UNIVERSITA' DEGLI STUDI DI PADOVA

Facoltà di Ingegneria

TESI DI LAUREA

NATURAL LANGUAGE PROCESSING E TECNICHE SEMANTICHE PER IL SUPPORTO ALLA DIAGNOSI: UN ESPERIMENTO

Corso di Laurea Specialistica in Ingegneria Informatica

Relatore: Prof. Mauro Bisiacco

Correlatori: Dr. Claudio Saccavini
Ing. Silvia Mancin

Laureanda: Raffaella Tono

28 giugno 2010 – anno accademico 2009/10

INDICE

INTRODUZIONE 1

CAPITOLO 1 - INFORMAZIONE STRUTTURATA E NON STRUTTURATA 3

1.1 Ottenere informazione dai documenti 3

1.2 Information Retrieval 4

1.3 Information Extraction 8

1.4 Informazione non strutturata nel web 12

1.5 Informazione non strutturata nel settore medico 13

1.6 Semantic web vision 19

1.7 I settori della computer science coinvolti nel semantic web 20

1.8 Documenti del web semantico 21

1.9 Dati e metadati 22

1.10 Ottenere un documento strutturato con i metadati 23

1.11 Ontologie 25

1.12 Ontologie nel web 29

1.13 Ontologie nel campo medico 34

CAPITOLO 2 – NATURAL LANGUAGE PROCESSING 36

2.1 Definizione e scopo 36

2.2 Fasi dell'NLP 39

2.2.1 Tokenization 41

2.2.2 Analisi morfologica e lessicale 42

2.2.3 Analisi sintattica e generazione parse tree 44

2.2.4 Named Entities Recognition 46

2.2.5 Analisi semantica 47

2.2.6 CoReference 48

2.2.7 Analisi pragmatica 48

2.3 Machine Learning 49

2.4 Clustering e Classificazione 52

2.4.1 Classificazione binaria 53

2.5 Principali classificatori 55
2.5.1 Decision Tree 55
2.5.2 Naive Bayes 56
2.5.3 *k*-nearest neighbor 57
2.5.4 SVM Support Vector Machine 58

CAPITOLO 3 – RISULTATI SPERIMENTALI 60
3.1 Contesto 60
3.2 Set di input 61
3.3 GATE 66
3.4 Generazione corpora 67
3.5 Annie 68
3.5.1 Document Reset 68
3.5.2 Tokeniser 68
3.5.3 Sentence Splitter 69
3.5.4 Part of Speech 69
3.5.5. Gazetteer 71
3.5.6 JAPE Transducer 75
3.6 Classificazione 79
3.7 Machine Learning in GATE 81
3.7.1 Preparazione corpora 82
3.7.2 Configurazione file 83
3.7.3 Fasi di training e application 84
3.8 Metriche di valutazione 85
3.9 Corpus Quality Assurance 87
3.9.1 Training set da 50 referti - Application set da 30 referti 89
3.9.2 Training set da 140 referti - Application set da 73 referti 90
3.9.3 Training set da 100 referti - Application set da 113 referti 91
3.10 Considerazioni sull'uso di GATE 93
3.11 Uso di NLP per tradurre informazioni cliniche da database 94
3.12 RadiO: application ontology 95
3.13 Conclusioni 97
Bibliografia 99
Sitografia 102

INDICE DELLE FIGURE

Figura 1	*pneumothorax* in SNOMED (browser di Bioportal)	15
Figura 2	*pneumothorax* con ICD-9-CM	16
Figura 3	*pneumothorax* con MeSH	17
Figura 4	*pneumothorax complication* in RadLex	19
Figura 5	rappresentazione ad albero di un documento XML	25
Figura 6	livelli semantici	29
Figura 7	ricerca del termine *gate* con hakia.com	31
Figura 8	ricerca del termine *gate* con kosmix.com	32
Figura 9	ricerca del termine *gate* con Powerset.com	33
Figura 10	ontologie consultabili su NCBO Bioportal	34
Figura 11	hierarchy to Root	35
Figura 12	network Neighborhood	35
Figura 13	Rosetta Translation	38
Figura 14	generazione sinomini con AdWords	39
Figura 15	Italian Parser	45
Figura 16	classificazione binaria lineare e non lineare	54
Figura 17	esempi di Voronoi tessellation	58
Figura 18	classificatore a margine massimo	59
Figura 19	distribuzione frequenze	65
Figura 20	ambiente di lavoro di GATE	67
Figura 21	creazione del corpus	68
Figura 22	risultato del tokeniser	69
Figura 23	risultati ottenuti con tokeniser, sentence splitter e POS tagger	70
Figura 24	utilizzo di annotazioni con GATE	72
Figura 25	individuazione dei lookup	75
Figura 26	individuazione degli attributi rilevanti	78
Figura 27	annotazioni usate per ML	82
Figura 28	corpus pipeline (sezione di destra)	83
Figura 29	annotation set di tutti gli algoritmi di ML	84
Figura 30	Corpus Quality Assurance	87

Introduzione

Lo studio condotto in questa tesi nasce da una collaborazione con il Consorzio Arsenàl.IT[1], Centro Veneto Ricerca ed Innovazione per la Sanità Digitale con sede a Treviso. Il consorzio è un centro studi a carattere regionale che opera nel settore dell'informatica e delle soluzioni ICT (Information and Communication Technology) per la Sanità e il Sociale ed è nato nel 2005 in seguito a una ricerca avviata dalla Regione Veneto sulle applicazioni di e-Health nel territorio regionale.
Oggi si occupa di ricerca per l'innovazione, formazione, ingegneria dell'offerta, standardizzazione e normalizzazione; esso svolge attività per conto e a vantaggio dei consorziati su tematiche e linee guida di carattere sovra-aziendale e di evidente interesse per la regione Veneto.

Le attività progettuali di cui si occupa si ispirano al modello dell'Health Technology Assessment (HTA), focalizzando la propria attenzione sull'applicazione dell'Informatica e della Telematica, sui processi di erogazione dell'assistenza socio-sanitaria *(e-Health Technology Assessment - eHTA)* e sulla valutazione sistematica delle diverse forme d'impatto – clinico, economico, organizzativo e sociale - generate dall'introduzione di tecnologie ICT nel sistema socio-sanitario regionale.

In particolare, la proposta di studio concordata con il consorzio riguarda la valutazione di tecniche da applicare a testi non strutturati per ottenere document classification. Nel caso specifico, analizzando un set di referti radiologici inerenti alla patologia dello pneumotorace, si tratta di capire se un approccio basato su tecniche semantiche ed elaborazione del linguaggio naturale possono rappresentare uno strumento efficace per la classificazione di tali referti in due categorie: quelli che dichiarano la presenza della patologia, e quelli che ne dichiarano la sua assenza, verificando l'affidabilità dei risultati ottenuti.

L'utilità di questa classificazione si inserisce in un contesto di supporto alla diagnosi: la consultazione di un grosso database di referti medici è più agevole se questi ultimi risultano già classificati in base a uno o più criteri, e nel caso specifico, se appartengono all'una o all'altra categoria. Già questa prima distinzione consente di far risparmiare tempo ed energie al radiologo che, per esigenze

1 http://www.consorzioarsenal.it/

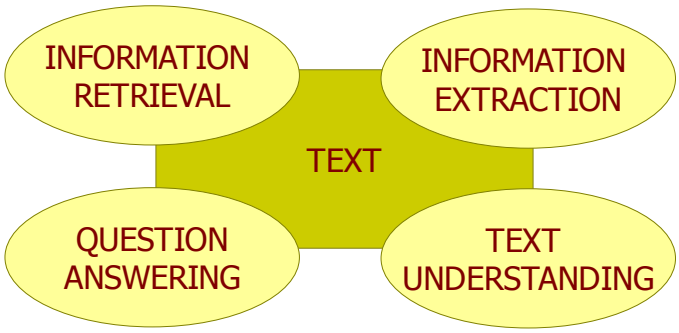
professionali, potrebbe voler consultare un insieme di referti appartenenti solo ad una delle due categorie per approfondire gli aspetti e le caratteristiche di una patologia.

Estendere questa document classification ad altre patologie mantenendo lo stesso criterio di classificazione, oppure fissare criteri di classificazione diversi e applicarli di volta in volta ad una patologia, può costituire un interessante sviluppo di studio per valutare, in termini ancor più generali, l'efficacia delle tecniche utilizzate che non escludono l'integrazione con altre metodologie ma, anzi, ne possono rappresentare un interessante miglioramento.

Capitolo 1
INFORMAZIONE STRUTTURATA E NON STRUTTURATA

1.1 - 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.



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[2]. 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.

1.2 - 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[3]**"**

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[4]**"**.

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

2 Marco Ernandes - *"Text processing – Basi di dati multimediali"* - 2005 *www.slidefinder.net/t/text_processing_information_extraction/4511739*

3 Gerald Salton - *Automatic Information Organization and Retrieval* -McGraw-Hill Inc. 1968

4 C. Manning, P. Raghavan, H. Schutze - *Introduction to Information Retrieval"*- Cambridge 2008

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[5])

- 1990’s:

Ricerca di documenti attraverso Internet (Lycos[6], Yahoo[7], Altavista[8]) e

5 http://www.nlm.nih.gov/pubs/factsheets/medline.html
6 http://www.lycos.it/
7 http://it.yahoo.com/
8 http://it.altavista.com/

categorizzazione automatica di documenti

- 2000's:

Link analysis (Google[9]), Question Answering, IR nei multimedia (immagini, video, audio), text summarization (scelta di frasi dal documento originale da presentare)

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 sinomini 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,

9 http://www.google.it/

valutando l'autorità/affidabilità delle fonti, eseguendo operazioni sui testi come l'indicizzazione, lo stemming[10]e la lemmatizzazione[11], operando l'espansione delle query mediante thesaurus[12] 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[13] o Vivisimo[14]) 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, video, immagini (Autonomy[15] o Recoll[16])
- *Blog search engine*, motore di ricerca specializzato nei blog. (Technorati[17] o Amatomu[18]) o *Bookmarks search engine* specializzato nei preferiti (delicious[19])

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

10 *Stemming:* funzione che consente di reperire documenti di testo contenenti parole con radici comuni alle keyword dell'interrogazione. Ad esempio, se la keyword è "danza", saranno restituiti anche documenti contenenti le parole "danzatore" o "danzatrice". Lo stemming a volte può causare ambiguità: se trasforma *gaseous* in *gas* può non essere facile distinguere il termine con significato *carburante* da quello di *stato* della materia

11 *Lemmatizzazione:* processo di riduzione di una forma flessa di una parola alla sua forma canonica detta lemma. Nell' elaborazione del linguaggio naturale, la lemmatizzazione è il processo algoritmico che determina automaticamente il lemma di una data parola. In italiano il verbo "camminare" può apparire nelle seguenti forme flesse: "cammina", "camminò", "camminando" e così via. La forma canonica, "camminare", è il lemma della parola ed è la forma di riferimento per cercare la parola all'interno di un dizionario. La combinazione della forma canonica con la relativa parte del discorso (part-of-speech) è detta *lessema* della parola.

12 *Thesaurus:* insieme delle parole chiave che danno accesso a un elenco di sinonimi

13 http://www.oracle.com/

14 http://vivisimo.com/

15 http://www.autonomy.com

16 http://www.lesbonscomptes.com/recoll/

17 http://technorati.com/

18 http://www.amatomu.com/

19 http://delicious.com/

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.

1.3 - 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[20].

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 obbiettivi ancora lontani[21]. 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 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.

20 http://it.wikipedia.org/wiki/ - Il *data mining* cerca patterns all'interno di grossi dataset ricavando informazione inizialmente sconosciuta; se fosse nota, sarebbe reperibile con opportune query

21 W.B. Croft, D. Metzler, T. Strohman - *"Search Engines"* - Pearson 2010

- *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.

Consideriamo per chiarezza il seguente esempio:
"L'innovativo profumo è stato lanciato Mercoledì. E' stato ideato da Coco Chanel che lo ha chiamato Chanel n. 5. Madame Chanel è proprietaria della Maison Chanel srl"

Si supponga di applicare le fasi sopra indicate:
(NE): "profumo", "Mercoledì", "Coco Chanel", "Chanel n. 5", "Maison Chanel srl"
(TE): profumo → "innovativo", "lanciato Mercoledì",…
(TR): "Madame Chanel" *è proprietaria di* "Maison Chanel srl"
(ST essenze femminili): profumo --> lanciato Mercoledì / ideato da Madame Chanel per conto della "Maison Chanel srl"
(CO): profumo → "lo", "Chanel n. 5"; "Coco Chanel" → "Madame Chanel"

Semplificando ulteriormente il processo, si può immaginare l'IE come somma di tre fasi:

IE =(segmentazione + classificazione + associazione)

Applicandole all'esempio di seguito riportato, si ottengono i risultati evidenziati in tabella:
"Il direttore dei grandi magazzini Luna, Domenico Zorro, stanco della crisi che abbatte da mesi le vendite, ha promosso un periodo di forti sconti. Da domani, nei negozi Luna, 150 articoli in promozione con sconto 50%. Zorro ha espresso ottimismo nelle vendite. Il vicedirettore del grandi magazzini Sole, Arturo Batman, ha dichiarato che tali iniziative dovrebbero essere concordate. I magazzini Sole, da sempre concorrenti dei magazzini Luna, temono infatti un improvviso calo delle vendite. Nella vicenda e' intervenuto il sindaco, Matteo Spider il quale...."

segmentazione	**classificazione**		**associazione**	
Direttore	Direttore	job	Direttore	job
Luna	*Luna*	nome	*Luna*	nome
Domenico Zorro	Domenico Zorro	persona	Domenico Zorro	persona
Luna	*Luna*	nome		
Zorro	Zorro	persona	*Luna*	nome
vicedirettore	vicedirettore	job	Zorro	persona
Sole	*Sole*	nome		
Arturo Batman	Arturo Batman	persona	vicedirettore	job

Sole Luna sindaco Matteo Spider	*Sole* *Luna* sindaco Matteo Spider	nome nome job persona	*Sole* nome Arturo Batman persona *Sole* nome *Luna* nome sindaco job Matteo Spider persona

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.

1.4 - INFORMAZIONE NON STRUTTURATA NEL WEB

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 perché 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[22]. 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.

1.5 - INFORMAZIONE NON STRUTTURATA NEL SETTORE MEDICO

Anche il settore medico sente fortemente il bisogno di strutturare l'informazione e di renderla facilmente accessibile: il numero di risorse disponibili ai ricercatori è enorme e così, a fronte di una richiesta inoltrata al motore di ricerca, l'utente si vede restituire una quantità considerevole di documenti.
Se da un lato la letteratura medica è particolarmente abbondante, dall'altro la prassi comune a medici e organizzazioni sanitarie di usare termini clinici differenti

22 G.Antoniu, F. Van Harmelen -*"A Semantic Web Primer"*- The MIT Press 2008

per indicare lo stesso concetto rende particolarmente complicato il processo di recupero di informazione relativo a quel concetto. Per esempio, se per un cardiologo i termini “attacco di cuore”, “infarto miocardico” e “MI” rappresentano la stessa cosa, per un computer si tratta di entità differenti.

Oltre ai problemi già accennati legati alla sinonimia, c'è l'esigenza di condividere a scopi di ricerca numerose banche dati in possesso dei vari organismi operanti nel settore: la possibilità di pubblicarle sul web (in un formato facilmente elaborabile) e di confrontarle tra loro può essere estremamente utile ma pone problemi di automazione e interoperabilità: *“La stabilità e l'interoperabilità attraverso sistemi informativi multipli, all'interno delle community e tra le community stesse, sta diventando una priorità urgente*[23]”
Gli sforzi compiuti nel settore tentano di superare le due principali barriere: da un lato la varietà di modi in cui gli stessi concetti sono espressi da persone diverse e in differenti risorse accessibili ai sistemi informatici e, dall'altro, la distribuzione di informazione utile su sistemi e database numerosi ma indipendenti. Tra gli esempi più importanti, in tal senso, è la creazione di vocabolari specializzati per i vari domini di applicazione. Vediamone alcuni in dettaglio.

UMLS® - **U**nified **M**edical **L**anguage **T**hesaurus[24]
Si tratta di una raccolta integrata di vocabolari controllati inerenti alle scienze biomediche. E' un progetto della National Library of Medicine (NLM – National Institute of Health, United States) nato nel 1986: esso fornisce una mappatura tra i vocabolari, consentendo di unificare la terminologia. Lo scopo è facilitare lo sviluppo di sistemi informatici che si comportano come se “comprendessero” il significato del linguaggio biomedico e sanitario. Proprio per tale motivo può essere visto come un esteso thesaurus e un'ontologia di concetti biomedici; inoltre, fornisce utilità per il Natural Language Processing (NLP). Il suo uso è principalmente rivolto agli sviluppatori di sistemi informatici dedicati al settore medico. L'UMLS si compone dei seguenti componenti: un *metathesaurus* ossia un insieme di concetti e termini provenienti da vari vocabolari controllati e le relazioni esistenti tra loro (rappresenta il database principale); un *semantic network* ossia un set di catogorie e relazioni utilizzate per classificare e correlare le entries del metathesaurus; alcuni *software* a supporto dedicati.

23 V.Kashyap, C.Bussler, M. Moran - *“The Semantic Web”* - Springer 2008
24 http://www.nlm.nih.gov/research/umls/about_umls.html

SNOMED CT - Systematized Nomenclature of Medicine -- Clinical Terms[25]

Come dice la descrizione stessa, si tratta di una collezione organizzata di termini medici processabile da un computer. L'iniziativa, cominciata nel 2002 grazie al College of American Pathologists, ha visto la fusione progressiva di più raccolte terminologiche; oggi rappresenta il più esteso vocabolario clinico con più di 344,000 concetti, integrando anche ICD-9-CM[26]. Essa copre numerose aree dell'informazione clinica, come malattie, diagnosi, procedure, microorganismi, farmaci, etc. Il suo pregio è di consentire una modalità di indicizzazione, memorizzazione, recupero e aggregazione di dati clinici. Inoltre aiuta la strutturazione dei contenuti presenti nei record medici mediante codifica ottenendo così un'informazione più fruibile sia per scopi di cura che di ricerca.
In aprile 2007 SNOMED CT è stata acquistata da IHTSDO (International Health Terminology Standard Development Organization).

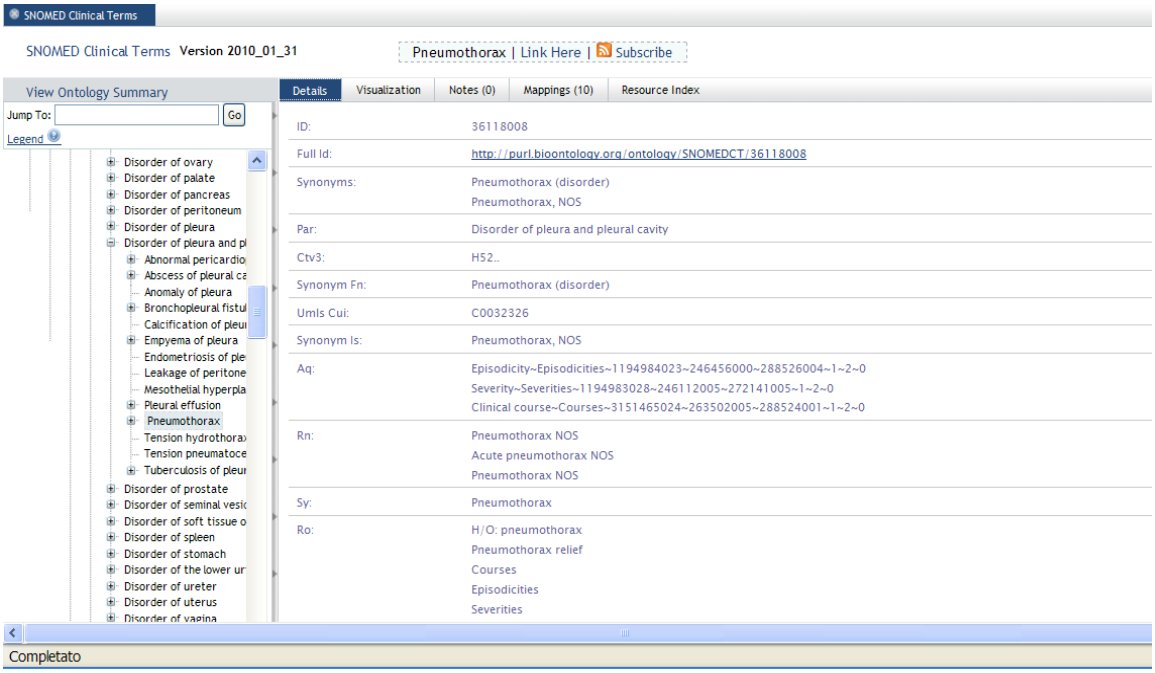
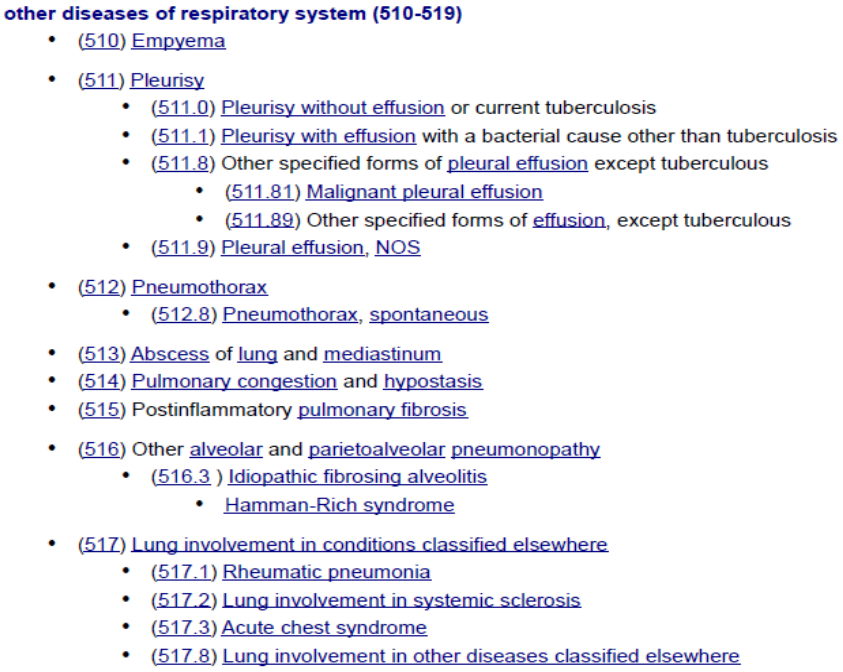


Figura 1 - *pneumothorax* in SNOMED (browser di Bioportal)

25 http://www.ihtsdo.org/about-ihtsdo/
26 http://en.wikipedia.org/wiki/SNOMED_CT

ICD-9-CM –
International **C**lassification of **D**iseases, **9th** revision, **C**linical **M**odification[27]

E' una classificazione finalizzata a tradurre in codici alfa-numerici i termini medici in cui sono espresse diagnosi di malattia, problemi di salute nonché procedure diagnostiche e terapeutiche. Si compone di tre volumi di cui i primi due contengono codici relativi alle diagnosi, mentre l'ultimo contiene codici relativi alle procedure. Il termine "clinical" è utilizzato per sottolineare le modificazioni introdotte: rispetto alla ICD-9, che è fortemente caratterizzata dall'orientamento a scopo di classificazione delle cause di mortalità, la ICD-9-CM è soprattutto orientata a classificare gli stati patologici. Infatti, le principali modificazioni introdotte sono finalizzate a consentire sia una classificazione più precisa ed analitica delle formulazioni diagnostiche, attraverso l'introduzione di un quinto carattere, sia l'introduzione della classificazione delle procedure. Dal 1 ottobre 2013 la classificazione ICD-9-CM sarà sostituita dalla ICD-10-CM che ne rappresenta la naturale evoluzione.



other diseases of respiratory system (510-519)

- (510) Empyema
- (511) Pleurisy
  - (511.0) Pleurisy without effusion or current tuberculosis
  - (511.1) Pleurisy with effusion with a bacterial cause other than tuberculosis
  - (511.8) Other specified forms of pleural effusion except tuberculous
    - (511.81) Malignant pleural effusion
    - (511.89) Other specified forms of effusion, except tuberculous
  - (511.9) Pleural effusion, NOS
- (512) Pneumothorax
  - (512.8) Pneumothorax, spontaneous
- (513) Abscess of lung and mediastinum
- (514) Pulmonary congestion and hypostasis
- (515) Postinflammatory pulmonary fibrosis
- (516) Other alveolar and parietoalveolar pneumonopathy
  - (516.3) Idiopathic fibrosing alveolitis
    - Hamman-Rich syndrome
- (517) Lung involvement in conditions classified elsewhere
  - (517.1) Rheumatic pneumonia
  - (517.2) Lung involvement in systemic sclerosis
  - (517.3) Acute chest syndrome
  - (517.8) Lung involvement in other diseases classified elsewhere

Figura 2 - *pneumothorax* con ICD-9-CM

27 http://en.wikipedia.org/wiki/ICD-9-CM#ICD-9-CM

MeSH - Medical Subject Headings[28]

E' un enorme vocabolario controllato ideato e gestito dalla U.S. National Library of Medicine (NLM) con l'obiettivo di indicizzare gli articoli e la lettura scientifica in ambito biomedico presenti nel database bibliografico di MEDLINE/PubMed e nel catalogo libri della NLM. La terminologia di MeSH consente di reperire informazione anche quando nel materiale scientifico è utilizzato un termine diverso da quello ricevuto in input, purchè naturalmente ci si riferisca al medesimo concetto.
Nei database Medline o PubMed il contenuto di ciascun articolo è indicizzato con 10-15 descrittori, di cui solo uno o due termini indicano l'argomento principale e che pertanto sono detti *major,* identificati con l'apposizione di un asterisco. L'utilità di queste operazioni può essere verificata nelle operazioni di ricerca in quanto il ricorso al vocabolario controllato MeSH e ai descrittori permette di essere molto selettivi e di ridurre enormemente il "rumore" che si otterrebbe se si utilizzassero solo le parole libere del linguaggio comune. La traduzione dei MeSH dalla lingua inglese a quella italiana, avviata nel 1986, è curata dall'Istituto Superiore di Sanità e rientra nell'ambito dell'UMLS.

Figura 3 - *pneumothorax* con MeSH

28 http://www.ncbi.nlm.nih.gov/mesh

GO - Gene Ontology[29]

Si tratta di un progetto simile a UMLS applicato però al contesto genetico. La realizzazione di un nuovo antibiotico, per esempio, richiede che il ricercatore si documenti su tutte le informazioni contenute nel DNA dei geni coinvolti nella sintesi di una proteina batterica.
Il progetto rappresenta uno sforzo collaborativo per raggruppare le descrizioni di "gene products" (materiale biochimico associato ai geni) presenti in differenti database. E' iniziato nel 1998 integrando tre database e successivamente ha incluso ulteriori database, compresi quelli principali inerenti a piante e animali. L'uso dei termini di GO facilita metodi di interrogazione uniforme su più database.

BioPAX - Biological PAthways eXchange[30]
Nato in Canada nel 2002, questo progetto ha lo scopo di sviluppare un formato di scambio comune per dati relativi a Biological Pathways, per facilitare la condivisione di pathway information tra database/utenti e per costruire una risorsa open source di pathway information. Un pathway è una rete biologica inerente ad uno specifico processo fisiologico: è costituita da componenti biologici che interagiscono l'uno con l'altro generando un singolo effetto biologico (il pathway metabolico è stato tra i primi ad essere rappresentato)

RADLEX[31]

La definizione data è "lessico per il reperimento e l'indicizzazione uniformi delle risorse informative radiologiche". Dato che le immagini radiologiche, i report sulle immagini e, più in generale, numerose informazioni di carattere medico si muovono online, diventa irrinunciabile l'esigenza di un linguaggio unificato per organizzare e recuperare i dati. Anche i radiologi, infatti, utilizzano una varietà di terminologia che pone serie difficoltà al recupero e allo scambio di informazione: ecco perché RadLex si propone come una singola risorsa unificata di termini radiologici.

29 http://www.geneontology.org/GO.doc.shtml
30 http://www.biopax.org/
31 http://www.radlex.org/viewer

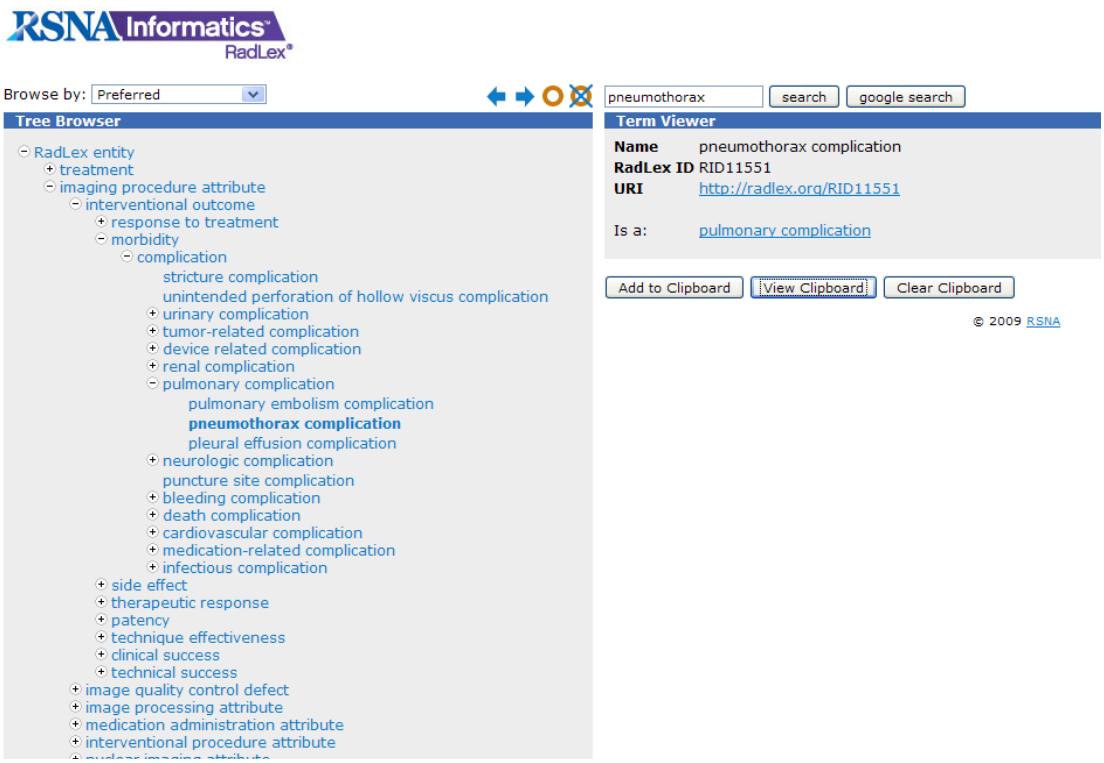


Figura 4: pneumothorax complication in RadLex

Tutti questi esempi rendono l'idea degli sforzi compiuti negli anni per cercare di integrare e utilizzare l'informazione. Tuttavia, l'utilizzo di vocabolari specializzati è solo il primo passo verso una effettiva fruizione dei numerosi dati a disposizione.
L'ideale sarebbe poter disporre di strumenti automatici capaci di elaborare l'informazione e di interagire con i programmi esistenti anche se questi ultimi sono stati disegnati in maniera totalmente indipendente.
La realizzazione della "Semantic Web Vision", dunque, sembra ancora lontana.

