ALMA MATER STUDIORUM - UNIVERSITÀ DI BOLOGNA

SCUOLA DI INGEGNERIA E ARCHITETTURA

DIPARTIMENTO DI INFORMATICA – SCIENZA E INGEGNERIA

CORSO DI LAUREA MAGISTRALE IN INGEGNERIA INFORMATICA

TESI DI LAUREA

in
Sistemi Intelligenti M

Realizzazione di un sistema di traduzione automatica di regole semantiche dall'italiano al francese

CANDIDATO

Sabioni Sara

RELATORE

Chiar.ma Prof.ssa Ing. Milano Michela

CORRELATORE

Varone Marco

Anno Accademico 2012/13

Sessione II

Indice

Introduzione

In Europa, Canada e in generale nel mondo, esistono alcune realtà in cui l'importanza della traduzione è decisamente rilevante. Proprio in questi ambiti, si è riscontrato che le competenze di traduttori professionisti sono spesso ristrette a pochi domini d'interesse (soprattutto per quanto riguarda le aree tecniche) e che conoscere la lingua come un parlante nativo non conferisce sicuramente l'abilità di traduttore professionista. L'interesse nella *machine translation* si è intensificata in modo considerevole negli ultimi vent'anni, durante i quali si è registrato un forte aumento di sistemi che servissero questo scopo, utilizzati in gran parte dalle autorità governative, nel business e nell'industria: già nel 1984 circa mezzo milione di pagine erano state tradotte da macchine. Alla luce di ciò, vale la pena soffermarsi a riflettere sul potenziale e sulle prospettive che possono avere questi sistemi. Gli accademici, solitamente, discutono su questi sistemi riguardo quali indici utilizzare per definire un sistema di "buona qualità" e "totalmente automatico": la "buona qualità" deve essere definita dall'utente in base ad una valutazione dell'output prodotto rispetto allo scopo di utilizzo del materiale tradotto; e se l'operazione nel suo complesso è efficace in termini di costo – beneficio.

Questo progetto di tesi si colloca quindi nell'ambito della machine translation, ed è stato realizzato all'interno dell'azienda Expert System, azienda che si occupa, nell'ambito della linguistica, della progettazione di sistemi semantici. Questi sistemi sono incentrati su Cogito®, una tecnologia che permette di effettuare analisi semantica approfondita del testo grazie alla mappatura della conoscenza all'interno di una ricca rete semantica, il Sensigrafo®. Attraverso Cogito®, è possibile trasformare automaticamente le informazioni non strutturate in dati strutturati. Cogito® possiede un linguaggio proprietario attraverso il quale linguisti ed esperti codificano la conoscenza definendo delle regole che la descrivono. Una delle particolarità di questa tecnologia è quella di effettuare la categorizzazione di documenti: definiti una serie di domini e dato un testo,

il sistema interpreta le regole e colloca automaticamente il testo all'interno della categoria corretta. Il lavoro di redazione delle regole è un compito piuttosto oneroso per i linguisti, e può capitare che alcuni progetti debbano essere redatti in lingue differenti.

Proprio per facilitare questo lavoro, nell'ambito di questo progetto di tesi, è stato studiato un sistema di traduzione automatico del contenuto delle regole di categorizzazione di Cogito®. L'applicativo realizzato si pone l'obiettivo di tradurre in un'altra lingua, il francese, tutti i termini che sono parte del progetto di Expert System di categorizzazione delle news italiane.

L'iter di presentazione di questo lavoro prevede un primo capitolo di introduzione ai sistemi di *machine translation* con presentazione dello stato dell'arte e delle soluzioni proposte in letteratura. A seguire verranno introdotte le tecnologie di Expert System sulle quali si basa l'applicativo sviluppato. Successivamente verrà introdotto in modo analitico l'obiettivo di questo lavoro, evidenziando alcuni problemi derivanti dalla traduzione e dall'utilizzo degli strumenti a disposizione. Infine verrà presentata la struttura del sistema di traduzione e l'implementazione del lavoro.

Traduzione automatica

Introduzione

All'interno di questo capitolo si vuole sottolineare l'importanza dei sistemi di traduzione automatica nell'ambito degli strumenti di supporto ai traduttori. La traduzione automatica è un problema difficile data la forte complessità del linguaggio parlato, molte parole hanno diversi tipi di significato e di conseguenza anche di traduzione. Nel capitolo vengono presentate le diverse tecniche seguite nella traduzione automatica e a seguire le tecniche linguistiche sulle quali si basano i traduttori automatici. Successivamente viene introdotta l'importanza del lessico di traduzione all'interno di questi sistemi, e vengono trattati i *non – compositional compounds*. L'ultima parte del capitolo spiega i diversi modelli di soluzione presenti in letteratura, con particolare attenzione a quelli che effettuano allineamento di testi, o utilizzano tecniche per reperire la traduzione dei singoli termini.

Machine Translation

È stato stimato, che la necessità di effettuare traduzioni sta crescendo ad un tasso ben oltre la presente o prevedibile offerta disponibile proveniente da traduttori professionisti. Per questo motivo, dagli anni cinquanta è stata proposta l'applicazione dei computer a questo compito. L'idea ha avuto un impatto immediato sul pubblico, ma l'interesse è scemato quando non sono comparsi sistemi effettivamente funzionanti. Negli ultimi anni, tuttavia, l'interesse è aumentato marcatamente grazie agli sviluppi in ambito commerciale di sistemi di traduzione primitivi, ma disponibili a costi ragionevolmente contenuti. Di conseguenza la ricerca ha posto nuovamente la sua attenzione all'ambito della traduzione automatica (MT – Machine Translation).

I sistemi di traduzione automatica possono essere classificati in tre gruppi differenti: quelli basati su regole linguistiche, quelli che utilizzano corpora testuali e quelli basati sul contesto.

Il primo, la traduzione automatica basata su regole, comprende la traduzione basata sul principio di formalizzazione delle conoscenze linguistiche rappresentate da dizionari e grammatiche. Lo schema classico utilizza tre moduli: un modulo per l'analisi della lingua sorgente, uno per il trasferimento e uno per la generazione delle frasi nella lingua di destinazione. In pratica la frase sorgente viene analizzata cercando di costruirne l'albero sintattico; utilizzando poi le regole di una grammatica contrastiva si cerca di creare la struttura sintattica equivalente nella lingua di destinazione e infine con un processo di sintesi si tenta di generare la frase corretta tradotta. Le parole vengono tradotte secondo un punto di vista prettamente linguistico scegliendo gli equivalenti linguistici più appropriati.

Il secondo, la tipologia di traduzione automatica che utilizza corpora linguistici paralleli, si basa sull'analisi di campioni reali e delle loro traduzioni corrispondenti. Un corpus è una collezione di testi selezionati e organizzati per facilitare le analisi linguistiche. Fra i sistemi che utilizzano corpora, quello principale è la traduzione automatica statistica, abbreviata in SMT (Statistical Machine Translation). L'obiettivo di questa tecnologia è la generazione di traduzioni a partire da metodi statistici basati su corpora di testi bilingue e monolingue. Affinché la SMT funzioni correttamente è necessario mettere a disposizione del sistema due banche dati piuttosto corpose: una di testi nella lingua sorgente con le relative traduzioni nella lingua di destinazione e un'altra di testi solo nella lingua di destinazione. Di fronte ad un nuovo testo da tradurre, il sistema genera possibili traduzioni delle sequenze di parole che trova nel testo stesso sulla base delle corrispondenze che riscontra nella prima banca dati. Tra le varie proposte di traduzione seleziona poi la migliore sulla base della seconda banca dati, quella relativa alla sola lingua di destinazione. Il vantaggio della traduzione automatica statistica è che, una volta impostato il sistema secondo le specifiche richieste dal cliente, questi ha a disposizione uno

strumento in grado di fornire una discreta qualità traduttiva di testi simili tra loro. Il lato negativo è che, affinché il sistema fornisca risultati di un certo livello, occorre mettere a sua disposizione un corpus molto sostanzioso di traduzioni esistenti e molto precise. Con questa tecnologia la qualità della traduzione aumenta all'aumentare della dimensione dei corpora linguistici nelle banche dati. Con un insieme sterminato di traduzioni esistenti e di testi nella lingua d'arrivo, si potrebbero ottenere risultati eccellenti con testi di argomento similare. Il primo programma di traduzione automatica statistica fu Candide, sviluppato da IBM. I sistemi di traduzione automatica statistica vengono usati quasi unicamente da agenzie governative e da multinazionali essendo piuttosto costosi e complessi, richiedono infatti dei corpus con un numero di parole superiore a due milioni per avere una buona traduzione.

Il terzo gruppo riguarda la traduzione automatica basata sul contesto. Si basa sulla ricerca della migliore traduzione di una parola prendendo in considerazione il resto delle parole che la circondano. Il sistema suddivide un testo in unità di parole (tipicamente sette o otto) e propone traduzioni di ogni sequenza nella lingua di destinazione eliminando le traduzioni che contengono frasi senza senso. Tale filtraggio utilizza un corpus nella lingua di destinazione, nel quale viene conteggiato il numero di volte in cui compare la frase cercata. In seguito la sequenza creata viene spostata di una posizione (una parola), ritraducendo la maggior parte dei vocaboli e filtrando nuovamente il testo in modo da lasciare solo le frasi coerenti. Tale procedimento viene ripetuto per tutto il testo. Nella fase finale vengono concatenati i risultati di ciascuna sequenza in modo da ottenere un'unica traduzione del testo. La traduzione automatica basata sul contesto presenta un grande vantaggio rispetto alle altre tecnologie di traduzione automatica basate su corpora linguistici: aggiungere nuove lingue è molto facile. Per inserire una nuova lingua, infatti, non è necessario tradurre milioni di parole come nei metodi statistici, sono sufficienti due corpora linguistici di dimensioni ridotte: un buon dizionario elettronico, contenente regole che permettano al sistema di coniugare correttamente i verbi e di accordare aggettivi e sostantivi in base al genere e al numero, e

un corpus nella lingua di destinazione, che può essere facilmente reperito su internet.

Oggi i software più sofisticati offrono la possibilità di fissare parametri specifici a seconda del settore in cui si esegue la traduzione per ottenere risultati più accurati. Ad esempio si può limitare la quantità di sostituzioni permesse oppure selezionare i nomi propri e aiutare il software a riconoscere la terminologia e la fraseologia più ricorrenti. Queste tecniche sono particolarmente utili in campi dove si impiega un linguaggio formale basato su moduli, come gli annunci nelle stazioni dei treni e negli aeroporti, i documenti legali e amministrativi o le previsioni del tempo. I traduttori automatici non possono sostituire i traduttori professionisti, ma possono contribuire a gestire in maniera organizzata una mole di lavoro maggiore e sono comunque utili per cogliere il senso generale di un testo e per verificare se il contenuto è di proprio interesse.

Tecniche linguistiche

Esistono diversi punti di vista per analizzare le tecniche di traduzione automatica. In particolare si evidenziano gruppi contrastanti di tecniche diverse: traduzione diretta e indiretta, approccio interlingua e con trasferimento, scopo locale e globale.

La **traduzione diretta** è una caratteristica di un sistema che nasce proprio per tradurre una determinata lingua in un'altra. I sistemi di traduzione diretta si limitano ad effettuare il minimo lavoro necessario alla realizzazione della traduzione: in questa tecnica la disambiguazione viene utilizzata solamente per reperire la conoscenza necessaria per la traduzione nella lingua di destinazione.

La **traduzione indiretta**, invece, è caratteristica di un sistema in cui l’analisi della lingua sorgente e di quella di destinazione sono processi totalmente indipendenti: la disambiguazione viene utilizzata per estendere la conoscenza determinando il significato delle parole nella lingua sorgente, senza considerare la lingua di destinazione.

L’**approccio interlingua** è caratteristico dei sistemi nei quali la rappresentazione del significato di una parola nella lingua sorgente è indipendente da qualsiasi lingua, e questa rappresentazione è utilizzata anche per sintetizzare la lingua di destinazione. In questo modo la rappresentazione di una determinata “unità di significato” (inteso come parola o insieme di parole nel caso di parole composte) potrebbe essere la stessa, senza preoccuparsi in quale lingua (o struttura grammaticale) quella unità verrà espressa. L’approccio definito con **trasferimento** è caratteristico di un sistema nel quale la rappresentazione del significato di un’unità grammaticale si differenzia a seconda della lingua dalla quale deriva e nella quale deve essere generata. Questo implica la presenza di un terzo step di traduzione nel quale la rappresentazione di un significato viene mappata nell’altra lingua: questo step si definisce trasferimento. Questi sistemi sono caratterizzati da tre fasi: analisi, trasferimento e sintesi e non sono applicabili a tutti i sistemi, ad esempio i sistemi MT diretti non usano nessuno dei due approcci, in quanto non tentano di rappresentare il significato.

Infine lo **scopo locale** caratterizza sistemi nei quali sono le parole a guidare l’analisi, e nei quali l’analisi è eseguita attraverso procedure separate per ogni parola per cercare di determinare parti del discorso, utilizzi idiomatici, e il senso della parola chiave della procedura. In questi sistemi, per esempio, gli omografi (ossia parole che sono scritte nello stesso modo, ma hanno significati diversi) rappresentano il problema maggiore, perché un’analisi della frase da sola non è attendibile. Lo **scopo globale** invece, caratterizza sistemi nei quali il significato di una parola è determinato dal suo contesto unito ad un’analisi della frase. In questi sistemi, gli omografi non creano problemi, perché il volume di contesto che viene preso in considerazione è di molto superiore al contesto considerato nei sistemi caratterizzati da scopo locale.

La funzione del lessico nella traduzione automatica

Il lessico di traduzione è un componente essenziale di qualsiasi sistema di traduzione automatica (MT). Il costo elevato di sviluppo del lessico e la sua manutenzione rappresentano un importante ostacolo per i nuovi potenziali fornitori nel mercato dei sistemi MT, oltre che un ostacolo alla crescita per i fornitori esistenti.

Per accelerare lo sviluppo del lessico, la soluzione maggiormente adottata è quella di incorporare metodi automatici per trovare possibili traduzione all'interno di corpora di testi. Tipicamente, questi candidati vengono presentati ad un esperto umano per essere validati. I metodi automatici devono essere accurati per essere efficaci, altrimenti l'esperto che li valida perderà più tempo a filtrare gli errori piuttosto che aggiungere informazioni all'interno del sistema. Metodi sufficientemente accurati sono nati intorno al 1995 e da allora lo stato dell'arte è avanzato considerevolmente.

Per capire come trarre vantaggio dai metodi per la traduzione automatica del lessico, è d'aiuto pensare al processo di traduzione con MT rispondendo a due domande:

- *Quali sono le possibili traduzioni riguardanti ogni singola parola?*
- *In quale contesto vengono usate le traduzioni trovate?*

I metodi automatici non sono ancora abbastanza precisi per poter rispondere alla seconda domanda in modo accurato. Possono comunque rispondere alla prima attraverso l'utilizzo di testi bilingue appropriati. Possono rispondere anche meglio di un sistema semi-automatico che comprende la presenza umana in uno degli step del processo, perché la presenza umana filtrerebbe i risultati sostituendoli in qualche modo con termini troppo dipendenti dal contesto.

L’integrazione di metodi automatici nel processo di sviluppo del lessico per l’MT ha il potenziale di migliorare non soltanto i costi ma anche l’accuratezza di una traduzione. Questo è vero specialmente quando un lessico esistente viene specializzato per un dominio specifico. In molti casi lo sviluppatore non sarà un esperto del dominio che si andrà ad integrare e non sarà in grado di predire quale traduzione dovrà essere aggiunta o modificata.

Esistono diversi tipi di traduzione, la traduzione probabilistica parola per parola è una di queste: il lessico viene creato senza considerare in che modo il contesto influenza la traduzione; il lessico è soltanto una lista di coppie di parole, contente una parola dalla lingua di partenza e la stessa parola nella lingua target. Il termine “probabilistico” significa che ad ogni entry del lessico viene associata una frequenza.

