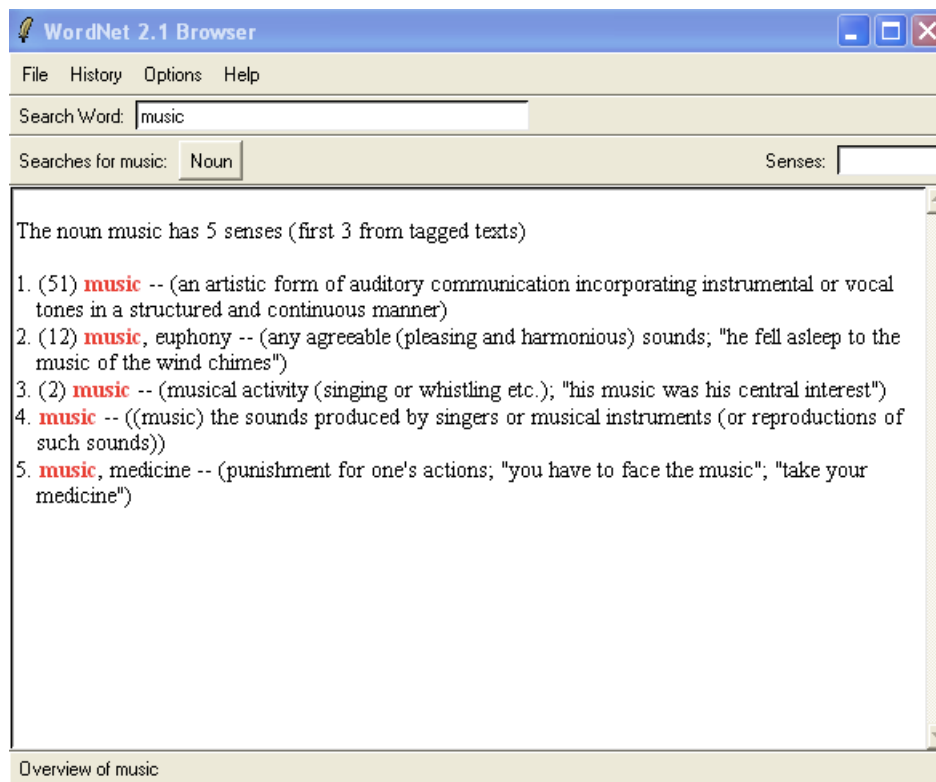


Figura A.0.1: Esempio WordNet usando la parola "music"

Le principali relazioni semantiche sono:

- **Sinonimia:** la relazione più importante in WordNet è senz'altro la sinonimia, visto che essa è alla base della costruzione della matrice lessicale. Secondo una definizione attribuita a Leibniz, due espressioni sono sinonime se la sostituzione di una per l'altra non cambia il valore della frase in cui avviene la sostituzione. Da tale definizione, risulta chiaro che i veri sinonimi sono molto rari. Una definizione meno forte esprime la sinonimia nel seguente modo: due espressioni sono sinonime in un contesto C se la sostituzione di una per l'altra in C non modifica il valore della frase. Questa definizione di sinonimia rende necessaria la divisione in nomi, verbi, aggettivi e avverbi di WordNet in modo che parole appartenenti a categorie diverse non possano essere sinonime visto che non possono essere intercambiabili.
- **Antonimia:** la relazione di antonimia associa ad un termine il suo contrario.

L'antonimo di una parola  $x$ , a volte è  $-x$ , ma non sempre: ad esempio *white* e *black* sono antonimi, ma dire che un oggetto non è bianco, non significa che sia necessariamente nero. Questo tipo di relazione può essere usato per coppie di termini appartenenti a tutte le categorie sintattiche in cui è suddiviso WordNet.

- **Iponimia/iperonimia:** un concetto rappresentato dal synset  $\{x, x', \dots\}$  è un iponimo del concetto rappresentato dal synset  $\{y, y', \dots\}$  se può essere costruita una frase del tipo *An  $x$  is a (kind of)  $y$* . Questo tipo di relazione è permesso solo per le categorie dei nomi e dei verbi; nel caso dei verbi, la relazione di iponimia viene chiamata "troponimia". La relazione di iponimia/iperonimia, chiamata anche relazione "IS-A", è transitiva e simmetrica; essa genera una struttura gerarchica semantica simile alla gerarchia di specializzazione/generalizzazione presente nei modelli relazionali. La radice di tale gerarchia è occupata dai concetti più generali, mentre al livello delle foglie ci sono i concetti specializzati. Come nei modelli relazionali e nei modelli ad oggetti, ogni iponimo eredita tutte le caratteristiche del concetto più generale e al più viene aggiunta una caratteristica che distingue l'iponimo dal concetto al livello superiore e dagli altri iponimi di quel concetto. Ad esempio la parola *apple* eredita tutte le caratteristiche del suo iperonimo *fruit* ma si distingue da tutti gli altri tipi di frutto per la sua forma, il colore, il gusto, etc.
- **Meronimia:** un altro tipo di relazione semantica importante per i lessicografi è la relazione di meronimia/olonimia, detta anche relazione parte-tutto; essa si applica solo alla categoria dei nomi. Un concetto rappresentato da un synset  $\{x, x', \dots\}$  è un meronimo del concetto  $\{y, y', \dots\}$  se può essere costruita una frase del tipo: *A  $y$  has an  $x$  (as a part)* oppure *An  $x$  is a part of  $y$* . La relazione di meronimia è transitiva e asimmetrica; essa può essere usata per costruire una gerarchia (bisogna considerare però che un meronimo può avere

più omonimi).

- **Implicazione:** questa relazione può essere utilizzata solo per i verbi. Un verbo  $x$  implica  $y$  se  $x$  non può essere fatto senza che  $y$  sia o sia stato fatto. Questo tipo di relazione non è simmetrico, infatti se  $x$  implica  $y$ , non è vero il contrario, tranne nel caso in cui i due verbi siano sinonimi.
- **Causale:** anche questa relazione si applica solo alla categoria dei verbi. La relazione causale lega due verbi, uno che produce una causa, che chiameremo per motivi di brevità  $C$ , e uno che “riceve” un effetto, che chiameremo  $E$ . Ad esempio: *show / see*; in questo caso *show* è il verbo  $C$ , mentre *see* è il verbo  $E$ .