1.6 - SEMANTIC WEB VISION

"I have a dream for the Web [in which computers] become capable of analyzing all the data on the Web – the content, links, and transactions between people and computers.
A 'Semantic Web', which should make this possible, has yet to emerge, but when it does, the day-to-day mechanisms of trade, bureaucracy and our daily lives will be handled by machines talking to machines.
The 'intelligent agents' people have touted for ages will finally materialize."
(Tim Berners-Lee, 1999)[32]

Il primo ad usare l'espressione "web semantico" è stato Tim Berners: esso immagina computer capaci di analizzare tutti i dati presenti nel web, di interagire tra loro e

32 http://en.wikipedia.org/wiki/Semantic_Web

di gestire molti aspetti della vita quotidiana. Ma cosa si intende per “semantico”?

La semantica si definisce come “lo studio del significato delle parole[33]” Tuttavia, da una prospettiva ingegneristica, la semantica può essere intesa come “significato e uso di un'informazione”. E infatti la premessa di base del web semantico è far sì che ogni “frammento di dato” abbia un significato ben definito a scopo di utilizzo.

I contenuti web disponibili si distinguono in due grandi tipologie:

- “di superficie”, quello che tutti conoscono, costituito da pagine web pubbliche
- “profondo”, costituito da speciali database accessibili via web e siti dinamici, generalmente sconosciuti all'utente medio, la cui quantità di informazione supera la prima di 400-550 volte[34].

Con il miglioramento della tecnologia dei motori di ricerca, molte pagine in formato non HTML (PDF, DOC, XLS,...) sono state rese visibili ai motori di ricerca che possono individuarle e indicizzarle. Lo scopo del web “profondo” consiste primariamente in questo: rendere i dati memorizzati nei database disponibili e elaborabili da applicazioni web che li convertono dinamicamente (“al volo”) in pagine web.
Il Web Semantico può essere immaginato allora come un'estensione del web corrente in cui ai dati sono associate informazioni e significati ben definiti, per consentire alle persone e ai computer di lavorare meglio insieme.

1.7 - I SETTORI DELLA COMPUTER SCIENCE COINVOLTI NEL SEMANTIC WEB

Per i dati finora prodotti e resi disponibili, si cerca di studiare tecniche sempre più efficienti per manipolarli ed estrarre contenuti da essi in modo affidabile; per l'informazione futura è auspicabile che essa rispetti le linee guida e la filosofia del web semantico, così da trarne tutti i vantaggi che esso offre.

I ricercatori dell'informatica e della computer science coinvolti dal web semantico sono numerosi[35]:

- *ricercatori esperti di databases e sistemi informatici*
  per sviluppare modelli di elaborazione che catturino la semantica dell'informazione mettendo a punto sistemi efficienti e scalabili di memorizzazione, indicizzazione e interrogazione dati
- *ricercatori nella rappresentazione della conoscenza*

33 Vocabolario Zingarelli – Zanichelli 2008
34 V. Kashyap, C. Bussler, M. Moran - *“The Semantic Web”* - Springer 2008
35 V. Kashyap, C. Bussler, M. Moran - *“The Semantic Web”* - Springer 2008

per sviluppare schemi e teorie che rappresentino la conoscenza. Questa comunità si concentra sull'implementazione di meccanismi di ragionamento e inferenza.

- *ricercatori nel reperimento dell'informazione*
  hanno sviluppato thesaurus e tassonomie per guidare la ricerca e l'esplorazione di documenti in vaste collezioni di documenti e sperimentato approcci statistici per catturare informazione nascosta tramite il calcolo delle frequenza di termini combinati
- *ricercatori nel settore del machine learning ed elaborazione del linguaggio naturale*
  si sono specializzati nell'annotazione semantica dei dati e dei documenti rispettando un ben definito set di categorie e concetti. Recentemente si sono ottenuti dei miglioramenti utilizzando ontologie di apprendimento.
- *ricercatori del peer-to-peer*
  La semantica dell'informazione si è rivelata un supporto efficiente per localizzare risorse rilevanti. Sono stati proposti approcci che usano annotazioni semantiche e mappe localizzate per individuare le risorse e integrare in modo performante i dati.
- *ricercatori dei servizi web*
  hanno proposto modelli di elaborazione e ontologie che consentano il riutilizzo l'interoperabilità delle applicazioni.

La forza del web semantico è data dal fatto che ormai ogni settore usa sistemi basati sul web per semplificare i processi e acquisire vantaggio competitivo nel mercato. Pertanto diventano sempre più prevalenti approcci che rendono le macchine in grado di “capire” i dati nel web.

1.8 - I DOCUMENTI DEL WEB SEMANTICO

Oggi i contenuti web sono formattati per essere ben compresi dalle persone piuttosto che dai programmi e il linguaggio predominante con cui sono scritte le pagine web è l'HTML, linguaggio che non prevede marcatori utili alla semantica. Un contenuto web semantico, invece, dovrebbe essere contemporaneamente comprensibile per l'utente e processabile per la macchina. Ciò richiede la possibilità di inserire varie annotazioni leggibili per l'utente e di specificare il significato del contenuto in modo preciso e disambiguo (un risultato si può ottenere tramite le varie specifiche XML come RDF). In pratica, il documento deve essere auto-descrittivo, e questa caratteristica si ottiene parzialmente producendo un linguaggio comune per specificare dati e metadati nel web. La standardizzazione è l'elemento chiave per migliorare la comprensione da parte degli elaboratori dei

dati e contenuti del web ed è fattibile tramite l'uso di termini di base nella creazione dei descrittori dei metadati.

1.9 - DATI E METADATI

L'informazione presente sul web può apparire sotto forma di dati *strutturati* (es. database relazionali), *semi-strutturati* (es. pagine scritte in XML), o *non-strutturati* (es. file di tipo .txt). Il concetto di *dato* è intuitivo: qualunque informazione rappresentata in modo da poter essere trattata da un calcolatore costituisce in pratica un dato.
Diverso invece il concetto di *metadato* che costituisce la caratteristica principale del web semantico e ha che il compito di descrivere il *dato:*

I *metadati e le annotazioni* si riferiscono al dominio specifico che vogliono descrivere e sono associati ai dati. Nel suo significato più generale, il metadato è definito come "informazione relativa al dato". Per database strutturati, il più comune esempio di metadato è lo schema del database. I metadati possono essere usati per memorizzare proprietà relative al formato, utili nell'accesso o nel recupero dei dati. Essi possono descrivere il contenuto informativo oppure rappresentarne un sommario quando i dati sono descritti in maniera molto analitica. Sono utili, inoltre, per rappresentare proprietà o relazioni tra oggetti appartenenti a tipologie eterogenee.

La funzione dei metadati è duplice:

- consentire l'astrazione dei dettagli di rappresentazione come il formato o l'organizzazione dei dati, catturando informazione sul contenuto degli stessi, indipendentemente dai dettagli di rappresentazione
- consentire la rappresentazione del dominio della conoscenza cui i dati appartengono. Tale conoscenza può essere usata per fare inferenze sui dati, per misurare la loro rilevanza e per identificare relazioni tra dati memorizzati in repository e siti web

Nel primo caso si parla di *metadati indipendenti dal contenuto* che, come dice il termine stesso, catturano informazioni che non dipendono dal contenuto del documento: per esempio, identificatori, numero di un record, numero per tracciare in laboratorio i campioni di un paziente, identificatori web-related come location, URI e mime types. Non c'è contenuto informativo in questi dati, ma sono comunque

utili per la gestione informatica dei dati.
Nel secondo caso invece si parla di *metadati basati sul contenuto*: essi catturano l'informazione inerente alla struttura del documento, oppure si riferiscono ad un particolare dominio di conoscenza. Per esempio “size” potrebbe indicare la dimensione di un'immagine radiologica; se pensiamo alla cartella clinica di un paziente, i metadati di dominio potrebbero essere: “esami eseguiti durante il ricovero”, “farmaci somministrati”, “medicazioni effettuate”, “allergie”, “controindicazioni” oppure i metadati che utilizzano termini di Medical Subject Headings (MeSH) per le annotazioni biomediche negli articoli di ricerca presenti in MEDLINE.

Se i metadati inerenti a un particolare dominio di conoscenza utilizzano vocabolari controllati che contengono termini e concetti di interesse tipico per un utente di quel dominio (come quelli visti in precedenza) si possono ottenere ontologie ben formate e riutilizzabili.

1.10 - OTTENERE UN DOCUMENTO STRUTTURATO CON I METADATI: un esempio

Si consideri questo esempio in HTML:

```
<h1>LUNA BEAUTY FARM</h1>
<p>Benvenuto nel mio sito. Qui puoi trovare molte informazioni su
    salute, cibo, cucce,  e accessori. Passa a trovarci: trattamenti speciali
    e su misura per combattere stress, pulci e zecche.
<h2>Staff</h2><p> Raffi assiste il tuo amico con grande esperienza (veterinario)
    mentre Lella ti aiuta a scegliere l'accessorio giusto per lui (negozio).
<h2>Orario di apertura</h2>
<p>tutti i giorni ore 15-17. Chiuso lunedì.
<h2>Sconti straordinari</h2>
<p> il primo lunedì di ogni plenilunio
```

Se per le persone l'informazione contenuta in queste poche righe risulta essere abbastanza chiara, per gli elaboratori essa pone qualche problema di interpretazione. Ad esempio, la ricerca in base a keyword troverà facilmente le stringhe 'Raffi' e 'Lella' ma non distinguerà quale delle due indica il veterinario e quale l'addetta alle vendite. Oppure se un cliente è interessato agli sconti straordinari, difficilmente la ricerca restituirà le date corrispondenti ai pleniluni. Questo perchè HTML prevede solo tag di formattazione e non riesce ad associare un significato alle parole.

Una soluzione potente ma ancora di difficile realizzazione potrebbe essere lo sviluppo di un ipotetico algoritmo particolarmente sofisticato che effettua la ricerca basandosi anche sul significato delle keyword; esiste però anche una soluzione più semplice: grazie all'XML, che si sforza si sostituire l'HTML, potremmo trasformare la pagina web in questo modo:

….

```
<company>
  <companyName>Luna Beauty Farm</companyName>
  <treatmentOffered>prodotti e servizi per cani</treatmentOffered>
  <staff>
        <veterinary>Raffa</veterinary>
        <shopGirl>Lella</shopGirl>
  </staff>
  ……..
</company>
```

….

Tale versione, pur visualizzando lo stesso contenuto all'utente è molto più "comprensibile" alle macchine e consente ricerche più mirate grazie all'informazione più ricca. Infatti, *XML consente una rappresentazione dell'informazione che è "accessibile" alle macchine,* mentre il documento HTML non contiene informazioni sulla struttura della pagina, sui singoli pezzi del documento e sulle loro relazioni. Se la pagina web descrive un libro, conterrà il nome dell'autore; ma senza un tag specifico che contrassegni la stringa relativa all'autore del libro, non sarà possibile recuperare tale informazione. Una macchina che elabora documenti XML, invece, dovrebbe "dedurre" automaticamente che l'elemento <autore> contenuto all'interno di un elemento <libro> fa riferimento proprio all'autore del libro.

Inoltre *XML separa il contenuto dalla formattazione:* la stessa informazione può essere visualizzata in modi diversi senza dover produrre copie diverse con lo stesso contenuto. Con HTML invece si deve specificare la formattazione perchè tale linguaggio è nato proprio per visualizzare l'informazione, non per elaborarla.

XML è un metalinguaggio per markup: a differenza dell'HTML, esso non ha un set prefissato di tag ma consente agli utenti di definirlo in base all'uso che ne devono fare. Tanto che numerose comunità hanno creato vocabolari specializzati in vari settori: MathML[36], BSML (Bioinformatic Sequence Markup Language), AML (Astronomy Language Markup) solo per citarne alcuni. Attraverso questa condivisione di terminologia, XML si presta come *formato di scambio dati uniforme*

36 http://www.w3.org/Math/

tra applicazioni: il recupero e l'aggiornamento di informazione *da* e *verso* database appartenenti a settori diversi diventa molto più agevole, senza bisogno di creare particolari software di elaborazione e di querying sviluppati specificatamente da ciascun partner che aumenterebbero la complessità tecnologica.

Un documento XML well-formed ha una struttura, definita di solito esternamente al documento stesso, che può essere rappresentata ad albero. L'albero fornisce un modello di dati formale per XML e deve avere precise caratteristiche:

- c'è esattamente una sola radice
- non ci sono cicli
- ciascun nodo, ad eccezione della radice, ha un unico parent
- ciascun nodo ha un'etichetta
- l'ordine degli elementi è importante

Come si vede nell'esempio, a differenza di una pagina HTML, i contenuti sono rappresentati in modo *chiaro per l'utente* e *in modo univoco per l'elaboratore*.

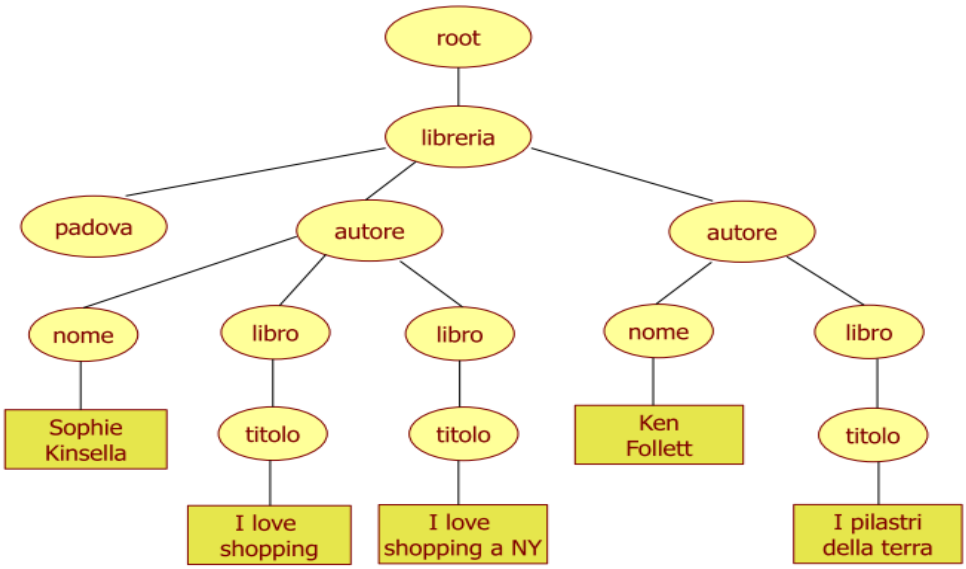


Figura 5 - rappresentazione ad albero di un documento XML

Il linguaggio XML rappresenta una buona soluzione per la codifica dei referti: dopo averli opportunamente annotatati con funzionalità che vedremo dettagliatamente in seguito, è possibile memorizzare il risultato ottenuto e trasformare il tutto in più documenti XML che ben si prestano ad eventuali ulteriori elaborazioni o interrogazioni.

La tecnica appena vista che fa uso di *metadati*, dati cioè relativi "al dato" o, in altre parole, dato che cattura parte del significato di un altro dato, da sola non è in grado di realizzare la "Web Semantic Vision" ma è sicuramente un primo passo

da intraprendere, neanche troppo complicato. Tuttavia, come spesso accade, la sfida maggiore non è scientifica, ma piuttosto l'adozione di tale tecnologie.

1.11 - ONTOLOGIE

Il concetto più innovativo nel web semantico è quello di *ontologia*: il termine ha le sue origini nella filosofia e in tale contesto ci si riferisce alla natura dell'esistenza, "allo studio dell'essere in quanto essere", come la definiva Aristotele o Sant'Anselmo d'Aosta.
Nel contesto della computer science, invece, ha assunto un significato diverso: anziché di "ontologia" si discute di "*un'ontologia*". Alcune autorevoli definizioni recitano:

- "L'ontologia è una specificazione esplicita e formale di una concettualizzazione"[37] dove per:
  concettualizzazione si intende un modello astratto e semplificato del mondo che si desidera rappresentare per qualche scopo (dominio di interesse). Ciò che esiste è esattamente ciò che può essere rappresentato.
  esplicita: la tipologia dei concetti e le relative restrizioni devono essere ben definiti
  formale: deve essere machine-readable
  condivisa: cattura conoscenza consensuale non ristretta a qualche individuo
- "L'ontologia nella computer science è una rigorosa ed esaustiva organizzazione di un dominio della conoscenza che è generalmene gerarchico e contiente tutte le entità rilevanti di quel dominio e le loro relazioni"[38]

L'ontologia può essere dunque pensata come un insieme di termini e relazioni che denotano i concetti rappresentativi di un particolare dominio di informazione; essi possono caratterizzare la conoscenza in una applicazione, in un dominio specifico o in un dominio generalizzato (upper ontologies). L'ontologia descrive una particolare conoscenza in modo univoco: non esiste ambiguità poiché termini e relazioni sono condivisi dall'intera comunità di utenti del dominio di riferimento e, al tempo stesso, assegna ai termini significati "machine-readable".

Le definizioni sopra citate indicano contestualmente le principali fasi per costruire un'ontologia: integrandole opportunamente, si possono così riassumere:

- determinazione di un ambito (dominio);

37 Thomas R. Gruber - *Translation Approach to Portable Ontology Specifications* – appared in Knowledge Acquisition, 5(2):199-220, 1993.
38 http://wordnetweb.princeton.edu

- valutazione di un eventuale riutilizzo, anche parziale, di ontologie già esistenti;
- identificazione dei termini base;
- definizione della tassonomia (gerarchia dei termini e loro relazioni);
- definizione delle proprietà e delle regole (specificandone dominio e range);
- individuazione di nuovi termini prodotti dalle regole;
- definizione di istanze (popolamento dell'ontologia);
- identificazione di eventuali inconsistenze.

Dunque, un'ontologia non include solo termini esplicitamente definiti in essa, ma anche quelli che possono essere derivati usando tali regole.

Un'ontologia può presentarsi in varie forme ma include sempre un vocabolario di termini e una descrizione dettagliata del loro significato. La sua caratteristica principale è la concettualizzazione di un dominio, ossia una rappresentazione formale della realtà così come è percepita e comunemente condivisa. Questo significa che, con riferimento ad una specifica realtà, si possono avere vocabolari diversificati ma un'unica concettualizzazione di quella realtà.

Un sistema terminologico può essere considerato una rudimentale forma di ontologia; in esso riconosciamo termini, relazioni e concetti su diversi livelli descrittivi:

- *Terminologia*: lista di termini che si riferiscono a concetti in un determinato dominio
- *Thesaurus*: i termini sono ordinati (p.e. alfabeticamente) e i concetti possono essere descritti da uno o più sinonimi
- *Vocabolari*: i concetti hanno definizioni, formali o in free text
- *Nomenclatura*: set di regole per comporre nuovi oggetti complessi
- *Classificazione*: i concetti sono arrangiati usando relazioni generiche (*is_a*)
- *Codifica*: i concetti sono individuati da codici.

Per costruire un'ontologia più complessa, invece, è necessario definire:

- *Classi* ossia concetti generali del dominio di interesse.
- *Relazioni* tra classi.
- *Proprietà* assegnate a ciascun concetto (attributi)
- *Restrizioni sulle proprietà* ossia tipologie di dati assunte dalla proprietà
- *Istanze* ossia specifici oggetti del mondo reale rappresentati mediante l'ontologia. Esse ereditano attributi e relazioni dalle classi.

Ontologie e istanze formano la *knowledge base*: i termini definiscono concetti importanti (classi di oggetti) del dominio. Le relazioni generalmente includono gerarchie di classi (X *subClassOf* Y), ma anche proprietà (X *cause* Y), vincoli sul

valore (X *hasValue* Y), statement disgiunti (X *is_a* Y), specifiche sulle relazioni logiche tra oggetti. In pratica, nel contesto web, le ontologie forniscono una *comprensione condivisa* di un dominio, superando le differenze nella terminologia.

Pur presentando delle analogie con gli schemi delle basi di dati, le ontologie sono più complesse di questi ultimi perchè:

- il linguaggio per la definizione delle ontologie è sintatticamente e semanticamente più ricco dei comuni approcci alle basi di dati;
- la terminologia è condivisa e consensuale in quanto destinata allo scambio;
- esse rappresentano una *descrizione formale* di un dominio e non un semplice *contenitore* di dati.

Esiste anche una categoria complessa di ontologie denominate *upper ontologies*: si tratta di ontologie che tentano di descrivere concetti estremamente generali presenti in tutti i domini. Lo scopo è poter accedere a numerose ontologie interfacciandosi con questa ontologia di livello superiore. Tra queste merita ricordare DOLCE[39], *"a Descriptive Ontology for Linguistic and Cognitive Engineering"* il primo modulo di un progetto più vasto denominato Wonder Web[40], portato avanti da un gruppo di ricercatori italiani del Laboratory for Applied Ontology (ISTC-CNR) di Trento.

Riassumendo quanto finora visto, possiamo quindi immaginare una stratificazione come quella illustrata di seguito[41]: la base è costituita dai *dati,* ossia da tutto ciò che è oggetto di elaborazione; al livello superiore si possono collocare i *metadati indipendenti dal contenuto* che aggiungono informazione ai dati ma con un apporto semantico ancora estremamente ridotto; i *metadati basati sul contenuto* rappresentano il primo livello di "interpretazione" dei dati, tentando di descrivere in maniera strutturata il contenuto di un documento o di fornire informazione con riferimento ad una precisa realtà; al vertice sono presenti le *ontologie*, punto di arrivo per la concettualizzazione di un dominio specifico ma anche punto di partenza per ontologie superiori e più complesse, indipendenti dal dominio.

In questo contesto, la standardizzazione dei contenuti dell'informazione è cruciale: infatti, la mancanza di un'*interpretazione* condivisa porta ad una limitatezza di comunicazione sia tra le persone, sia tra gli applicativi software, a scapito dell'interoperabilità, della condivisione e del riutilizzo.

39 http://www.loa-cnr.it/DOLCE.html
40 http://wonderweb.man.ac.uk/
41 Immagine rielaborata da V.Kashyap, C.Bussler, M. Moran - *"The Semantic Web"* - Springer 2008

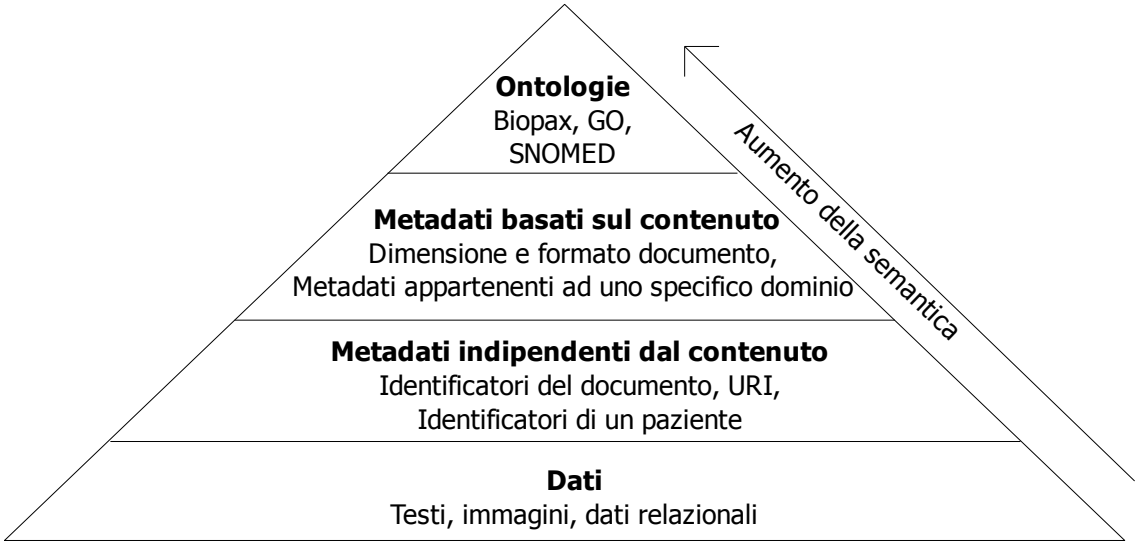


Figura 6 - Livelli semantici

1.12 - ONTOLOGIE NEL WEB

Le ontologie sono molto utili per l'organizzazione e la navigazione dei siti web: molti siti web espongono sulla sinistra della pagina i top level di una gerarchia concettuale di termini; cliccando su di essi, l'utente espande le sottocategorie e accede alle varie risorse disponibili.
Le ontologie sono molto utili anche per migliorare l'accuratezza delle ricerche web: i motori di ricerca possono cercare pagine che si riferiscono ad un preciso concetto presente in una ontologia invece che collezionare tutte le pagine nelle quali certe keywords sono presenti. In questo modo le differenze nella terminologia tra pagine web e query possono essere superate.

I motori di ricerca, in più, possono sfruttare la generalizzazione o la specializzazione dell'informazione. Se una query non restituisce alcun documento rilevante, il motore di ricerca potrebbe suggerire all'utente una query più generale. Oppure se i risultati sono troppi, il motore potrebbe suggerire una ricerca più raffinata.

Per descrivere le ontologie del web, sono stati sviluppati linguaggi specifici, tra cui:
RDF: data model per oggetti ("risorse") e relazioni tra loro. Fornisce una semantica semplice e i dati possono essere rappresentati in sintassi XML.
RDF Schema: è un linguaggio per la descrizione del vocabolario, specifica proprietà e classi di risorse RDF, con una semantica basata su generalizzazione gerarchica.
OWL: l'Ontology Web Language è un linguaggio per una descrizione "arricchita" del

vocabolario, in quanto specifica proprietà e classi come relazioni tra classi (p.e. “disgiunzione”), cardinalità (“esattamente uno”), uguaglianza, caratteristiche delle proprietà (p.e. “simmetria”) e classi numerate.

Resource Description Framework (RDF) e *Web Ontology Language (OWL)* sono stati scelti dal progetto “Ontologie archivistiche” promosso dal Ministero per i Beni e le Attività culturali – Direzione Generale per gli Archivi – progetto che si propone come *“sistema collaborativo di analisi e descrizione ontologica di sistemi archivistici nazionali e di una base concettuale condivisa*[42]*”*, ideato e sviluppato in linea con le proposte del W3C per il Web Semantico. L'obiettivo fondamentale è la definizione di un'ontologia di due sistemi archivistici nazionali[43] e di un'ontologia “esterna” di concetti archivistici ritenuti rappresentativi della tradizione descrittiva nazionale, proposta ed utilizzata come modello concettuale di riferimento rispetto al quale rapportarsi.

Nel mondo dei motori di ricerca la sfida è aperta: di seguito si fa riferimento ad interessanti esperimenti, forse poco noti, che adottano strategie diverse ma sempre ispirate a modelli ontologici.

42 http://www.archivi.beniculturali.it/servizioII/progetti/ontologie.html

43 *SIUSA – Sistema Informativo Unificato delle Sovrintendenze Archivistiche* e *Sistema Guida generale degli Archivi di Stato italiani*

hakia.com[44]

E' un motore di ricerca nato nel 2004 e si basa su tre tecnologie:

OntoSem: repository di relazioni concettuali, ossia un database dove le parole sono categorizzate specificando i vari significati che assumono

QDEX: tecnica che, basandosi su OntoSem, estrae tutte le possibili query relative al contenuto di un documento; tali query diventano la porta di accesso al documento originale. Questo processo riduce sensibilmente il data set che gli indicizzatori devono trattare mentre fanno query su dati on-the-fly.

SemanticRank è l'algoritmo che, analizzando più frasi, decide la posizione del documento nella visualizzazione dei risultati all'utente.

Quando l'utente sottopone una query, *hakia.com* restituisce un set di risultati selezionati in diverse aree: Web, News, Blogs, hakia Galleries, Credible Sources, Video, and Images.

Figura 7 - ricerca del termine *gate* con hakia.com

44 http://hakia.com/

Kosmix.com[45]

"We'll show you content that you may never have discovered otherwise". Così almeno promette questo motore di ricerca in cui gli utenti possono arricchire la categorizzazione dei concetti tramite un dashboard. Il motore focalizza le ricerche basandosi su topic più che su una particolare risposta o un URL. Così l'utente può cercare una specifica informazione o navigare tra i siti che già conosce e, allo stesso tempo, usare *Kosmix.com* per vedere cosa c'è nel web, oltre tali siti, inerenti allo allo stesso topic.

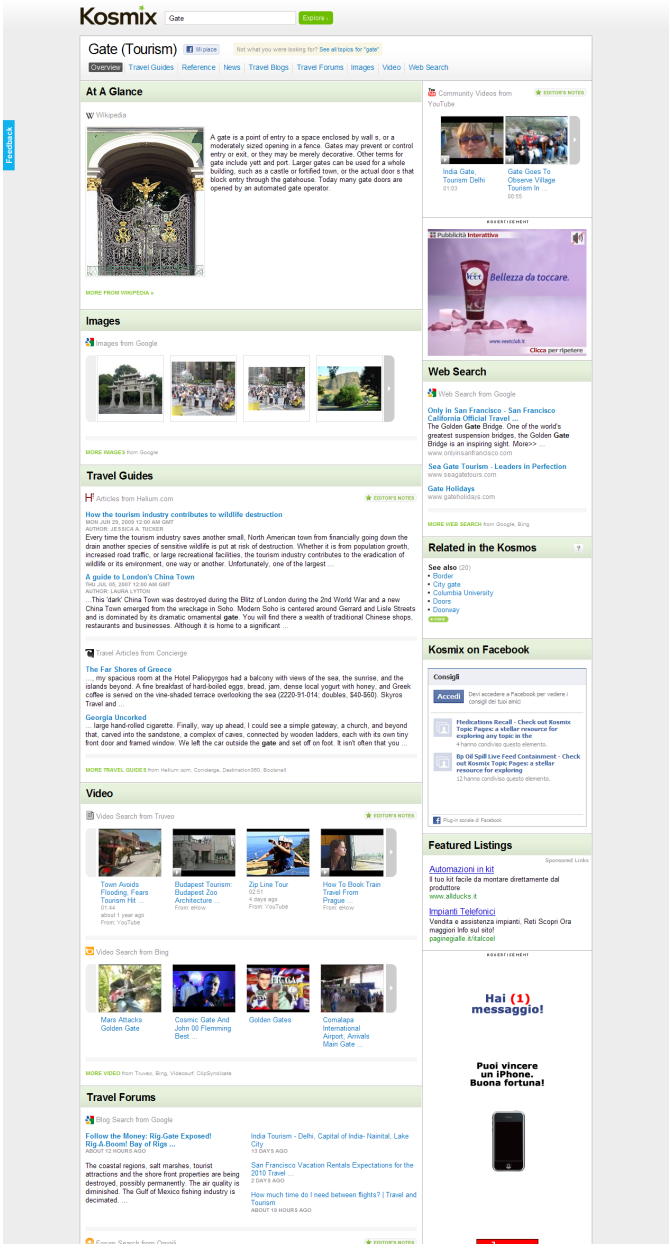


Figura 8 - ricerca del termine *gate* con kosmix.com

45 http://www.kosmix.com/

Powerset.com[46]

Questo motore è nato nel 2005 e dal 1 agosto 2008 è di proprietà della Microsoft. *Powerset.com* applica il Natural Language Processing per migliorare i risultati della ricerca, cercando di interpretare i significati racchiusi nel linguaggio umano.
L'utente può inserire keywords, frasi o semplici domande e il motore restituirà i risultati cercando di rispondere direttamente alla domanda e aggregando l'informazione presente in più articoli.