Associare la probabilità ad ogni termine porta con sé numerosi vantaggi: se non c’è abbastanza tempo o denaro per migliorare il lessico manualmente, allora la traduzione più frequente nel lessico costruito automaticamente sarà sicuramente meglio che nessuna traduzione. Anche in un lessico migliorato manualmente, l’informazione sulla frequenza di utilizzo di un termine può essere usata come prima strategia di selezione della parola in un contesto linguistico non familiare.

Non-compositional compounds

Uno degli aspetti più sfidanti nello sviluppo della traduzione del lessico è quello di trovare la minima unità di contenuti del linguaggio in questione. Questo task è particolarmente difficile in lingue come il cinese, dove la forma scritta non utilizza spazi tra parole; invece gli spazi nel testo in una lingua come l’inglese, o l’italiano, offrono una prima approssimazione molto facile. Questa approssimazione però, confonde quelli che vengono definiti i non-compositional compound (NCC), che sono parole formate da più token, come ad esempio “*hot dog*” oppure “*swimming pool*”. Esistono diversi tipi di non-compositionality: sequenze di parole come “*ivory tower*” e “*banana republic*” sono non-compositional guardandole da una

prospettiva monolingue. Anche la loro definizione in un dizionario monolingue non è data dal significato delle singole parole. In ogni caso però, la loro traduzione corretta in francese, ad esempio, è data dalla traduzione delle singole parole: "*tour d'ivoire*", "*republique bananière*".

Altre sequenze di parole invece, hanno la caratteristica opposta. Per esempio, la frase inglese "*tax system*" sembra una semplice parola composta guardandola dalla prospettiva inglese, ma la sua traduzione canonica in francese è "*régime fiscale*". Le NCC la cui traduzione non corrisponde alla traduzione delle singole parole, sono le più importanti dal punto di vista dello sviluppo dei sistemi MT, perché l'incapacità di trattare correttamente NCC come unità atomiche ostacola anche il tipo più semplice di traduzione uno a uno, per non parlare degli approcci più sofisticati.

Se le NCC non sono tradotte parola per parola, allora un modo per scoprirle è quello di guardare le sequenze di parole nei bitext[1] che sono collegate in modo inusuale. Per esempio, supponiamo che il nostro bitext contenga il seguente segmento:

Inglese: The head of the company works at the head office.

Francese: Le chef de la société travaille au siège.

Molte istanze della parola inglese "*head*" nel bitext sono collegate sia con "*tête*" che con "*chef*". In questa istanza però non lo sono. Allo stesso modo, in questa istanza la parola "office" non è legata alla parola "bureau", come dovrebbe. Il verificarsi di questi due eventi, suggerisce che in questo caso la parola "head office" deve essere tradotta in modo non-compositionally. Più volte capita con una certa sequenza di parole, più siamo certi di aver trovato una NCC.

Queste intuizioni sono state formalizzate e codificate in un algoritmo iterativo per scoprire le NCC [1]. In ogni iterazione, l'algoritmo produce un numero enorme di parole candidate ad essere NCC, e poi esegue dei test teorici sulle informazioni che ha prodotto per prevedere quali dei

[1] Bitext è un'abbreviazione per "bilingual text" ossia un insieme di testi che comprende coppie di testi in due lingue diverse, dove ogni testo è la traduzione dell'altro.

candidati selezionati può andare a migliorare la precisione della traduzione. L'algoritmo può trovare centinaia di NCC ogni iterazione. Una valutazione obiettiva su una semplice traduzione parola per parola, ha confermato che un'accurata individuazione delle NCC incrementa la precisione dell'algoritmo di costruzione del lessico, e quindi anche del risultato della traduzione stessa.

Alcuni modelli di soluzione

A partire dalla pubblicazione di Brown et al. del 1990 [2] sulla traduzione statistica automatica, c'è stato un interesse scientifico verso l'allineamento delle frasi e delle parole nei testi tradotti. Molti studi dimostrano che, per corpora bilingue paralleli, ben formati, la precisione ottenuta può essere fino al 99% sia per allineamenti di frasi che di parole.

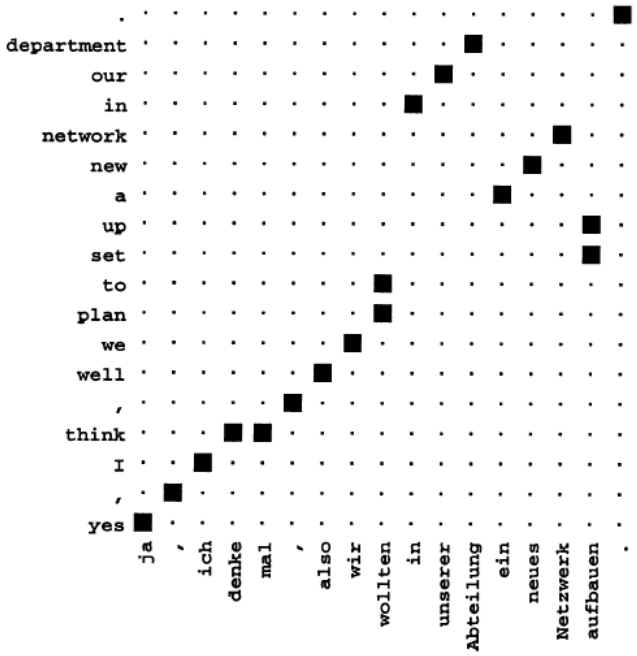


Figura 1 – Esempio di allineamento di parole (Verbmobil task).

Ovviamente, algoritmi di trasposizione, inserimento e sostituzione nel processo di traduzione, con testi reali, possono essere esposti a qualsiasi tipo di problema e perciò la robustezza continua a rappresentare un

ostacolo; inoltre in generale i problemi di ricerca per la traduzione automatica statistica sono problemi NP – completi [3], perciò in generale non ci si può aspettare di sviluppare algoritmi che hanno la garanzia di risolvere il problema senza incorrere in errori. Per le applicazioni pratiche è comunque accettabile considerare la presenza di qualche errore, quindi lo sviluppo di un algoritmo di ricerca comprende il compito di trovare un'approssimazione accettabile e un'euristica che permetta di effettuare ricerche efficienti senza commettere troppi errori. Molti degli approcci descritti in letteratura non incontrano gli obiettivi di questo tipo di ricerca:

- La ricerca A* o la programmazione lineare intera per la traduzione automatica non garantiscono una ricerca efficiente per frasi di discreta lunghezza;
- L'algoritmo di ricerca greedy tipicamente commette numerosi errori e quindi non è affidabile;
- Altri approcci per risolvere il problema di ricerca con tempo polinomiale, introducono delle semplificazioni come alleggerire le strutture di ricomposizione o limitare lo spazio di ricerca, e anch'esse portano con sé numerosi errori concettuali.

Nonostante ciò, i risultati raggiunti con questi algoritmi sono utili per la costruzione di dizionari, per controllare la consistenza dell'uso terminologico nelle traduzioni, oppure per agevolare il lavoro sulla terminologia di traduttori e interpreti.

Molti degli algoritmi proposti, inizialmente effettuano l'allineamento delle frasi, collegando le frasi che sono una la traduzione dell'altra. In un secondo step, viene effettuato un allineamento sulle parole analizzando la corrispondenza dei termini in ogni coppia di frasi. Gli algoritmi sono basati solitamente su uno dei seguenti indizi statistici:

1. Corrispondenza di parola e ordine della frase;
2. Correlazione tra la frequenza delle parole;
3. Utilizzo delle similitudini (*cognates*): lo spelling simile di parole in lingue diverse.

Tutti questi indizi solitamente funzionano bene per testi paralleli; ma purtroppo, nonostante gli sforzi che vengono fatti nella compilazione di corpora paralleli, la disponibilità di corpus paralleli sufficientemente grandi per uno specifico dominio e coppia di lingue rappresenta un'eccezione.

	English → French			French → English		
Corpus size	Precision	Recall	AER	Precision	Recall	AER
0.5K	73.0	83.8	23.1	68.5	79.1	27.8
8K	77.0	88.9	18.7	76.0	88.5	19.5
128K	84.5	93.5	12.1	84.6	93.3	12.2
1470K	89.4	94.7	8.6	89.1	95.2	8.6

Figura 2 – La tabella mostra l'effetto della dimensione del corpus di training sulla traduzione in termini di precision, recall e alignment error rate (AER). I dati riguardano esperimenti riportati nel paper di Och e Ney [30]

Siccome l'acquisizione di corpora monolingue è molto più facile, sarebbe desiderabile avere un programma che può determinare la traduzione di parole da confronti o possibili testi monolingue, non relazionati tra loro, di due lingue. Questo è quello che solitamente fanno i traduttori o gli interpreti quando preparano la terminologia in uno specifico campo: leggono testi relativi a quell'argomento in entrambe le lingue e traggono le loro conclusioni sulla corrispondenza delle parole dall'utilizzo dei termini. Ovviamente, traduttori e interpreti possono capire i testi, mentre i programmi possono soltanto basarsi su degli indizi statistici.

Per i testi che non sono paralleli, il primo indizio, che solitamente è il più affidabile, non è applicabile. Il secondo è generalmente meno affidabile del primo data l'ambiguità di molte parole nel linguaggio naturale, e molte ambiguità sono diverse in lingue diverse. Nonostante ciò, questo indizio è applicabile nel caso di testi confrontabili, sebbene abbiano un'affidabilità minore rispetto ai testi paralleli. Tuttavia, nel caso di testi indipendenti, la sua utilità può essere vicino a zero. Il terzo indizio è generalmente limitato all'identificazione di coppie di parole con spelling simile ed è solitamente utilizzato in combinazione con il primo indizio. Siccome il primo indizio non funziona con testi non paralleli, il terzo è inutile per l'identificazione

della maggior parte delle coppie; e non è comunque applicabile per lingue che non sono in relazione. In questa situazione Rapp [4] propone di utilizzare un indizio diverso dai tre menzionati sopra: la co-occorrenza che intende, si basa sull'assunzione che c'è una correlazione tra pattern di co-occorrenze in lingue diverse. Per esempio, se le parole *teacher* e *school* co-occorrono più spesso di quello che ci si aspetta in un corpus inglese, per esempio, allora la traduzione in tedesco delle stesse parole, *leher* e *schule*, dovrebbe co-occorrere più volte di quelle che ci si aspetta in un corpus tedesco.

Se consideriamo invece, corpora paralleli, uno dei modelli proposti è quello che si basa sulle co-occorrenze. Due parole co-occorrono se ricorrono in segmenti allineati. Per rispondere al problema delle possibili traduzioni, dobbiamo estrarre la traduzione di X dalle parole che co-occorrono con X. Supponiamo di voler trovare le traduzioni possibili della parola inglese "*head*". Consideriamo il bitext seguente:

1) *Inglese*: The minister nods his head.

   Francese: Le ministre hoche la tête.

2) *Inglese*: The head of the company has a big head.

   Francese: Le chef de la société a une grosse tête.

3) *Inglese*: My office is smaller than my boss's office.

   Francese: Mon bureau est plus petit que le bureau de mon chef.

L'algoritmo non dovrebbe aver bisogno di una vasta conoscenza delle lingue inglese e francese per intuire che le traduzioni di "*head*" sono "*tête*" e "*chef*".

Si parte dalla considerazione che le parole che sono una la traduzione dell'altra hanno più probabilità di co-verificarsi di altre coppie di parole. Questa intuizione può portare a un'euristica molto semplice per collegare le parole con le loro traduzioni: si collegano le coppie di parole che hanno una probabilità proporzionale alla loro relativa frequenza di co-occorrenza. Ovviamente non è così facile, soprattutto perché le parole più frequenti hanno una buona probabilità di co-occorrere con tutte le parole, e

non soltanto con la loro traduzione. Per esempio: "*head*" co-occorre con "la" tanto quanto co-occorre con "*tête*". Per considerare questa ambiguità, nei conteggi di co-occorrenza generalmente non sono comprese le frequenze delle parole marginali (come ad esempio sono gli articoli). Esistono molte soluzioni in letteratura riguardo questo problema, ma tutte si riducono ad un iniziale punteggio somiglianza *S(u, v)* tra le parole *U* e *V* contenute in segmenti opposti di un bitext.

Questo punteggio viene calcolato:

$$S(u,v) = f(\frac{\mathrm{c(u,v)}}{c(u)\ \mathrm{O}\ c(v)})$$

Dove *c(u,v)* è la frequenza in cui *U* e *V* co-occorrono; *c(u)* e *c(v)* sono la frequenza di conteggi di U e V rispettivamente e O è sia addizione che moltiplicazione e *f* è una funzione di scala positiva.

Se *Q* e *V* co-occorrono molto più spesso di quanto previsto, allora ogni metrica di somiglianza ragionevole, li riterrà essere mutue traduzioni. Se *Q* e *V* sono davvero traduzioni reciproche la loro tendenza a co-verificarsi viene definita *associazione diretta*.

Supponiamo che le parole Q e R spesso co-occorono nella stessa lingua. V ed R co-occorreranno più spesso quanto previsto dal caso. La freccia tra *V* e *R* in figura 3 rappresenta un'*associazione indiretta*, poiché l'associazione tra *V* e *R* esiste solo in virtù dell'associazione che esiste tra loro e *Q*. Traduzioni di lessico che nella costruzione dei metodi non tengono in considerazione le associazioni indirette, tendenzialmente non sono molto accurate.

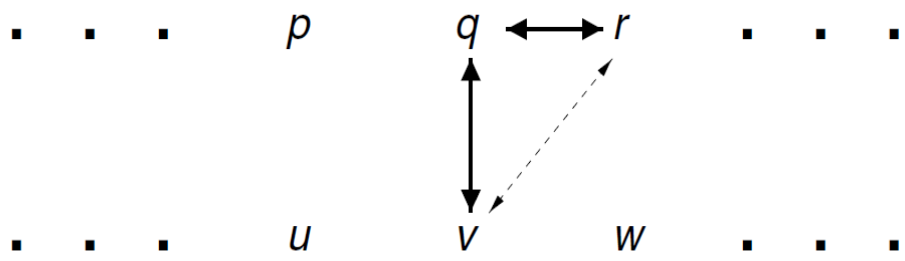


Figura 3 – Esempio di associazione indiretta

Utilizzo dei *cognates* nella costruzione del lessico di traduzione

Nel contesto della traduzione automatica, il termine *cognates* denota parole in lingue differenti che sono simili nella loro forma fonetica o grafica e sono possibili traduzioni l'una dell'altra. La somiglianza è dovuta solitamente ad una relazione genetica tra le parole (e.g. inglese *night* e tedesco *nacht*) oppure a causa di un prestito di parole da una lingua all'altra (e.g. inglese *sprint* e giapponese *supurinto*). In generale, i *cognates* includono non solo le parole relazionate tra loro geneticamente e prestiti tra lingue diverse, ma anche nomi, numeri e punteggiatura. Praticamente, tutti i bitexts contengono qualche sorta di *cognates*.

I *cognates* vengono utilizzati per un gran numero di compiti riguardanti la traduzione, come ad esempio l'allineamento delle parole o la traduzione delle parole, per formare un lessico e migliorare la qualità dei modelli di traduzione automatica. I *cognates* sono particolarmente utili quando non sono presenti dizionari bilingue. È stato dimostrato che, aggiungendo probabili *cognates* al bitext di training, il livello di incertezza della traduzione si abbassa notevolmente e migliora l'allineamento delle frasi insieme al modello di traduzione e quindi anche alla traduzione stessa.

Nella pubblicazione di Tiedemann [5] vengono descritti tre metodi per estrarre la misura di similarità adatta al riconoscimento dei *cognates* in due lingue specifiche. Questi metodi sono basati sul **LCSR** – *longest common subsequence ratio*. Consistono in una collezione di pesi per ogni mapping di un carattere nella prima lingua in un carattere della seconda lingua. I pesi sono appresi da una collezione di coppie di *cognates*. Una coppia di parole può essere identificata se il ratio della lunghezza delle parole è compreso tra 7/10 e 10/7 ed entrambe le parole sono lunghe almeno 4 caratteri.

Un esperimento riportato nel paper "Statistical Machine Translation" [6], mostra come l'utilizzo dei *cognates* sia stato necessario per allargare e rendere più completo il dizionario utilizzato. Siccome solitamente non si

ha a disposizione una collezione nota di coppie di *cognates*, si istruisce la funzione del peso su tutto il dizionario. Inutile dire che molte delle entry del dizionario non sono coppie di *cognates*, ma la speranza è che il rumore derivante dalle coppie estranee faccia in modo che le coppie si escludano da sole. Si prendono poi le coppie con il punteggio più alto dal dizionario, considerandole come una collezione di "cognati noti", scegliendo un valore di soglia opportuno che separa i veri affini dai falsi positivi. Si deve notare che nella collezione derivante dall'esperimento, non erano presenti falsi amici perché tutte le coppie provenivano da un dizionario, ma si suppone che, in generale, non ce ne possano essere. Questo esperimento mostra il valore del dizionario e dei *cognates* nella costruzione dei modelli di traduzione, e indica che il trattamento di queste informazioni come dati o il loro utilizzo, come vincoli, nei modelli di addestramento sono due modi effettivi e non dispendiosi (in termini computazionali) di migliorare il modello di traduzione. Inoltre, metodi imprecisi ed economici di trovare *cognates* sembrano fornire un miglioramento sul modello di base e modelli che utilizzano soltanto il dizionario.

In letteratura sono state sperimentate, inoltre, tre diverse misure di somiglianza: la condizione di Simard, il coefficiente di Dice e LCRS.

La **condizione di Simard** afferma che due parole sono da considerare *cognates* se sono formate da almeno 4 caratteri e i loro primi 4 caratteri sono identici.

Il **coefficiente di Dice** invece è definito anche come indice di similarità e può essere utilizzato per misurare quanto sono simili due parole in termini di coppie di lettere adiacenti comuni. Viene calcolato attraverso il rapporto tra il numero delle sillabe comuni e il numero totale delle sillabe in entrambe le parole.

$$s = \frac{2 * |A \cap B|}{|A| + |B|}$$

Per esempio date due parole A = colour e B = couleur hanno in comune tre sillabe (co, ou e ur), quindi il loro coefficiente di Dice è 6/11.

L'**LCSR** (longest common subsequence ratio) di due parole è analizzato dividendo la lunghezza della loro sequenza comune più lunga per la lunghezza della parola più lunga. Per esempio, LCSR(colour, couleur)= 5/7, siccome la loro sequenza comune più lunga è "c-o-l-u-r".

Per identificare le collezioni di possibili *cognates* in una frase allineata, questa viene divisa in parole e tutte le coppie di parole vengono immagazzinate in un file. I numeri e la punteggiatura non sono considerate, siccome richiederebbero un approccio più specifico. Dopo averli ordinati e rimossi i duplicati, rappresenta tutti i possibili allineamenti uno ad uno tra le parole del bitext. Per ogni coppia di parole, viene calcolato l'indice di somiglianza e il file viene poi ordinato nuovamente, questa volta, secondo l'indice. Se la misura ritorna un valore di similitudine non binario, i *cognates* considerati "validi" sono molto frequenti vicino alla cima della lista, e diventano meno frequenti verso la fine. La collezione di possibili *cognates* si ottiene selezionando tutte le coppie con una somiglianza sotto una certa soglia. Tipicamente, abbassando la soglia si aumenta la recall mentre si abbassa la precisione.

Allineamento fonetico e similarità

L'abilità di quantificare la somiglianza fonetica sia diacronica che sincronica tra due parole è importante in molte applicazioni di traduzione automatica. Uno studio (Heeringa et al. [7])conferma che metodi word – based per il confronto funzionano meglio che metodi corpus – based che ignorano i confini delle parole. Questi metodi solitamente stimano la similarità tra parole come la somma tra le similarità dei i segmenti fonetici corrispondenti e perciò il risultato dipende in modo cruciale dall'allineamento corretto tra le parole.

Al contrario della similarità tra parole, che è una nozione soggettiva, possiamo stabilire l'allineamento corretto con un alto grado di confidenza; una valutazione oggettiva è più semplice quindi per un algoritmo di allineamento che per un algoritmo di similarità. Entrambi gli algoritmi però, contengono due componenti principali: una metrica per misurare la

distanza tra i segmenti fonetici e una procedura per trovare l'allineamento ottimale.

Nel paper di Kondrak [8] vengono presentati diversi approcci per calcolare l'allineamento fonetico o la similarità riportati in letteratura. L'allineamento è un aspetto che viene quasi sempre dimenticato nell'ambito dei metodi di confronto automatici, sebbene sia uno step non necessariamente banale.

Per esempio l'allineamento corretto della parola latina *do* con la parola greca *didomi* è il seguente:

```
- - d ō - -
d i d ōm i
```

Figura 4 – Allineamento corretto tra le parole

E non

```
d ō - - - -      d - - ō - -      - - - - d ō
d i d ōm i      d i d ōm i      d i d ōm i
```

Figura 5 – Allineamento scorretto

Questo dimostra che esiste un'alta probabilità di effettuare un allineamento scorretto.

Convington [9] ha sviluppato un algoritmo per l'allineamento dei *cognates* sulla base della similarità fonetica. In prima battuta l'algoritmo era stato sviluppato solamente per due lingue, ma esteso poi, in un secondo momento, anche ad altre. Solitamente in letteratura per realizzare questo obiettivo si ricorre a metodi di programmazione dinamica, ma Covington specifica di volerli evitare a causa del fatto che prima di tutto le parole da allineare sono relativamente corte, quindi l'efficienza della programmazione dinamica non è necessaria; secondo, la programmazione dinamica normalmente restituisce soltanto un allineamento per ogni coppia di stringhe (effettua una ricerca nello spazio di tipo breath – first), ma un metodo completo di allineamento potrebbe necessitare di *n* migliori

alternative, o di quelle che sono conformi a certi criteri. Il suo algoritmo consiste in una metrica di valutazione progettata appositamente e in una procedura di ricerca depth – first per trovare l'allineamento di costo minimo. La metrica di valutazione è una funzione che specifica il costo di ogni coppia di segmenti e un costo inserimento/cancellazione (abbreviato come *indel*) dipendente dal contesto. Il costo di un allineamento è dato dalla somma dei costi di tutte le sostituzioni e *indel*.

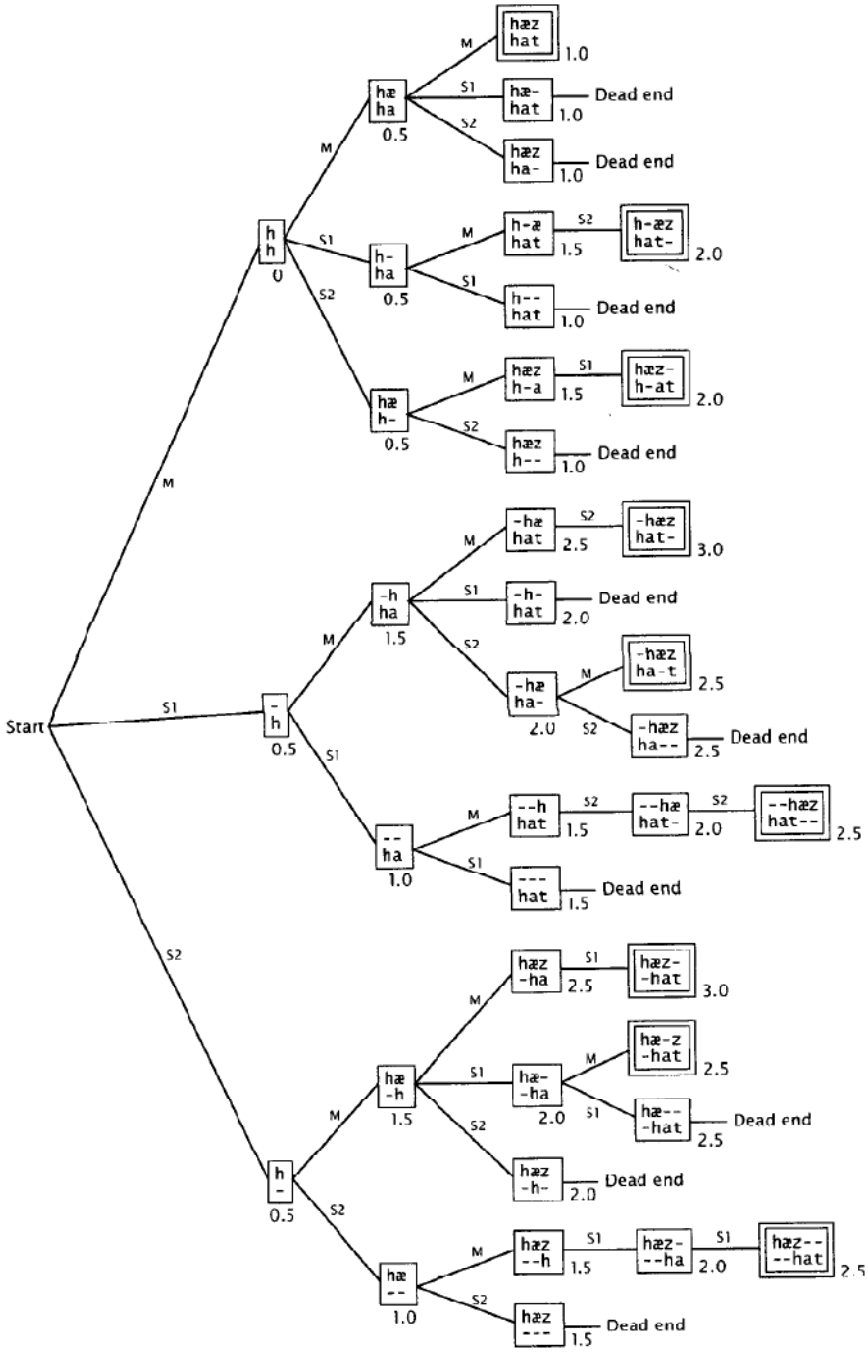


Figura 6 – Spazio di ricerca dell'algoritmo di Covington per l'allineamento delle parole inglese *hæt* e tedesco *hat*

Somers [10] ha proposto un algoritmo speciale basato su un albero di derivazione, per allineare i dati articolati figli con i propri modelli adulti. L'algoritmo dipende pesantemente dall'allineamento delle vocali accentate e ha prestazioni simili all'algoritmo di Convington.

Nel sistema di Gildea e Jurafsky [11] invece, le regole fonetiche vengono generate direttamente da un corpus di notevoli dimensioni di parole corrispondenti. Gli autori hanno scoperto che un pre – allineamento delle parole migliora notevolmente le performance del sistema.

Kessler [12] ha testato una serie di approcci differenti per calcolare la distanza tra i diversi dialetti irlandesi. I dialetti erano rappresentati sottoforma di liste di parole, ognuna delle quali contenente circa 50 concetti. I metodi più sofisticati impiegano dodici caratteristiche fonetiche multi valore. La differenza tra i segmenti fonetici è calcolata come la differenza media tra tutte le dodici caratteristiche. Kessler ha dimostrato però che un metodo basato sulle caratteristiche funziona peggio rispetto ad un metodo che si basa sui fonemi.

Nerbonne e Heeringa [7] hanno sperimentato il problema di misurare la distanza fonetica tra dialetti olandesi. La distanza è misurata prendendo la somma delle distanze di Levenshtein[1] tra due set di parole corrispondenti. Il costo dell'*indel* viene settato ad un valore pari alla metà della media di tutte le sostituzioni. La distanza calcolata viene normalizzata dividendo il suo valore per la lunghezza della parola più lunga. Gli autori hanno concluso che per misurare la distanza tra due fonemi sulla base delle caratteristiche, la distanza Manhattan è preferibile alla distanza di Euclide e alla correlazione di Person.

Il programma di Oakes [13], JAKARTA, contiene un algoritmo di allineamento fonetico, il cui obiettivo finale è quello di scoprire i cambiamenti dei suoni regolari. La similarità tra i segmenti fonetici è stimata controllando solo l'identità dei valori delle caratteristiche. Tipicamente le caratteristiche valutate sono: la posizione, il modo e la voce, dei quali i primi due sono espressi attraverso più valori.

[1] Vedi paragrafo successivo "Distanza di Levenshtein", pag. 27.

L’allineamento di due parole avviene attraverso la definizione di punti di ancoraggio (*anchor point*) tra le stringhe da confrontare. Bisogna fare attenzione a questi algoritmi perché alcuni hanno un’imperfezione importante: supponiamo che le stringhe da allineare siano *tewos* e *divut*. Anche se i segmenti corrispondenti sono leggermente diversi, l’allineamento attraverso un algoritmo di tipo greedy cerca subito il matching migliore, allineando erroneamente le due *t*.

t	e	w	o	s		-	-	-	-	t	e	w	o	s
d	i	v	u	t		d	i	v	u	t	-	-	-	-

Figura 7 – Allineamento corretto e sbagliato di un’ipotetica coppia di *cognates*.

A causa della sua natura ricorsiva, l’algoritmo non ha possibilità di rimediare l’errore; un algoritmo che non fa mai backtracking non ha la possibilità di garantire un allineamento corretto. Nemmeno la vocale su cui cade l’accento è da considerare un valido punto di ancoraggio, in quanto l’accento è troppo instabile per dipendere da esso; anche i dialetti della stessa lingua possono avere diverse regole riguardanti l’uso degli accenti.

Distanza di Levenshtein

Nella teoria dei linguaggi, la distanza di Levenshtein è una misura per la differenza tra due stringhe, in particolare serve a determinare quanto due stringhe siano simili. Viene solitamente applicata in semplici algoritmi di controllo ortografico e per fare ricerca di similarità tra immagini, suoni, testi, software per tradurre il linguaggio naturale basato sulla memoria di traduzione.

La distanza di Levenshtein tra due stringhe A e B è il numero minimo di modifiche elementari che consentono di trasformare la stringa A nella stringa B. Per modifica elementare si intende:

- Cancellazione di un carattere;
- Sostituzione di un carattere con un altro;

- ➥ Inserimento di un carattere.

Per esempio, per trasformare "bar" in "biro" occorrono due modifiche:

1. "bar" -> "bir" (sostituzione di 'a' con 'i')
2. "bir" -> "biro" (inserimento di 'o')

Non è possibile trasformare la prima parola nella seconda con meno di due modifiche, quindi la distanza di Levenshtein fra "bar" e "biro" è 2.

Solitamente si tende a confrontare soltanto stringhe di lunghezza ridotta, in quanto il costo impiegato nella computazione è circa proporzionale al prodotto della lunghezza delle due stringhe; questo rende l'algoritmo impraticabile in certi casi, per motivi di efficienza.

Simili alla distanza di Levenshtein esistono altri algoritmi, tra cui il già citato LCSR, che permette soltanto inserimenti e cancellazioni, ma non sostituzioni; la distanza di Damerau – Levenshstein, che oltre alle tre operazioni aggiunge anche la trasposizione; la distanza di Hamming, anch'essa utilizzata per rilevare la distanza tra le stringhe, che permette soltanto sostituzioni ed è quindi applicabile solamente a parole che hanno la stessa lunghezza.

Integrazione di metodi automatici e manuali

Se consideriamo che il costo del tempo di lavoro di un computer è insignificante confrontato con il costo del tempo di lavoro di una persona umana, allora il processo di costruzione del lessico più efficiente è quello che minimizza lo sforzo umano. Per essere d'aiuto, il processo automatico deve aiutare lo sviluppatore del lessico ad inserire entry più velocemente. Inoltre, il tempo risparmiato accelerando l'inserimento delle entry, non deve però essere sprecato eliminando falsi candidati.