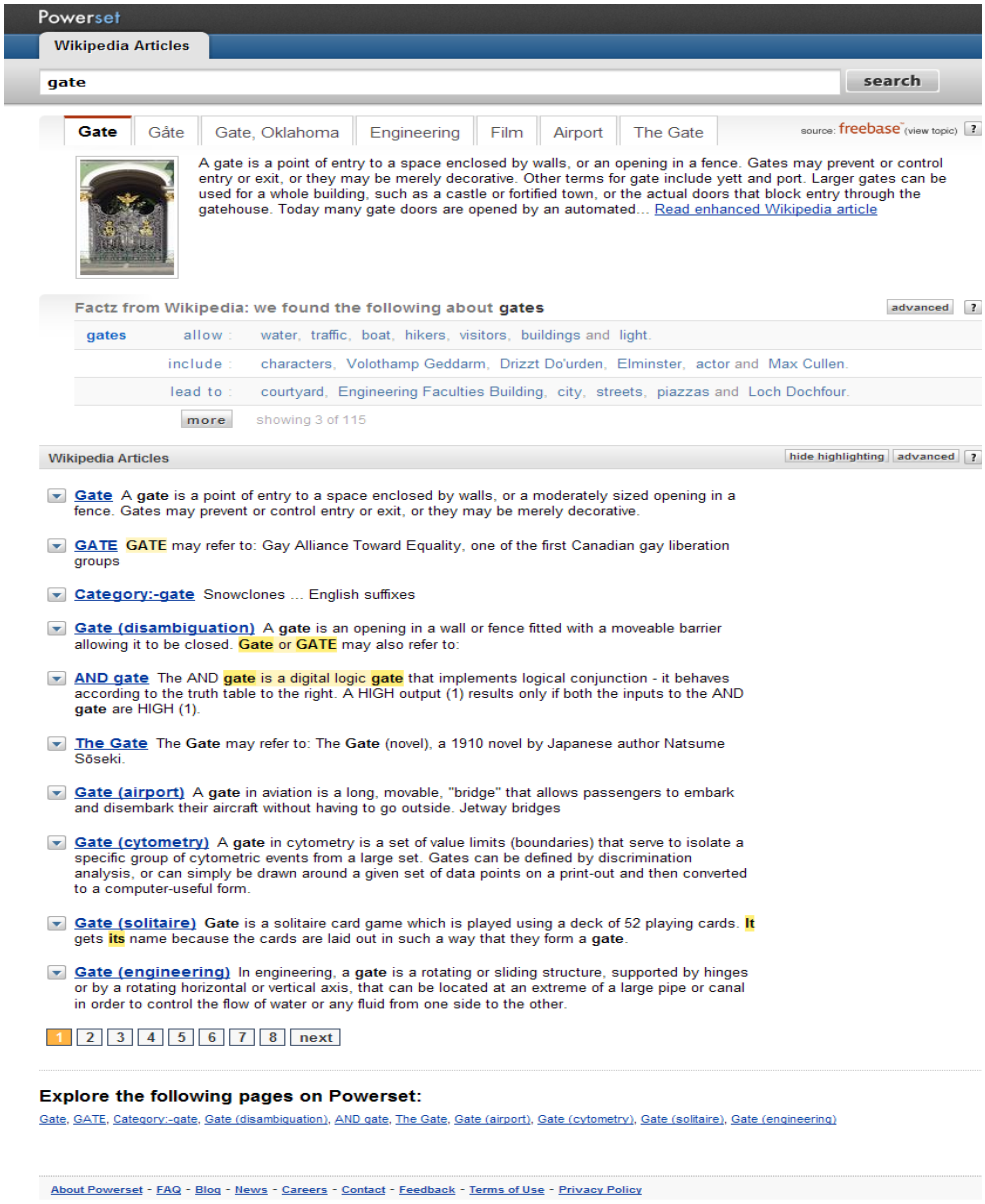


Figura 9 - ricerca del termine *gate* con Powerset.com

46 http://www.powerset.com/

1.13 - ONTOLOGIE IN CAMPO MEDICO

In campo medico le ontologie sviluppate sono numerose: esiste un sito[47] sul quale è possibile esplorare e visualizzare contemporaneamente più ontologie condivise inserendo il termine chiave, navigare all'interno di una specifica ontologia, oppure recuperare risorse inerenti a quel termine.

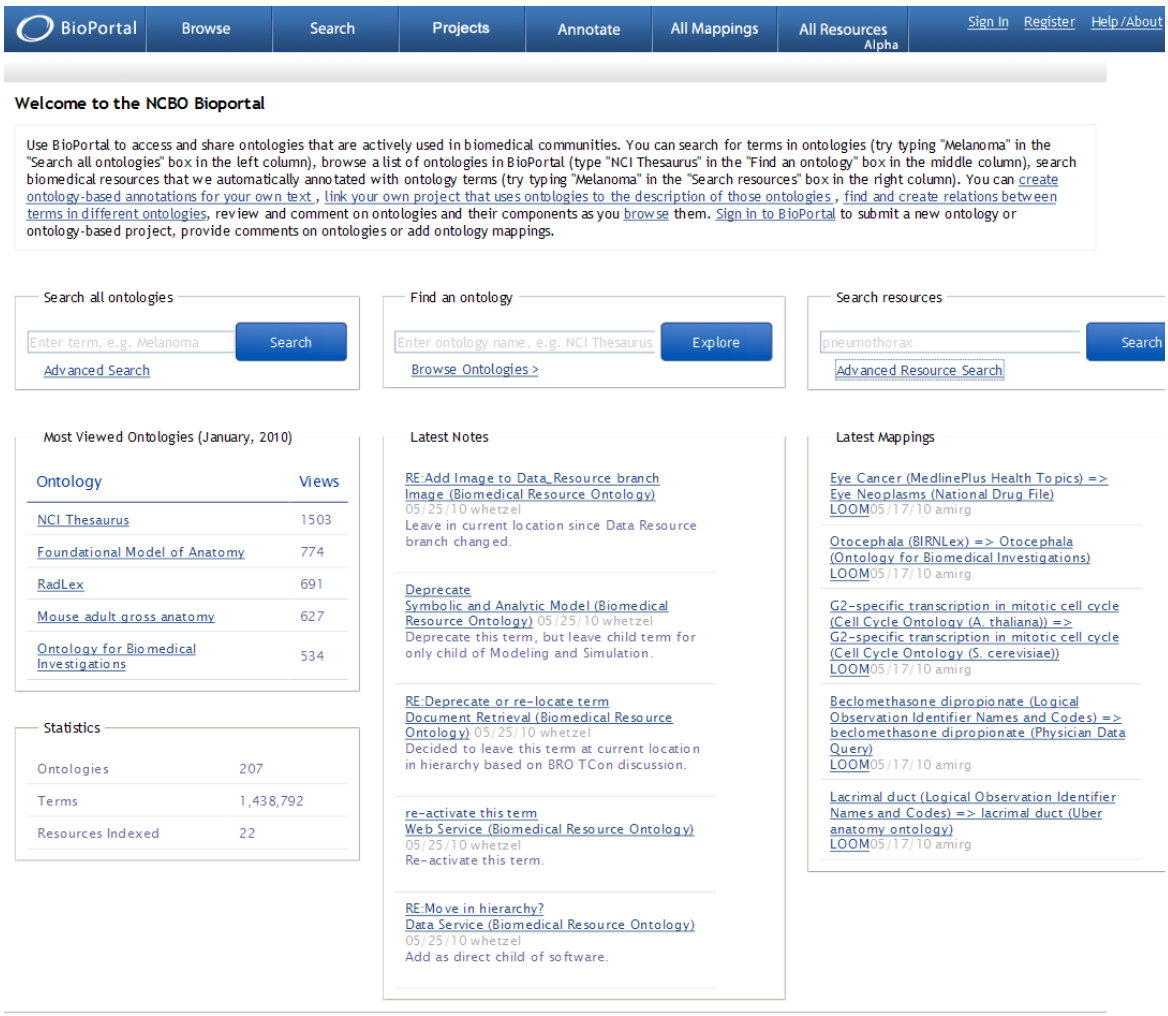


Figura 10 - Ontologie consultabili su NCBO Bioportal

Se per esempio l'utente inserisce la parola “pneumothorax” il sito restituisce un lungo elenco specificando in ogni riga l'ontologia in cui quel termine è presente e consentendone l'esplorazione. Selezionando l'ontologia di SNOMED CT con visualizzazione “Hierarchy to root”, ottieniamo il contesto in cui è inserito il termine cercato: guardando l'immagine riportata di seguito, notiamo i numerosi livelli di gerarchia che collegano l'entità *Pneumothorax* con la radice *Clinical finding* (evidenziati dagli archi *PAR - parent)* e le relazioni di sottoclasse (evidenziati da *subClassOf*)

47 http://bioportal.bioontology.org/ontologies

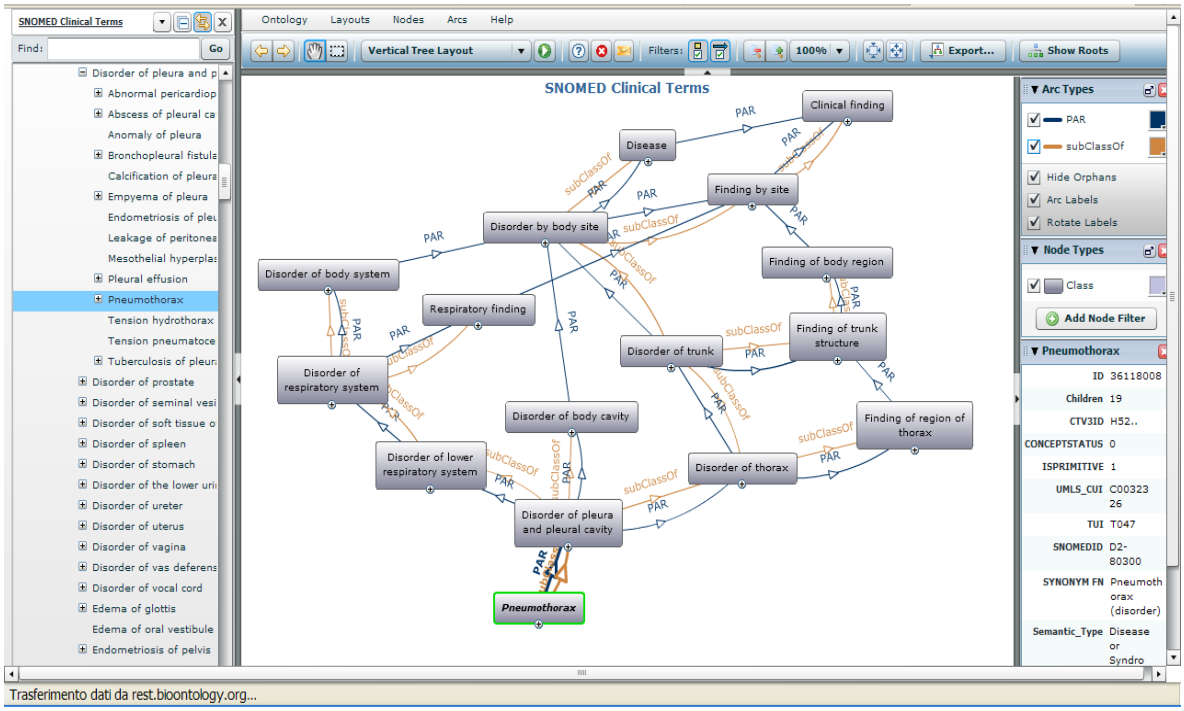


Figura 11- Hierarchy to Root

Se invece si seleziona l'opzione “Network Neighborhood”, si ottengono i livelli immediatamente superiore e inferiore collegati a “pneumothorax”: come mostrato nelle seguente figura, il livello di dettaglio, la completezza terminologica e allo stesso tempo la chiarezza di visualizzazione raggiunti sono particolarmente apprezzabili.

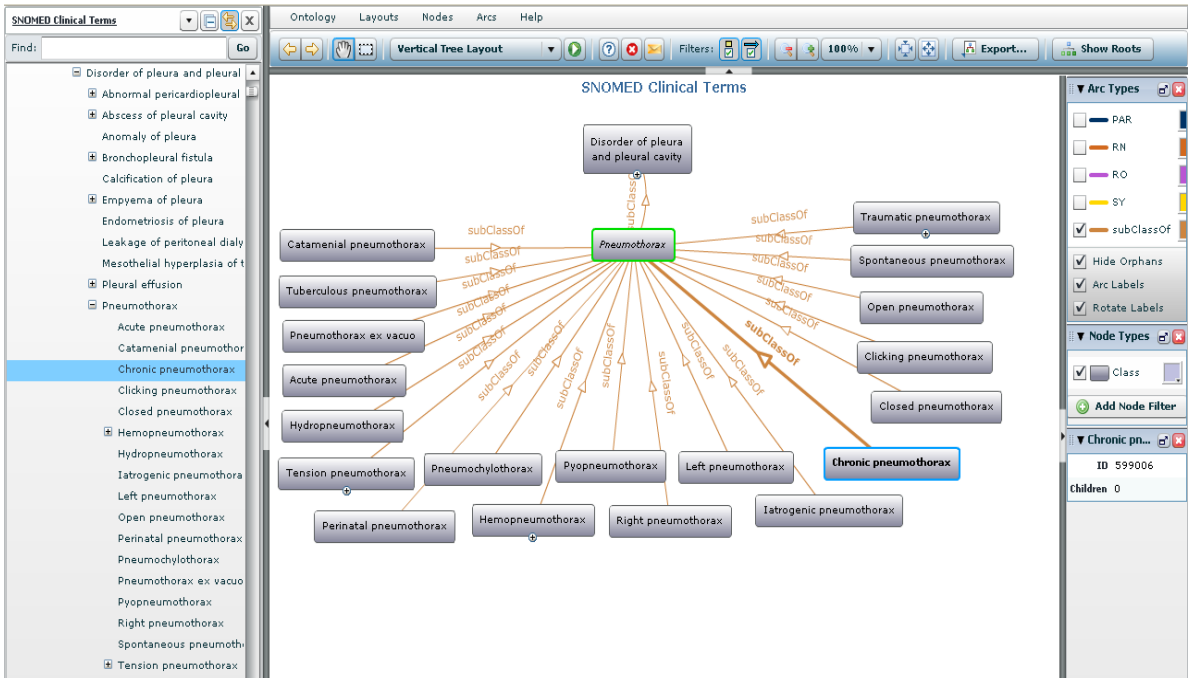


Figura 12 - Network Neighborhood

Capitolo 2
NATURAL LANGUAGE PROCESSING (NLP)

2.1 - DEFINIZIONE E SCOPO

Il Natural Language Processing (NLP) è un settore della computer science che si occupa delle interazioni tra computer e linguaggio umano naturale; esso studia i problemi connessi alla generazione automatica e alla comprensione del linguaggio umano, scritto o parlato. Il termine "Natural Language" è usato per distinguere il linguaggio umano usato dalle persone (italiano, inglese, spagnolo...) dal linguaggio dei computer (Java, C++, XML). La sua attenzione è dunque rivolta al trattamento automatico della lingua utilizzando l'analisi morfologica, sintattica e semantica come strumento di approssimazione del significato di un testo. Nel nostro caso specifico, l'elaborazione consiste in un processo di estrazione di informazione dai referti scritti in linguaggio naturale.

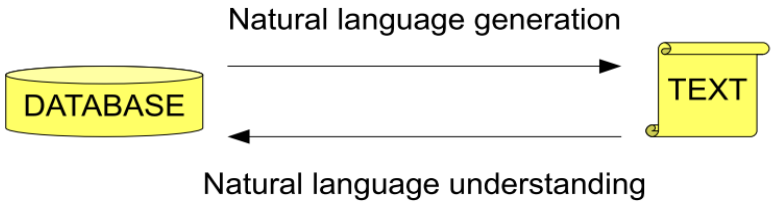
Lo studio del linguaggio naturale avviene per fasi eseguite in una precisa sequenza e caratterizzate da una valenza semantica crescente:

- *articolare e decodificare i suoni di una lingua*
  (identificazione di suoni e lettere)
- *conoscere le parole di una lingua, la loro struttura (plurale/singolare) e la loro organizzazione (sostantivo, verbo, aggettivo)*
  (l'analisi lessicale individua i lessemi che compongono il linguaggio e che trovano una definizione nel dizionario; l'analisi morfologica individua la struttura plurale/singolare, modo verbale, tempo verbale, e assegna ad ogni parola la propria classe morfologica intesa come aggettivo, sostantivo, verbo)
- *composizione di parole in costituenti complessi (part-of-speech)*
  (la sintassi individua le parti del discorso come soggetto, predicato, complemento, oppure gruppi di parole con significato unico come hot-dog, o ancora la parte nominale e verbale con derivazione dell'albero sintattico completo)
- *assegnazione di significati alle espressioni linguistiche semplici e complesse*
  (la semantica cerca di individuare il significato delle parole in funzione del contesto)
- *utilizzo delle frasi nei contesti, situazioni e modi appropriati agli scopi*

comunicativi (la pragmatica osserva come e per quali scopi la lingua viene utilizzata, distinguendo se si tratta di narrazione, dialogo, metafora, ect.)

I risultati ottenuti sono poi applicati alle due principali categorie di NLP:

- *natural language generation* che si occupa della conversione di informazione da database a linguaggio umano leggibile
- *natural language understanding* che trasforma il linguaggio umano in forme di rappresentazione facilmente manipolabili dai programmi.



Il NLP affronta numerose problematiche: *speech segmentation,* conversione di una traccia vocale in caratteri e in parole di senso compiuto; *text segmentation,* riconoscimento delle singole parole in testi scritti con ideogrammi anziché lettere (cinese, giapponese, thai....); *part-of-speech tagging,* individuazione degli elementi grammaticali di una frase come sostantivo, aggettivo, verbo, pronome; *word sense disambiguation*: deduzione dal contesto del significato di un termine normalmente utilizzato per indicare più concetti; *imperfect or irregular input*: riconoscimento e correzione di eventuali accenti regionali, errori di battitura, errori prodotti da strumenti di Optical Character Recognition.

NLP, dunque, indica qualsiasi applicazione che elabora il linguaggio umano. La linguistica computazionale descrive metodi per il NLP che possano "dotare" il computer di conoscenze linguistiche affinchè programmi e sistemi informatici siano in grado di assistere l'uomo in compiti linguistici. I principali task in cui fornisce supporto sono:

- strumenti di scrittura e lettura in lingua straniera
- suggerimento ortografico o correzione ortografica
- controllo della grammatica
- individuazione di sinonimi e suggerimento degli stessi all'utente consentendogli di raffinare il risultato di una query (detta anche *generazione di sinonimi)*
- information extraction
- assegnazione automatica di metadati
- automatic summarization, ossia la creazione automatica, a partire da un documento, di un riassunto che contenga i punti più importanti del testo originale.

- classificazione o categorizzazione dei documenti (si parla di *classificazione* quando le categorie sono due, di *categorizzazione* quando le categorie sono più di due)
- traduzione simultanea di linguaggio umano da una lingua all'altra (machine translation)
- question answering, ossia strumenti che, da una richiesta espressa in linguaggio umano, producono risultati tradotti in linguaggio umano.
- speech recognition, traduzione simultanea di un discorso in un testo scritto.

Nell'ambito del machine translation, per esempio, esistono sistemi come Rosetta[48] o Systran[49] basati su tecnologie diverse che consentono di tradurre grosse quantità di documenti in svariate lingue. Il primo sistema contiene un analizzatore semantico che, partendo dal parse tree di una frase scritta in una lingua, tenta di estrarne il significato per poi rigenerare l'albero sintattico secondo le regole di composizione della seconda lingua.
Systran, invece, pur basandosi sempre sul parse tree, non tenta di estrarre il significato della frase originale bensì di "tradurre" direttamente l'albero sintattico stesso secondo le regole grammaticali della seconda lingua, basandosi su un'analisi morfo-sintattica parziale (shallow parsing) e su "scorciatoie" per le espressioni idiomatiche o le parole composte.

Figura 13 – Rosetta Translation

Per quanto riguarda invece la generazione di sinonimi, un esempio evidente e' il servizio offerto dai motori di ricerca oppure l'AdWords[50] offerto da Google con il quale un'azienda sottopone la keyword di interesse e il sistema restituisce un lungo elenco di "sinonimi" (e la relativa frequenza di ricerca) in modo che l'utente possa

48 http://www.rosettatranslation.com/
49 http://www.systran.it/
50 www.google.it/adwords

scegliere la keyword che garantisce maggior visibilita' al proprio sito:

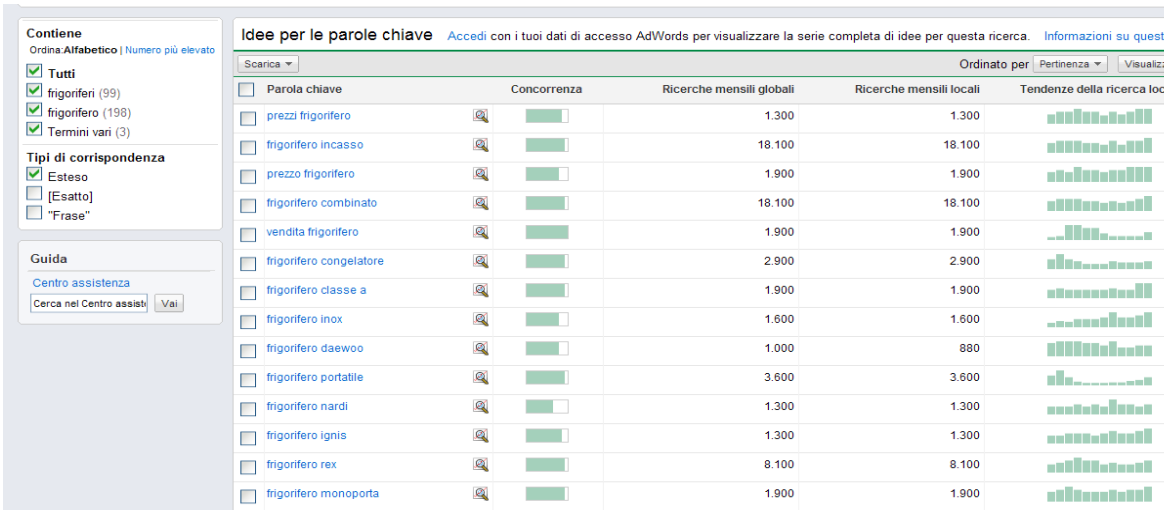


Figura 14 – generazione sinomini con AdWords

Le difficoltà di elaborazione in ambito linguistico si spiegano anche considerando le caratteristiche più evidenti del linguaggio naturale stesso: la *flessibilità*, perchè usa modi diversi per affermare lo stesso fatto; l'*ambiguità* in quanto la medesima affermazione può avere più di un significato; la *dinamicità* data dalla continua creazione di parole nuove.

Proprio a causa di queste particolarità, la comprensione del linguaggio naturale è spesso considerata un problema IA-completo, ossia un problema la cui risoluzione è paragonata equivalente alla creazione di una intelligenza artificiale "realistica". Infatti la comprensione dei testi necessita la comprensione dei concetti ad essi associati, e quindi una conoscenza estesa della realtà e una grande capacità di manipolarla. Per le persone, la comprensione del linguaggio è frutto di un processo mentale che non può essere riprodotto in una macchina; inoltre, il linguaggio è una forma di comunicazione e interazione tra persone che "riflette la superficie del significato"[51] e consente alle persone di capirsi mentre un elaboratore, per quanto sofisticato possa essere il software di cui è dotato, si basa comunque su procedimenti determinati a priori.

Per questa ragione, la definizione di "comprensione" è uno dei maggiori problemi dell'elaborazione del linguaggio naturale.

51 Marti Hearst – *Applied Natural Language Processing* – 2004 - http://www2.sims.berkeley.edu/courses/is290-2/

2.2 - FASI DEL NLP

Come anticipato poco fa, l'elaborazione del linguaggio naturale comporta una successione di fasi che tentano di superare le ambiguità del linguaggio umano. Si tratta di un processo particolarmente delicato a causa delle complesse caratteristiche del linguaggio stesso. Se per esempio consideriamo la regola della grammatica inglese che posiziona l'aggettivo prima del nome, possiamo avere le seguenti combinazioni tranne l'ultima, sebbene rispetti l'ordine richiesto; oppure se si volesse tradurre in inglese il verbo “girare” si avrebbe una serie di termini legati al contesto:

✔	A big black dog	*girare per negozi*	to go round
✔	A big black scary dog	*girare una cambiale*	to endorse
✔	A big scary dog	*girare un film*	to shoot
✔	A scary big dog	*girare a destra*	to turn
✗	A black big dog	*girare la domanda*	to refer

Proprio per ridurre il piu' possibile gli errori, il processo di elaborazione viene suddiviso in numerose fasi[52].:

tokenization	scomposizione di un'espressione linguistica in *token* (parole, spazi, punteggiatura, frasi)
analisi morfologica	forme di flessione e regole di composizione (part-of-speech)
analisi lessicale	individuazione dei vocaboli
analisi sintattica	creazione di una struttura sintattica ad albero (*parse tree*)
named entity recognition	identificazione di entità con significato semantico
analisi semantica	assegnazione di un significato alla struttura sintattica e, di conseguenza, all'espressione linguistica
co-reference	individuazione di pronomi e parafrasi
analisi pragmatica	distinzione del testo in dialogo, discorso, metafora

52 *Lessico:* insieme dei vocaboli che costituiscono una lingua. *Morfologia*: scienza che studia le regole di composizione delle parole nella loro formazione e flessione. *Sintassi:* branca della linguistica che studia le regole in base alle quali le parole si posizionano all'interno di una frase. *Semantica:* branca della linguistica che studia il significato delle parole (semantica lessicale), degli insiemi di parole (semantica frasale) e dei testi. *Pragmatica:* studio del linguaggio in rapporto all'uso che ne fa il parlante. *Grammatica:* disciplina che studia le regole fonetiche, ortografiche, morfologiche, lessicali e sintattiche di una lingua.

Poiché queste fasi caratterizzano una tipica elaborazione di NL e sono applicate in gran parte nell'esperimento condotto sui referti, si forniscono alcuni dettagli sulla loro funzionalità e sul tipo di ambiguità che si prestano a risolvere.

2.2.1 – Tokenization

L'elaborazione di un testo inizia con la sua scomposizione in *token* corrispondenti a spazi, parole, punteggiatura, frasi. Il task non e' particolarmente complesso rispetto agli altri ma presenta comunque alcune problematiche: per esempio, se si considera il punto come fine *sentence* si rischia di sbagliare frequentemente in quanto il punto potrebbe riferirsi ad una abbreviazione o ad una data o ad un link. Con opportune regole che gestiscono la struttura delle abbreviazioni si possono ottenere risultati ancora piu' accurati. Pur essendo di per se' un task indipendente dalla lingua, non sempre si ottengono i risultati sperati. Qualche esempio:

Citazioni	`>>> tagger.tokenize('This is a "simple" example.')`
	`['This', 'is', 'a', '"', 'simple', '"', 'example', '.']`
	`>>> tagger.tokenize('"This is a simple example."')`
	`['"', 'This', 'is', 'a', 'simple', 'example', '."']`

Parole che contengono caratteri speciali	`>>> tagger.tokenize('Parts-Of-Speech')`
	`['Parts-Of-Speech']`
	`>>> tagger.tokenize('amazon.com')`
	`['amazon.com']`

Punteggiatura	`>>> tagger.tokenize('Quick, go to amazon.com.')`
	`['Quick', ',', 'go', 'to', 'amazon.com', '.']`
	`>>> tagger.tokenize('Live free; or die?')`
	`['Live', 'free', ';', 'or', 'die', '?']`

Tolleranza per punteggiatura non corretta	`>>> tagger.tokenize('Hi , I am here.')` `['Hi', ',', 'I', 'am', 'here', '.']`
Forme grammaticali particolari	`>>> tagger.tokenize("my parents' car")` `['my', 'parents', "'", 'car']`

Numeri	`>>> tagger.tokenize("12.4")`
	`['12.4']`
	`>>> tagger.tokenize("-12.4")`
	`['-12.4']`
	`>>> tagger.tokenize("$12.40")`
	`['$12.40']`

Date	`>>> tagger.tokenize("10/3/2009")`
	`['10/3/2009']`
	`>>> tagger.tokenize("3.10.2009")`
	`['3.10.2009']`

2.2.2 - Analisi morfologica e lessicale (part-of-speech)

La morfologia fornisce informazioni sulla forma di ciascuna parola e sul suo ruolo all'interno di una frase. Il lessico (o vocabolario) è il complesso delle parole e dei modi di dire (locuzioni) di una lingua. L'analisi morfologica e lessicale prevedono la consultazione di apposite liste di lemmi e loro derivazioni (liste che possono essere eventualmente integrate con termini specifici relativi al dominio che si sta studiando), la risoluzione delle forme di flessione (come la coniugazione per i verbi e la declinazione per i nomi) e la classificazione delle parole basata su determinate categorie (come il nome, pronome, il verbo, l'aggettivo). Per esempio:

Analisi morfologica				**Analisi lessicale**
libro	sostantivo	maschile	singolare	*libro*
simpatiche	aggettivo	femminile	plurale	*simpatico*
urlano	verbo	terza persona	plurale	*urlare*

L'analizzatore morfologico tenta di individuare la radice e gli affissi (prefissi e suffissi) di ogni parola ottenendo morfemi, ossia le unità linguistiche più piccole con significato semantico, e generando le seguenti scomposizioni:

portato	port-ato	(participio passato)
indipendentemente	in-dipend-ente-mente	(prefisso-radice-suffissi)
pittrici	pitt-ric-i	(radice- suffisso-plurale)

Analizzatore lessicale e morfologico possono essere implementati separatamente, ma spesso costituiscono un unico task. Il risultato di questo processo è più spesso noto come *Part of Speech Tagging (PoS)* e il tagset più diffuso è Penn che distingue 36 parti di speech (48 se si include la punteggiatura).

Qualche esempio concreto aiuta ad individuare e far comprendere le più frequenti tipologie di problemi legati a questa fase e le funzionalità offerte dai tagger PoS[53]:

>>> tagger('These are simple examples.')	**PennTreebank tagset**
[['These', 'DT', 'These'],	DT determiner
['are', 'VBP', 'are'],	VBP verb, pres.tense, not 3th person singular
['simple', 'JJ', 'simple'],	JJ adjective
['examples', 'NNS', 'example'],	NNS noun, plural common
['.', '.', '.']]	

53 Marcus M., Santorini B, Marcinkiewicz M.A. *The Penn Treebank* – Computational Linguistic 1993

Regole	

`>>> tagger('Ikea')`
`[['Ikea', 'NN', 'Ikea']]`
`>>> tagger('Ikeas')`
`[['Ikeas', 'NNS', 'Ikea']]`
	Correggi nomi di default

`>>> tagger('. Police')`
`[['.', '.', '.'], ['police', 'NN', 'police']]`
`>>> tagger('Police')`
`[['police', 'NN', 'police']]`
`>>> tagger('. Stephan')`
`[['.', '.', '.'], ['Stephan', 'NNP', 'Stephan']]`
	Distingui nomi propri
	da nomi ad inizio frase

`>>> tagger('The fox can jump')`
`[['The', 'DT', 'The'],`
`['fox', 'NN', 'fox'],`
`['can', 'MD', 'can'],`
`['jump', 'VB', 'jump']]`
 	
`>>> tagger('The fox can really jump')`
`[['The', 'DT', 'The'],`
`['fox', 'NN', 'fox'],`
`['can', 'MD', 'can'],`
`['really', 'RB', 'really'],`
`['jump', 'VB', 'jump']]`
	Determina il verbo modale e il relativo verbo successivo

`>>> tagger('examples')`
`[['examples', 'NNS', 'example']]`
`>>> tagger('stresses')`
`[['stresses', 'NNS', 'stress']]`
`>>> tagger('cherries')`
`[['cherries', 'NNS', 'cherry']]`
	Normalizza forme plurali

`>>> tagger('men')`
`[['men', 'NNS', 'men']]`
`>>> tagger('feet')`
`[['feet', 'NNS', 'feet']]`
	In alcuni casi, non fare nulla

Già in questa fase emergono le prime difficoltà legate all'ambiguità, in questo caso lessicale: per certi vocaboli il task non sa attribuire correttamente la giusta categoria morfologica. Consideriamo per esempio *"La vecchia legge la regola"* : si possono avere due possibili varianti di classificazione morfologica associata ai *token.*

La	articolo		articolo
vecchia	sostantivo		aggettivo
legge	verbo	*oppure*	sostantivo
la	articolo		pronome
regola	sostantivo		verbo

In questo esempio si nota come solo l'articolo rimanga inviariato mentre gli altri token siano associati a categorie totalmente diverse.

2.2.3 - Analisi sintattica e generazione di parse tree

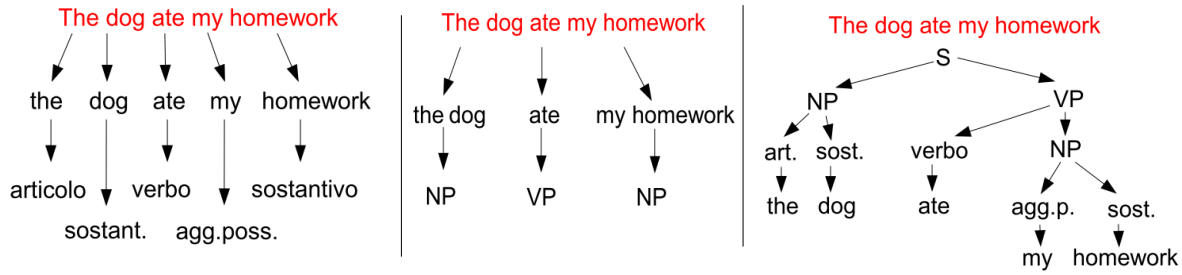
La sintassi contiene la conoscenza necessaria per spiegare come sequenze di parole hanno senso compiuto mentre altre sequenze, pur contenendo esattamente le stesse parole ma in ordine non corretto, siano prive di senso. Per esempio, la frase "il sole tramonta a ovest" ha senso compiuto; "il tramonta ovest a sole" non ha alcun senso. L'analisi sintattica tenta, dunque, di individuare le varie parti che danno un significato alla frase (soggetto, predicato, complementi) e di costruire la giusta posizione delle parole.

Generalmente la struttura sintattica è rappresentata come *Parse Tree*. Si consideri la frase *"The dog ate my homework"* e si supponga di aver risolto l'ambiguità lessicale con il giusto tag di PoS (anche qui *dog* potrebbe indicare un sostantivo o il verbo pedinare); l'analizzatore sintattico rielabora il documento e crea la struttura ad albero sulla base di alcune regole:

The	det		PoS
dog	noun		PoS
	det + noun	NP (noun frase)	applicazione regola
ate	verb		PoS
my	poss. adj		PoS
homework	noun		PoS
	verb + NP	VP (verbo phrase)	applicazione regola
	NP + VP	S (sentence)	applicazione regola

Il *parsing*, dunque, tenta di capire le relazioni sintattiche tra gli elementi del testo. Esso può essere implementato a vari livelli di complessità: il *full parsing (o parsing completo)* è molto dettagliato ma comporta tempi eccessivi di elaborazione e percentuali di errore non trascurabili. Lo *shallow parsing (o parsing superficiale),* invece, oltre ad essere più veloce, ottiene buoni risultati, anche perchè l'informazione essenziale di solito è presente in poche porzioni dell'albero.

I parsing per la frase di esempio potrebbero dunque essere i seguenti, dove S = simple declarative clause, NP = Noun Phrase e VP = Verb Phrase:



Part-of-speech **Shallow parsing** (parsing superficiale) **Full parsing** (parsing completo)

Esistono numerose versioni di questa funzionalità, tuttavia sono sempre sviluppate per la lingua inglese e, in rari casi, ad altre lingue. In particolare per l'italiano esiste una versione beta, il *Tanl Italian Parser*[54], per ora fornito solo come servizio web: inserendo una frase, si ottiene l'immediato parse tree.

Figura 15 - Italian Parser

Anche in questa fase l'ambiguità, stavolta di tipo sintattico (detta anche *ambiguità locale* perchè riferita ad una parte della frase), comporta non poche difficoltà. Si consideri questo esempio:

"Egli vide una donna con gli occhiali"

Non è ben chiaro se *l'uomo ha gli occhiali e vede una donna* oppure se *la donna vista dall'uomo indossa gli occhiali*. A volte il contesto aiuta a risolvere tali

54 http://paleo.di.unipi.it/parse

ambiguità, ma in presenza di frasi articolate si genera una proliferazione di alberi sintattici con evidente aumento della complessità.

2.2.4 - Named Entities Recognition

La fase successiva vede poi l'individuazione, tramite regole o approcci statistici di machine learning, di parole o gruppi di parole che specificano entità appartenti a diverse categorie. E' spesso suddivisa in due step: segmentazione e classificazione. Con la segmentazione si individuano i confini (start e end offset) delle entità, con la classificazione si assegna loro un'annotazione con validità semantica. Un esempio è quello riferito all'individuazione dei nomi propri come “Mario Rossi”: gestirli con apposite liste è troppo dispendioso a causa dei continui aggiornamenti che richiederebbero, senza contare che appartengono a troppi domini diversi. Le entità possono appartenere ad alcune categorie standard, generalmente riconosciute valide in ogni dominio come Luoghi o Organizzazioni, a categorie comuni come Date, Misure, Indirizzi oppure a specifici domini (nomi farmacologici, banche..).

Per riconoscere le entità si utilizzano generalmente tre approcci:

- *Lookup List o Gazetteers*: veri e propri elenchi suddivisi per categoria.
- *Rule-based (pattern-matching)*: le entity sono individuate utilizzando espressioni regolari che analizzano il contesto e/o alcune caratteristiche delle entity stesse come l'ortografia, la categoria di POS o altre features ad esse collegate. Anche con questa modalità non mancano i casi di ambiguità: General Motors potrebbe indicare un'entità appartenente alla categoria Organizzazione oppure essere considerata un titolo militare.
- *Machine-trainable*: in questo modalità, la tecnologia che si è affermata maggiormente grazie ai risultati raggiunti è quella di Hidden Markov Model (HMM), un modello probabilistico che tratta sequenze di parole. Introdotto da Church e DeRose[55] e importato dalla comunità di speech recognition, il modello parte da corpus annotato per ricavare le probabilità di transizione da uno stato all'altro (macchina a stati finiti). Tale processo probabilistico genera sequenze di stati e di simboli in output che rappresentano una frase. In pratica, definisce le probabilità inerenti a sequenze di parole.

L'uso dei *gazetteers* è certamente il sistema più semplice ma non sempre è applicabile perchè richiederebbe elenchi troppo lunghi, come evidenziato in questo esempio[56]:

55 Church K. *A stochastic parts program an Noun Phrase Parser for Unrestricted Texts* – Proc. 2nd Conference of ANLP (1988)

56 Marti Hearst – *Applied Natural Language Processing* – 2006 - http://www2.sims.berkeley.edu/courses/is256/

establish	creare
establishment	creazione
disestablishment	distruzione
antidisestablishment	antidistruzione
antidisestablishmentarian	antidistruzionista
antidisestablishmentarianism	antidistruzionismo - filosofia politica che si oppone alla separazione chiesa-stato

Il riconoscimento delle entità basato su *regole* ha sicuramente il vantaggio della semplicità in fase di sviluppo, ma richiede tempi lunghi per verificare i risultati sui corpora e raffinare in più step la regola stessa. E' veloce in fase di esecuzione e normalmente si applica su testi standard ottenendo buone prestazioni.
Il machine-trainable, infine, ha il vantaggio di essere più generale e applicabile senza variazioni a più domini ma richiede moltissimi dati per calcolare le probabilità e finora ottiene risultati inferiori rispetto alle altre due metodologie.

2.2.5 - Analisi semantica

L'analisi semantica si propone di estrarre il significato di una intera frase partendo dal significato di ciascun termine che la compone e dalle relazioni esistenti tra gli stessi. Infatti il significato di una frase non è dato solo dalle parole ma anche dalla conoscenza delle regole che decidono il loro significato in base alla combinazione, all'ordine in cui appaiono, ai legami che le legano ad altri termini interni o esterni alla frase.

Un esempio di regola semantica può essere quella che presuppone la presenza del complemento oggetto quando nella frase è utilizzato il verbo *dare*. Oppure può attribuire una annotazione semantica che inserisce il termine in un preciso contesto: *cane* → *animale*. Questa operazione consente di risolvere alcuni casi di ambiguità globale, ossia riferita all'intera frase, comunque presente anche quando il PoS tagging restituisce la corretta categoria morfologica di un vocabolo e il parser costruisce la giusta struttura della frase. Per esempio:

Maria ha un forte interesse per i computer.
Maria ha un alto tasso di interesse sul suo conto corrente.

In questo esempio *interesse* è sempre sostantivo ma il significato che trasmette nelle due frasi è molto differente. Inoltre *forte interesse* e *alto tasso di interesse* sono entrambi complemento oggetto ma hanno significati totalmente diversi nelle

due frasi. Rimane invece difficile distinguere il significato di frasi come queste:

Qual è il senso di "vita"? da *Qual è il senso della vita?*

oppure risolvere ambiguità relative alla quantificazione:

Roberto regalò agli amici una birra

in cui non è chiaro se Roberto ha distribuito una birra a ciascun amico oppure se quella birra era da spartire. Ma forse l'esempio più famoso è quello composto nel 1957 da Noam Chomsky per dimostrare l'inadeguatezza degli algoritmi di allora nell'elaborazione del linguaggio naturale:

"Colorless green ideas sleep furiously"

liberamente tradotta in "idee verdi senza colore dormono furiosamente": sebbene ciascun termine abbia un suo significato e sebbene la struttura grammaticale del periodo sia corretta, la frase nel suo insieme non ha alcun significato.

2.2.6 - CoReference

Questo task, particolarmente complesso, tenta di risolvere pronomi e parafrasi, identificando nei testi relazioni di identità tra entità individuate da Named Entity o referenze anaforiche ad esse attinenti. In una prima fase identifica le parti nominali del testo usando NE, numero, genere e ruolo sintattico. Successivamente effettua un controllo di consistenza a ritroso e determina tutti i possibili precedenti fino ad una distanza predefinita. Dopodichè elimina i precedenti che non sono correlati e infine ordina per vicinanza gli antecedenti tentando di distinguere i pronomi intra-frase, intra-paragrafo, o intra-documento.

Esiste poi un task ulteriore che risolve le ellissi, ossia frasi prive di alcuni elementi sintattici che devono essere dedotti dal contesto. Per esempio:

Docenti e studenti finirono gli esami e se ne andarono. Anche Mario.

L'ellisse in questo caso è rappresentata dalla seconda frase: essa non ha una costruzione regolare e acquista senso solo se la si considera legata alla precedente.

2.2.7 - Analisi pragmatica

La pragmatica tende a fare distinzione tra *significato dell'enunciato* e *intenzione del parlante*. Il significato dell'enunciato è il suo significato letterale, mentre l'intenzione del parlante è il concetto che il parlante tenta di trasmettere.

Consideriamo questo esempio:

Io non ho mai detto che lei ha rubato i miei soldi.
*Lo dice qualcun altro, ma non **io***
Io **non** ho **mai** detto che lei ha rubato i miei soldi.
*Io semplicemente non l'ho **mai** detto.*
Io non ho mai **detto** che lei ha rubato i miei soldi.
*Posso averlo lasciato intendere ma non l'ho mai **detto** esplicitamente*
Io non ho mai detto che **lei** ha rubato i miei soldi.
*Ho detto che qualcuno ruba, ma non ho detto che era **lei***
Io non ho mai detto che lei ha **rubato** i miei soldi.
*Ho detto che probabilmente li ha presi **in prestito***
Io non ho mai detto che lei ha rubato i **miei** soldi.
*Ho detto che ha rubato i soldi a **qualcun altro***
Io non ho mai detto che lei ha rubato i miei **soldi**.
*Ho detto che ha rubato **qualcosa** ma non i miei soldi.*

Pur considerando sempre la stessa frase, abbiamo posto l'enfasi su parole diverse mettendo in luce le possibili intenzioni dell'interlocutore. Si tratta di un task particolarmente difficile che comunque non trova la sua ideale applicazione nei referti, soprattutto per la document classification che si vuole ottenere.

Quello che conta, alla luce delle problematiche emerse e degli esempi riportati, è sottolineare quanto sia importante risolvere l'ambiguità, non solo per ottenere precisione nei risultati ma anche per risparmiare sulla complessità dell'intera analisi, dato che più interpretazioni differenti comportano una ripetizione esponenziale dei task.

2.3 - MACHINE LEARNING

Il *Machine Learning* rappresenta una delle aree fondamentali dell' Intelligenza Artificiale e si occupa della realizzazione di sistemi che si basano su osservazioni o esempi come dati per la sintesi di nuova conoscenza (classificazioni, generalizzazioni, riformulazioni).
Il *Machine Learning (ML)* o *Apprendimento automatico* è una disciplina scientifica

che progetta e sviluppa algoritmi che consentono agli elaboratori di evolvere il proprio comportamento basandosi su dati empirici. Il principale obiettivo di ricerca in ambito di Machine Learning è "imparare" a riconoscere automaticamente pattern complessi ed effettuare scelte intelligenti basandosi su dati già analizzati.
La necessità di ricorrere al ML nasce dal fatto che prevedere a priori l'intero set di possibili comportamenti in base all'input, costruendo per esempio manualmente un set di regole, è troppo complesso da descrivere in un linguaggio di programmazione.
Parallelamente, la difficoltà di tale metologia risiede nell'incertezza con cui si individua una corrispondenza tra input e output: essa si basa su un meccanismo parametrico per la generazione dei dati, di cui però non si conoscono a priori valori esatti dei parametri.

Caratteristica del ML è l'*induzione*, ossia l'estrazione di leggi generali a partire da un insieme di dati osservati. Essa si contrappone alla *deduzione* in cui, a partire da leggi generali, si prevede il valore di un insieme di variabili.
L'induzione parte dall'osservazione per misurare un insieme di variabili e per poi effettuare previsioni su ulteriori dati. Questo processo complessivo nel quale, a partire da un insieme di osservazioni, si vuole effettuare previsioni su nuovi dati prende il nome di *inferenza*.

Le situazioni di difficile soluzione mediante algoritmi tradizionali sono numerose e dovute tipicamente alla presenza di:

- *Difficoltà di formalizzazione*: per esempio ognuno di noi sa riconoscere se una certa immagine raffiguri il volto di un amico ma probabilmente nessuno sa descrivere una sequenza di passi computazionali che, eseguita sui pixel, consenta di rispondere alla domanda.
- *Elevato numero di variabili in gioco*: quando si considera ad esempio l'elaborazione di documenti in linguaggio naturale, la specifica di tutti i parametri che si pensa possano essere coinvolti può essere particolarmente complessa. Inoltre, la stessa formalizzazione applicata in un medesimo contesto ma su corpora differenti potrebbe rivelarsi inadeguata e richiedere una nuova rielaborazione.
- *Mancanza di teoria:* si immagini di dover prevedere con esattezza l'andamento dei mercati finanziari in assenza di leggi matematiche specifiche.
- *Necessità di personalizzazione:* la distinzione tra documenti "interessanti" e "non interessanti" dipende significativamente dalla percezione del singolo utente.

Gli algoritmi di apprendimento automatico sono tradizionalmente divisi in tre

principali tipologie[57]:

- *Apprendimento supervisionato*: quando l'utente fornisce esempi (e controesempi) di quello che si deve apprendere. E' il problema più studiato nel machine learning. Esso si pone l'obiettivo di prevedere, dato un elemento di cui si conoscono un insieme di parametri (features), il valore di un diverso parametro di output relativo all'elemento stesso.
- *Apprendimento non supervisionato*: parte da osservazioni non preclassificate
- *Apprendimento con rinforzo*: tecnica di programmazione che si basa sul presupposto che l'algoritmo possa ricevere stimoli dall'esterno a seconda delle scelte fatte.

Dato che il ML è stato utilizzato per l'esperimento sui referti e ha richiesto l'esplicitazione di una serie di parametri, si forniscono alcuni elementi per la comprensione della logica di funzionamento di questi algoritmi, senza la pretesa di approfondire l'argomento.
Il problema è definito a partire da un universo di elementi: ciascun elemento **x** è descritto dai valori assunti da un insieme di features considerate come input del problema. Ad ogni **x** è associato un valore *y* di output (o target). A partire dalla conoscenza di un insieme **T** di elementi (denominato training set) in cui ogni elemento è descritto da una coppia (x_i, y_i), con x_i = vettore dei valori delle *d* features $x_i^{(1)}$, $x_i^{(2)}$, ….$x_i^{(d)}$ e y_i = valore di output, si vuole derivare un modello delle relazioni sconosciute tra features e valori di output, che, dato un nuovo elemento **x**, consenta di predire il corrispondente valore di output *y*.

Un esempio di problema che può essere trattato con ML potrebbe riguardare la previsione in merito alla presenza o meno di una malattia sulla base dei risultati di un insieme di analisi cliniche, oppure la stima di una quotazione sulla borsa valutata sull'andamento dei giorni precedenti, o ancora indovinare il possibile gradimento di un film da parte di uno spettatore a partire dalle preferenze rispetto ad altri film già visti, etc.

I valori assunti dalle singoli features possono essere di varia natura:
- *Quantitativi*: quando forniscono la misura di una grandezza
- *Qualitativi*: se specificano una classe di appartenenza
- *Qualitativi ordinati*: quando specificano l'appartenenza ad un intervallo

Allo stesso modo, i valori di output possono essere:
- *Quantitativi*: in questo caso il valore restituito è la predizione di una misura, e si parla di regressione (valori continui).
- *Qualitativi*: in questo caso il valore restituito è l'assegnazione ad una classe

57 http://it.wikipedia.org/wiki/

(categoria), e si parla di classificazione o pattern recognition. In particolare, se il numero di possibili classi è due, si parla di classificazione binaria, altrimenti di classificazione multi-classe.
• *Qualitativi ordinati*: in questo caso si parla di regressione ordinale.

2.4 - CLUSTERING E CLASSIFICAZIONE

La classificazione, come accennato, si ha quando i valori di output *y* che si vogliono predire non sono continui ma possono assumere soltanto valori appartenenti ad un insieme discreto, in cui ogni valore denota una classe. Nel caso più diffuso, le classi sono disgiunte, e quindi ogni input va assegnato ad una e una sola classe: lo spazio di input è quindi diviso in *regioni di decisione* delimitate da *confini*.

Il machine learning ha un ruolo importante nei problemi di classificazione che può avvenire con diversi livelli di granularità: classificazione di interi documenti (text classification), identificazione e classificazione di piccole unità presenti nei documenti (IE), identificazione e classificazione di nomi propri, luoghi, organizzazioni, particolari entità relative ad uno specifico contesto (sottoinsieme di IE).

Esistono due modi per assegnare la classe ad un elemento:
Classificazione: si conoscono a priori le classi in cui assegnare gli elementi e si hanno a disposizioni un set di esempi per ciascuna classe.
Clustering: si assume che esista una naturale suddivisione degli elementi in categorie ma non si sa a priori quante e quali siano le categorie. In questa ipotesi non ci sono naturalmente esempi espliciti per ciascuna categoria.

Proprio per la particolare tipologia dei due metodi, la classificazione si basa su dati annotati e avviene in modalità supervised, ossia si allena il sistema su dati con classificazione nota. Il clustering invece avviene in modalità unsupervised e si applica a corpora non annotati. Quando le dimensioni dei corpora sono particolarmente elevate, si tenta di applicare una modalità semi-supervised che rappresenta una generalizzazione delle due e che consiste nell'annotazione parziale del training set, lasciando alcune istanze non annotate.

Se per esempio si chiede di determinare l'esatta Part-of-Speech di una parola (che quindi rappresenta l'istanza), la funzione (classifier) prima apprende da una collezione di istanze appartenenti alle relative categorie morfologiche ed etichettate con risposta corretta, poi predice per la nuova istanza (con PoS sconosciuta) il valore nominale, così definito in quanto appartenente ad un set

finito di valori possibili. Con il clustering, invece, non conoscendo a priori questo set di valori possibili, la funzione tenterà di eseguire la categorizzare senza alcuna istanza di riferimento.

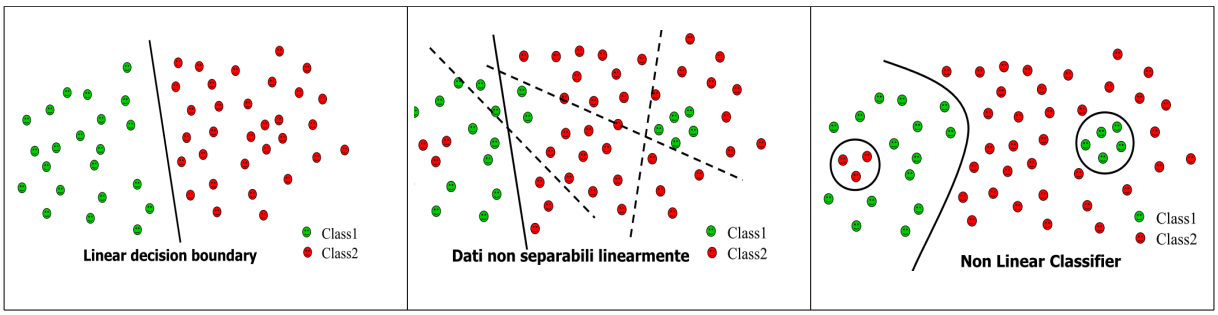
La scelta della modalità *supervised* o *unsupervised* si basa sui vantaggi e svantaggi di entrambe: la modalità *supervised* riesce a predire la giusta classe per le istanze appartenti al test set ma richiede una consistente quantità di istanze annotate e questo può rappresentare un processo costoso se effettuato manualmente.
La modalità *unsupervised* tipica del clustering, invece, ha il vantaggio di non richiedere un training già annotato (situazione particolarmente frequente quando si ricorre al ML) ma difficilmente etichetta correttamente il cluster e ottiene una precisione più scarsa rispetto al primo metodo nell'associare le istanze ai cluster corretti.

I principali approcci di classificazione sono due[58].
In un *modello parametrico*, il modello stesso è preventivamente caratterizzato da un vettore θ di parametri: si suppone quindi che esista una relazione tra features e input e che tale relazione sia rappresentabile all'interno di una famiglia di relazioni parametriche rispetto a θ che costituisce un modello; in altre parole, un'assegnazione di valori al vettore θ definisce una specifica relazione della famiglia. Gli elementi nel training set sono utilizzati proprio per derivare tale assegnazione di valori ai parametri (o una distribuzione di probabilità per tali valori), dopo di che non sono più utilizzati.
In un *modello non parametrico*, invece, il numero di parametri cresce con la dimensione del training set: sostanzialmente, ogni singola previsione, in questo caso, richiede l'utilizzo dell'intero training set. Un esempio di approccio non parametrico sono i classificatori di tipo nearest neighboor, in cui si determina l'elemento $\mathbf{x}_i$ del training set più vicino a $\mathbf{x}$ e si impone il valore della nuova previsione y associata a $\mathbf{x}$ uguale al valore y_i dell'elemento $\mathbf{x}_i$.

2.4.1 – Classificazione binaria

E' una tipologia di classificazione in cui le istanze appartengono in maniera esclusiva ad una classe o categoria (positiva) oppure all'altra (negativa). Geometricamente parlando, si tratta di individuare un separatore tra le istanze. Se tale separatore è una linea retta che classifica correttamente le istanze appartenenti alle due classi, si parla di *classificazione lineare*; se ciò non è possibile, si parla di *classificazione non lineare*.

58 Giogio Gambosi – *Introduzione al Machine Learning* – Materiale didattico del corso tenuto all'Università degli Studi di Roma Tor Vergata a.s. 2009-10



Quando si considerano solo due features, la classificazione binaria è rappresentabile su un piano: nella figura sottostante i due assi potrebbero riportare valori di insulina e colesterolo per un insieme di pazienti osservati e le classi da assegnare potrebbero essere “Sano” (bianco) e “Malato” (rosso). A sinistra si nota un esempio di classificatore lineare: la separazione delle due classi si può ottenere tracciando una linea retta anche se tre elementi del training set (due rossi e un bianco) appartengono alla classe sbagliata. A destra, invece, le classi sono separate da una curva più complessa: non ci sono errori di classificazione, ma la separazione potrebbe essere eccessivamente dipendente dal training set (in tal caso si parla di overfitting).

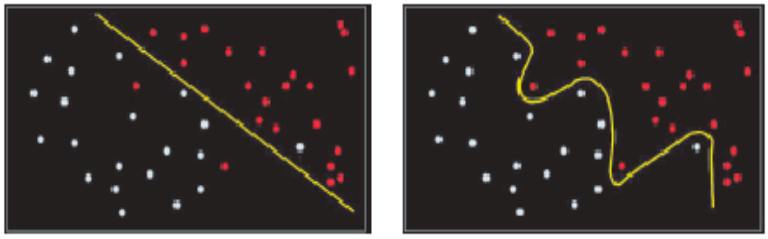


Figura 16 : classificazione binaria lineare e non lineare

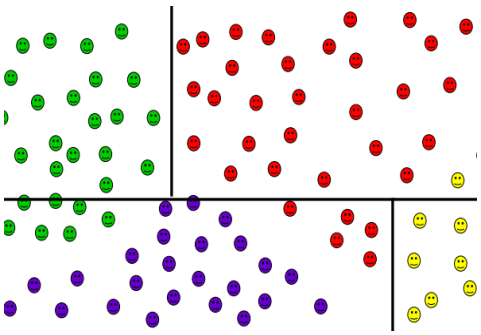
Esistono vari algoritmi di classificazione, i più noti sono:

- decision tree: lineare – multiclasse - confini di regione paralleli
- Naive Bayes: lineare – multiclasse - confini di regione non paralleli
- KNN: non lineare – multi-classe
- SVM: lineare - binaria

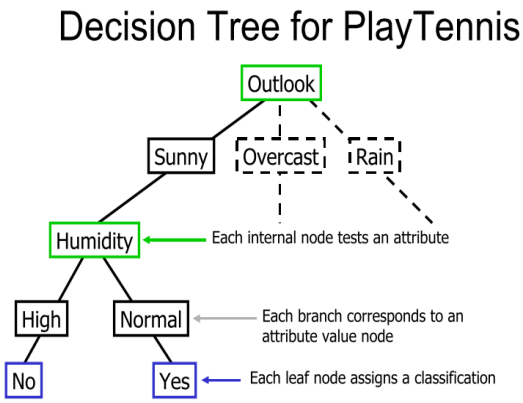
Per ognuno si fornisce una breve panoramica in quanto applicati all'esperimento.

2.5 - PRINCIPALI CLASSIFICATORI[59]

2.5.1. - DECISION TREE



Classificatore lineare con confini di regione paralleli



Si tratta di un classificatore con struttura ad albero, in cui ogni nodo può essere o *foglia* o *nodo interno:* se foglia, indica il valore della classe assegnata all'istanza; se nodo interno, specifica il test effettuato su un attributo. Per ciascun valore assunto da un attributo in un test, l'algoritmo crea un ramo e il relativo sottoalbero.

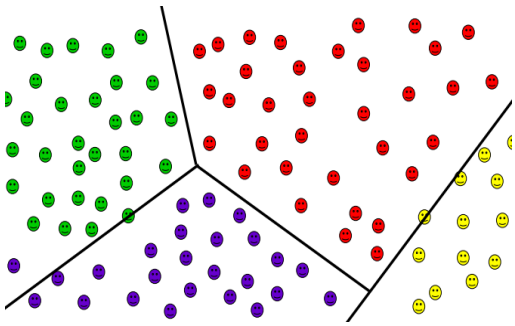
Il focus principale dell'algoritmo di crescita del decision tree è come scegliere gli attributi da testare in ciascun nodo interno dell'albero. L'obiettivo è selezionare gli attributi più utili per classificare le istanze di training attraverso una strategia top down, che consiste in una ricerca greedy degli attributi senza tornare a riconsiderare le precedenti scelte.

Il criterio di split (suddivisione) con cui crea un nuovo nodo si basa sul massimo guadagno di informazione (info gain). In pratica sceglie l'attributo che riesce a dividere "meglio" le istanze appartenenti a classi diverse (detto anche criterio di massima riduzione di incertezza). Quando tutti gli elementi in un nodo hanno la medesima classe, l'algoritmo non procede con ulteriore split (criterio di stopping). Per evitare overfitting, l'algoritmo inizia una eventuale fase di pruning (potatura): individua gli attributi che non hanno contribuito ad una consistente suddivisione delle istanze ed elimina i rispettivi nodi riunendo le istanze al livello superiore. Quando l'algortimo termina, è possibile percorrere l'albero dalla radice e, seguendo il persorso risultante dai singoli test presenti su ogni nodo interno, si ottiene la classificazione dell'istanza (nodo foglia).

Il decision tree non funziona bene quando la classificazione prevede numerose classi e un numero relativamente piccolo di esempi. Inoltre la fase di training può essere computazionalmente costosa perchè deve confrontare tutti i possibili split ed eventualmente effettuare il pruning, anch'esso molto costoso.

59 Le immagini sono tratte da Barbara Rosario – *Applied Natural Language Processing* – corso ottobre 2004 e adattate

2.5.2 - NAIVE BAYES



Classificatore lineare con confini di regione non paralleli

The Naive Bayes è un classificatore lineare particolarmente semplice, basato sul teorema di Bayes e su una forte assunzione di indipendenza, tanto che a volte è noto come "modello a feature indipendenti". In pratica si assume che la presenza (o l'assenza) di una particolare feature di una classe *non* sia correlata alla presenza (o assenza) di altre features. Dunque il contributo di ogni feature è considerato indipendente dagli altri.
Il classificatore considera un'istanza come un vettore di valori relativi a *m* attributi: $\mathbf{x} = (x_1, x_2, \ldots x_m)$.
La probabilità $p(\mathbf{x},y)$ di un'istanza **x** con label *y* può essere espressa come[60]:

$$p(\mathbf{x},y) = p(y)p(x_1|y)\ldots.p(x_m|y, x_1, x_2, \ldots x_{m-1})$$

Grazie proprio all'assuzione di indipendenza dei parametri, sono determinate solo le varianze delle variabili di ciascuna classe. Sebbene questa assunzione sia spesso falsa, il classificatore si dimostra paradossalmente efficiente nel supervised learning, senza contare l'enorme beneficio in termini di complessità computazionale.