Nel processo di creazione del lessico rimangono fondamentali la conoscenza pre-esistente sulla traduzione e la presenza umana per validare

i collegamenti. Una possibile strategia di sviluppo del lessico potrebbe essere la seguente:

1) Avviare l'algoritmo di traduzione automatica per la costruzione del lessico sui bitext a disposizione;
2) Ordinare le entry in output rispetto al punteggio di similarità;
3) Presentare ad un linguista la lista per la validazione insieme ai bitext che forniscono un contesto;
4) Il linguista dovrà continuare a validare la lista finché il rapporto tra entry vere e false non scende sotto una soglia ragionevole;
5) Sistemare il punteggio di similarità a meno infinito per le entry che sono state scartate: in questo modo si previene che queste vengano ricandidate ad un successivo passaggio dell'algoritmo;
6) Ripetere dallo step 1 finché le entry valide non diventano così rare da preoccuparsi.

La semantica

Introduzione

In questo breve capitolo si vogliono spiegare alcuni concetti, fornendo la base teorica su cui è costruita la tecnologia di Expert System. Elemento fondamentale del business di Expert System è l'analisi semantica automatica.

Semantica

La semantica è quella parte della linguistica che studia il significato delle parole, degli insiemi delle singole lettere e delle parole, delle frasi e dei testi.

Le tecnologie che si basano sulla semantica effettuano un'analisi linguistica approfondita, che arriva fino al riconoscimento del significato (semantica). Eseguono l'analisi morfologica elaborando le keyword, l'analisi grammaticale riconoscendo il lemma base (la forma del termine così come compare nel vocabolario), l'analisi logica di periodo (identificando soggetto, verbo, complemento oggetto, proposizioni reggenti, subordinate, ecc.) e con memoria del contesto nell'ambito del documento. Raggiungono in questo modo il livello di interpretazione concettuale, distinguendo fra le varie accezioni dei termini e riconoscendo i diversi modi per esprimere la stessa informazione. Il risultato dell'analisi è la costruzione di una mappa cognitiva e concettuale del testo. L'utilizzo di queste tecnologie porta inoltre i seguenti benefici:

- comprendere implicitamente la capacità di costruire indici sulle keyword, in quanto primo passo dell'analisi linguistica;
- gestire in maniera completa la lingua, senza trarre vantaggio dall'utilizzo congiunto con altre tecnologie o rendere necessario l'intervento umano;

- riconoscere i lemmi e quindi gestire tutte le flessioni dei termini; arrivando anche all'identificazione dei concetti distinguendo ad esempio fra *tempo* sinonimo di clima e *tempo* inteso come epoca storica.

Disambiguazione

La **disambiguazione** (in inglese **Word Sense Disambiguation**) è l'operazione con la quale si precisa il significato di una parola o di un insieme di parole (frase), che denotano significati diversi a seconda dei contesti. Il problema della disambiguazione, mediante appositi algoritmi, riveste particolare importanza nelle ricerche sull'intelligenza artificiale e, in particolare, sull'elaborazione del linguaggio naturale. Specificamente, i campi in cui si prevedono benefici della disambiguazione sono:

- **Machine translation**: la disambiguazione del senso è essenziale per effettuare la traduzione corretta di una parola. Ad esempio la parola francese *grill*, dipendente dal suo contesto, può essere tradotta come *railings, gate, bar, grid, scale, scheduel,* ecc…;
- **Information retrival e hypertext navigation**: quando si effettua una ricerca di una parola specifica, sarebbe preferibile visualizzare solamente i documenti dove la parola viene utilizzata nel senso che intendiamo, eliminando i risultati dove la parola viene utilizzata con un senso inappropriato;
- **Contenuto e analisi tematica**: un approccio comune al contenuto e all'analisi tematica è quello di analizzare la distribuzione di determinate categorie di parole predefinite (ad esempio: parole indicative di un determinato concetto, idea, tema, ecc…);
- **Analisi grammaticale**: la disambiguazione è utile per etichettare parti del discorso. Per esempio in alcune lingue capire a che significato della parola ci si riferisce, permette di capire il genere della parola, la preposizione che deve precederla o di effettuare l'analisi sintattica della frase;

- **Elaborazione del parlato**: la disambiguazione è richiesta per effettuare la corretta fonetizzazione della parole nella sintesi del parlato;
- **Elaborazione del testo**: la disambiguazione è necessaria per la correzione degli errori ortografici.

La disambiguazione presuppone la presenza di un'ontologia lessicale dalla quale sia possibile reperire le informazioni riguardo ai concetti e alla relazioni che vi intercorrono.

Ontologia e tassonomia

Insieme al concetto di semantica, spesso ritroviamo anche i concetti di ontologia e tassonomia.

In informatica, un'ontologia indica la descrizione formale dei concetti che costituiscono un dato dominio; ne identifica le classi, le organizza in una gerarchia specificando anche le loro proprietà (che caratterizzano gli oggetti appartenenti alla classe stessa) e indica le relazioni fondamentali tra tali classi. Un'ontologia informatica in pratica definisce il modo in cui la macchina, intesa come calcolatore, modella ciò che conosce del mondo in cui è inserita e contiene le regole e i costrutti di base necessari a compiere riflessioni su di esso.

La tassonomia invece (dal greco taxic, taxis, "ordinamento", e nomoc, nomos, "norma" o "regola"), in campo informatico, ma anche in contesti più generalizzati, è una scienza che si occupa della classificazione di elementi. Col termine tassonomia ci si può riferire sia alla classificazione gerarchica di concetti, sia al principio stesso della classificazione.

Precision e Recall

Precision e recall [14] sono comuni classificazioni statistiche utilizzate in diversi ambiti e di particolare importanza nel campo dell'information

retrival e del natural language processing. Nell'ambito della semantica vengono utilizzati per valutare l'esattezza del processo di disambiguazione.

La *precision* può essere vista come una misura di *esattezza* o fedeltà, mentre la *recall* è una misura di *completezza*.

Tipicamente la recall indica quanti casi vengono rilevati rispetto a quelli rilevanti; mentre la precision indica tra i casi rilevati quanti sono corretti.

Si può quindi comprendere che sono grandezze inversamente proporzionali: maggiore è la precision di una ricerca, minore sarà la recall e viceversa. Consegue quindi, che ad esempio motori di ricerca perfetti, ossia che trovino tutti e soli documenti pertinenti ad una particolare ricerca, non esistono. Si tratta di effettuare un bilanciamento tra le due grandezze.

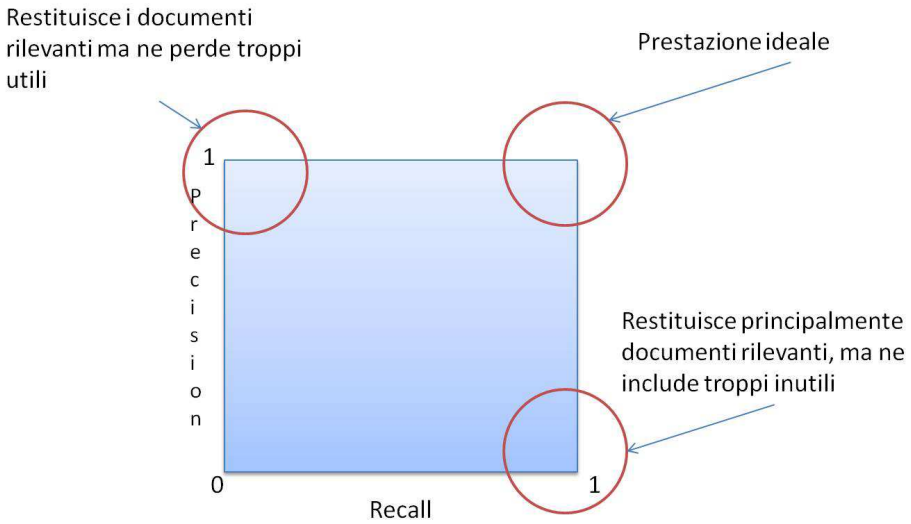


Figura 8 – La figura mostra nella scala da 0 a 1, le conseguenze che derivano dalla calibrazione di questi due parametri.

La tecnologia di Expert System

Introduzione

Questo capitolo è rivolto all'introduzione degli strumenti di Expert System che sono stati utilizzati per la realizzazione di questo progetto di tesi. Partendo dagli elementi fondamentali quali il Sensigrafo® e il motore di disambiguazione semantica, andremo poi ad introdurre Cogito® Studio, strumento per lo sviluppo di regole di elaborazione di testi, per poi finire con Trek, motore di indicizzazione dell'azienda.

Il Sensigrafo®

Il Sensigrafo® è uno degli elementi principali dell'architettura di Expert System. Si tratta di una rete semantica di notevoli dimensioni, organizzata per concetti di una lingua, rappresentati attraverso dei nodi. Una rete semantica infatti, è una forma di rappresentazione della conoscenza, attraverso un grafo orientato o non orientato formato da vertici che rappresentano concetti della lingua in esame, e archi che rappresentano le relazioni semantiche tra i concetti. Le reti semantiche sono un tipo comune di dizionario comprensibile da una macchina; organizzato, non nel classico modo in ordine alfabetico, ma attraverso relazioni tra gli elementi.

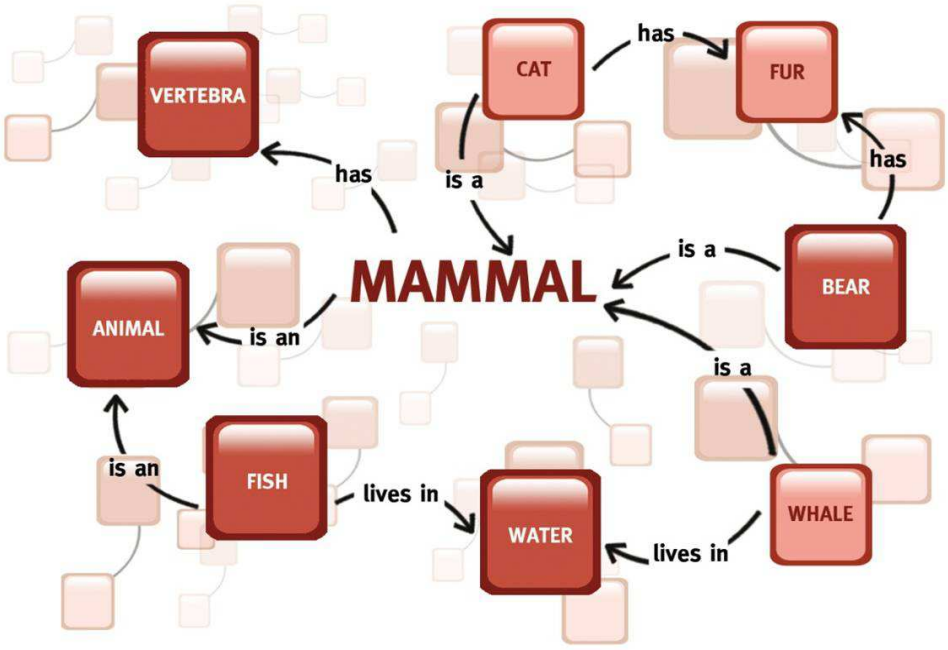


Figura 9 – La figura riporta un esempio semplificato di un nodo di rete semantica.

Un esempio di rete semantica è WordNet, un database semantico – lessicale per la lingua inglese elaborato dal linguista George Armitage Miller presso l'Università di Princeton, che si propone di organizzare, definire e descrivere i concetti espressi dai vocaboli [15].

Alcuni tipi di relazione semantica esistenti tra i termini rappresentati da una rete semantica possono essere:

- **Meronimia**: A è parte di B, quindi B ha A come sua parte, ad esempio *ruota* è meronimo di *automobile* perché è parte di essa;
- **Olonimia**: B è parte di A, quindi A ha B come sua parte, è il concetto opposto di meronimia, ad esempio *l'automobile* è un'olonimia di *motore*;
- **Iponimia**: A è subordinata a B; A è un tipo di B, ad esempio *automobile* è iponimo di *veicolo*;
- **Iperonimia**: A è subordinata a B, ad esempio *veicolo* è iperonimo di *automobile*, *camion*, *autobus* ecc;
- **Sinonimia**: A denota la stessa cosa di B, ad esempio *automobile* è sinonimo di *auto*;

- **Antonimia**: A denota il concetto opposto a B, ad esempio *dirottare* è contrario di *guidare*;
- **Troponimia**: è il corrispondente dell'iponimia dei nomi, ma riguarda i verbi. Il verbo V1 è troponimo del verbo V2 se V1 indica un caso particolare del più generico verbo V2; ad esempio *passeggiare* è troponimo di *camminare*.

Ogni elemento del Sensigrafo® è un syncon, ossia un identificatore univoco di un concetto. Ogni syncon è rappresentato da uno o più lemmi. È possibile che a uno stesso lemma siano associati più syncon id, poiché il syncon rappresenta in dettaglio un particolare concetto lessicale e non la sequenza di simboli utilizzata per rappresentarlo.

Ad esempio, la parola "*calcio*" è associata ad almeno due nodi del Sensigrafo®: uno per il significato di "*elemento chimico*", uno per "*sport*".

Il nodo del syncon è collegato agli altri in una struttura gerarchica a ereditarietà. I lemmi di uno stesso syncon sono quindi sinonimi e hanno lo stesso significato. Il significato, a sua volta, sarà determinato e rappresentato dall'insieme di sinonimi che costituisce il syncon.

Il significato di un syncon è dato non solo dal concetto espresso dai suoi lemmi, ma anche dalle relazioni che intercorrono con gli altri syncon, cioè dalle relazioni semantiche che la rete permette di stabilire tra i suoi nodi. Ogni nodo è arricchito dalle caratteristiche e dal significato dei nodi vicini. Ogni syncon nel Sensigrafo® è definito dal tipo grammaticale, dalla relazione semantica (link), dagli attributi, dalla glossa e dagli esempi.

Il disambiguatore semantico

Il Sensigrafo® è necessario per la rappresentazione della conoscenza linguistica di un idioma, ma un altro elemento fondamentale di questa tecnologia è il **motore di disambiguazione semantica**. Il motore permette l'analisi di un testo e, attraverso la conoscenza fornita dal Sensigrafo®, è in grado di associare ogni parola ad un syncon id, di effettuare ragionamenti

e analisi sul testo di partenza, comprendendo approfonditamente il significato di ogni singola frase. Per un essere umano, il significato di una parola è chiaro perché gli elementi che la circondano lo aiutano a comprenderne il senso ed il contesto in cui questa è usata. Un software richiede un'interpretazione delle parole priva di ambiguità, ma che allo stesso tempo restituisca un'esperienza cognitiva paragonabile a quella umana.

Figura 10 – Per un riconoscitore automatico è difficile capire a quale significato della parola calcio si fa rifermento.

Se correttamente istruito, sulla base del modello umano, il computer può unire la comprensione logica delle parole alla potenza di memorizzazione e computazione. Il disambiguatore può analizzare le singole frasi, o interi documenti, alla ricerca del corretto significato per ciascun elemento che compone il testo, eliminando ogni ambiguità. Viene svolto un ragionamento atto ad identificare i diversi significati di tutti gli elementi del testo e il contesto di riferimento.

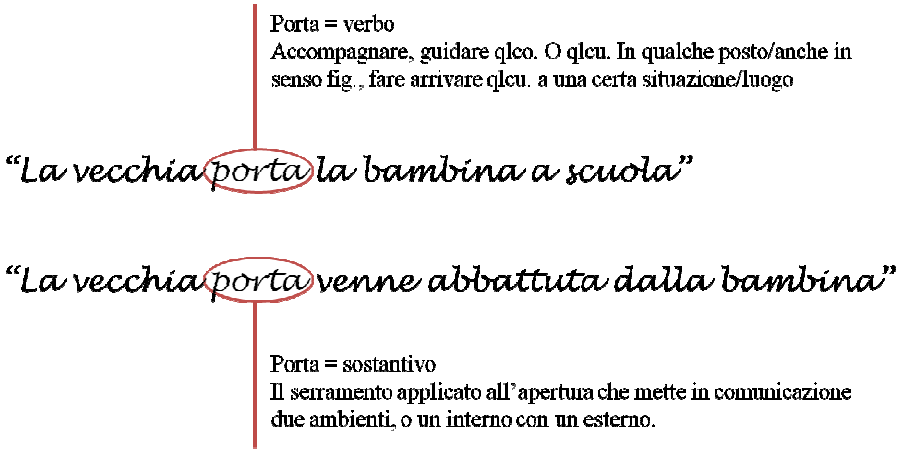


Figura 11 – Esempio di disambiguazione

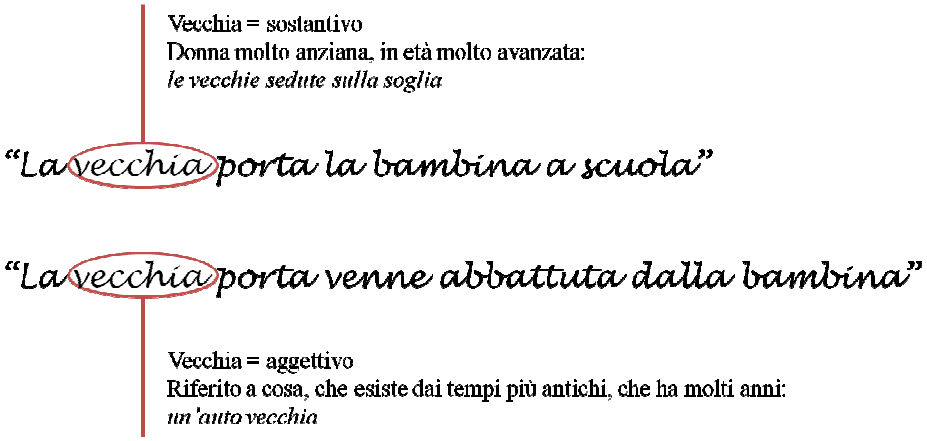


Figura 12 - In figura 11 e 12 sono rappresentati alcuni esempi di disambiguazione.

Il processo di disambiguazione parte dall'analisi linguistica di un testo scritto in linguaggio naturale e ne restituisce in output le entità e relazioni che le legano, contenute nel testo originale; tale rappresentazione concettuale è detta stringa di disambiguazione. La disambiguazione avviene tramite blocchi di analisi successive e può essere suddivisa in tre macro fasi: grammaticale, logica e del periodo.

Durante la prima fase il testo è sottoposto ad un'operazione di pulizia per semplificare l'analisi (eliminazione di spaziature, resa uniforme di caratteri equivalenti, de-capitalizzazione di blocchi...), per poi essere processato dal blocco di parser. L'output generato a questo punto viene analizzato nuovamente con meccanismi euristico – statistici per migliorare riconoscimenti equivalenti (es: sequenze ambigue sostantivo-aggettivo), per passare poi alla parte in cui si può attribuire a queste sequenze tag grammaticali. In ultimo gli elementi grammaticali riconosciuti sono uniti in gruppi logici per agevolare la fase successiva.

Nella seconda fase, sui gruppi logici viene eseguita un'analisi logica minimale che, appoggiandosi ad algoritmi euristici e a informazioni statistico – semantiche estratte dal Sensigrafo®, attribuisce relazioni tra verbi e soggetti logici. In questo passaggio viene anche realizzata una divisione e classificazione in proposizioni.

Nella terza fase ad ogni elemento significativo vengono associati i possibili syncon, ordinati per un valore di frequenza che è memorizzato nel relativo database.

A questo punto interviene la disambiguazione vera e propria che avviene secondo logiche di raffinamento successive applicate a tutti gli elementi in gioco. La disambiguazione di primo livello avviene automaticamente durante l'assegnazione dei significati presi dal Sensigrafo®. L'ordine di frequenza dei significati, viene utilizzato per assegnare un punteggio iniziale proporzionato: più un significato è frequente, più il relativo punteggio è alto. Un secondo livello si realizza tramite l'analisi statistica dei domini, recuperati dai syncon, proporzionati alla frequenza dei significati, per il loro tipo grammaticale e il loro ruolo logico. L'analisi dei domini viene compiuta prima estraendo il loro elenco completo (con le percentuali di riferimento) su tutto il testo, poi passando ad una analisi, frase per frase, e attribuendo un punteggio in base all'appartenenza dei significati al gruppo di domini principali. Un livello successivo analizza i contesti pertinenti ai singoli significati analizzandone l'intorno e pesando l'eventuale match sia sulla distanza, sia sulla profondità semantica del legame.

Si procede poi con altri algoritmi di raffinamento che controllano ulteriormente l'uniformità delle scelte fatte e, infine, si applica un passaggio di selezione euristico nel caso alcuni elementi del testo non siano stati attivati e siano ancora ambigui (ad esempio quando si hanno più significati con lo stesso punteggio).

Per quanto riguarda la tecnologia semantica, implementata in Expert System, esistono anche moduli software che fanno da contorno alle elaborazioni del disambiguatore, sia in fase di input dei documenti, ad esempio preparando questi ultimi ad una elaborazione ottimale, oppure in fase di output o post-processing per correggere alcuni errori che possono essere commessi in fase di elaborazione.

Cogito® Studio

Cogito® Studio è lo strumento di Expert System che permette di effettuare analisi semantica dei testi. In particolare Cogito® Studio permette di eseguire categorizzazione di documenti in relazione ad una determinata tassonomia o estrazione di informazioni specifiche da testi scrivendo delle regole nel suo linguaggio proprietario. La tecnologia alla base di Cogito® permette di catturare tutti gli aspetti morfologici, sintattici e lessicali di un testo attraverso un'accurata analisi linguistica e sceglie il concetto giusto delle parole attraverso la comprensione della giusta semantica.

In Cogito® Studio esistono tre tipi di linguaggi:

- *Linguaggio C*: viene utilizzato per i compiti di categorizzazione e per attribuire punteggi ai domini della tassonomia; viene utilizzato nei file di estensione `*.efo`;
- *Linguaggio D*: per gestire i punteggi associati alla tassonomia dei domini ed utilizzato nei file con estensione `*.efd`;
- *Linguaggio E*: utilizzato per i compiti di estrazione di dati strutturati ed utilizzato nei file con estensione `*.efe`.

Ad alto livello, lavorare con Cogito® Studio, implica sviluppare il progetto in modo ciclico, passando attraverso step successivi che si ripetono:

- Creazione del progetto;
- Scrittura delle regole;
- Testing delle regole;
- Deployment.

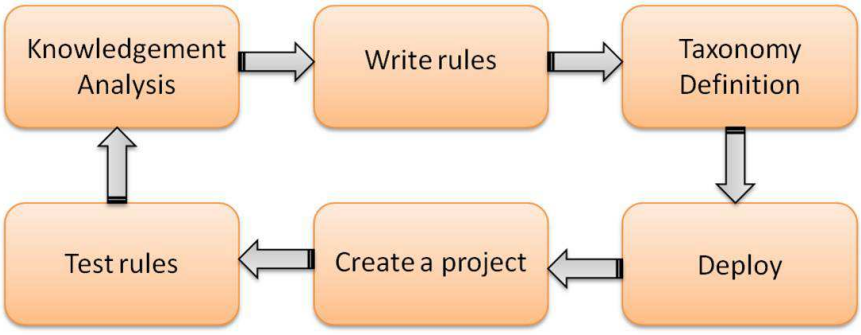


Figura 13 - Approccio nello sviluppo di un progetto in Cogito® Studio

Linguaggio C

Il linguaggio C è un linguaggio dichiarativo che definisce delle regole per selezionare uno o più domini da associare ad un documento, con lo scopo di categorizzarlo automaticamente. Una regola di linguaggio C definisce un insieme di condizioni che devono verificarsi in seguito all'analisi linguistica del testo in ingresso. Se si verificano le condizioni definite nella regola, questa assegna un punteggio al dominio associato; il punteggio che il dominio riceve dipende da una serie di elementi quali la tipologia del punteggio scelta, la sintassi della regola e la posizione del documento in cui è stata riconosciuta. Lo scopo finale è ottenere una classifica di domini sulla base dei punteggi ottenuti.

```
SCOPE SENTENCE
{
    // produzione agricoltura
    DOMAIN(04001000:NORMAL)
    {
        LEMMA("produzione")
        >>
        LEMMA("agricolo","agrario")
    }
}
```

Figura 14 – Esempio di regola di categorizzazione

La sintassi seguita nella scrittura della regola è definita come segue:

```
SCOPE extension
{
    DOMAIN( domain [:score] )
    {
        expressions
    }
}
```

La sua costruzione avviene, prima di tutto, attraverso la definizione di `SCOPE` ed `EXTENSION`: il primo definisce l'area di documento in cui la regola deve essere applicata, mentre il secondo rappresenta la porzione di documento (minore o uguale allo `SCOPE`) entro cui tutti i parametri della regola devono essere verificati affinché questa scatti. Nel caso

illustrato in figura 14, lo scope della regola è l'intero documento, mentre l'extension è rappresentata dalla singola frase definita dall'attributo SENTENCE. I valori dell'extension possono essere molteplici: una o più frasi, oppure uno o più paragrafi (PARAGRAPH); allo stesso modo il valore dello scope può essere definito invece che sull'intero documento soltanto sul titolo oppure su un atomo (es. SCOPE SENTENCE IN SECTION (TITLE)).

Successivamente la regola richiede la definizione di un dominio attraverso la keyword DOMAIN. Questo permette l'assegnazione del punteggio a quel dominio specificato, quando la regola viene riconosciuta nel testo. Nell'esempio riportato in figura 14, al dominio "*04001000*" verrà assegnato un punteggio NORMAL (10pt). Il punteggio assegnato può essere anche HIGH (15pt) o LOW (3pt). Esistono anche altri due tipi di comandi: SELECT, attraverso il quale si può far risultare vincitore il dominio al quale viene applicato (cioè il dominio risulta il primo nella classifica finale dei domini), si utilizza solitamente per le regole fondamentali; DISCARD, invece, serve ad ottenere l'effetto contrario, ovvero a fare in modo che se la regola viene riconosciuta, la categoria definita non compare nell'output.

All'interno del corpo del dominio è presente la regola vera e propria, *l'espressione* (o *condizione*), che definisce cosa dovrà essere riconosciuto nel testo affinché la regola scatti. L'espressione può essere costituita da uno o più *attributi* connessi tra loro da *operatori*.

Gli attributi

Un **attributo** permette di identificare un token, cioè un'espressione linguistica nel testo, definendo una caratteristica che questo token dovrebbe avere per far scattare la regola.

Gli attributi utilizzabili nel linguaggio C sono:

- SYNCON: sono definibili come set di termini sinonimi. Quando si cerca un syncon, significa cercare tutti i termini presenti al suo interno. Token diversi ma sinonimi, vengono identificati dalla

stessa regola, attraverso l'id del syncon che lo comprende. Ad esempio: se identifichiamo il lemma '*tentativo*' con il suo syncon scrivendo `SYNCON(11630)`, non si individuerà soltanto '*tentativo*', ma anche i suoi sinonimi come '*prova*' ed '*esperimento*';

- `ANCESTOR`: serve per identificare un syncon e tutti i suoi nodi figli. Sempre prendendo come esempio '*tentativo*', oltre ai lemmi appena visti con `SYNCON`, ci permette di individuare tutti i figli nodo presenti nel Sensigrafo® come '*semitentativo*' oppure '*ritentamento*';
- `LEMMA`: l'attributo permette di identificare un token attraverso la sua forma base. Usare il lemma di un verbo significa, quindi, individuare qualsiasi forma flessa del verbo che ricorre nel testo preso in analisi;
- `LIST`: permette di identificare un token attraverso i lemmi di un syncon o attraverso una catena del Sensigrafo®. Un token è identificato se una delle sue forme base è inclusa all'interno della lista di syncon o della catena di syncon. La funzione è equivalente all'utilizzo dell'attributo `LEMMA` con all'interno la lista completa dei lemmi associati a quel syncon. Il numero di livelli va da 0 a 99, se è omesso a default viene considerato 0. Esempio: nell'esempio del syncon id 1045, per far match con l'attributo, vengono considerati due livelli seguendo la relazione supernomen/subnomen.

```
LIST(1045:2:supernomen/subnomen)
```

- `KEYWORD`: è l'attributo con cui identificare un token in base alla sua forma esatta all'interno del testo (incluse lettere maiuscole). Ha un utilizzo molteplice, e può essere applicato nel riconoscimento di tutto ciò che non si trova nel Sensigrafo® come modi di dire oppure nomi di associazioni, sigle particolari;
- `TYPE`: con questo attributo si riesce ad identificare un token in base alla tipologia grammaticale di cui fa parte. Alcuni esempi

sono: ART (articolo), SOS (sostantivo), NPH (nome proprio di persona);

- ROLE: identifica un token in base alla funzione logica che possiede all'interno della frase;
- POSITION: con questo attributo è possibile rintracciare un token in base alla sua posizione all'interno del documento analizzato; alcuni esempi sono: BEGIN SENTENCE (oppure PARAGRAPH/SECTION), END SENTENCE (oppure PARAGRAPH/SECTION);
- PATTERN: si utilizza per identificare dei token attraverso delle espressioni regolari. Utilizzato in particolare per le date, codici o stringhe alfanumeriche;
- RELEVANT: permette di identificare un token che si trova in almeno una delle liste contenente gli elementi rilevanti; esistono tre liste contenenti gli elementi rilevanti: LEMMAS, SYNCON, e KEYWORD.

Gli operatori

Tutti questi attributi possono essere combinati fra loro all'interno di un blocco domain, attraverso degli **operatori** di diverso tipo, così come si può vedere in figura 13. In particolare il simbolo riportato tra le due regole LEMMA ">>" indica che i token devono trovarsi esattamente nella sequenza indicata, senza che nulla si interponga tra loro. Questo operatore fa parte degli **operatori posizionali di sequenza**, che considerano dei gruppi di token riferendosi alla loro posizione reciproca nel testo. Altri tipi di operatori di sequenza sono: ">" oppure "<" che indica che un token deve seguire (o precedere) un altro ma ammettendo che possano esistere elementi interposti tra loro e "<>" che invece indica una sequenza flessibile dove tutti gli elementi fino alla fine della frase sono accettati. È possibile specificare anche sequenze di token in base alla logica grammaticale con cui sono legati, ad esempio: se voglio cercare una frase in cui i token "dog" e "cat" sono entrambi soggetti, dovrò scrivere:

```
LEMMA(“dog”) &SS LEMMA(“cat”)
```

Gli **operatori di relazione logica** sono:

- &VS: verbo e soggetto;
- &VO: verbo e oggetto;
- &SV: soggetto e verbo;
- &SO: soggetto e oggetto;
- &OV: oggetto e verbo;
- &OS: oggetto e soggetto;
- &SS: soggetto e soggetto;
- &OO: oggetto e oggetto.

Attraverso i simboli + o - è possibile combinare infatti più attributi insieme. Nel caso di + il token deve fare match con il primo attributo e con tutti quelli preceduti dal +, ma non quelli preceduti da -.

Esistono anche attributi che permettono di caricare liste di valori da uno o più file di testo esterni. Questi sono LOAD e EXPAND e devono essere utilizzati all’interno degli attributi stessi. Ad esempio:

```
DOMAIN(GEOGRAPHY)
{
     SYNCON(
          LOAD "list_cities.txt",
          LOAD "list_districts.txt"
     )
}
```

Oltre a tutti questi operatori, esistono anche i classici **operatori booleani** per combinare tra loro gli attributi.