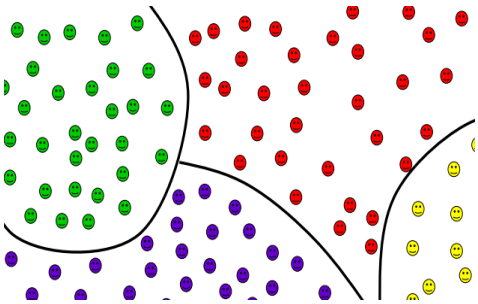
I vantaggi di questo algoritmo comprendono la semplicità del modello stesso facile da implementare, l'efficienza ottenuta soprattutto con la text categorization, la velocità sia in fase di training che di classificazione, lo spazio di memoria richiesto abbastanza contenuto, la preparazione di un set training non troppo vasto.

Lo svantaggio principale è dato dal fatto che, assumendo gli attributi indipendenti tra loro, non considera la sequenza ordinata delle parole, ma solo la loro presenza (*bag of words*), assunzione inappropriata se ci sono forti dipendenze tra le variabili. Sebbene tale modello non sia del tutto "corretto", lavora bene in un sorprendente gran numero di casi perchè spesso si è interessati all'accuratezza della classificazione più che all'accuratezza delle stime di probabilità.

60 Steven Abney – *Semi-supervised Learning for Computational Linguistics* - Chapman & Hall/CRC, 2008

2.5.3 - *K*-NEAREST-NEIGHBORS CLASSIFIER

Anche questo classificatore è considerato tra i più semplici del machine learning: esso memorizza le istanze del training, poi, basandosi su un criterio di vicinanza, mette in relazione l'istanza da classificare con alcune istanze del training set presenti nello spazio delle feature. In pratica, l'istanza è classificata "a maggioranza" in base alla classe più comune tra le *k* istanze più vicine del training.



Classificatore non lineare

Tutto il lavoro è fatto dal classificatore in runtime. Data una nuova istanza **x** da classificare, il classificatore cerca i *k* esempi del training che sono più "simili" a **x** e guarda le loro etichette. Qualsiasi label ricorra più frequentemente tra le *k* label più vicine è scelta per assegnare la classe a **x**. Il parametro *k* è un intero positivo, tipicamente non molto grande e dispari nella classificazione binaria, per evitare di ritrovarsi in situazioni di parità.
La migliore scelta di *k* dipende dalla tipologia di dati: grandi valori di *k* riducono l'effetto rumore nella classificazione (ossia la classificazione è meno influenzata da istanze "strane") ma rendono i confini tra le classi meno definiti.
Considerando solo le label dei *k* oggetti vicini, se una classe predomina sensibilmente sull'altra l'errore di classificazione aumenta. In questa circostanza può essere utile associare dei pesi alle istanze vicine in modo che l'algoritmo dia la "giusta" importanza alla distanza tra l'oggetto considerato e i vicini, senza lasciarsi influenzare totalmente dalla quantità.

Scegliendo *k* =1 risulta semplice applicare una misura di similarità tra istanze basata sulla distanza euclidea. Le istanze corrispondono a punti in uno spazio definito da assi corrispondenti a feature e sono rappresentate da vettori di dimensione pari al numero delle feature. Ogni punto *P* del training è circondato da una regione: i punti dello spazio che racchiude sono quelli più vicini a *P* rispetto a qualsiasi altra istanza del training. I confini delle regioni sono ricavati calcolando la bisettrice della linea che collega una coppia di punti; alcuni di questi confini inividuano le regioni di decisione che separano istanze appartenenti a classi diverse. L'insieme di confini che suddivide lo spazio di feature in porzioni crea la "Voronoi tessellation", illustrata in figura.

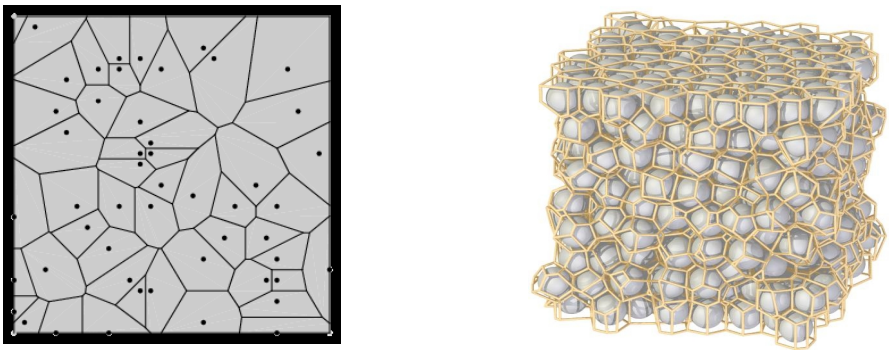


Figura 17 - esempi di Voronoi tessellation

Per classificare una nuova istanza, l'algoritmo cerca nel training set l'istanza “più simile” a quella data. Poi consulta le k istanze più vicine e sceglie la classe maggioritaria assegnandola alla nuova istanza. L'accuratezza dell'algoritmo può subire un forte calo per la presenza di feature irrilevanti.
E' un algoritmo robusto, computazionalmente costoso sebbene contettualmente semplice, che ottiene spesso buoni risultati. Tuttavia la performance è molto condizionata dalla misura di similarità usata e trovare una buona misura di similarità può essere difficile.

In generale esiste un compromesso tra la complessità di un modello e l'accuratezza della classificazione (fit). Il kNN rappresenta l'estremo della complessità: pur essendo un algoritmo con buone prestazioni, cattura anche il rumore. Il Naive Bayes si pone invece all'estremo opposto: la sua assunzione di indipendenza lo rende un modello computazionale semplice ma il confine tra due regioni di decisione è una linea retta che, come tale, ottiene un fit minore.

Idealmente, un algoritmo dovrebbe saper catturare gli attributi che generalizzano il training senza lasciarsi trarre in inganno dagli attributi accidentali che generano rumore: un algoritmo che cattura troppi dettagli rischia di essere molto complesso (*overfit*); un algoritmo che ne cattura troppo pochi, è più semplice da eseguire ma rischia di perdere in generalizzazione (*underfit*)

2.5.4 - SVM Support Vector Machine

Senza voler entrare in dettagli formali, l'idea principale di questo classificatore consiste nel rappresentare gli esempi del training come punti nello spazio mappati in modo tale che punti appartenenti a classi differenti siano separati dal più ampio gap possibile. I punti che mappano il test set saranno assegnati ad una categoria o all'altra in base al lato del gap su cui cadono. Più specificatamente, SVM costruisce un iperpiano ed esegue una buona separazione quando l'iperpiano ha la più ampia

distanza dai punti di training più vicini di ciascuna classe. Ci sono molti iperpiani che potrebbero classificare il dato. La miglior scelta è quella di selezionare l'iperpiano che rappresenta la più ampia separazione, o margine, tra due classi, ossia l'iperpiano tale che la distanza tra esso e il punto più vicino su ciascun lato sia massima. Se esiste, tale iperpiano è noto come il *massimo margine di iperpiano* e il classificatore lineare è definito come *classificatore a margine massimo*. Se i dati non sono linearmente separabili, il vincolo non è soddisfacibile: in tal caso, si introduce nel modello un rilassamento del vincolo (variabile "slack") il cui valore indica l'entità della violazione.

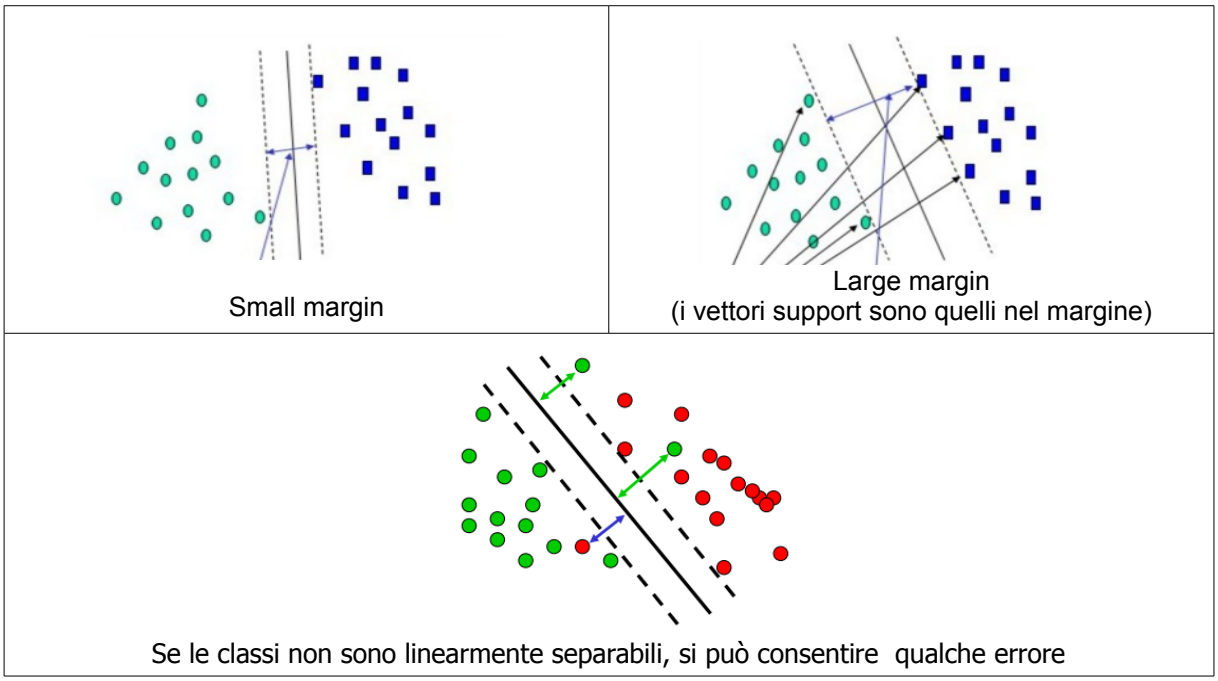


Figura 18: classificatore a margine massimo

Capitolo 3
DESCRIZIONE ESPERIMENTO

3.1 - CONTESTO

Come specificato nell'introduzione, scopo di questo studio è valutare l'affidabilità di un sistema di classificazione automatica di documenti costituiti, in questo caso, da referti radiologici inerenti alla patologia dello pneumotorace. Per poter eseguire tale classificazione (che in questo caso è binaria in quanto dovrà suddividere i referti che dichiarano la presenza della patologia da quelli che ne dichiarano l'assenza) il sistema dovrà conoscere il contenuto dei referti stessi e, in base a questo, decidere a quale categoria poter assegnare ciascun referto. Dunque, tenendo conto di quanto approfondito nei capitoli precedenti, si tratta di utilizzare strumenti di analisi linguistica (NLP) per realizzare un processo di Information Extraction.

Perchè scegliere la patologia dello pneumotorace?

Per capire il contesto in cui si inserisce questo esperimento e del perchè si è scelta questa patologia, si forniscono alcune nozioni di base sulla patologia, senza alcuna pretesa di rigore medico. Lo *pneumotorace* si definisce come presenza di aria nel cavo pleurico[61]: l'aria entra nella cavità pleurica e non riesce più ad uscire, creando un accumulo d'aria che può schiacciare il polmone stesso. Può essere spontaneo, secondario, traumatologico o iatrogeno: nel primo caso, lo pneumotorace è provocato dalla rottura di una o più bolle di enfisema del polmone; è secondario se provocato da una malattia polmonare; di origine traumatica se si sviluppa come conseguenza di un trauma (per es. ferita da parte di coste fratturate); infine iatrogeno se conseguenza di manovre diagnostiche. Il medico che sospetta la presenza di tale patologia sulla base di sintomi come dolore toracico localizzato, di intensità variabile, con esordio improvviso, ingravescente, esacerbato dal respiro e associato ad altri sintomi, sottopone il paziente ad un esame radiografico: solo dall'analisi di quest'ultimo, il medico può avere conferma o meno della presenza di tale patologia.

Dunque, se l'esame viene eseguito proprio per dissipare tale dubbio, il referto

61 Teschendorf - *Diagnostica differenziale radiologica* - McGraw-Hill libri Italia - Milano 1993, p.408

dovrebbe essere caratterizzato da un'elevata chiarezza: il radiologo, oltre a descrivere il contenuto della lastra, specifica se la patologia “c'è o non c'è”. In linea di principio quindi non ci dovrebbero essere dubbi circa la classe da assegnare in fase di classificazione. La scelta della patologia e dei referti ad essa collegati, è stata fatta proprio sulla base di questa considerazione.

Tuttavia, leggendo i referti con attenzione, si notano casi particolari in cui la chiarezza ai fini della classificazione viene meno, soprattutto se si considera che un algoritmo non ha capacità di comprensione e quindi non può mettere in atto un processo di “interpretazione” tipicamente umano. Può succedere, infatti, che sottoponendo il paziente a ripetuti esami per tenere sotto controllo l'evoluzione della patologia, la bolla gassosa scompaia dal punto dove era inizialmente presente (suggerendo quindi la guarigione) e contestualmente appaia in un punto diverso (denotando quindi l'insorgenza della patologia in una nuova sede). In tal caso il referto dichiara entrambe le condizioni: sia la presenza, sia l'assenza dello pneumotorace, ma, dal punto di vista del paziente, ha senso classificare il referto tra quelli che dichiarano la presenza della patologia.
Oppure si notano spesso frasi in cui si dichiara la presenza di “falda pleurica” ma non di “falda di pneumotorace” (o viceversa): lo pneumotorace è una patologia della pleura ma non è una falda pleurica. Esistono quindi ambiguità che possono trarre in inganno.

Sarà interessante notare come si comporterà il sistema in questi casi.

3.2 - SET DI INPUT

Il set di input è costituito da 213 referti radiologici estratti dal RIS (Radiology Information System) della sezione di Radiologia di una struttura sanitaria di cui si omette il nome e la sede per motivi di privacy dei referti. Il sistema è basato interamente su tecnologia open source (server web Apache, database relazionale MySQL, linguaggio di script Php) e contiene i referti relativi a diverse diagnostiche (RX - radiografia tradizionale, RXS - radiografia speciale, TC - tomografia computerizzata, US - ecografia, RM - risonanza magnetica, AN – angiografia) effettuate dal 1999 ad oggi. I referti sono stati estratti con una query che estrae tutti i referti contententi il termine *pneumotorace*. Quest'ultimo non è l'unico termine usato per indicare la patologia: come specificato in seguito, esiste un lungo elenco di sinonimi, e per ottenere tutti i referti inerenti a tale patologia si sarebbe dovuto ripetere la query per ogni termine equivalente. Già in questa fase si nota come l'uso di un'ontologia sarebbe stata particolarmente utile per reperire tutti i referti inerenti allo pneumotorace, basandosi sia sul termine chiave che sui

suoi sinonimi; in questo modo si sarebbero reperiti anche eventuali referti contenenti termini poco noti o in disuso che però sono attinenti alla patologia considerata. Tuttavia, ai fini della classificazione che si vuole eseguire, non è fondamentale ripetere la query per tutti i sinomini: il metodo usato può essere facilmente esteso anche ad altri referti specificando opportunamente i sinonimi in appositi elenchi descritti in seguito.

La query ha restituito un elenco di record che sono stati successivamente convertiti in singoli file di testo con estensione *.txt* e numerati; ogni file contiene l'esito dell'esame radiologico. Se ne riportano tre di esempio con alcune brevi note:

955206 Nei confronti del precedente esame del 18.06.2009: reperto invariato, in particolare non più riconoscibili le falde di pneumotorace. *Referto molto breve. Si nota il riferimento ad un esame precedente e all'assenza della patologia, senza alcun altra informazione aggiuntiva.*
957292 Campi polmonari normoespansi, con aumento della trama interstiziale. Ombre ilari e cardiaca nei limiti. Aortosclerosi. Emidiaframmi liberi. Non evidenti segni di pneumotorace. Legacci metallici sternali e immagine di protesi valvolare contro l'ombra cardiaca. Immagine di pacemaker contro le parti molli sottoclaveari di sinistra da cui si diparte filo elettrostimolatore che, senza apparente soluzione di continuo, per via anonimo-cavale, giunge con estremo profondo in ventricolo destro. *Referto di media lunghezza. Si noti come il radiologo ha dichiarato l'assenza della patologia in un contesto ricco di informazioni aggiuntive. Inoltre, lo stile di scrittura tende a isolare i concetti inserendoli in frasi separate.*
955644 Addensamento parenchimale coinvolge il segmento superiore ed il parenchima basale posteriore del lobo inferiore di sinistra; omolateralmente concomitano falda pleurica con spessore massimo di 13 mm alla base ed assai esile falda di pneumotorace apico-basale. Sottile falda di versamento pleurico basale anche a destra, con parziale atelettasia del parenchima polmonare contiguo. Posizionata sonda di Argyle con estremo profondo al terzo medio del campo polmonare di sinistra, in piani dorsali.....(prosegue con la descrizione di numerosi dati clinici) *Referto molto lungo. L'informazione considerata utile al nostro scopo è presente nelle prime 5 righe. Si noti la presenza concomitante di falda pleurica e falda di pneumotorace.*

Già da questi esempi si nota una prima notevole differenza tra i referti: alcuni sono particolarmente lunghi e contengono numerosi dati diagnostici; altri invece sono molto brevi, composti da una o due frasi. La lunghezza del referto potrebbe indicare un livello di dettaglio maggiore rispetto a quello fornito dai referti estremamente corti, caratteristica utile qualora si intendesse ricavare delle

relazioni tra pneumotorace e altre caratteristiche cliniche desumibili dal referto.

Altra caratteristica frequente è il riferimento ad un esame precedente. Questa peculiarità potrebbe essere sfruttata e se si volesse, per esempio, studiare l'evoluzione dello pneumotorace nel tempo, dopo quanto tempo in genere scompare, quali fattori possono ritardare o accelerare la sua scomparsa in base alle informazioni cliniche evidenziate nel referto.

Ai fini della classificazione, si è rivelata particolarmente utile un'ulteriore doppia caratteristica: i concetti presenti nel referto possono essere molteplici ma il radiologo tende a specificarli in frasi separate; inoltre, quasi mai il medesimo concetto viene ripetuto in più frasi. Nel primo esempio tale peculiarità non è evidente in quanto il referto dichiara un unico concetto, ossia l'assenza dello pneumotorace. Nel secondo referto invece si nota bene come i concetti di “campi polmonari”, “ombre ilari e cardiaca”, “aortosclerosi”, “emidiaframmi”, “legacci metallici sternali e protesi”, “pacemaker” e così via appartengano ciascuno ad un'unica frase distinta. Nel terzo referto, invece, la prima parte della caratteristica non è perfettamente rispettata perchè falda pleurica e falda di pneumotorace appartengono alla stessa frase, mentre la seconda parte lo è. Questa ultima caratteristica consente di operare una semplificazione sui referti: sebbene possano essere particolarmente lunghi, ai fini della classificazione è sufficiente selezionare solo la frase inerente alla patologia, frase che quasi sempre è singola. Nello specifico si tratterà di individuarla per poi analizzarne ogni singolo termine.

Altro elemento chiave ai fini della classificazione è l'identificazione di quegli elementi che dichiarano o meno la presenza della patologia. Ritornando agli esempi di referti sopra riportati e leggendone altri, si nota l'uso frequente di termini definiti *negativi* come “non”, “non più”, “scomparso” quando il radiologo vuole escludere la patologia, mentre usa termini *positivi* come “persiste” o “comparso” per indicarne la presenza. In alcuni casi, però, la parola “pneumotorace” non è accompagnata da alcun elemento *positivo*: il solo fatto di nominare la patologia lascia intendere la sua presenza. Questo significa che non si può contare su quest'ultima tipologia di termini; conviene invece considerare solo quelli *negativi,* individuando i referti che dichiarano l'assenza della patologia e assegnando gli altri, per esclusione, alla categoria dei referti che dichiarano la presenza della patologia.

L'analisi dei referti potrebbe continuare nell'individuazione di altri elementi importanti: per esempio, su quale lato o in quale posizione specifica insorge lo pneumotorace, sia per esigenze statistiche o per inferire delle relazioni con altri elementi del referto. Sono solo esempi di come si potrebbe fare Information

Extraction sui referti: non si tratta solo di ricercare stringhe presenti in un testo, ma attribuire un “valore” agli elementi che l'utente ritiene importanti.

Per valutare con maggior accuratezza il set considerato, si procede innanzitutto con un'analisi delle occorrenze dei singoli termini, in modo da evidenziare eventuali elementi utili alla classificazione. Utilizzando TextSTAT[62], si ottiene in output un elenco dei termini presenti nel corpus in ordine decrescente di occorrenza. Si considerino solo i primi 56 termini[63]:

di	724	confronti	151	base	65	persiste	50
del	409	il	130	mm	62	sede	49
destra	234	2009	122	si	62	basale	47
a	221	non	119	estremo	60	sonda	46
con	221	polmonare	91	alla	59	lobo	44
in	210	destro	89	circa	56	versamento	44
falda	206	al	86	contro	56	limiti	42
pneumotor	204	07	78	superiore	55	pleurico	42
e	201	apicale	78	polmonari	54	che	40
nei	196	spessore	75	delle	53	livello	40
la	173	massimo	70	drenaggio	53	ridotta	40
esame	169	le	67	piÃ¹	53	09	39
sinistra	169	della	66	cm	52	ed	39
precedente	154	per	66	profondo	52	08	38

Togliendo le preposizioni, gli articoli e, in generale, i termini non strettamente significativi per la classificazione, si nota che, a parte il termine “pneumotorace”, che dovrebbe essere presente in tutti i referti proprio per il metodo con cui sono stati selezionati, il primo termine potenzialmente utile potrebbe essere “non”, presente al 18° posto con una frequenza di 119; al 43° posto troviamo “persiste” con una frequenza sensibilmente minore pari a 50; infine “ridotta” al 53° posto con una frequenza di 40 (si indicano questi termini come *potenzialmente* utili perchè, con l'analisi del contesto, si tenterà di trovare una loro eventuale relazione con la patologia).
In pratica, la classificazione non può contare su termini caratteristici particolarmente frequenti. Infatti osservando la distribuzione di tutte le parole si ottiene il seguente grafico: escludendo le zone uniformi corrispondenti a termini non significativi (di, del, destra, a,....) e quella relativa a “pneumotorace” (in quanto ci aspettiamo che sia presente in tutti i referti), rimane una porzione di grafico fortemente frammentato. L'unico termine che riesce a rendersi visibile è proprio “non”, indicato dalla freccia, e che pertanto verrà considerato con

62 http://neon.niederlandistik.fu-berlin.de/en/textstat/ - Freie Universitat Berlin
63 Si noti come questa analisi consente di rilevare termini inesatti: nella tabella si nota “più” in cui la vocale accentata è stata tradotta dalle elaborazioni in caratteri non comprensibili a causa delle diverse codifiche utilizzate dalle procedure. Osservando il file in altri punti si possono evidenziare anche errori di ortografia come per esempio “penumotorace”, fortunatamente a fine elenco con frequenza 1.

particolare riguardo.

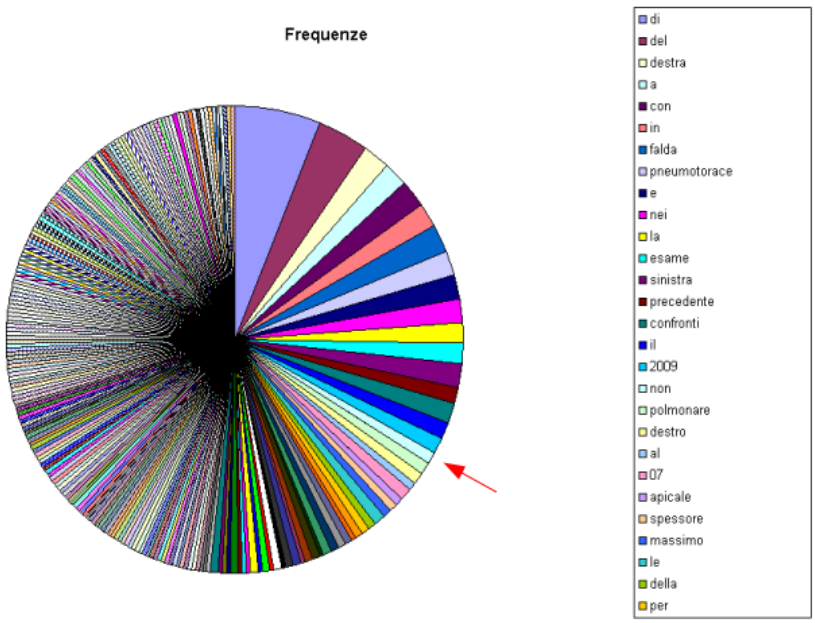


Figura 19 - Distribuzione frequenze

Il processo di estrazione dell'informazione si concretizza tipicamente mediante l'uso di annotazioni apposte su parole-chiave, frasi, concetti utili e, limitatamente alla classificazione, sull'intero documento, grazie a tools che consentono di elaborare testi non strutturati sulla base delle specifiche fornite dall'utente. I software di elaborazione del linguaggio umano (dunque strumenti NLP) più diffusi e disponibili come open source sono:
GATE – basato su Java, con una ricca libreria di componenti open source – (University of Sheffield)
NLKT[64] – basato su Python, fornisce tokeniser, stemmer, tagger, parser e altre funzionalità. Ha una documentazione molto ricca.
Minor Third[65] - basato su Java, fornisce classi per memorizzazione e annotazione del testo, nonché learning per estrazione entità e categorizzazione testi.
Clairlib[66] - basato su Perl, fornisce moduli per semplificare una serie di task del NLP, dell'IR e del Network Analysis – (University of Michigan)

Tra questi è stato scelto GATE, sia perchè offre numerose funzionalità che vedremo man mano sia perchè è disponibile una corposa documentazione per il suo utilizzo costituita da una user guide, video dimostrativi, wiki, mailing list, tutorial, classi JAVA ed esempi di programmazione in JAPE.

64 http://www.nltk.org/
65 http://sourceforge.net/apps/trac/minorthird/wiki
66 http://www.clairlib.org/index.php/Main_Page

3.3 - GATE – General Architecture Text Engineering [67]

GATE può essere pensato come un'architettura sofware per il Language Engineering (LE). A sua volta il LE può essere definito come "la disciplina o l'arte dei sistemi software in grado di eseguire task che implicano l'elaborazione del linguaggio umano. Sia i processi di costruzione che i suoi output sono misurabili e predicibili."[68]

Il contesto in cui si inserisce è più chiaro se si considerano le seguenti definizioni:

- **Computational Linguistics (CL)**: scienza del linguaggio che usa il calcolo come strumento investigativo.
- **Natural Language Processing (NLP):** scienza che si occupa di strutture dati e algoritmi per elaborare il linguaggio umano
- **Language Engineering (LE):** costruisce sistemi NLP i cui costi e risultati siano misurabili e predicibili

GATE si pone come **Software Architecture for Language Engineering (SALE**), ossia come infrastruttura software per integrare e sviluppare strumenti per CL, NLP e LE.

Come definito nel sito stesso, cui si rimanda per ogni approfondimento, GATE è un open source free software che comprende un ambiente di sviluppo integrato (IDE - Integrated Development Environment) per l'utilizzo di componenti di NLP già presenti nell'applicativo e un insieme di utili plugin da installare all'occorrenza.
Le principali funzionalità di GATE comprendono tokeniser, gazetteer, sentence splitter e parser; l'applicativo fornisce inoltre strumenti per analisi di testi scritti in varie lingue sebbene, purtroppo, non ancora un plugin per testi in italiano, consente di utilizzare algoritmi di machine learning e di costruire ontologie. In particolare fornisce un sistema di IE, chiamato ANNIE, che comprende una serie di componenti normalmente utilizzati in una tipica elaborazione di NL e che si sono rivelati particolarmente utili. Per il test è stata utilizzata la 5.1, scaricata dal sito ufficiale e installata senza alcuna difficoltà.

Prima di descrivere i moduli utilizzati e il procedimento attuato, si definiscono i seguenti concetti su cui si basa GATE, necessari per la comprensione di quanto descritto successivamente:

- *Applications Resources (AR):* consentono di definire sequenze di processing resources che possono elaborare singoli documenti oppure l'intero corpus
- *Language Resources (LR)*: consentono di visualizzare il documento o il corpus

67 http://gate.ac.uk/

68 H. Cunningham. *A Definition and Short History of Language Engineering.* - Journal of Natural Language Engineering, 5(1):1–16, 1999.

sul quale si sta lavorando

- *Processing Resources (PR)*: visualizzano i moduli attivati dall'utente che possono essere selezionati e inseriti nelle sequenze di cui al punto uno
- *Datastores (DS):* rende persistente il corpus e le annotazioni create dall'utente sui documenti mediante memorizzazione in un datastore.
- *Documento*: elemento testuale oggetto di elaborazione
- *Corpus o corpora:* set di documenti che saranno elaborati in modo uniforme dai vari moduli selezionati dall'utente.
- *Annotation:* annotazione che sarà creata analizzando il documento.
- *Annotation set:* gruppo di annotazioni tra loro correlate

3.4 - GENERAZIONE DEI CORPORA

La schermata di avvio si presenta così:

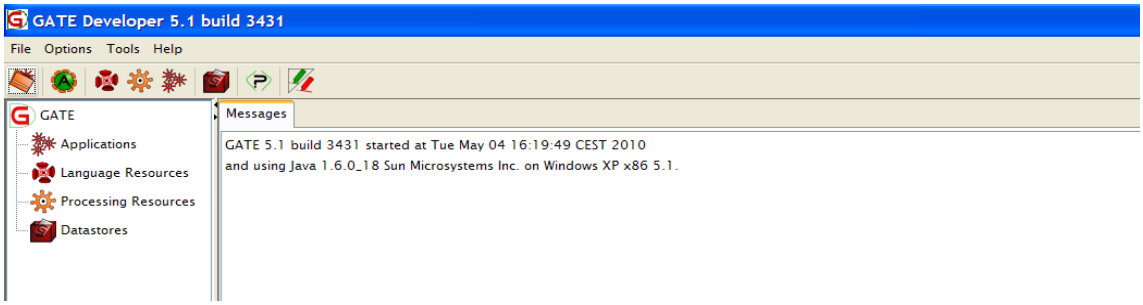


Figura 20 – ambiente di lavoro di GATE

Come si può notare, la grafica è molto essenziale: le poche icone presenti sulla destra e in alto consentono di attivare i singoli componenti di GATE (AR, LR, PR, DS) e di aggiungere eventuali plugin.

Innanzitutto è necessario caricare i documenti da elaborare: selezionando LR, si carica un documento (nel nostro caso un referto), si crea un corpus contenente quel referto e infine si popola il corpus aggiungendo tutti gli altri referti. Si ottiene così un insieme di documenti gestiti come fossero un unico grande documento. Per far sì che le annotazioni create sui documenti rimangano "persistenti", ossia memorizzate, è necessario definire un datastore: delle due tipologie possibili, è stato scelto un SerialDataStore basato su un sistema di serializzazione di Java. All'utente, il datastore appare come due directory destinate a contenere copia dei referti annotati e l'elenco degli stessi (denominato corpus).