Questi sono:

- NOT
- XOR
- AND
- OR

Gli operatori possono essere combinati a catena gestendo le precedenze tramite parentesi tonde.

Linguaggio D

Il linguaggio D è un linguaggio di programmazione a tutti gli effetti.

Si possono scrivere programmi attraverso istruzioni che vengono eseguite in sequenza dalla prima all'ultima. Ogni istruzione può modificare oggetti, chiamati variabili, che possono cambiare di stato durante l'esecuzione del programma. Un programma quindi può essere visto come un automa a stati finiti che attraversa un insieme di stati fino a raggiungere quello finale.

Ci sono due elementi fondamentali che possono essere manipolati dal linguaggio D: i domini e i set. I domini, che sono il risultato dell'applicazione delle regole scritte in linguaggio C, sono identificati dal loro nome, dal punteggio e dalle relazioni che hanno con gli altri elementi dell'albero dei domini. Un set invece è una lista di domini che possono essere manipolati come entità singola.

Le operazioni sui domini non vanno ad agire sui punteggi, ma sulle relazioni che intercorrono tra i domini stessi.

Nel linguaggio D sono di particolare importanza gli ACT, ossia gruppi di istruzioni che, una volta eseguite, ricavano un'immagine dello stato iniziale del motore per capire cosa è accaduto durante l'esecuzione; vengono scritti attribuendo ad ognuno di loro un numero crescente.

È possibile compiere operazioni utilizzando degli algoritmi predefiniti, ossia funzioni a default che eseguono delle operazioni sui set creati.

Linguaggio E

Il linguaggio E permette di effettuare l'estrazione da testi di informazioni non strutturati di dati, invece, strutturati, ponendoli all'interno di strutture predefinite.

In un sistema di sintesi i processi di estrazione vengono chiamati **ETL** [16], espressione che definisce tre processi differenti: *extraction*, *transform* e *load*. L'informazione infatti viene estratta dal contesto (tipicamente un file di testo), trasformata e caricata in tabelle del sistema

di sintesi. Durante la trasformazione i dati vengono consolidati, cioè resi omogenei e aderenti alla logica di business del sistema di analisi per cui viene sviluppato.

Simile al linguaggio di categorizzazione, include regole e template. I template, possono essere definiti all'inizio del programma o introdotti tra i blocchi di regole. Le regole devono essere racchiuse in blocchi che definiscono lo scopo e l'estensione, e devono sempre riferirsi a template già definiti.

Template

I template rappresentano la descrizione della tabella che riceverà i dati generati. La sintassi del template è la seguente:

```
TEMPLATE
{
     fields
     [merge rule]
     [attraction rules]
}
```

dove `fields` è una sequenza di definizioni di campi separati da virgole; `merge rule` è una regola di unione; `attraction rules` è una sequenza di regole di collegamento. Ogni template è identificato da un nome, che consiste in una sequenza di caratteri alfanumerici.

Ogni `field` rappresenta una colonna nella tabella, ed è definito attraverso nome e attributo. La sua sintassi è la seguente:

```
@fieldname [(attributes)]
```

Gli attributi che gli sono applicati possono essere:

- [N]ecessary: una regola di estrazione genera un record se riesce a completare almeno tutti i campi che sono contrassegnati da N;
- [C]ardinal: due record che hanno gli stessi valori nel campo cardinale, possono essere combinati tra loro, se sono compatibili;

- [V]alidating: alla fine del processo, un record appare tra i risultati solo se possiede dei valori per tutti i campi contrassegnati da V;
- [S]olitary: il campo viene considerato solo se c'è corrispondenza con la posizione.

Essex

Essex (*Expert System Semantic Engine eXtended server*) è il punto di accesso per l'integrazione della tecnologia di Expert System. Attraverso Essex è possibile integrare la disambiguazione semantica e analisi del testo; e può essere inoltre utilizzato in qualsiasi contesto applicativo, incontrando le necessità dei programmatori o dei clienti che necessitano della tecnologia.

Architettura

Ad alto livello, Essex, espone una collezione di semplici interfacce che permettono l'accesso alle funzioni di analisi base, alla categorizzazione e all'estrazione. Preso un testo, è in grado di fornire informazioni tipo: dati semantici, informazioni di sintesi come concetti rilevanti e domini, dati riguardo la struttura del testo, dati di categorizzazione ed estrazione. Dietro a tutto il processo c'è il disambiguatore semantico che è in grado di attuare un'analisi semantica completa di un testo in input utilizzando i concetti immagazzinati nel Sensigrafo®.

Utilizzando i dati semantici, generati dal disambiguatore semantico, è possibile scrivere regole di categorizzazione ed estrazione personalizzate, per soddisfare particolari requisiti.

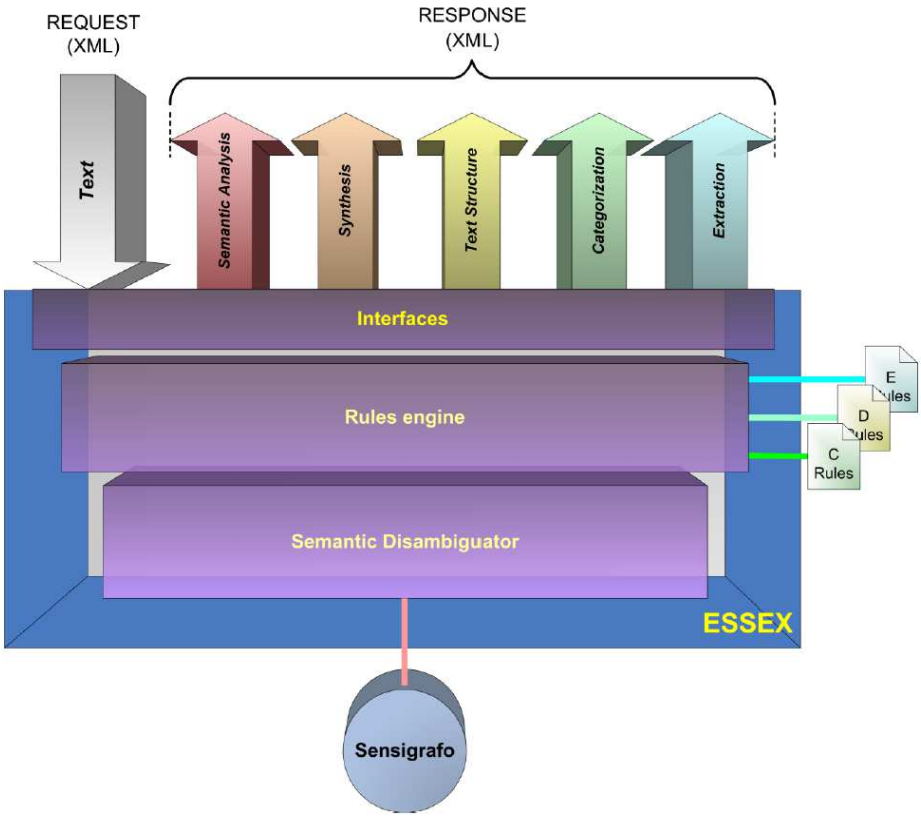


Figura 15 – Architettura di Essex

Per realizzare l'integrazione con altri sistemi, sono state implementate diverse interfacce. Tutte queste interfacce sono molto semplici ed effettuano una precisa sequenza di chiamate ad alto livello:

1) *Init*: per istanziare una nuova istanza di ESSEX;
2) *Analyze*: per spedire ad Essex una richiesta XML contenente il testo in input da essere analizzato e per ottenere una risposta XML contenente i risultati di estrazione;
3) *Destroy*: per eliminare l'istanza di Essex.

La fase di analisi lavora con documenti di tipo XML, la cui struttura è vincolata ad un formato proprietario contenente:

- Il contenuto del documento da analizzare, in una o più sezioni CDATA;

- Una collezione di opzioni di input che definisce i risultati che sono richiesti per l'applicazione specifica.

Essex è stato sviluppato in modo tale da soddisfare la necessità di creare un sistema solido che può essere facilmente esteso per sopportare aumenti di carico. Come le applicazioni distribuite, la modularità è un importante tratto distintivo ed essenziale di Essex insieme alla scalabilità, alla distribuzione della potenza di carico e alla tolleranza ai guasti. Per questa ragione le funzioni semantiche di Essex sono rese disponibili attraverso un servizio che comunica con protocollo TCP/IP.

Lo step finale che segue la creazione di un nuovo progetto Cogito® Studio e lo sviluppo di regole è quello di realizzare un nuovo GSL Essex.

GSL Server

Lato server si trova l'infrastruttura di Essex – **GSL Server**. GSL è un acronimo che sta per "*General Semantic Layer*", un protocollo internet che appartiene al livello applicativo. Il GSL server espone diverse funzioni, è in grado di elaborare documenti fornendo risposte semantiche, ed esegue su un nodo della rete. Per bilanciare il carico è possibile installare diverse istanze dello stesso GSL nell'ambiente di deployment dello stesso GSL server in diversi sistemi.

Un GSL può fornire due tipologie di risposta:

- Risposte dipendenti dal progetto eseguito sul server;
- Risposte non dipendenti dal server.

Le risposte dipendenti dal server sono quelle che discendono dall'origine dello stesso: in particolare, dalle regole di categorizzazione (domini) e/o da quelle di estrazione (record) del progetto Cogito® che l'ha generato.

Viceversa esistono delle risposte standard, che ogni server GSL è in grado di fornire: è in grado di individuare, ad esempio, i nomi di persona, numeri di telefono, luoghi.

Una richiesta di analisi ad Essex è formata da un documento XML suddiviso in quattro aree:

- Area del documento: contiene il testo che deve essere analizzato;
- Area delle richieste: contiene le specifiche di analisi del documento;
- Area dei flags: opzioni varie;
- Area metadati: è un'area jolly dove vengono inseriti dati trasportati che rimangono immutati.

La risposta è nuovamente una stringa XML ben formata.

Alcune delle analisi disponibili sono:

- `SEGMENTS`: fornisce i segmenti individuali del testo;
- `CATEGORIZATION`: fornisce l'output dell'elaborazione derivante dalle regole di categorizzazione presenti in quel GSL;
- `EXTRACTION`: fornisce l'output delle regole di estrazione presenti nel server GSL;
- `DISAMBIGUATION`: fornisce informazioni relative alla disambiguazione semantica effettuata sul testo;
- `SYNTHESIS`: fornisce informazioni di sintesi sul documento, come entità, nomi, luoghi, organizzazioni, etc;

GSL Dispatcher

Il GSL dispatcher è un servizio che si comporta come un gateway tra l'applicazione client e uno o più GSL server, stabilendo una connessione TCP/IP. Il client stabilisce una connessione permanente con il dispatcher e comunica sempre soltanto con lui.

I messaggi che vengono spediti al GSL server, hanno un header seguito da dati formattati per lo specifico server. L'header del messaggio contiene le coordinate del servizio e la funzione richiesta. Questi dati identificano il tipo di GSL e non la specifica istanza, quindi se nello stesso sistema esistono più GSL, è necessario che ognuno venga identificato da un diverso ID, in questo modo il dispatcher potrà scegliere la risorsa e

l'istanza disponibile e quindi eseguire un vero bilanciamento del carico di richieste. Il client per comunicare con il server deve quindi conoscere soltanto la porta in cui è in ascolto il dispatcher e l'id del server con il quale desidera mettersi in comunicazione.

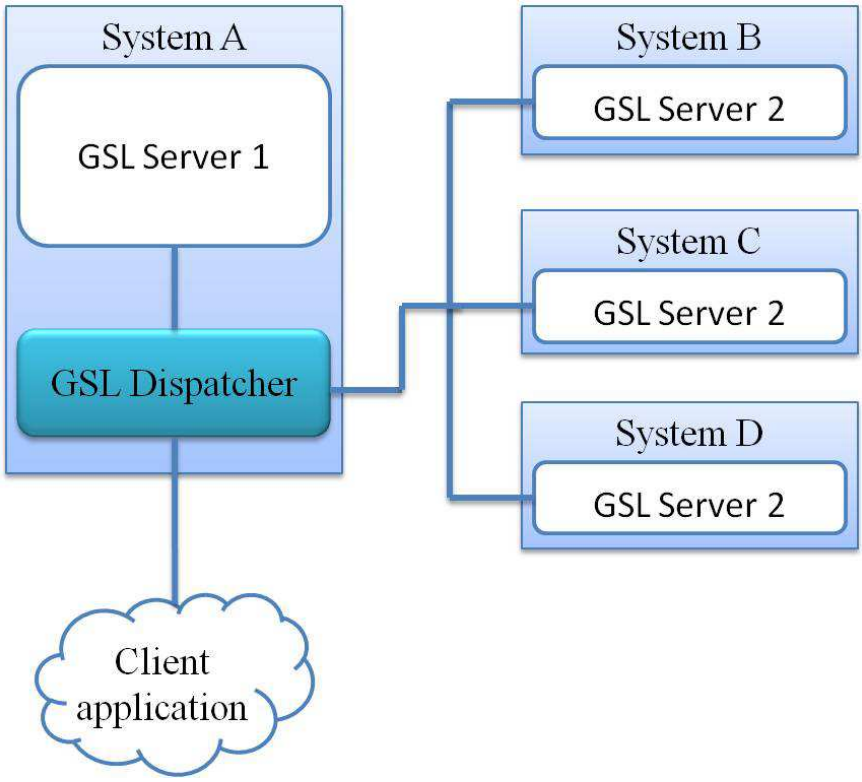


Figura 16 –Architettura GSL Server e dispatcher

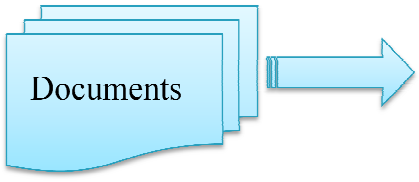
Grazie al dispatcher, l'infrastruttura è molto robusta rispetto ai guasti dei GSL stessi, come ad esempio problemi hardware; il danno, infatti, viene gestito dal dispatcher, che provvede a ridistribuire il traffico verso i GSL rimanenti. Resta comunque lo svantaggio che un problema al dispatcher compromette tutto il sistema. La struttura è, inoltre, piuttosto idonea per le soluzioni cluster native, dove il sistema operativo abilita il server GSL ad essere disponibile sul nodo passivo, quando quello attivo smette di funzionare.

Trek

Trek è il motore di indicizzazione semantica di Expert System; consente l'implementazione di servizi di ricerca potenziati dalla tecnologia Cogito®.

In generale per *indicizzazione* si intende la scansione di una pagina dalla quale se ne estraggono i contenuti che si salvano attraverso parole chiave in un apposito database. In particolare i motori di indicizzazione collezionano, analizzano e immagazzinano dati per facilitare e affinare le procedure di information retrival.

Trek basa la sua indicizzazione su *indici inversi*, il che significa che per ogni parola, viene memorizzato l'elenco degli identificativi dei documenti che contengono tale termine. Questo permette, non solo di localizzare i documenti contenenti determinate parole, ma anche di ordinare questi documenti secondo rilevanza ed accedere quindi al documento in modo estremamente performante.



Term	Doc#
I	1,2,6
Did	3,7,12
The	2,3
Let	12
Told	2,3,4
So	1,6,7
You	3
Killed	2,3,12

Figura 17 – In figura è rappresentato il funzionamento di un indice inverso, dai documenti analizzati vengono estratte le singole parole e vengono affiancate dall'id del documento in cui si trovano.

Questa tipologia di indice non registra però, nessun tipo di informazione riguardante la frequenza delle parole oppure la loro posizione nel testo; viene perciò considerato un indice di tipo booleano. L'indice inverso crea, in fase di memorizzazione, una matrice sparsa a causa del semplice fatto che non tutte le parole sono presenti in ogni documento.

Trek è un sistema *document oriented*: si basa quindi sullo store di documenti nel database in un formato predefinito, che nel caso di Trek è XML – based. I documenti sono indirizzati con una chiave univoca che li rappresenta, e che può quindi essere utilizzata nelle ricerche. La caratteristica principale di questo sistema è quella di poter ricercare i documenti in base al loro contenuto.