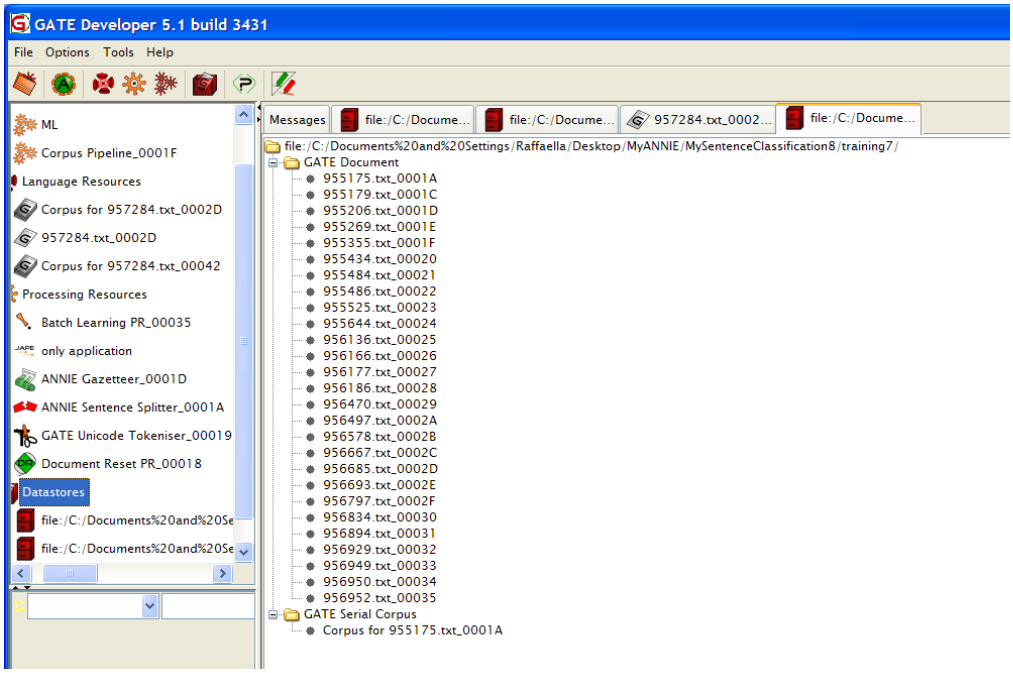


Figura 21 - creazione del corpus

A questo punto si può procedere con la creazione delle annotazioni.

3.5 ANNIE – A Nearly New Information Extraction System

GATE include un sistema completo di Information Extraction chiamato ANNIE (a Nearly-New Information Extraction System). Tale sistema non è caricato automaticamente dal programma ma deve essere selezionato dall'utente come plugin, configurato ed eventualmente settato per il caricamento automatico ad ogni avvio di GATE. Esso comprende una serie di moduli di seguito brevemente illustrati in quanto utilizzati per il test.

3.5.1 - Document Reset

Questa risorsa consente di resettare le annotazioni aggiunte al documento nelle varie elaborazioni ed è particolarmente utile quando si eseguono ripetute elaborazioni sullo stesso corpus per sperimentare i differenti risultati; restano invariate eventuali annotazioni originali. L'utente ha facoltà di impostare dei parametri per decidere un reset parziale delle annotazioni.
Tale risorsa è stata inserita in ogni test prima di utilizzare le altre Prs in modo tale che le annotazioni aggiunte da elaborazioni precedenti non influenzino i risultati ottenuti man mano.

3.5.2 - Tokeniser

Scopo di questa risorsa è suddividere il testo in "pezzi" come numeri, punteggiatura e parole. Come output si ottengono annotazioni di tipo *SpaceToken* oppure *Token* a cui sono associati di default gli attributi come *string*, *orth*, *category*, *length*, etc.

Per esempio, se il documento da processare è: *"The name of dog is Moon."*, il tokeniser restituirà i valori evidenziati nella seguente schermata:

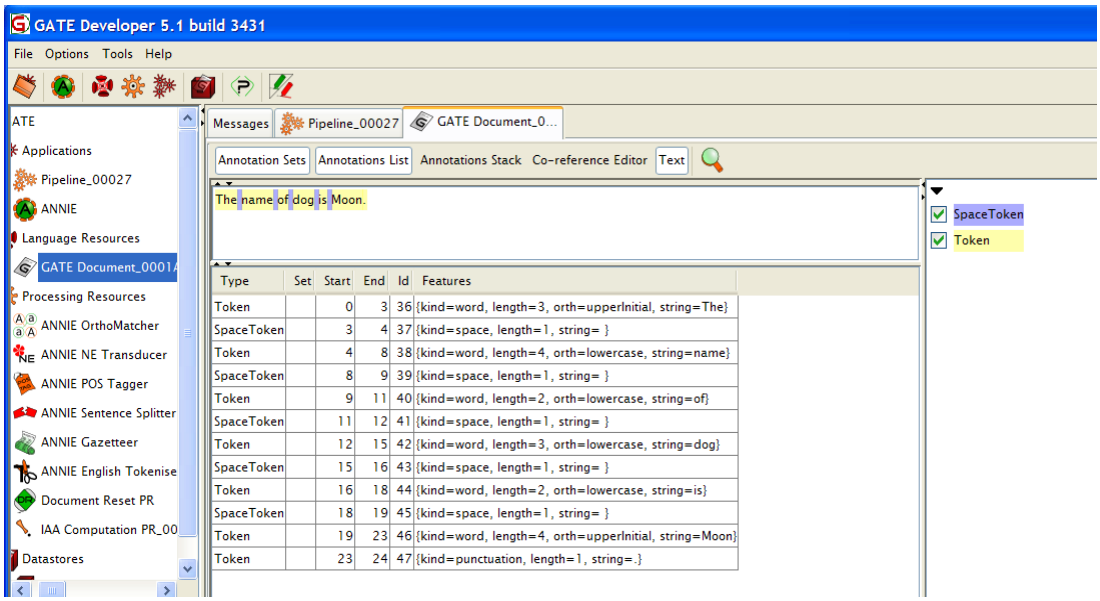


Figura 22 - risultato del tokeniser

Gli attributi associati ai *Token* e agli *SpaceToken* hanno i seguenti significati:

kind per distinguere se si tratta di *word*, *space* o *punctuation*;

length per indicare la lunghezza in caratteri del token

orth per indicare come è scritto il *Token*. Può avere i seguenti valori:
lowerCase: se tutte le lettere sono scritte in minuscolo
upperInitial: se la prima lettera è maiuscola e le successive minuscole
allCaps: se tutte le lettere sono maiuscole
mixedCaps: tutte le altre combinazioni non previste dalle prime tre

string stringa che riporta i caratteri di cui si compone il *Token*

GATE offre due tipi di tokeniser: quello fornito di default da ANNIE e uno più generico (GATE Unicode Tokeniser). Il primo è utilizzabile esclusivamente per testi in lingua inglese in quanto riconosce particolari regole grammaticali, tra cui le forme contratte. Per tutte le altre lingue, è necessario usare quello generico pertanto anche per i referti è stato utilizzato il GATE Unicode Tokenizer.

3.5.3 - Sentence Splitter

La PR Annie Sentence Splitter divide il documento elaborato in segmenti corrispondenti a frasi. Essa aggiunge due annotazioni di facile comprensione: *Sentence* (che individua il testo della frase) e *Split* (che rappresenta il punto di fine frase).

Esiste anche un modulo alternativo: RegEx Sentence Splitter. Tuttavia, quando si è provato a testarlo più volte su un documento, ha reso il sistema instabile. Per l'elaborazione, si è preferito quindi scegliere quello di ANNIE.

3.5.4 - Part Of Speech Tagger

La funzionalità fornita da questo tipo di tagger è già stata approfondita in precedenza. L'effetto di questa PR sul testo considerato consiste nell'assegnazione a ciascun *Token* di un ulteriore attributo *category* valorizzato sulla base di una codifica predefinita la cui lista completa è disponibile sulla user guide di GATE e di cui si riportano a titolo di esempio solo i codici relativi al documento analizzato:

DT *determiner: Articles including 'a', 'an', 'every', 'no', 'the', 'another', 'any', 'some', 'those'.*
NN *noun - singular or mass*
IN *preposition or subordinating conjunction*
VBZ *verb - 3rd person singular present*
NNP *proper noun – singular (all words in names usually are capitalized but titles might not be.)*
. *literal period*

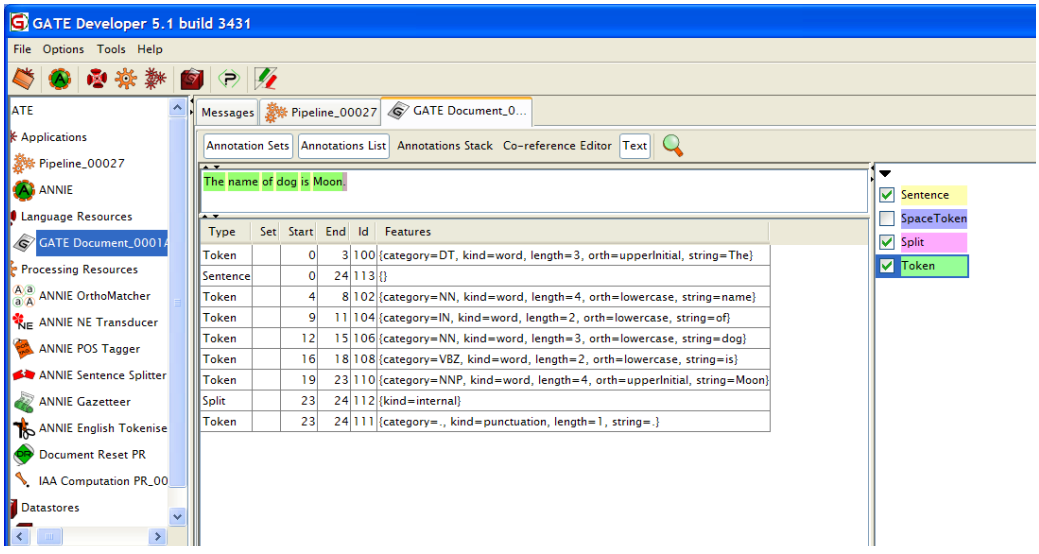


Figura 22 - risultati ottenuti applicando tokeniser, sentence splitter e POS tagger

Come si può facilmente intuire questa funzionalità è molto utile se si utilizzano documenti scritti in lingua inglese mentre non ha senso applicarla su testi in lingua italiana in quando darebbe risultati inaffidabili. Inoltre, come già accennato, per

l'italiano esiste al momento solo un servizio web di POS tagging e non un modulo integrabile con GATE. Tuttavia, dato che sono già disponibili dei plugin per alcune lingue europee diverse dall'inglese, si spera che lo staff di GATE rilasci a breve anche quello specifico per i testi in italiano.

3.5.5 - Gazetteer

Il ruolo di questa PR è identificare entità nel testo basandosi su apposite liste.
Questa fase, da un punto di vista pratico, non presenta particolari difficoltà; tuttavia richiede notevole accuratezza nella predisposizione delle liste se si vogliono ottenere risultati soddisfacenti.

Si supponga di voler estrarre più informazioni possibili da un documento e come esempio si consideri il referto n. 957284:

> *Esame eseguito in urgenza.*
> *Nei confronti del precedente esame del 29/06/2009: non più evidente il livello idroaereo al terzo medio del campo polmonare di sinistra per riduzione parziale della falda pleurica omolaterale.*
> *Persiste invariata la falda di pneumotorace apico-basale.*
> *Rimossa la sonda di drenaggio contro l'ipocondrio sinistro.*

Definendo "entità" un elemento significativo, da testo considerato si potrebbero ricavare le seguenti informazioni:

- *Esame eseguito in urgenza.*
  esclude l'ipotesi che si tratti di un esame di controllo o programmato
- *Nei confronti del precedente esame del 29/06/2009:*
  il paziente possiede già uno o più referti radiologici nella stessa struttura medica
- *non più evidente*
  presuppone la presenza di un'entità nel precedente esame
- *livello idroaereo*
  prima entità che si reputa significativa
- *al terzo medio*
  individua la precisa locazione della prima entità...
- *campo polmonare*
  ... all'interno di quest'altra entità
- *sinistra*
  individua a quale dei due polmoni ci si sta riferendo in questo caso,
- *riduzione parziale*
  individua una situazione che si evolve verso la normalizzazione
- *falda pleurica*
  rappresenta una terza entità (la cui evoluzione, in questo caso, ha influenzato la prima)

- *omolaterale*
  individua la locazione della seconda entità
- *Persiste*
  individua un'entità già presente nel referto precedente
- *invariata*
  specifica che tale entità non ha manifestato evoluzioni
- *falda di pneumotorace*
  rappresenta una quarta entità
- *apico-basale*
  locazione della quarta entità
- *Rimossa*
  in contrapposizione a "posizionata"
- *sonda di drenaggio*
  quinta entità da considerare
- *contro ipocondrio*
  locazione della quinta entità
- *sinistro*
  ulteriore specifica della locazione riferita alla quinta entità.

Se si volesse raccogliere tutte le informazioni sopra specificate, una possibile soluzione potrebbe essere quella rappresentata in figura:

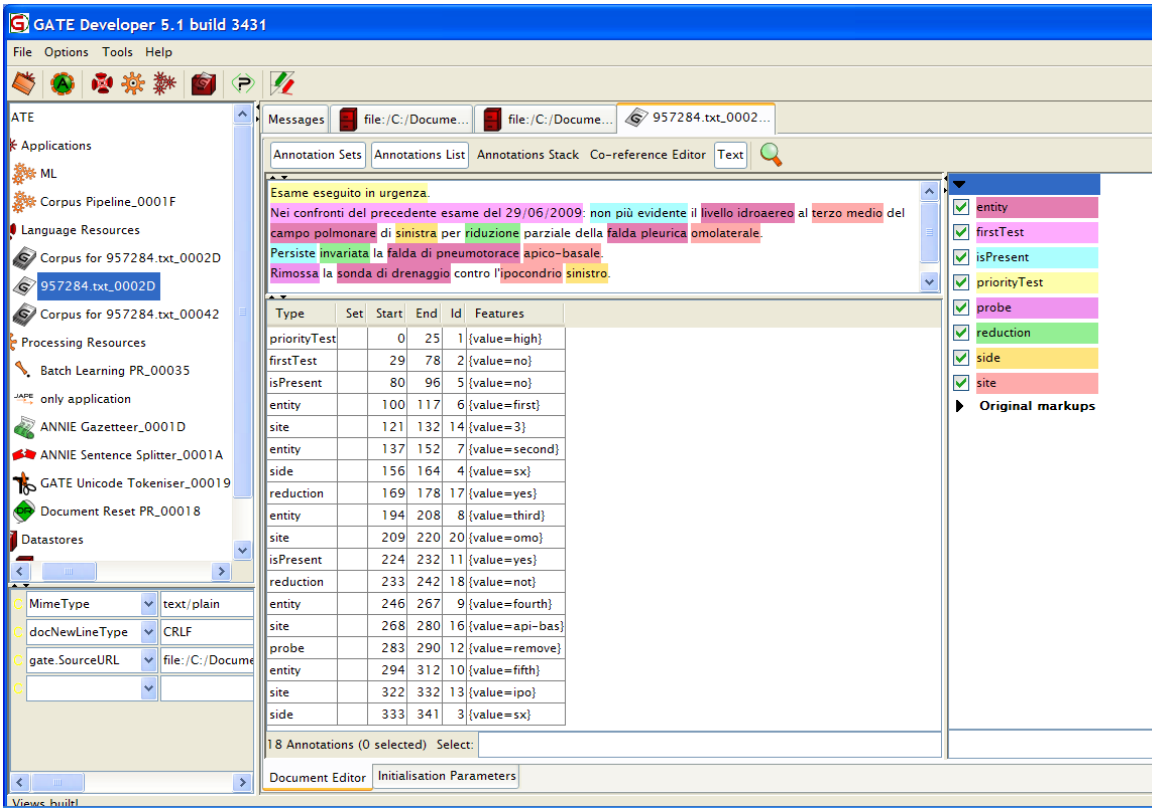


Figura 24 - utilizzo di annotazioni con GATE

Tutte queste annotazioni possono essere ottenute mediante apposite liste di Gazetteer, ottenendo un primo livello di struttura. Il vantaggio, già a questo livello,

è quello di ottenere in modo agevole e “abbastanza” veloce una serie di dati utili per impostare i primi criteri di analisi. Per esempio, è possibile selezionare solo i referti che, nel loro testo, fanno riferimento alla sonda, valutare quanti referti sono inerenti a patologie presenti sul lato destro o su quello sinistro, quali sono quelli inerenti ad esami eseguiti in urgenza e quali no, studiare in quanti referti la falda pleurica e la falda di pneumotorace sono contemporaneamente presenti, etc. Tale approccio, tuttavia, richiede un'analisi non banale relativa alla scelta più opportuna delle annotazioni e degli attributi ad esse associate, soprattutto se si pensa ad una loro combinazione per dedurre ulteriori elementi informativi.

Per quanto concerne la classificazione che si vuole ottenere, non è richiesto un livello di annotazione dettagliato come nell'esempio precedente ma sono comunque necessarie alcune accortezze. Di sicuro sarà presente il termine “pneumotorace” ma molto spesso la patologia viene indicata con la sequenza di termini “falda di pneumotorace”: dunque è più corretto individuare quest'ultima sequenza, quando presente, che non il singolo termine. Altro termine simile è “idropneumotorace”: la query che ha agito sul database ha sicuramente estratto referti contenenti tale termine in quanto “pneumotorace” è sottosequenza di “idropneumotorace”; in fasi di annotazione, tuttavia, è preferibile annotare quest'ultimo termine e non il primo. Se si immagina poi di avere a disposizione tutti i referti inerenti a tale patologia, indipendentemente dalla terminologia usata, allora diventa fondamentale fornire al sistema un elenco completo di termini medici equivalenti che, nel caso considerato, è il seguente:

pneumo.lst[69]

```
Falda di pneumotorace
falda di pneumotorace
Pneumotorace spontaneo
pneumotorace spontaneo
Pneumotorace post-traumatico
pneumotorace post-traumatico
Pneumotorace iperteso
pneumotorace iperteso
Pneumotorace chiuso
pneumotorace chiuso
Pneumotorace aperto
pneumotorace aperto
Pneumotorace a valvola
pneumotorace a valvola
Pneumotorace
pneumotorace
PNX
pnx
presenza di gas nello spazio pleurico
Presenza di gas nello spazio pleurico
presenza di aria nello spazio pleurico
Presenza di aria nello spazio pleurico
presenza di immagine gassosa nello spazio pleurico
```

69 Elenco gentilmente fornito da Dr. Roberto Silverio - Ospedale della Misericordia – Grosseto

```
Presenza di immagine gassosa nello spazio pleurico
presenza di immagine gassosa
Presenza di immagine gassosa
falda di aria
Falda di aria
falda di aria libera
Falda di aria libera
immagine ipertrasparente
Immagine ipertrasparente
idropneumotorace
Idropneumotorace
```

Tale file con estensione *.lst* costituisce il primo elenco fornito al Gazetteer che lo userà per trovare corrispondenze nei referti. Da notare che ogni termine è stato ripetuto scrivendolo sia con l'iniziale maiuscola che minuscola: questo per considerare sia i casi in cui la parola-chiave è contenuta all'interno di una frase sia quando ne rappresenti l'inizio.

Allo stesso modo sono stati creati ulteriori due file per i termini “positivi” e i termini “negativi”, ossia elenchi di verbi e avverbi normalmente utilizzati per decretare la presenza o l'assenza di “qualcosa”. Infine, per eventuali ricerche più accurate, sono stati previste anche le parole-chiave inerenti alla posizione della patologia. Gli elenchi ottenuti sono questi:

presenza.lst	**assenza.lst**	**lato.lst**	**posizione.lst**
persiste	non	destro	apicale
Persiste	Non	Destro	Apicale
comparsa	non pi	destra	apico-basale
Comparsa	Non pi	Destra	basale
aumentata	non più	dx	Basale
Aumentata	Non più	sinistro	
presente	non piu'	Sinistro	
Presente	Non piu'	sinistra	
	scomparsa	Sinistra	
	Scomparsa	sx	
		bilateralmente	
		Bilateralmente	

Tutti i file creati dall'utente (ed eventuali file resi disponibili da GATE che però in questa circostanza non sono utilizzabili visto che i referti sono scritti in italiano e non in inglese) devono essere specificati in un file riassuntivo, accoppiando a ciascuno nome una variabile, come di seguito illustrato:

list.def *(estratto del file)*

```
...
pneumo.lst:diseaseD
presenza.lst:positiveD
assenza.lst:negativeD
lato.lst:sideD
posizione.lst:siteD
```

. . .

Le variabili saranno utilizzate dalla successiva PR. L'esecuzione di ANNIE Gazetteer comporta la creazione di annotazioni di tipo *Lookup* con un attributo di tipo *majorType* che prenderà il valore della variabile da noi specificata.

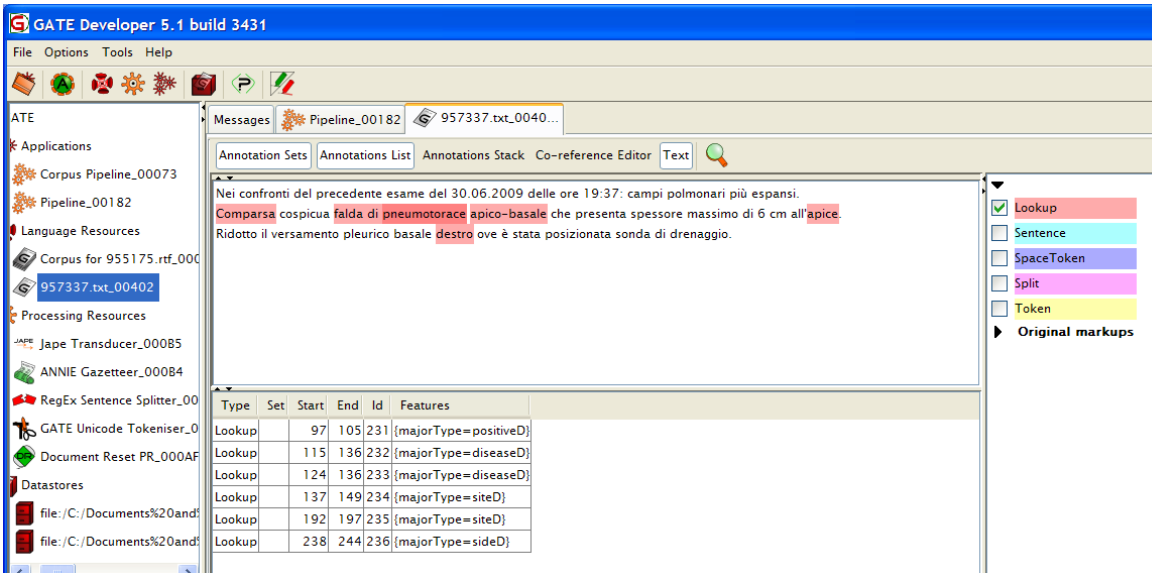


Figura 25: individuazione dei lookup

Come si evince dall'esempio, i termini evidenziati corrispondono a 4 tipologie di *Lookup* provenienti dalle liste: essi tuttavia non si riferiscono necessariamente alla patologia. Da qui l'opportunità di impostare delle “regole” per combinare i termini estratti con l'informazione che si desidera ottenere.

3.5.6 - JAPE Transducer

Si consideri ora la fase che utilizza i *Lookup* ottenuti con ANNIE Gazetteer. JAPE (Java Annotation Patterns Engine) è un linguaggio pattern-matching, una modalità che consente di scrivere alcune regole in base ad una precisa sintassi, che poi verranno tradotte in codice JAVA ed eseguite secondo una precisa sequenza, producendo come output ulteriori annotazioni, modifiche su quelle esistenti oppure modifica/creazione di attributi.

Una JAPE grammar consiste in un *main.jape* in cui si specifica secondo un preciso ordine logico un set di fasi, nel nostro caso quattro. La sequenza di fasi è giustificata dalla necessità di creare annotazioni temporanee che si trasformano in input nelle fasi successive per essere combinate tra loro o trasformate. Ciascuna fase è costituita da regole che individuano dei pattern ed eseguono delle azioni su di essi e poiché sono proprio le regole a dare significato ai dati finora estratti, se ne descrive il funzionamento a grandi linee:

```
Phase: preprocessing2
Input: Lookup
Options: control = appelt

Rule: Preprocess7
(
{Lookup.majorType==disease}
):isDisease

-->
:isDisease
  {
    gate.AnnotationSet matchedAnns= (gate.AnnotationSet) bindings.get("isDisease");
    gate.FeatureMap newFeatures= Factory.newFeatureMap();
    newFeatures.put("rule","Preprocess7");
    annotations.add(matchedAnns.firstNode(),matchedAnns.lastNode(),"disease", newFeatures);
}
```

Le prime righe di ciascuna grammatica costituiscono l'head e contengono il nome della fase (*preprocessing2*), gli input (*Lookup*), le modalità di match (*control*). L'input indica il *tipo* di dati, ottenuti da precedenti elaborazioni, su cui si vogliono eseguire delle azioni. Per il matching sono previste cinque modalità di esecuzione:

brill Se una o più regole individuano lo stesso match, sono tutte attivate contemporaneamente. Si avrà pertanto la creazione di più annotazioni all'interno di una stessa fase sulla medesima porzione di testo. Se più regole trovano match con uguale posizione iniziale ma lunghezza diversa, la ricerca di match proseguirà a partire dalla posizione in cui finisce il match più lungo.

all È simile a brill ma la regola prosegue la ricerca di match a partire dalla fine del match corrente.

first Appena una regola trova un match si attiva senza curarsi di un eventuale match più lungo

once Appena una regola è stata attivata, l'intera fase di JAPE si conclude dopo il primo match

appelt Solo una regola può essere attivata nella stessa regione di testo, secondo precise regole di priorità:

- tra tutte le regole che trovano un match con posizione iniziale identica, si attiva solo quella che corrisponde al match più lungo
- se una o più regole trovano un match su una stessa porzione di documento, si attiva quella con priorità più alta
- se c'è più di una regola con la stessa priorità, si attiva quella definita per prima

Con riferimento alla modalità *appelt*, si può dichiarare un parametro opzionale di priorità associato a ciascuna regola che di solito è un numero intero positivo. Un numero più alto rispetto ad un altro corrisponde ad una priorità maggiore. Se la

priorità non è dichiarata, per default tutte le regole hanno priorità -1.

La quarta riga dichiara il nome della prima regola: la presenza di più regole in una fase è consentita ma è bene fare attenzione all'ordine con cui sono scritte ed eseguite per evitare risultati inattesi.

Subito dopo aver specificato il nome della regola, si distinguono due parti: Left-Hand Side (LHS) e Right-Hand Side (RHS), separate dalla sequenza “-->”.

La LHS della regola specifica i pattern che devono essere trovati (matched) nel documento: nel caso considerato, cercherà tutti i *Lookup* il cui attributo sia uguale a *disease,* creando la variabile *isDisease*. Si noti che nella LHS si dichiarano come match da trovare annotazioni già esistenti ottenute da fasi precedenti a quella in cui è presente la regola in corso di esecuzione. Tramite un'opportuna sintassi, si può specificare un match che tiene conto anche degli attributi associati all'annotazione e dei relativi valori.

Se un'annotazione viene specificata in una LHS ma per errore non viene indicata nell'input, la regola sarà ignorata. Ogni annotazione da trovare è racchiusa in parentesi graffe mentre il pattern da individuare è racchiuso in parentesi tonde e associato ad una variabile, in questo caso *SentDis*.
Spesso si è rivelato utile ricorrere a operatori contestuali:

```
(
   {Sentence contains disease}
):SentDis
```

La presente regola consente di individuare la frase all'interno del referto che si riferisce esplicitamente alla patologia. L'uso di questo operatore, tuttavia, si è dimostrato molto difficoltoso, anche perchè non bene documentato quando le condizioni da dichiarare erano più di una.

Quando non si è in grado di definire a priori il numero di occorrenze di un'annotazione da individuare, si può ricorrere agli operatori di unione e Kleene particolarmente utili e frequentemente usati:

|	or
*	Zero o più occorrenze
?	Zero o una occorrenza
+	Una o più occorrenze

Tutte le corrispondenze trovate sulla base delle specifiche fornite dall'utente nella LHS sono “passate” alla RHS tramite la variabile *isDisease* e “depositate” in un

oggetto Java di tipo AnnotationSet. Da questo set, si estraggono man mano i dati contenuti e ad ognuno si associa una nuova annotazione completa di eventuali attributi ad essa collegati (nel nostro caso crea l'annotazione *disease* con attributo *rule*). A volte è utile impostare come valore dell'attributo la stringa stessa cui si riferisce l'annotazione. In questo caso la RHS estrae i valori dell'offset iniziale e finale che identificano la posizione della stringa all'interno del documento, e copia nell'attributo i caratteri della stringa. L'esempio precedente si trasforma quindi come segue:

```
:isDisease
{   gate.AnnotationSet matchedAnns= (gate.AnnotationSet) bindings.get("isDisease");
       //estrai i pattern richiesti dalla LHS della regola

    int begOffset = matchedAnns.firstNode().getOffset().intValue();
    int endOffset = matchedAnns.lastNode().getOffset().intValue();
       //ricava il valore degli offset in base alla posizione della stringa nel testo

    String myDoc = doc.getContent().toString();
    String matchedString = myDoc.substring(begOffset, endOffset);
       //usa gli offset ottenuti in precedenza per estrarre la stringa di testo

    gate.FeatureMap newFeatures= Factory.newFeatureMap();
    newFeatures.put("valore", matchedString);
       //crea un nuovo attributo "valore" = "stringa"

    annotations.add(matchedAnns.firstNode(),matchedAnns.lastNode(),"disease", newFeatures); }
       //aggiungi un'annotazione "disease" con attributo "valore" uguale a "stringa"
```

In questa stessa fase sono state impostate con uguale modalità regole per ricavare le annotazioni *positive*, *negative*, *site* e *side*.