Trek possiede un proprio file system basato su corpus, chiamato CFS. Ogni corpus ha la sua cartella dove vengono immagazzinati tutti gli indici inversi; l'elaborazione produce anche un albero di ricerca, gestito dal CFS, tramite il quale si possono eseguire ricerche efficienti sui testi.

Trek è composto da due moduli principali: Trek Index e Trek Searcher.

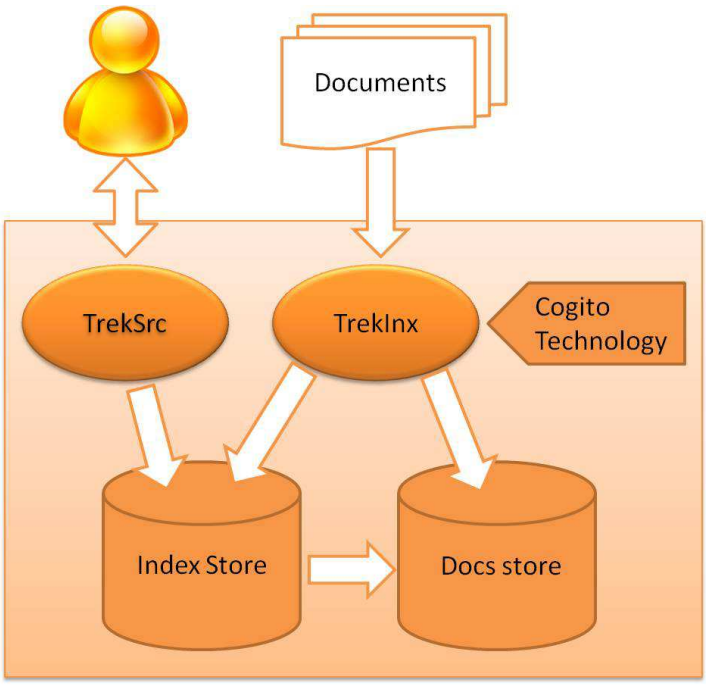


Figura 18 – Schema delle interazioni di Trek

Trek Index

Trek Index è un servizio presente in ogni CFS che gestisce il processo di indicizzazione. Durante il processo di indicizzazione, l'input di Trek Index è un XML contenente la risposta di Essex, e attraverso questa riesce a disambiguare creando indici inversi su lemmi, keyword, syncon, persone, etc. L'organizzazione dei documenti viene scelta dal client che richiede l'indicizzazione. Esiste inoltre, la possibilità di utilizzare KEYVALUE per i dati dinamici che devono essere modificati frequentemente.

Trek Searcher

Trek Searcher, invece, è un servizio che gestisce tutte le query sugli indici. Ogni CFS ha il proprio Searcher, ma la ricerca sull'albero CFS è completa in quanto ogni istanza del Searcher comunica con le altre attraverso un protocollo interno, in modo da effettuare una ricerca completa in tutto il sistema di corpus.

Esistono tre tipi principali di query:

- *Boolsearch*: attraverso la quale cerco esattamente il termine specificato nella query;
- *Proximity*: che permette di selezionare una collezione di documenti con una boolean search ottenendo in output una lista di token che sono contenuti nei documenti presi in analisi;
- *Ranked*: che permette di ricercare alcuni termini attraverso un algoritmo che cerca il match migliore (molto simile ad una ricerca di Google).

Le query sono suddivise in due grandi gruppi: le query di analisi e quelle standard (quest'ultime verranno approfondite in appendice A).

Query di analisi

Le query di analisi possibili in Trek sono *proximity* e *relations.* Nell'ambito di questo progetto è stata utilizzata la query di analisi di tipo *proximity* per ricavare la frequenza di un lemma all'interno di documenti facenti parte di una determinata categoria.

Le query di analisi e quelle standard si comportano in modo analogo, ovvero vengono eseguite in modo sincrono alla chiamata “executeQuery”, e restituiscono alla fine dell’elaborazione o il messaggio “PROCESSED” oppure un messaggio di errore in caso di malfunzionamento.

In generale la ricerca tramite *proximity* [17], nell’elaborazione di un testo, cerca i documenti in cui due o più termini separati che fanno matching tra loro, si trovano al di sotto di una distanza specificata, dove per distanza si intende il numero di parole o di caratteri intermedi. Oltre alla vicinanza, in alcuni implementazioni presenti in letteratura, viene anche imposto un vincolo di ordine tra le parole stesse. Viene considerata una forma di ricerca avanzata, in quanto va oltre il semplice matching delle parole. Per esempio, una ricerca può essere usata per cercare “red brick house” e può far match con frasi quali “red house of brick” oppure “house made of red brick”. Limitando la ricerca con la proximity, queste frasi fanno match tra di loro evitando documenti in cui le parole sono sparse nella pagina e non correlate tra loro.

La proximity applicata a questo progetto richiede l’analisi dei corpus per restituire il syncon associato al lemma o alla keyword più frequente.

Di seguito è riportata la query XML con la quale si interrogano i corpus.

```
<?xml version="1.0" encoding="UTF-8"?>
<command name="proximity">
     <param name="cp" format="xml">
          <corpus>
            <c>/archiviocorriere_201303151700</c>
            <c>/webnews_201303151232</c>
            <c>/test_iptc_201304150922</c>
          </corpus>
     </param>
     <param name="scope_type" format="txt">T</param>
     <param name="lo" format="txt">1</param>
     <param name="hi" format="txt">1</param>
     <param name="token" format="xml">
          <boolean>
           <rules>
           <rule table="EXTRA" field="L"opl="AND">
               <value><![CDATA[soggetto]]></value>
             </rule>
             <rule table="EXTRA" field="100">
```

```
                <value><![CDATA[02001000]]></value>
              </rule>
            </rules>
          </boolean>
     </param>
     <param name="output_list" format="xml">
          <extra>
               <e>2</e>
          </extra>
     </param>
</command>
```

Il tag `<corpus>` contiene i corpus sui quali si vuole eseguire la proximity. La ricerca che si vuole effettuare ha come scope i token (T), e viene specificato all'interno del tag `<param>`. Nel caso di scostamento T la booleana potrà contenere solo K (keyword), L (lemma), S (sentence) ed EXTRA tutti legati strettamente ad un token della frase. Nel caso riportato come esempio all'interno del tag `<boolean>` è presente la ricerca vera e propria: in questo caso stiamo cercando il lemma "soggetto" in AND con il codice di categoria "02001000" (contenuto nel progetto). Il limite massimo e minimo dello scostamento della proximity è definito all'interno dei tag `<param>` chiamati `"hi"` e `"lo"` e in questo caso è 1 (si vuole infatti cercare di individuare la parola stessa che stiamo cercando, ma lo scostamento minimo è 1). Per configurare una determinata lista di output, si utilizza il tag `<extra>`, in particolare in questo caso la lista da calcolare è la 2 ossia i lemmi.

Il progetto IPTC

Introduzione

All'interno di questo capitolo viene descritto il progetto oggetto di questa tesi. Partendo dalla presentazione di IPTC e descrivendo qual è la sua funzione; viene quindi introdotto il progetto di Expert System, punto di partenza di questo lavoro di tesi. A seguire verrà descritto il progetto vero e proprio, descrivendone gli obiettivi.

L'IPTC

L'IPTC, **International Press Telecommunications Council**, è un consorzio che include le agenzie di stampa più importanti del mondo, editori, fornitori di notizie del settore. Sviluppa e mantiene le norme tecniche per lo scambio di notizie che vengono utilizzate da quasi tutte le organizzazioni del mondo. L'IPTC ha creato e mantiene dei set di concetti o tassonomie, che vengono assegnati come metadati alle news, qualsiasi sia il loro formato (di testo, fotografie, file audio o video, stream). Questo permette di avere una codifica consistente delle notizie che rimane coerente anche nel corso del tempo. Ad oggi l'IPTC viene considerato come standard nella categorizzazione delle news e un modello di interscambio di informazioni.

Cenni storici

È stato sviluppato dall'omonimo consorzio negli anni novanta per accelerare lo scambio internazionale di notizie tra i quotidiani e le agenzie di stampa. La specifica IIM (Information Interchange Model) completa comprende una struttura di dati complessi e un insieme di definizioni di metadati.

Fin verso la fine degli anni settanta [19] le attività di IPTC si sono principalmente concentrate sullo sviluppo e pubblicazione di standard di

settore per lo scambio di informazioni. Il primo standard, l'IPTC 7901, ha fatto da ponte tra le ere di telescriventi e computer. Negli anni ottanta lo sviluppo iniziò su uno standard (il Modello di interscambio di informazioni) destinato a lavorare meglio con i nuovi sistemi di editing computerizzati. In particolare, IPTC definisce un insieme di attributi di metadati IIM che possono essere applicati alle immagini. Questi sono stati definiti originariamente nel 1979 e modificati in modo significativo nel 1991 per far parte della IIM (Information Interchange Model), ma il concetto in realtà avanza nel 1994 quando Adobe Systems definisce una specifica pratica per l'incorporamento dei metadati in un file di immagine digital, creando le "Intestazioni IPTC". Nel 2001, Adobe introduce "Extensible Metadata Platform" (XMP), che è uno schema XML per gli stessi tipi di metadati come IPTC, ma è basato su XML/RDF ed è quindi intrinsecamente estensibile. Lo sforzo ha generato una collaborazione con IPTC, producendo alla fine l'IPTC Core Schema for XMP, che unisce i due approcci per incorporare i metadati.

La tassonomia IPTC

La tassonomia è formata da **News Code**, ossia codici univoci che rappresentano un concetto usato per categorizzare il contenuto delle notizie. Per l'industria giornalistica un requisito fondamentale è quello di essere in grado di inserire in una notizia dei metadati riguardanti il contenuto della singola notizia pubblicata. Per questo motivo è nata la necessità di avere dei codici univoci che fossero uguali per tutti e non soggettivi, che portassero con sé anche la semantica. I codici possono essere utilizzati in qualsiasi lingua, l'unica cosa che necessita di traduzione è la descrizione del codice. Le modifiche al codice avvengono soltanto su proposta formale dei membri dell'IPTC, ma chiunque utilizzi la tassonomia e sia interessato a creare una modifica può proporla.

Un codice rappresenta **un concetto**. Un concetto è qualsiasi cosa alla quale ci si vuole riferire; i concetti si dividono in due grandi categorie: i concetti generici e nomi di entità (che in particolare possono essere: una

persona, un luogo, un evento, un'organizzazione, il nome di un prodotto, il nome di un oggetto o qualsiasi altra notizia correlata).

Ogni codice è espresso attraverso un **QCode**, un formato speciale utilizzato dall'IPTC che prevede tipicamente una stringa seguita da due punti e da un'altra stringa, ad esempio: "short name for the controlled vocabulary":"code of the concept" come subj:06011000.

Il tipo di un concetto permette di raggruppare in modo logico tutti i concetti simili, indipendentemente dal tipo di vocabolario al quale il concetto appartiene. Esempi di tipo di concetto potrebbero essere: persona, organizzazione, lingua, settore di attività o geografia. Il vocabolario è un insieme di codici.

La tassonomia IPTC relativa ai subject code è formata da 1400 voci suddivise in un albero a 3 livelli.

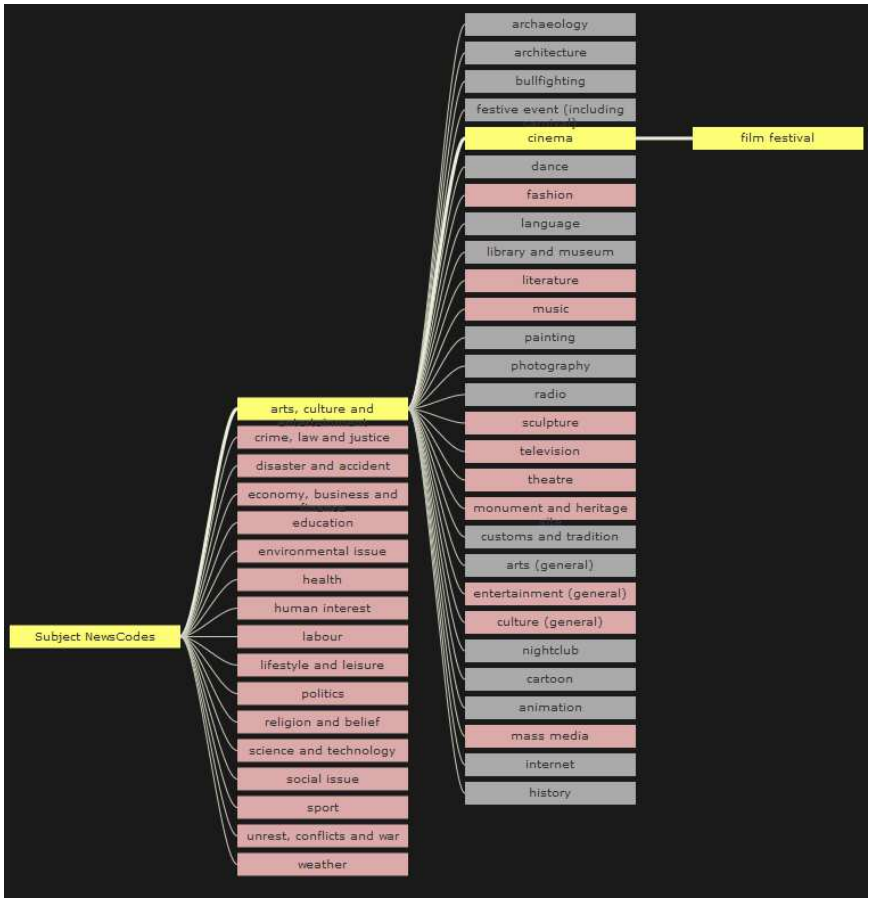


Figura 19 – Nell'immagine l'albero delle categorie IPTC: si può notare come l'albero sia organizzato su tre livelli.

Expert System ha sviluppato un progetto per la categorizzazione delle notizie in italiano sulla base di questa tassonomia. In questo modo riesce, attraverso la tecnologia Cogito® a classificare qualsiasi news che gli si chiede di analizzare, definendo immediatamente l'argomento.

Chiaramente un progetto di simile portata è in continua evoluzione a causa della natura intrinseca delle notizie e richiede un notevole sforzo da parte dei linguisti che scrivono le regole.

Per agevolare quindi la creazione dei progetti in nuove lingue sarebbe opportuno trovare un metodo di traduzione del contenuto delle regole stesse; in questo modo il team di linguisti che deve creare il nuovo progetto avrà la possibilità di partire da una struttura nota, ponendo l'attenzione sull'analisi delle notizie più dettagliate.

Requisiti del progetto

L'obiettivo di questo progetto di tesi è quello di sperimentare una procedura per tradurre automaticamente, dall'italiano ad un'altra lingua, le regole di categorizzazione, create con Cogito® Studio, del progetto IPTC che classifica il contenuto delle news in lingua italiana. La lingua destinazione della traduzione è al momento il francese. I file oggetto della traduzione sono tutti i file "`.efo`" facenti parte del progetto in questione.

Non sono stati esplicitati requisiti in termini di occupazione di risorse e tempo di elaborazione; è invece richiesta precisione nella traduzione del contenuto delle regole, in modo tale che i sorgenti tradotti nella nuova lingua ed inseriti in un progetto possano essere compilati ed utilizzati per la categorizzazione di articoli in francese.

Non è richiesta la creazione di un'interfaccia grafica, la procedura dovrà infatti produrre in output solamente i file sorgenti di categorizzazione tradotti in francese.