Il risultato ottenuto al termine di questa fase è il seguente:

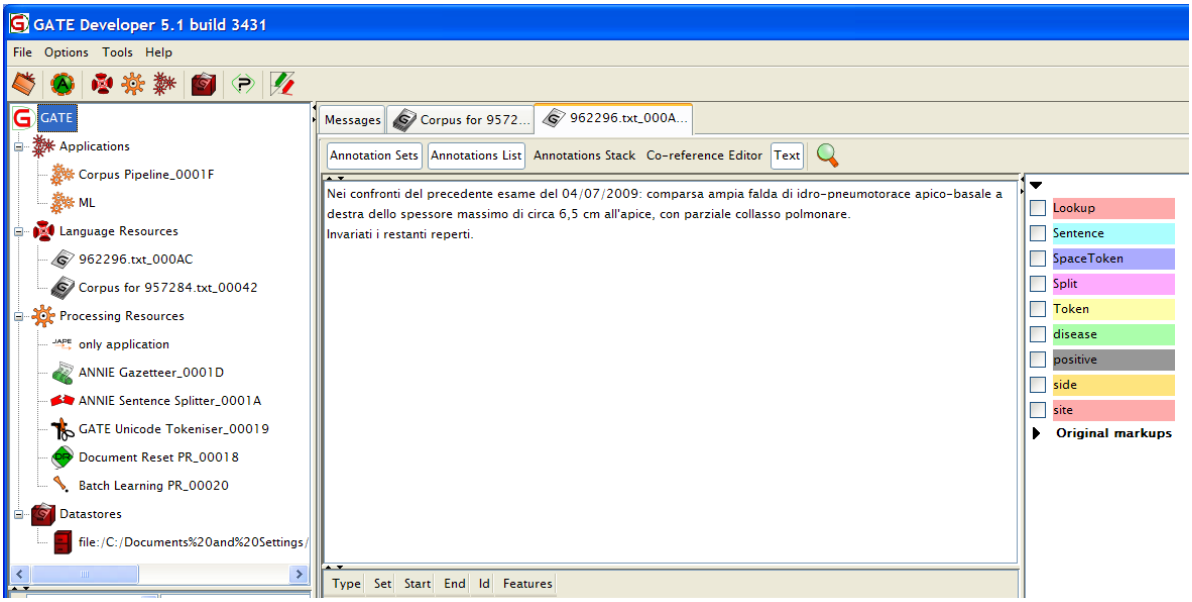


Figura 26 - individuazione degli attributi rilevanti

Sulla base poi delle motivazioni scaturite dall'analisi dei referti e già accennate, si ritiene utile impostare un'ulteriore regola per individuare la frase contenente l'annotazione *disease* e annotarla con *SentenceDisease*.

3.6 - CLASSIFICAZIONE

Per semplicità, si definiscono le seguenti classi:

Classe P: insieme dei referti che dichiarano la PRESENZA della patologia
Classe NP: insieme dei referti che dichiarano l'ASSENZA della patologia

Con le annotazioni a disposizione si è tentato di applicare una regola basata sulle seguenti assunzioni:

- tutti i referti hanno un'annotazione di tipo *disease*
  dunque tutti i referti hanno un'annotazione di tipo *SentenceDisease*
- se l'annotazione *SentenceDisease* contiene anche un'annotazione di tipo *negative*, allora quel referto dichiara l'assenza della patologia
- se la precedente condizione non è verificata, allora il referto dichiara la presenza della malattia.

Esso si basa sulla caratteristica precedentemente accennata dei referti: essere costituiti principalmente da frasi "brevi" e normalmente riferite a singoli concetti, come può essere appunto la presenza o meno della patologia.

Com'era naturale attendersi, i risultati non sono stati soddisfacenti ma l'esperimento è stato utile per individuare quali referti presentano maggiore difficoltà di classificazione e la casistica che fa crollare il ragionamento.

Il primo esempio è questo:

Evidente falda di pneumotorace apicale; non versamento pleurico

Utilizzando le PR finora descritte, GATE individua le seguenti annotazioni:

- *disease* ("falda di pneumotorace");
- *negative* ("non")
- *SentenceDisease* (in questo caso la frase completa);

ed assegna il referto alla *Classe NP* poiché *SentenceDisease* contiene *negative*. Naturalmente ciò non corrisponde a verità, perché *negative* fa riferimento ad un altro concetto, non alla patologia. Una possibile soluzione potrebbe essere quella di suddividere la frase in due parti: la prima conterrebbe il testo che precede la punteggiatura ";" mentre la seconda conterrebbe il testo rimanente. In questo

modo *SentenceDisease* sarebbe assegnata solo alla prima parte. Tale approccio, tuttavia, è comunque rischioso e troppo dipendente dal testo pertanto non si procede a raffinare la regola.

Un altro caso particolare si ha con il seguente referto:

> *Presenza penumotorace iperteso a destra con collasso polmonare e verosimile presenza di enfisema mediastinico.*

Il termine riferito alla patologia è scritto in modo sbagliato: "penumotorace". In questo caso la regola non riesce ad individuare *disease*, non crea *SentenceDisease*, quindi non assegna una classe al referto. Questa situazione non è di facile soluzione in quanto, da un punto di vista pratico, non è pensabile prevedere a priori liste del Gazetteer con i più frequenti errori di battitura e, tramite inferenza, non si riuscirebbe a desumere lo pneumotorace da altri fattori presenti nel referto (come potrebbe accadere per altre patologie).

Altro caso non raro è rappresentato dalla seguente tipologia di referto che possiamo immaginare divisa in due parti separate dal simbolo ";":

> *...non più apprezzabile la falda di PNX apicale bilateralmente; comparsa minima falda di pneumotorace al segmento basale anteriore del lobo inferiore di destra."*

In questo caso i problemi sono due: il primo riguarda la dichiarazione di scomparsa della patologia bilateralmente e l'insorgenza della stessa a destra. Con le regole approntate, per far sì che ad ogni referto sia attribuita una classe in maniera esclusiva, la frase contenente l'annotazione *negative* ha la precedenza e quindi l'intero referto viene assegnato alla classe NP. Il che, dal punto di vista del paziente, non è corretto.
Il secondo problema, più generale, riguarda l'uso dei sinonimi. Si supponga che il referto fosse stato questo:

> *...non più apprezzabile la falda di pneumotorace apicale bilateralmente; comparsa minima falda di PNX al segmento basale anteriore del lobo inferiore di destra."*

(in pratica sono state invertiti i termini "PNX" e "pneumotorace"). Se non fosse stato fornito l'elenco dei termini equivalenti riferiti alla patologia la seconda parte del referto non sarebbe stata considerata significativa; in tal caso, il referto sarebbe stato assegnato alla classe NP mentre è evidente che quella corretta è la classe P.

Per superare queste difficoltà, o perlomeno, per tentare di ottenere buoni risultati senza basarsi troppo sulle caratteristiche dei singoli referti, si è deciso di utilizzare algoritmi di Machine Learning.

3.7 - MACHINE LEARNING IN GATE

GATE offre la possibilità di sperimentare il machine learning utilizzando le annotazioni "su misura" create dall'utente. Riferendosi alla terminologia del Machine Learning, occorre definire una corrispondenza che lega i concetti di quest'ultima e i concetti di GATE.

	Machine Learning	**Interpretazione di GATE**
Istanza	*Esempio del fenomeno studiato.* L'algoritmo di ML apprende un modello da istanze note (TD) e lo applica a ulteriori istanze non note (AD).	*Annotazione.* Corrisponde ad un'annotazione creata in precedenza con JAPE. Il tipo di annotazione da scegliere come istanza è fondamentale e varia a seconda del tipo di elaborazione. Per i referti si userà un'annotazione che copre l'intero documento.
Attributo	*Caratteristica dell'istanza.* Ogni istanza è definita dai valori dei suoi attributi. Il set dei possibili attributi è ben definito ed è lo stesso per tutte le istanze del TD che del AD	*Valore attributo.* Valore di un particolare attributo associato ad una specifica annotazione che è in relazione con l'istanza considerata. L'annotazione può (parzialmente) coprire sia il tipo di annotazione usato come istanza oppure un altro tipo di annotazione correlata a quello considerata. Il valore dell'attributo può riferirsi all'istanza corrente od ad un'altra istanza situata sia in una specifica locazione rispetto a quella corrente oppure avente una relazione speciale con la corrente istanza.
Classe	*Attributo* Attributo per il quale il valore è disponibile nel TD in fase di learning, ma che non è presente nell'AD. Il ML è usato proprio per valorizzare questo attributo nell'AD.	*Attributo.* qualsiasi attributo riferito alla corrente istanza può essere contrassegnato come attributo di classe da valorizzare.

Convenzione: TD = training dataset; AD = application dataset

Per utilizzare questi algoritmi, sono state create con JAPE ulteriori annotazioni basate su queste regole:

- individua i *Token* in *SentenceDisease* e riannotali in *TokenD,* in modo da distinguerli dai token appartenenti ad altre frasi.
- se le annotazioni *negative e positive* appartengono a *SentenceDisease* riannotale in *negativeD* e *positiveD* per distinguerle da quelle esterne a *SentenceDisease*
- crea l'annotazione *myFindD* che copre l'intero referto e che funzionerà da istanza. A questa annotazione è associato l'attributo *class* da valorizzare tramite l'algoritmo di ML.

Un referto a fine elaborazione presenterà dunque le annotazioni di seguito evidenziate:

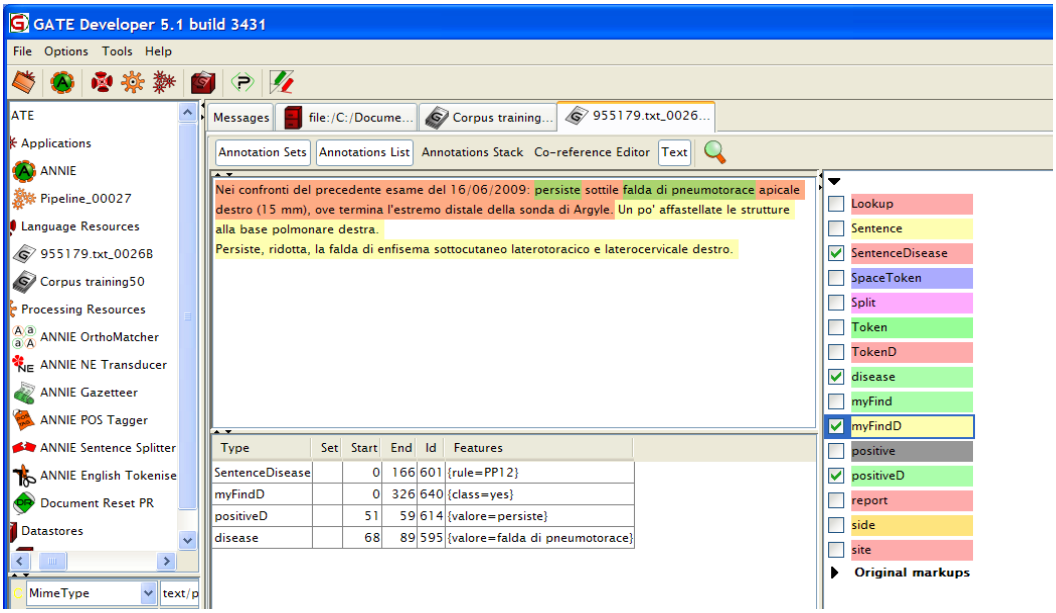


Figura 27 - annotazioni usate per ML

3.7.1 - Preparazione corpora

Gli algoritmi di ML richiedono due corpus su cui sono applicare due fasi distinte di funzionamento:

- *Training:* relativa alla costruzione di un modello basato su dataset di istanze già classificate
- *Application:* utilizzo del modello risultante dal training per classificare nuove istanze

I dataset sono stati preparati utilizzando Corpus Pipeline (selezionabile tra le Application Resources) che consentono di definire in sequenza le PR da applicare ai corpus, specificando per ciascuna eventuali file di configurazione.

Nello specifico, i dataset sono stati elaborati con GATE Unicode Tokeniser, ANNIE Sentence Splitter, ANNIE Gazetteer (specificando il file *list.def)* e JAPE Transducer

(specificando il file *main.jape)*. Nel file *main.jape* usato per la codifica del training set è presente la fase che crea l'annotazione *myFindD* sull'intero referto con attributo *class;* manualmente, poi, tale attributo è stato valorizzato (con *yes* o *no*) creando così il *gold standard*, ossia un dataset con istanze correttamente classificate. I referti dell'application set, invece, sono elaborati senza creare l'annotazione *myFindD*.

I corpus così preparati sono poi memorizzati nei rispettivi datastore.

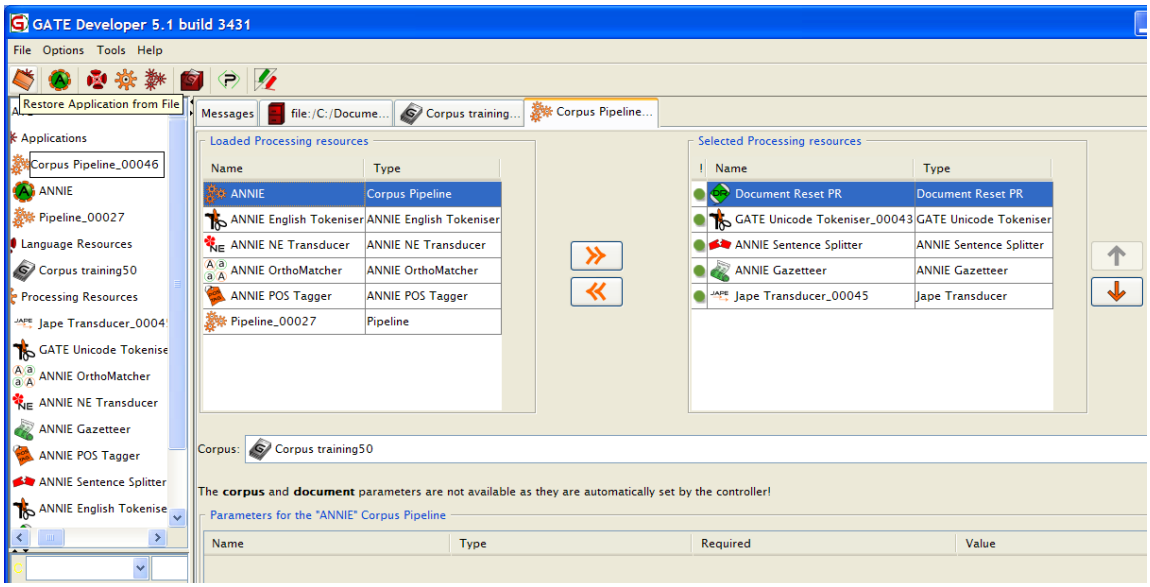


Figura 28: Corpus Pipeline (sezione di destra)

3.7.2 - Configurazione file

Ciascun algoritmo di Machine Learning richiede un file di configurazione in XML dove si possono distinguere due sezioni principali: la prima riguarda i parametri specifici dell'algoritmo, la seconda invece è uguale per tutti e riguarda la scelta degli attributi con cui costruire il modello di learning.

Questa seconda parte richiede la massima attenzione: è qui che si decide quali sono le istanze, le annotazioni e gli attributi che si considerano essenziali per una corretta classificazione dei documenti. Purtroppo, l'individuazione di questi elementi rappresenta la fase più incerta e difficile del Machine Learning, pertanto spesso si procede per tentativi cercando di migliorare man mano i risultati ottenuti. Nella classificazione dei referti, si è scelto di selezionare solo le annotazioni *TokenD, positiveD, negativeD* e *myFindD* secondo la tabella indicata:

Parametro	**Annotazione**	**Attributo**
Istanza	*myFindD*	
Attributo 1	*TokenD*	*valore*

Attributo 2	*negativeD*	*valore*
Attributo 3	*positiveD*	*valore*
Attributo 4	*disease*	*valore*
Classe	*myFindD*	*class*

Trattandosi di classificazione di documenti, l'istanza è rappresentata dal referto stesso; il primo attributo serve per considerare tutti i token appartenenti alla *SentenceDisease;* il secondo, il terzo e quarto attributo segnalano la presenza di attributi ritenuti significativi; il quinto attributo specifica la classe il cui valore sarà appreso nel training set e generato nell'application set.

3.7.3 - Fasi di training e application

In fase preliminare, gli algoritmi di ML sono stati utilizzati su corpora ridotti e precisamente su un training set composto da 50 referti e su un'application set da 30 referti, rispettando la proporzione di 2/3 e 1/3 suggerita dalla documentazione e dalla letteratura. Ogni algoritmo è stato prima eseguito sul training set, poi sull'application set fornendo ogni volta un nome diverso allo schema di annotazioni in output. In questo modo, su uno stesso documento dell'application dataset si possono confrontare i valori attribuiti a *class* dai diversi algoritmi.

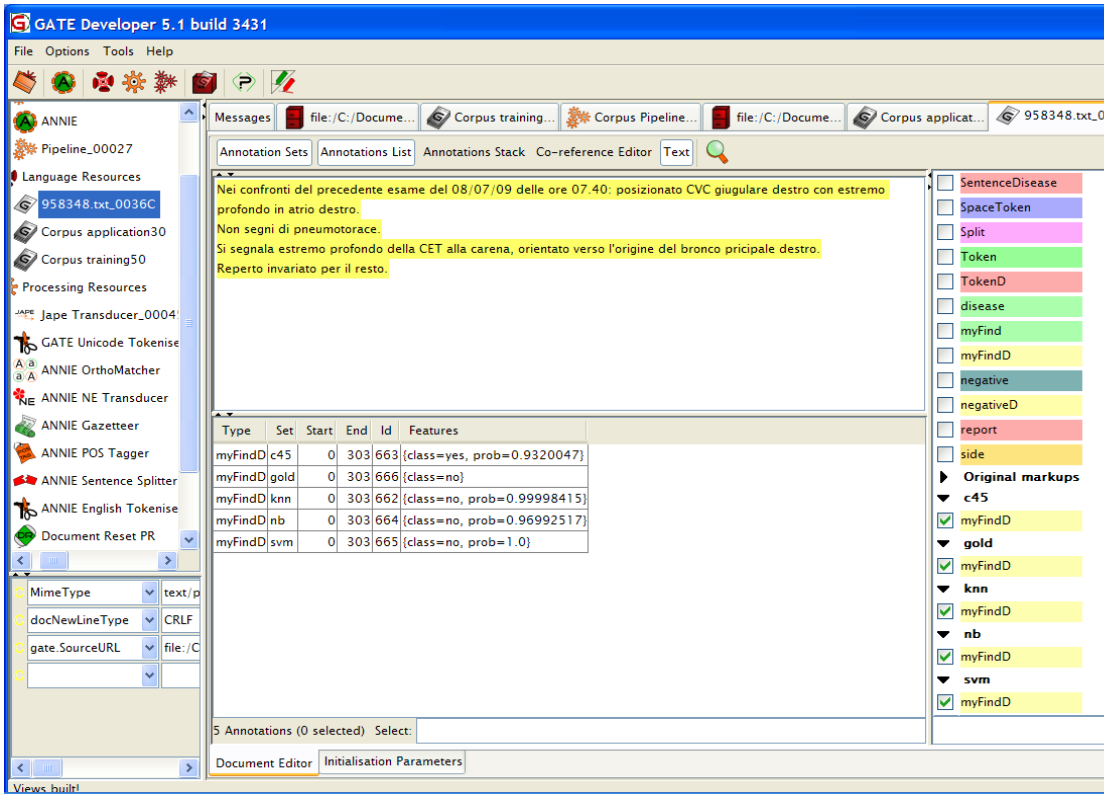


Figura 29: annotation set di tutti gli algoritmi di ML

Osservando l'immagine è possibile notare sulla destra l'elenco dei nomi attribuiti in

output dalle varie elaborazioni:

gold: corrisponde all'annotazione manuale
c45 corrisponde all'esecuzione dell'algoritmo C4.5 implementato in Weka
knn corrisponde all'esecuzione dell'algoritmo kNN con k = 5 implem. in Weka
nb corrisponde all'esecuzione dell'algoritmo Naive Bayes implem. in Weka
svm corrisponde all'esecuzione dell'algoritmo SVM implementato in Java

Nell'esempio considerato si può notare come tutti gli algoritmi abbiano attribuito il giusto valore a *class* tranne C4.5Weka. Tuttavia sarebbe troppo laborioso aprire ciascun documento e verificare i risultati ottenuti; per una valutazione generale delle prestazioni si ricorre ad un'altra funzionalità di GATE: il Corpus Assurance Quality.

3.8 – METRICHE DI VALUTAZIONE

Quando si desidera valutare la perfomance di una PR come Tokeniser, POS tagger o di un'intera applicazione, normalmente si effettua il confronto tra il risultato ottenuto dal software e quello che viene definito un “gold standard”, ossia il corpus annotato manualmente da un esperto. Tuttavia non sempre è facile e ovvio come dovrebbe essere il gold standard, nel senso che persone diverse potrebbero avere opinioni diverse su cosa sia corretto o meno.

Nel caso specifico, questo problema di ambiguità è superabile in quanto si tratta di decidere semplicemente a quale categoria appartiene un referto. Per misurare la bontà di un classifier su un intero dataset di dati, GATE offre la possibilità di confrontare le annotazioni ottenute con due sistemi diversi: il primo sarà sempre lo schema di annotazione *gold* (quello manuale e quindi corretto) e il secondo quello ottenuto dagli algoritmi di ML.

In Information Retrieval i criteri di valutazione per riflettere l'efficacia e l'efficienza di un sistema in termini quantitativi si basano tradizionalmente su:

Precision: n° documenti rilevanti reperiti rispetto al totale documenti reperiti
Recall: n° documenti rilevanti reperiti rispetto al totale documenti rilevanti
F-measure. media pesata di precision e recall: 2 (P x R) / (P+R)

Se anziché documenti si confrontano annotazioni, non si parla più di documenti rilevanti e reperiti, ma si considera una casistica più complessa che tiene conto sia del *tipo* che dello *span* dell'annotazione. Se si considera *key* un'annotazione di riferimento e *response* un'annotazione di cui si vuole valutare la corrispondenza, esse saranno definite:

correct: quando *tipi* e *span* coincidono rispettivamente

partial: quando i *tipi* coincidono mentre gli *span* si sovrappongono senza essere identici

missing: si applica solo a *key* e quando gli *span* non coincidono e non si sovrappongono oppure quando uno o più dei suoi attributi non sono presenti in *response* o hanno valore diverso

spurious: si applica solo a *response* e quando gli *span* non coincidono e non si sovrappongono oppure quando uno o più dei suoi attributi non sono presenti in *key* o hanno valore diverso

Precision e *Recall* si trasformano allora come segue:

$$\text{Precision} = \frac{\text{Correct} + \frac{1}{2}\,\text{partial}}{\text{Correct} + \text{spurios} + \text{partial}}$$

$$\text{Recall} = \frac{\text{Correct} + \frac{1}{2}\,\text{partial}}{\text{Correct} + \text{missing} + \text{partial}}$$

$$\text{F-Measure} = \frac{(b^2 + 1)\,(P \times R)}{(b^2\,R) + P}$$

Impostando b=1 e tenendo conto che l'annotazione *myFindD* su cui si effettua la valutazione ha uno *span* esteso sull'intero documento e che l'attributo *class* è unico ed è sempre presente sia nella *key* che nella *response,* le annotazioni *partial* spariscono, pertanto i rapporti precedenti si semplificano in:

$$\text{Precision} = \frac{\text{Correct}}{\text{Correct} + \text{spurios}}$$

$$\text{Recall} = \frac{\text{Correct}}{\text{Correct} + \text{missing}}$$

$$\text{F-Measure} = \frac{2\,(P \times R)}{R + P}$$

Nei problemi di classicazione, è utile ragionare in termini di *observed agreement* che si basa su una tabella di contingenza così definita quando vi sono due sole categorie (o classi):

	Annotator 2		
Annotator 1	*cat-1*	*cat-2*	Marginal sum
cat-1	a	b	**a + b**
cat-2	c	d	**c + d**
Marginal sum	**a + c**	**b + d**	**a+ b + c + d**

a = documenti assegnati a cat-1 da entrambi gli annotator
b = documenti assegnati a cat-1 da Ann-1 e a cat-2 da Ann-2
c = documenti assegnati a cat-2 da Ann-1 e a cat-1 da Ann-1
d = documenti assegnati a cat-2 da entrambi gli annotator

L'*observed agreement* è dato quindi dal rapporto:

$$O = \frac{(a+d)}{(a + b + c + d)}$$

e misura quanto il classificatore di ML si è avvicinato al *gold*.

Tuttavia, per la particolare tipologia del problema affrontato e per le semplificazioni che esso introduce in questi parametri di misura, è facile osservare come i valori di fatto si riducano per tutte le metriche ad un rapporto dato dal numero di match sul totale dei documenti considerati.

3.9 - CORPUS QUALITY ASSURANCE

Corpus Quality Assurance è una funzionalità di GATE che consente di valutare i risultati ottenuti in termini di precision, recall e F1-measure tra due set di annotazioni presenti in un corpus.

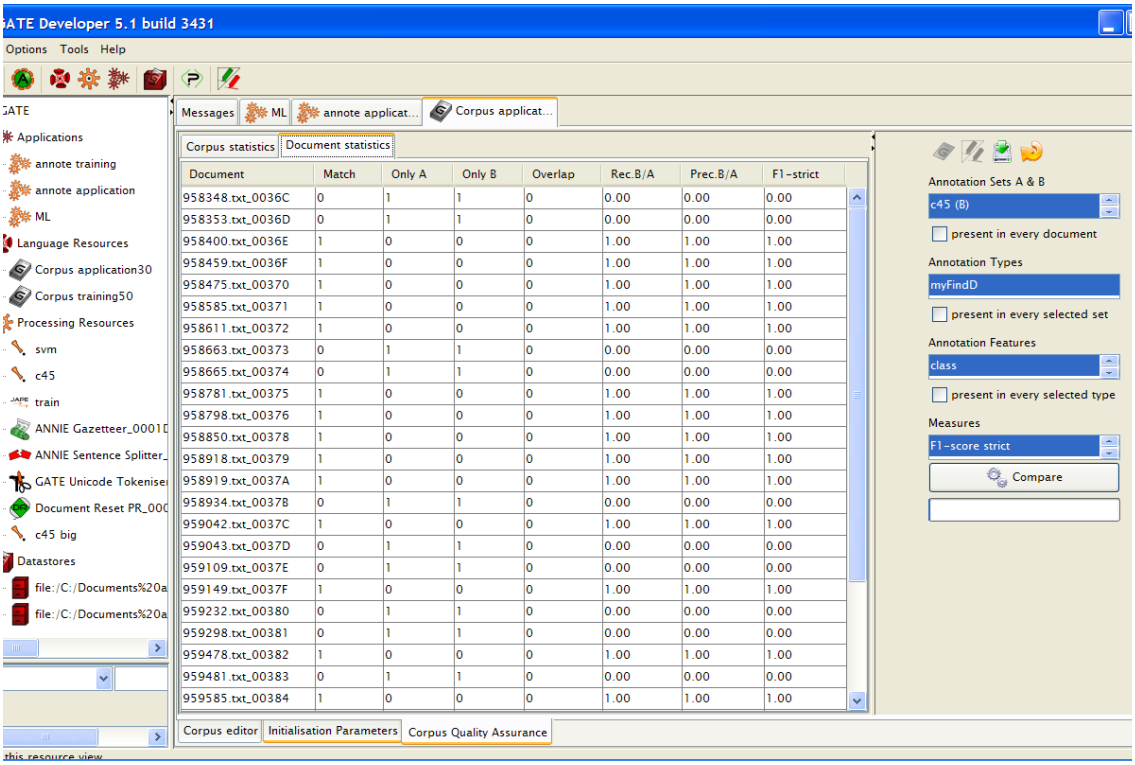


Figura 30 – Corpus Quality Assurance

Come evidenziato in figura, sulla destra sono posizionate alcune caselle con le quali si selezionano i seguenti parametri:

- Annotation set A: *gold* corrisponde al gold standard e rimane fisso
- Annotation set B: *c4.5* corrisponde in questo caso al set di C4.5 ma sarà di volta in volta modificato con gli altri set di ML.
- Annotation Type: *myFindD* annotazione che contiene l'attributo *class*
- Annotation Features: *class* il suo valore decreta l'esito della classificazione.

Cliccando sul tasto *Compare*, GATE esegue il confronto tra i due set di annotazioni: se la feature *class* ha valori differenti sui due set di annotazioni in corrispondenza del medesimo documento, allora l'algoritmo di ML ha attribuito la classe sbagliata. Le colonne che compaiono a video mostrano:

- *colonna 1*: nome del documento
- *colonna 2*: numero di match riscontrati tra i due annotation set
- *colonna 3*: numero di annotazioni presenti solo sull'Annotation set A
- *colonna 4*: numero di annotazioni presenti solo sull'Annotation set B
- *colonna 5*: numero di overlap, ossia di sovrapposizioni tra annotazioni
- *colonna 6*: valore di Recall
- *colonna 7*: valore di Precision
- *colonna 8*: valore di F1-measure.

Considerando la particolare tipologia di problema affrontato, valgono le seguenti semplificazioni:

- *colonna 3 e colonna 4* avranno sempre uguale valore che può essere:
  0 - se il valore di *class* sulle due annotazioni è concorde (match)
  1 - se il valore di *class* sulle due annotazioni è discorde (mismatch)
  n - conta il totale delle discordanze sull'intero corpus
  In caso di mismatch il valore è *1* perchè i due set hanno ciascuno un'unica annotazione.
- *colonna 5* avrà sempre valore zero; poiché l'annotazione *myFindD* copre tutto il documento e non solo una sua parte, non potrà mai esserci parziale sovrapposizione tra le annotazioni.
- *colonne 6, 7 e 8*, pur riferendosi a misure diverse, avranno anch'esse valore uguale, proprio per via dei particolari valori inseriti nei rispettivi rapporti che le definiscono.
- *F1-score strict, F1-score lenient e F1-score average* avranno anch'esse lo stesso valore in quanto non esistono annotazioni “parzialmente corrette”. Questa misura pur essere pensata come il rapporto tra il numero di match e il totale delle istanze: più alto è il valore, migliore è la classificazione.

Nelle considerazioni che seguono, la valutazione degli algoritmi non deve intendersi assoluta ma relativa, nel senso che si tratta di prestazioni riferite ad un particolare dataset, con una specifica scelta di attributi e in base a precisi parametri di configurazione.

3.9.1 Training set da 50 referti – Application set da 30 referti

Il primo test, condotto su corpus ridotti, ha ottenuto i seguenti risultati:

RISULTATI

	REFERTO N.	REALI		c45		knn		nb		svm	
		sì	no	sì	no	sì	no	sì	no	sì	no
51	348		x	E			x		x		x
52	353		x	E			x		x		x
53	400	x		x		x		x		x	
54	459	x		x		x		x		x	
55	475	x		x		x		x		x	
56	585	x		x		x		x		x	
57	611	x		x		x		x		x	
58	663		x	E			x		x		x
59	665		x	E		E		E			x
60	781	x		x		x		x			E
61	798	x		x		x		x		x	
62	840	x		x		x		x		x	
63	850	x		x		x		x		x	
64	918	x		x		x		x		x	
65	919	x		x		x		x		x	
66	934		x	E		E		E		E	
67	959 042	x		x		x		x		x	
68	043		x	E			x	E			x
69	109		x	E			x	E			x
70	149	x		x		x		x		x	
71	232		x	E			x		x		x
72	298		x	E			x	E			x
73	478	x		x		x		x		x	
74	481		x	E		E		E			x
75	585	x		x		x		x		x	
76	624	x		x		x		x		x	
77	627	x		x		x		x		x	
78	831		x	E			x	E			x
79	853	x		x		x		x		x	
80	864	x		x		x		x		x	

Con Corpus Quality Assurance sono stati ricavati i valori riportati in tabella:

	C4.5	**kNN (k=3)**	**KNN (k=5)**	**Naive Bayes**	**SVM**
Match	19	27	27	23	28
Only A/B	11	3	3	7	2
F1-score	0.62	0.90	0.90	0.77	0.93

Considerazioni: SVM e KNN ottengono le prestazioni migliori con un F1-score che è pari o superiore a 0.90; Naive Bayes e C4.5 invece evidenziano un coefficiente molto basso, con un rilevante numero di mismatch. Bisogna tener conto, tuttavia, che il training set in base al quale hanno costruito il proprio modello, pur rispettando la proporzione di 2/3 del totale corpus, è particolarmente ridotto, pertanto non è da escludere che, aumentando la dimensione del training, le prestazioni possano migliorare.
Osservando nel dettaglio il listato dei risultati si notano alcuni esempi curiosi:

- SVM è l'unico algoritmo che classifica correttamente i referti 958665 e 959481; tuttavia si sbaglia clamorosamente sul referto 958781, classificato invece correttamente dagli altri algoritmi, compresi quelli meno accurati.
- Tutti gli algoritmi si sbagliano sul referto 958934[70] sebbene sembri uno dei più semplici da classificare.
- Modificare il parametro di kNN da *k=3* a *k=5* non ha prodotto alcun miglioramento
- Un mismatch di Naive Bayes corrisponde ad un mismatch di C4.5 (ma non viceversa): in pratica i referti su cui sbaglia Naive Bayes sono un sottoinsieme dei referti su cui sbaglia C4.5

3.9.2 – Training set da 140 referti – Application set da 73 referti

Si riesegue ora il test sull'intero set di 213 referti, mantenendo la proporzione di 2/3 per il training set e di 1/3 per l'application set. Dato che la dimensione dei corpora è sensibilmente aumentata rispetto a prima, si tiene conto anche dei tempi di esecuzione del learning e della classificazione. Il Corpus Quality Assurance restituisce i seguenti valori:

	C4.5	**KNN (k=3)**	**KNN (k=5)**	**Naive Bayes**	**SVM**
Tempi di training (sec.)	14.875	12.562	15.734	13.484	12.297
Tempi di application (sec.)	13.703	12.797	13.109	12.703	13.125
Match	62	67	71	56	73
Only A/B	11	6	2	17	0
F1-score	0.85	0.92	0.96	0.77	1

Considerazioni: SVM si rivela il miglior classificatore su questo corpus con una prestazione eccellente, classificando correttamente tutti i 73 referti. Anche kNN si è dimostrato performante con un F1-score al di sopra di 0.90; in particolare, con

70 "Nei confronti del precedente esame del 10/07/2009 delle ore 15:32 eseguito in altra sede, non piu' evidente la falda di pneumotorace basale a destra; a tale livello posizionata sonda di Argyle."

k=5 ha sbagliato solo due referti, ottenendo un valore pari a 0.97. L'aumento dimensionale del dataset ha sensibilmente migliorato i risultati di C4.5 che ha raggiunto un F1-score pari a 0.85 e ha superato in accuratezza il Naive Bayes; tuttavia rimane ancora sotto la soglia di 0.90. Non bene invece il Naive Bayes che ha riportato esattamente lo stesso valore di F1-score, senza quindi trarre profitto in fase di apprendimento, dal maggior numero di istanze.

Anche in questo caso si notano alcune curiosità:

- SVM, oltre ad aver ottenuto punteggio pieno, ha eseguito il training in minor tempo rispetto agli altri.
- KNN con k=5 ha avuto il tempo di training più lungo, addirittura più di C4.5, e un tempo di application nella media. Il vantaggio è l'accuratezza raggiunta, ben superiore al Naive Bayes con il miglior tempo di application.
- KNN con k=5 ha sbagliato solo due referti: 963928 e 974741. Mentre il secondo presenta difficoltà di classificazione poiché solo SVM è riuscito ad assegnargli la giusta classe, il primo sembra "abbastanza facile" da classificare in quanto anche il Naive Bayes, che in questo test ha ottenuto le prestazioni peggiori, è riuscito a classificarlo correttamente.
- Anche C4.5 ha assegnato la classe sbagliata al referto 963928, sebbene in generale abbia prestazioni superiori a Naive Bayes. Tenendo conto della considerazione fatta nel test precedente riguardo alla tendenza degli algoritmi di sbagliare sugli stessi referti, questo è uno dei rari casi in cui un referto con classificazione sbagliata non appartiene al sottinsieme di referti erroneamente classificati dall'algoritmo con peggiori prestazioni.
- Il Naive Bayes è un algoritmo relativamente veloce rispetto agli altri ma i risultati raggiunti sono scarsi. Il corpus più esteso non ha contribuito a maggiore precisione, tanto che il valore di F1-score è rimasto invariato.

3.9.3 – Training set da 100 referti – Application set da 113 referti

Dato che SVM nel precedente test ha ottenuto punteggio pieno, si vuole verificare se una diversa ripartizione del corpus suddiviso in circa 1/2 + 1/2 ha effetti sul risultato finale e in che misura. Ragionando su corpus molto vasti, infatti, la dimensione del training diventa un fattore cruciale. Dopo aver eseguito ripetutamente learning ed application variando l'algortimo di ML, il Corpus Quality Assurance ha restituito i seguenti valori:

	C4.5	KNN (k=3)	KNN (k=5)	Naive Bayes	SVM
Tempi di training (sec.)	9.812	8.435	8.594	8.438	8.578
Tempi di application (sec.)	21.125	19.975	20.188	12.703	19.860
Match	88	105	102	85	111
Only A/B	25	8	11	28	2
F1-score	0.78	0.93	0.90	0.75	0.98

Considerazioni. Osservando i tempi si nota subito un fatto evidente: a parità di dimensione dei set, circa 100 per entrambi, i tempi di application sono in pratica il doppio. Tra questi, risalta quello di C4.5 che presenta anche il tempo più lungo di training. Le prestazioni invece degli algoritmi hanno subito variazioni, come era naturale attendersi, rispetto al precedente test rilevabili dal valore di F1-score: C4.5 ha perso 0.07, KNN (k=5) ha perso 0.07, Naive Bayes ha perso 0.02 e SVM ha perso 0.02. In controtendenza KNN (k=3) che ha guadagnato 0.01. In pratica la riduzione del corpus si è fatta sentire maggiormente su C4.5 e KNN (k=5), ha diminuito di poco l'accuratezza di SVM, mentre ha lasciato quasi indifferente Naive Bayes. L'analisi dei singoli referti rivela altre particolarità:

- ancora una volta SVM sbaglia la classe su un referto che invece è correttamente classificato dagli altri algoritmi (referto 961111). Mantiene comunque un'accuratezza molto alta.
- Sugli 11 referti sbagliati da KNN (k=5), 5 sono correttamente classificati dagli altri algoritmi, compresi C45 e Naive Bayes con rendimento ben più basso.
- Il miglioramento ottenuto da KNN (k=3) non è in sé particolarmente significativo; conta tuttavia il fatto che ha ottenuto prestazioni simili al test precedente costruendo il proprio modello di learning su una quantità di dati ben inferiore.

I test potrebbero proseguire ulteriormente cambiando le dimensioni dei dataset, modificando gli attributi specificati nel file di configurazione degli algoritmi di ML o, infine, creando nuove annotazioni basate sull'intento di estrarre ulteriore informazione dai referti. Probabilmente un esperto di Machine Learning potrebbe fornire una spiegazione di questi comportamenti e suggerire opportuni miglioramenti nell'impostazione dei parametri per ottenere risultati più performanti. Tuttavia, scopo di questa tesi non è approfondire il funzionamento degli algoritmi di Machine Learning ma capire l'utilità che se ne può trarre, pur senza conoscerne a fondo il funzionamento, impostando parametri non banali e verificando la perfomance di tali algoritmi sul corpus a disposizione.

Inoltre, come già accennato, non esiste un algoritmo di classificazione perfetto: si tratta di trovare la giusta combinazione (algoritmo + parametri + attributi) che consenta di ottenere risultati soddisfacenti che, nel caso considerato, si basano sull'accuratezza della classificazione ma in altri contesti possono tenere conto della velocità di elaborazione su corpus molto estesi, dello spazio di memoria richiesto nella fase di running, della robustezza dell'algoritmo in mancanza di determinati valori, della gestione dell'overfitting, della facilità d'uso, della facilità di interpretazione del modello di learning ottenuto, e cosi via.

3.10 - CONSIDERAZIONI SULL'USO DI GATE

La scelta di GATE è già stata motivata: una presentazione dell'applicativo ben fatta, un sito internet essenziale nella forma ma ricco nei contenuti, un'impressione generale positiva e, soprattutto, una documentazione corposa e facilmente consultabile, hanno dirottato la scelta verso questo open source. Purtroppo il suo utilizzo è stato fin dall'inizio difficoltoso: la grafica è a volte “troppo” semplice nel senso che non consente un uso intuitivo dell'applicativo ma richiede una lettura attenta e rigorosa della documentazione.

Altra nota dolente è proprio la documentazione: in alcuni punti è chiara e ben comprensibile, in altri fornisce informazioni essenziali, senza esempi a supporto o appena accennati. Capita allora di dedicare molto tempo, per non dire troppo, a tentativi frustranti per cercare di far funzionare una risorsa secondo le esigenze dell'utente. Per esempio, alcune istruzioni JAPE indicate nella documentazione e descritte con un esempio, funzionano bene con una singola condizione, ma se l'utente ha necessità di specificarne più di una secondo le regole da loro stessi definite, rischia di dedicare ore in innumerevoli tentativi pensando si tratti di un errore di sintassi e non invece di un preciso vincolo non dichiarato. Quando poi scopre che è un problema noto e molti altri hanno avuto la stessa difficoltà, si auspica che la documentazione venga integrata quanto prima. I moduli di ANNIE (tokeniser, gazetteer, sentence splitter....) sono ben documentati ma solo ricorrendo ad un tutorial non incluso nella guida si è potuto ordinarli nella corretta sequenza. Molto scomoda si è rivelata la scrittura delle regole in JAPE: trattandosi di testo scritto con WordPad o simili, non prevede un editor specifico né un debug che aiutasse nell'individuazione degli errori.

Un'altra funzionalità molto difficoltosa da utilizzare è stato il machine learning: per chi non ha mai utilizzato in precedenza questi strumenti, non è semplice capire come funzionano, quali e quanti attributi indicare, i parametri da impostare nel

file di configurazione, le modalità di running. La documentazione sembra chiara al riguardo, perchè fornisce anche qualche esempio, ma quando poi si effettuano i primi tentativi, ci si rende conto che non è così semplice ed immediato come sembra.

Una volta acquisita pratica, l'applicativo si rivela un buon strumento di NLP e un suo punto forte è sicuramente la ben dotata libreria di plugin. Sarebbe interessante provare in futuro prodotti alternativi per confrontarne la facilità d'uso e le funzionalità offerte. L'esperienza comunque ha dato la possibilità di approfondire argomenti mai affrontati in corsi universitari e, da un punto di vista personale, molto interessanti.

Per mancanza di tempo, in parte dovuta proprio alle difficoltà incontrate nell'uso dell'applicativo, non si è potuta approfondire la funzionalità relativa alle ontologie: GATE offre la possibilità di crearle e uno sviluppo futuro potrebbe essere proprio un'elaborazione dei referti tramite le ontologie. Nel settore medico, e in particolare con riferimento ai report radiologici, esistono interessanti progetti che si basano su NLP e ontologie applicati al reporting radiologico: tra questi ne sono stati scelti due, per dare un'idea delle potenzialità offerte da questo nuovo modo di organizzare l'informazione.

3.11 - USO DI NLP PER TRADURRE INFORMAZIONI CLINICHE DA DATABASE[71]

Il primo interessante studio è stato condotto dai dipartimenti di Radiologia e Informatica Medica della Columbia University di NY nel 2002. Scopo della ricerca era valutare la "traduzione" di report relativi a radiografie polmonari usando tecniche di natural language processing.

La pratica medica genera un'abbondante quantità di dati clinici in forma narrativa, ma la mancanza di standardizzazione ne ostacola l'uso per analisi aggregate da parte di sistemi automatizzati real-time. In questa circostanza il NLP è di aiuto in quanto può trasformare, almeno in parte, il contenuto dei referti in testo strutturato: il processo di codifica dei referti radiografici ottenuto con queste tecniche può raggiungere livelli di accuratezza comparabili a quelli umani, come dimostrato in studi precedenti[72,73].

71 G. Hripcsak, J. Austin, P. Alderson, C. Friedman – *Use of Natural Language Processing to translate clinical information fron a database of 889,921 chest radiographic reports* – Radiology 2002; 224: 157-163

72 G. Hripcsak, C. Friedman, P. Alderson e altri – *Unlocking clinical data from narrative reports: a study of NLP* – Ann Intern Med 1995; 122: 681-688

73 M. Fiszman, WW Chapman, D. Aronsky, R. Evans, P. Haug – *Automatic detection of acute bacterial pneumonia from chest x-ray reports* – J. Am Med Inform Assoc 2000; 7: 793-604

La sfida raccolta riguardava la codifica di ben 889.921 referti radiologici scritti a mano o con l'aiuto di software (speech recognition) prodotti dal 1989 al 1998 e memorizzati in un repository. Come nell'esperimento descritto in questa tesi, sono stati considerati referti "omogenei" (escludendo quindi tecniche investigative diverse dalla radiografia) e "anonimi", ossia privati di ogni riferimento relativo all'identità del paziente.

Mediante l'utilizzo di MEDLEE, un software di NLP non commerciale sviluppato dalla Columbia University, l'informazione contenuta nei referti è stata codificata sulla base di un vocabolario controllato e strutturata in modo da poter eseguire delle inferenze su di essa.
Per verificare l'attendibilità del sistema ottenuto, gli autori hanno cercato di dimostrare "fatti noti" analizzando quanto ottenuto dalla codifica. I risultati hanno confermato, per esempio, la frequenza espressa nel rapporto 3:2 con cui il cancro al polmone colpisce il lato destro rispetto al lato sinistro[74]. Oppure hanno rispecchiato la frequenza con cui ricorrono certe condizioni cliniche correlate all'effusione pleurica. L'analisi nel tempo, poi, dei referti relativi a ferite da arma da fuoco e accoltellamento hanno mostrato una diminuzione in percentuale che corrispondeva alla diminuzione stimata della criminalità. Infine, con riferimento ai referti relativi allo pneumotorace, il grado di accuratezza ottenuto nella loro codifica sembra essere addirittura superiore a quello ottenuto manualmente usando un sistema di codifica ICD-9.

3.12 - RadiO[75]: APPLICATION ONTOLOGY

Il secondo progetto riguarda RadiO, un prototipo di application ontology che si inserisce in un'ottica di supporto al reporting radiologico. E' sviluppato in Protégé e comprende tre livelli:

1. *livello di report radiologico* che cattura le osservazioni fatte sull'esame cui si è sottoposto il paziente tramite l'uso di un vocabolario controllato di termini inerenti specificatamente le immagini radiografiche
2. *una ontologia* che rappresenta la conoscenza riferita alle entità presenti nelle immagini e alle loro caratteristiche
3. *una ontologia di riferimento anatomica (FMA)* che rappresenta la conoscenza anatomica canonica

74 J. Goldman e altri – *Term domain distribution analysis: a data mining tool for text databases* – Methods Inf Med 1999; 38: 96-101

75 D. Marwede, M. Fielding, T. Kahn – *RadiO: a prototype application ontology for radiology reporting tasks* – AMIA 2007 Symposium Proceedings Page 513 - 517

Scopo di questo prototipo è supportare l'identificazione di specifici attributi per ciascuna entità rilevante contenuta nell'immagine radiologica (*images features of image entities*) e il loro uso nella rappresentazione diagnostica, fornendo al contempo le basi per una applicazione di reporting strutturato nel dominio delle immagini mediche.

RadiO si interfaccia tra due ontologie già esistenti: Radlex (vocabolario controllato usato per il reporting delle immagini radiografiche) e FMA[76] (Foundational Model of Anatomy - ontologia di riferimento per l'anatomia). Nel dominio delle immagini mediche, essa ha lo scopo di costruire una base di conoscenza relativa alle conclusioni tratte dagli imaging findings e alla loro interpretazione come diagnosi.

Il valore diagnostico di un esame effettuato con tecniche ad immagini, infatti, dipende da due fattori: il tipo di patologia e se le caratteristiche della patologia sono facilmente desumibili dall'analisi dell'immagine stessa. Per esempio, lo pneumotorace è facile da osservare nelle immagini, mentre un carcinoma bronchiale può essere difficile da diagnosticare in quanto non presenta caratteristiche ben evidenti in un'immagine così come non presenta sempre le stesse caratteristiche per individuarlo. Inoltre, se da un lato gli strumenti di esame sono sempre più sofisticati e computerizzati, dall'altro il contenuto del report radiologico, che serve come base per comunicare i risultati della diagnosi, è abbastanza estraneo a tecnologie di gestione dell'informazione come le ontologie.

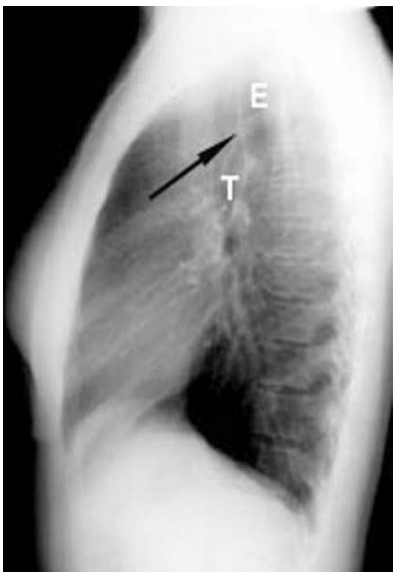
RadiO si basa su due tipi di ontologie:

- *reference ontology*: intesa come risorsa generica e riutilizzabile, disegnata per rispondere alla necessità di informazione strutturata richiesta da una qualche applicazione
- *application ontology*: accede alla reference ontology, è costruita per un particolare gruppo di utenti e serve per task specifici inerenti ad un particolare dominio.

Da un punto di vista pratico, il principale compito del radiologo, nella pratica clinica, è interpretare le immagini del corpo per fini diagnostici. Per ciascuna immagine, egli deve determinare se essa rappresenta una condizione normale o patologica per il paziente, documentando i segni associati. Se patologica, il radiologo deve riportare quali caratteristiche specifiche presenti nell'immagine evidenziano la diagnosi. Pertanto, le asserzioni fatte dal radiologo si riferiscono primariamente all'immagine stessa, e solo secondariamente agli organi rappresentati nelle immagini. Nella seguente immagine, per esempio, riconosciamo

76 http://sig.biostr.washington.edu/projects/fm/

quattro elementi essenziali:



- il lato posteriore della trachea
- il lato anteriore dell'esofago
- lo spazio tra di essi
- la vista laterale
  (poiché è solo da questa prospettiva che si nota la diversa conformazione delle due parti anatomiche)

Come detto prima, poiché non tutte le patologie sono evidenziabili tramite immagini, un organo può apparire normale in un'immagine ma non esserlo nella realtà. Nell'application ontology utilizzata per reporting di immagini è importante, quindi, separare il dominio delle entità del corpo da quello che descrive il modo in cui appaiono, domini che pur essendo correlati sono tuttavia distinti.

Lo studio condotto, al quale si rimanda per ogni approfondimento, mette in luce come nel settore radiologico, sebbene siano stati realizzati modelli per il contenuto dei report, non sia stata ancora sviluppata alcuna ontologia relativa alla diagnosi di malattia. E lo scopo di RadiO è proprio questo: costruire una conoscenza basata sulle caratteristiche delle parti anatomiche (così come appaiono nelle immagini) e una loro interpretazione per fini diagnostici.

3.13 - CONCLUSIONI

Questo studio è nato con lo scopo di sperimentare l'efficacia di un sistema di classificazione in grado di dividere un set di referti medici in due categorie: quella che dichiara la presenza di una determinata patologia e quella che ne dichiara l'assenza.

La difficoltà di tale realizzazione è motivata dalla tipologia dei referti stessi: essi sono stati estratti da un database e si presentano in forma narrativa, ossia scritti in linguaggio umano, usando una terminologia medica non standard e privi di struttura che li renda facilmente processabili.

L'idea di affrontare questa problematica nasce dall'esigenza di riuscire a manipolare i *dati,* sempre più numerosi, presenti in vasti repository e spesso strutturati in database caratterizzati da rigide modalità di accesso e di querying.

Dati che, in un preciso contesto e interpretati attraverso la conoscenza, si trasformano in *informazione*. L'ambito medico riflette bene questa situazione: la quantità di dati disponibili è considerevole, sia per quanto riguarda la letteratura medica sia a livello di dati diagnostici, e la classificazione, in particolare dei referti, è un modo per selezionare informazione e renderla poi utile a scopi di ricerca e come supporto alla diagnosi.

L'analisi dei referti ha evidenziato le problematiche relative allo stile con cui sono scritti, l'uso di una vocabolario non controllato e la mancanza di parole frequenti utili alla classificazione. Per tentare di superare queste difficoltà, si è ricorsi al Natural Language Processing, utile per estrarre, mediante analisi lessicale, sintattica e semantica, informazione ritenuta utile allo scopo prefissato. In questa fase è stato usato un applicativo open source che, pur non essendo predisposto per l'analisi di testi scritti in lingua italiana, si è rivelato utile per estrarre entità e relazioni sotto forma di annotazioni, aggiungendo così valore semantico agli elementi di un referto.

Le annotazioni così ottenute sono state utilizzate come parametri per il Machine Learning: sono stati messi a confronto vari algoritmi, basati su logiche differenti, per valutare la loro efficacia nella classificazione. Il risultato ottenuto impostando determinati parametri, attributi e vincoli sul dataset considerato, ha permesso di ottenere la corretta classificazione di tutti i referti, dimostrando le potenzialità di questo metodo per l'esecuzione di task più sofisticati.

La sfida futura in molti contesti, dal web al settore medico, è riuscire a strutturare ulteriormente l'informazione creando ontologie di dominio o addirittura ontologie superiori mediante le quali concettualizzare una particolare conoscenza in modo univoco, migliorare l'accuratezza delle ricerche, superare le differenze della terminologia, agevolare l'interoperabilità dei sistemi informatici e inferire relazioni che, in certi casi, possono rivelarsi apportatrici di nuova informazione. In un simile contesto, è impensabile un intervento “manuale” diretto nella strutturazione e manipolazione dei dati: Information Extraction, Natural Language Processing e Machine Learning rappresentano dunque un contributo importante in questo processo evolutivo, e la classificazione automatica dei referti ottenuta ne è la dimostrazione.

BIBLIOGRAFIA

1. W.B. Croft, D. Metzler, T. Strohman - *"Search Engines"* - Pearson, 2010.
2. G. Antoniou, F. van Harmelen - *"A Semantic Web Primer"* - The MIT Press, 2008.
3. V. Kashyap, C. Bussler, M. Moran - *"The Semantic Web"* - Springer, 2008.
4. C. D. Manning, P. Raghavan, H. Schutze - *"Introduction to Information Retrieval"* -
5. Cambridge University Press, 2008.
6. C. D. Manning, H. Schutze - *Foundations of Statistical Natural Language Processing* - MIT Press, 1999.
7. D. Jurafsky, J. H. Martin - *Speech and Language Processing* - Prentice Hall, 2000.
8. S. Abney – *Semisupervised Learning for Computational Linguistics* – Chapman & Hall/CRC, 2008.
9. G. Hripcsak, J. Austin, P. Alderson, C. Friedman – *Use of Natural Language Processing to translate clinical information fron a database of 889,921 chest radiographic reports* – Radiology 224: 157-163, 2002.
10. G. Hripcsak, C. Friedman, P. Alderson e altri – *Unlocking clinical data from narrative reports: a study of NLP* – Ann Intern Med 122: 681-688, 1995.
11. M. Fiszman, WW Chapman, D. Aronsky, R. Evans, P. Haug – *Automatic detection of acute bacterial pneumonia from chest x-ray reports* – J. Am Med Inform Assoc 7: 793-604, 2000.
12. J. Goldman e altri – *Term domain distribution analysis: a data mining tool for text databases* – Methods Inf Med 1999; 38: 96-101, 1999.
13. D. Marwede, M. Fielding, T. Kahn – *RadiO: a prototype application ontology for radiology reporting tasks* – AMIA Symposium Proceedings, 513 – 517, 2007.
14. H. Cunningham. *A Definition and Short History of Language Engineering.* - Journal of Natural Language Engineering, 5(1):1–16, 1999.
15. Teschendorf - *Diagnostica differenziale radiologica* - McGraw-Hill libri Italia - Milano, p.408, 1993.
16. L. B Koski, M. W Gray, B. F. Lang, G. Burger - *AutoFACT: An Automatic Functional Annotation and Classification Tool* - BMC Bioinformatics, 6:151, 2005.
17. A. Hindle, D. M. German, M. W. Godfrey, R. C. Holt - *Automatic Classification of Large Changes into Maintenance Categories* - IEEE 17th International Conference on Program Comprehension, pp. 30-39, 2009.
18. B. Motika, I. Horrocks, U. Sattlerb - *Bridging the gap between OWL and relational databases* - WWW '07: Proceedings of the 16th international conference on World Wide Web, pp. 807-816, 2007.
19. A. Ruttenberg et al. - *Advancing translational research with the Semantic Web* - BMC Bioinformatics, 8(Suppl 3):S2, 2007.
20. S. Noh et al. - *Classifying Web Pages Using Adaptive Ontology* - IEEE International Conference on Systems, Man, and Cybernetics (SMC03), IEEE Press, 2003.

21. J. L. Leidner - *Current Issues in Software Engineering for Natural Language Processing* - Proc. of the Workshop on Software Engineering and Architecture of Language Technology Systems (SEALTS), 2003.
22. F. Wei, L.I. Solberg,, J.C. Butler, K.J. Palattao, C.A. Vinz, M.A. Marshall - *Effects of Electronic Decision Support on High-Tech - Diagnostic Imaging Orders and Patients* – American Journal of Managed Care, 16(2):102-6, 2010.
23. S. Sahay, B. Li, E. V. Garcia, E. Agichtein, A. Ram - *Domain Ontology Construction from Biomedical Text* - International Conference on Artificial Intelligence (ICAI'07), Las Vegas, Nevada, USA, CSREA Press, 2007.
24. H. S. Goldberg et al. - *Evaluation of a Commercial Rule Engine as a Basis for a Clinical Decision Support Service* - American Journal of Roentgenology (AJR), 143:509-517, 1984.
25. J.L.V. Mejino, D.L. Rubin, J.F. Brinkley - *FMA-RadLex: An Application Ontology of Radiological Anatomy derived from the Foundational Model of Anatomy Reference Ontology* - AMIA Symposium Proceedings, 465-469, 2008.
26. N. Madnani - *Getting Started on Natural Language Processing with Python* – Crossroads, Volume 13, Issue 4, ACM Press, 2007.
27. I.H. Witten, K.J. Don, M. Dewsnip, V. Tablan - *Text mining in a digital library* - International Journal on Digital Libraries, Volume 4, Number 1, Springer, 2004.
28. E.S. O'Neill, N. M. Dluhy, P.J. Fortier, H. E. Michel - *Knowledge acquisition, synthesis, and validation: a model for decision support systems* - Journal of Advanced Nursing 47(2), 134–142, 2004.
29. W. Ceusters, P. Elkin, B. Smith - *Referent Tracking: The Problem of Negative Findings* - Stud Health Technol Inform, 2006.
30. N. McIntosh et al. - *Clinical Diagnosis of Pneumothorax Is Late: Use of Trend Data and Decision Support Might Allow Preclinical Detection* – Pediatric Research Vol. 48, No. 3, 2000.
31. V. Kashyap, A. Morales, T. Hongsermeier - *On Implementing Clinical Decision Support: Achieving Scalability and Maintainability by Combining Business Rules and Ontologies* - AMIA Symposium Proceedings, 414-418, 2006.
32. A. Katifori et al. - *Ontology Visualization Methods—A Survey* - ACM Computing Surveys, Vol. 39, No. 4, Article 10, 2007.
33. C. K. Cheng, X. S. Pan, F. Kurfess - *Ontology-based Semantic Classification of Unstructured Documents* – Proceedings of Adaptive Multimedia Retrieval (AMR), LNCS 3094, Springer, 2004.
34. S. Sahay et al. - *Semantic Annotation and Inference for Medical Knowledge Discovery* – Proceedings of the Symposium on Next Generation of Data Mining and Cyber-Enabled Discovery for Innovation, NSF, 2007.
35. M. Motterlini - *Cognizione, decisioni ed errori in medicina: calibration, overconfidence e hindsight bias* - Networks 5: 116-127, 2005
36. The International Health Terminology Standards Development Organisation -

SNOMED Clinical Terms® - Technical Reference Guide, International Release, 2008.
37. The International Health Terminology Standards Development Organisation - *SNOMED Clinical Terms® - User Guide*, International Release, 2008.
38. N. Ireson et al. - *The Evaluation of Machine Learning for Information Extraction* - Proceedings of the 22nd international conference on Machine learning (ICML), pp. 345-352, 2005.
39. J. Opitz, B. Parsia, U. Sattler - *Using Ontologies for Medical Image Retrieval -An Experiment,* Proceedings of OWL: Experiences and Directions (OWLED), 2009.
40. E. Gabrilovich, S. Markovitch - *Wikipedia-based Semantic Interpretation for Natural Language Processing* - Journal of Artificial Intelligence Research 34,443-498, 2009.

SITOGRAFIA CONSULTATA A MAGGIO 2010

1. http://www.consorzioarsenal.it/
2. http://www.nlm.nih.gov/pubs/factsheets/medline.html
3. http://it.yahoo.com/
4. http://www.lycos.it/
5. http://it.altavista.com/
6. http://www.google.it/
7. http://trec.nist.gov/
8. http://www.autonomy.com
9. http://vivisimo.com/
10. http://www.oracle.com/
11. http://www.lesbonscomptes.com/recoll/
12. http://technorati.com/
13. http://www.amatomu.com/
14. http://delicious.com/
15. http://wonderweb.man.ac.uk/
16. http://www.loa-cnr.it/DOLCE.html
17. http://www.ce.unipr.it/people/bianchi/
18. http://www.archivi.beniculturali.it/servizioII/progetti/ontologie.html
19. http://hakia.com/
20. http://www.kosmix.com/
21. http://www.powerset.com/
22. http://www.cognition.com/
23. http://www.rosettatranslation.com/
24. http://www.systran.it/http://www2.sims.berkeley.edu/courses/is256/
25. http://en.wikipedia.org/wiki/SNOMED_CT

26. http://en.wikipedia.org/wiki/ICD-9-CM#ICD-9-CM
27. http://www.ncbi.nlm.nih.gov/mesh
28. http://www.nlm.nih.gov/research/umls/about_umls.html
29. http://www.geneontology.org/GO.doc.shtml
30. http://www.biopax.org/
31. http://www.radlex.org/viewer
32. http://www.loa-cnr.it/DOLCE.html
33. http://pypi.python.org/pypi/topia.termextract/
34. http://paleo.di.unipi.it/parse
35. http://gate.ac.uk/
36. http://sourceforge.net/apps/trac/minorthird/wiki
37. http://www.nltk.org/
38. http://www.clairlib.org/index.php/Main_Page
39. http://neon.niederlandistik.fu-berlin.de/en/textstat/
40. http://sig.biostr.washington.edu/projects/fm/
41. http://www.uniroma2.it/didattica/WmIR/
42. http://www.uniroma2.it/didattica/MGRI/deposito/ml_intro.pdf/
43. http://www.informatica.uniroma2.it/upload/2009/ML/introduzione.pdf