Analisi dei requisiti

Il sistema richiesto rientra nel dominio dei sistemi di traduzione automatica. La differenza fondamentale che separa i sistemi di traduzione classici da quello richiesto è che in questo caso non si andrà a tradurre un interamente un testo, nel quale ogni parola è contestualizzata; ma si avrà a che fare con un linguaggio di programmazione che dovrà rimanere invariato. La traduzione infatti dovrà riguardare solamente il contenuto delle regole di categorizzazione, che di fatto è rappresentato da id e termini.

Non essendo necessaria un'interfaccia grafica, l'utente non svolge un ruolo attivo sul sistema, ma si limita a selezionare i file che devono essere oggetto di traduzione, e prelevare quelli tradotti.

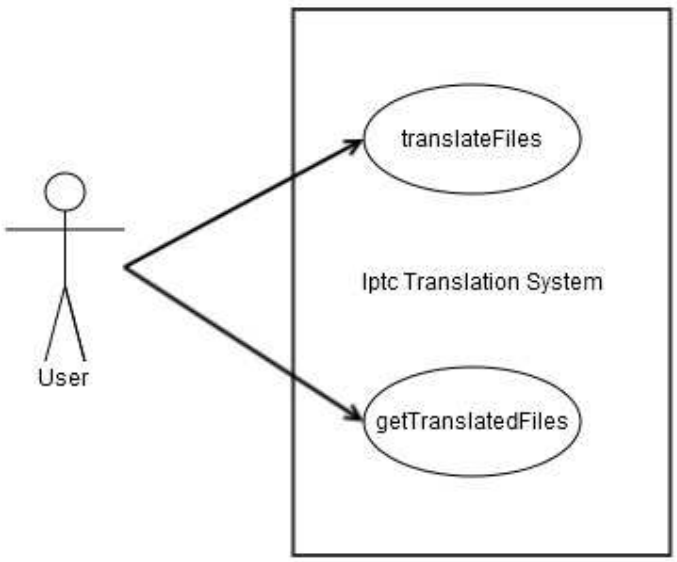


Figura 20 – L'utente non interagisce con l'applicazione se non richiedendo la traduzione di sorgenti e prelevando i file tradotti.

La lingua di destinazione non presenta particolari problemi di rappresentazione, essendo comunque una lingua europea che presenta delle affinità con l'italiano, a causa della discendenza comune.

Per la realizzazione del progetto, è necessario considerare il vantaggio messo a disposizione anche dagli strumenti di Expert System. Di particolare rilevanza è il Sensigrafo® della lingua francese. Nella realizzazione del Sensigrafo® francese infatti, viene fatto un mapping molto accurato tra i syncon italiani e quelli francesi. Il Sensigrafo® italiano esprime sempre un punto di partenza fondamentale per l'evoluzione delle altre lingue, vista la sua completezza e l'importante investimento in anni uomo per la sua realizzazione.

Analisi del problema

Per effettuare la traduzione in esame bisogna partire dall'analisi dei file sorgenti `.efo` che contengono le regole di categorizzazione: questi file infatti rappresentano l'input del sistema, e dopo l'elaborazione devono presentare la stessa struttura di partenza, con contenuto modificato.

Figura 21

La cartella contenente i file sorgenti, possiede un file per ogni argomento dell'albero di primo livello di IPTC. Le regole contenute nei file, presentano la stessa struttura descritta nel capitolo riguardante Cogito® Studio, in figura 14. Ogni News Code viene modellato da molte regole diverse all'interno dello stesso file.

Osservando le regole, si può notare che il News Code della tassonomia IPTC è contenuto all'interno del dominio, e identifica la categoria a cui deve essere associata la news quando la regola scatta. All'interno di un blocco domain ci possono essere uno o più attributi (già introdotti

all'interno del capitolo riguardante Cogito® Studio), il cui contenuto sarà oggetto della traduzione.

Di seguito viene riportato un esempio di regola estratta dal progetto, la regola riguarda il dominio 01005000, che corrisponde all'argomento "cinema" nella tassonomia IPTC. All'interno del dominio sono presenti le regole oggetto della traduzione, in particolare tutto ciò che è colorato in grigio e l'id all'interno dell'attributo ANCESTOR.

```
SCOPE SECTION (BODY)
{
      // attore
      DOMAIN(01005000:NORMAL)
      {
            !LEMMA("giocatore")
            <
            ANCESTOR(13271)
            –
            LEMMA("protagonista")
            >
            !LEMMA("squadra") // cineasta
            AND NOT
            LEMMA("teatro","episodio","teatrale")
      }
}
```

Come si può notare anche dalla regola qui riportata, sono spesso presenti all'interno dei sorgenti commenti single-line, che specificano, soprattutto per i syncon id, il concetto preso in considerazione. Esistono inoltre numerosi commenti a blocco, in quanto il sorgente è in continua evoluzione e per comodità dei linguisti le regole vengono mantenute sotto forma di commento per essere reintrodotte su necessità. Proprio per questo motivo è stato deciso di tradurre anche il contenuto delle regole commentate, in modo tale che chiunque si trovi ad aggiornare o sistemare le regole sorgenti sia in grado di capire l'evoluzione che hanno avuto le regole attuali e inoltre poter introdurre alcune regole commentate in precedenza.

Le regole di categorizzazione di Cogito® si possono suddividere in due grandi insiemi: le regole che hanno come argomenti i syncon id del Sensigrafo® e le regole che utilizzano lemmi o le keyword.

Sappiamo anche che esiste un mapping diretto i syncon id del Sensigrafo® italiano e quello di lingua francese, e che questi mapping sono accessibili all'interno di un file di testo a disposizione. Da questo si può facilmente dedurre che la traduzione nella nuova lingua dei termini contrassegnati da un syncon id è semplice: basta controllare nel file di mapping la corrispondenza di ogni syncon id italiano con quello nella lingua di destinazione. Ogni id italiano corrisponde ad uno o più id nella lingua di destinazione.

Per quanto riguarda invece lemmi e keyword la traduzione è più complessa: si tratta infatti di capire quale significato del termine italiano stiamo considerando.

Approccio risolutivo

La soluzione parte sicuramente dall'approccio adottato per i syncon id: grazie al file di mapping, tutti i syncon id italiani all'interno delle regole possono essere mappati con i loro corrispondenti francesi con massima precisione.

Per la risoluzione che riguarda invece lemmi e keyword, come già introdotto, la traduzione è più complessa: si tratta infatti di capire quale significato del termine italiano stiamo cercando. Ad esempio: se all'interno del listato delle regole ho la parola "calcio" a quale significato di calcio si sta riferendo la regola? In questo caso non possiamo infatti affidarci al mapping tra i sensigrafi, occorre trovare una strategia differente.

La soluzione adottata pone le sue basi sul dizionario bilingue italiano – francese: ricercando il termine all'interno del dizionario, posso sicuramente tradurre con precisione i termini che hanno solamente una traduzione possibile. Lo stesso approccio si può applicare anche a quei termini che, ricercati all'interno del Sensigrafo®, hanno soltanto un syncon id associato: risalendo all'id e poi controllando il suo corrispondente nel

file di mapping (che deve essere uno soltanto, altrimenti la traduzione non viene accettata), riusciamo a tradurre con una buona precisione il termine.

Si tratta quindi di capire cosa fare con i termini che hanno più di una traduzione: lo scoglio da superare è quello di trovare una strategia valida per permettere alla procedura di disambiguare correttamente, senza sbagliare il significato. Osservando le regole, l'unico contesto al quale possiamo fare affidamento è quello che deriva dal News Code presente all'interno del DOMAIN: infatti all'interno dei sorgenti scritti per la categorizzazione non esistono altri riferimenti per decidere a quale tipo di contesto fare riferimento. Per dargli un contesto si è pensato di creare dei corpus di articoli di giornale, indicizzati tramite Trek e classificati nelle categorie definite da IPTC. Ricercando all'interno di questi corpus, prendendo la categoria specificata nel dominio, possiamo trovare il lemma o la keyword associata al syncon id più frequente. In questo modo si riesce a cercare il termine contestualizzandolo nella categoria di cui dovrebbe far parte l'articolo e prendendo il syncon id più frequente riesco a disambiguare il significato. Dopo aver estratto il syncon id, ritorno al file di mapping dove l'id è collegato ai corrispondenti francesi. Una volta estratto l'id francese, interrogando il Sensigrafo®, si possono estrarre i termini francesi che gli sono associati. In questo modo ho ottenuto una traduzione contestualizzata dal dominio di appartenenza del termine. Le traduzioni estratte però, vengono confrontate con quelle estratte dal dizionario: con questo ulteriore controllo si evita di estrarre parole obsolete o incorrette e la traduzione viene fatta sempre con il termine più appropriato riportato appunto nel dizionario.

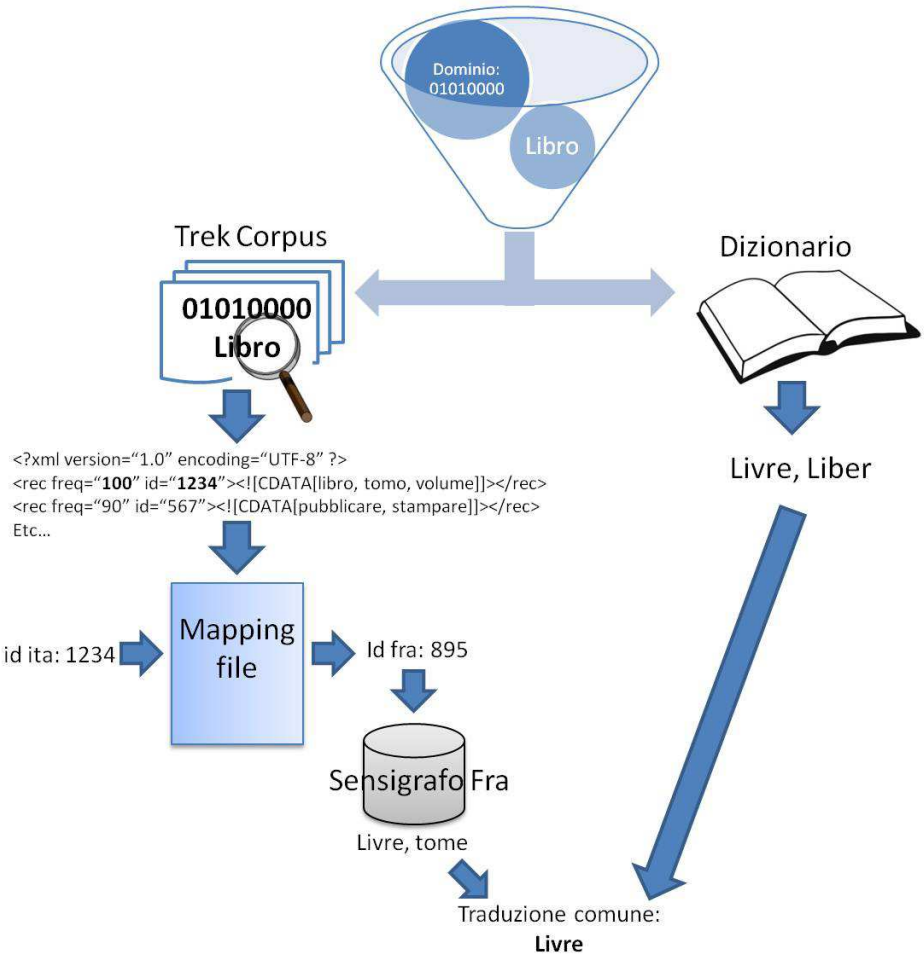


Figura 22 – Schema del processo di traduzione proposto come soluzione.

Occorre precisare che è stata considerata questa strada per trovare un contenuto significativo, in quanto la tecnica non presuppone la necessità di tradurre testi o di creare corpus bilingue precisi. Infatti, l'alternativa per determinare la traduzione corretta dal contesto della parola è quella di avere a disposizione un corpus bilingue, tradotto in modo molto preciso, e determinare l'allineamento tra la parola italiana e la sua traduzione, come avviene con google translate.

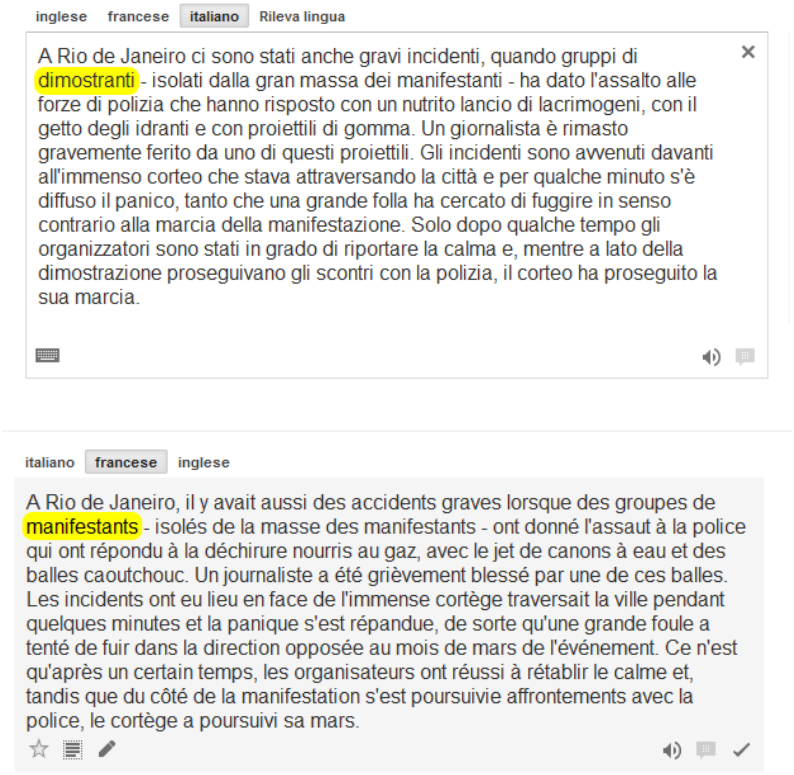


Figura 23 – La figura mostra l'allineamento che effettua Google translate.

Questa tecnica oltre a presentare numerosi svantaggi, quali la necessità di disporre di traduttori professionisti per la creazione del corpus e di creare un corpus di notizie così ampio da poter coprire la quantità di termini presenti; non è nemmeno del tutto applicabile. Infatti è molto difficile riuscire a capire di quale contesto fa parte la parola contenuta nella regola, se non sfruttando il suo dominio di appartenenza. Quindi questa tecnica di traduzione è stata proposta per ovviare ad una serie di problemi derivanti dagli approcci classici, e per adattare al meglio la soluzione al problema sfruttando gli strumenti a disposizione.

Il dizionario bilingua italiano – francese

Per quanto riguarda il dizionario, è stato richiesto di utilizzare i file di testo estratti dal database che lo contiene. Estraendo con una query opportuna i dati, notiamo che per avere tutte le accezioni dei termini e le

locuzioni di accezioni abbiamo la necessità di suddividere i termini su tre file differenti quali: traduzioni di accezione, traduzioni locuzioni accezione e traduzioni locuzioni lemma. Osservando i file notiamo che, anzitutto i termini italiani contengono gli accenti fonici, che nei file sorgenti di categorizzazioni non sono chiaramente presenti; inoltre notiamo che nei file che contengono le locuzioni di accezione e quelle di lemma, la traduzione contiene sia simboli speciali per la corretta visualizzazione delle parole, sia il trattino nella posizione dove si dovrebbe trovare la parola stessa: ad esempio

```
abito n.m.  NULL  prendere{\plain ,} vestire l'–  prendre
l'habit
```

Quindi ricercare l'accezione della locuzione "abito", "vestire l'abito", sarebbe impossibile.

È opportuno quindi effettuare una pre – elaborazione dei file di testo contenenti le parole, dove si eliminano gli accenti fonici e si ripulisce il testo da simboli speciali utili solo alla visualizzazione (all'interno del'applicativo del dizionario). Inoltre i file delle locuzioni di accezione devono essere riorganizzati, eliminando campi superflui e ponendo al posto del "-" la parola stessa. Questo renderà più agevole la lettura dei file in fase di traduzione, e permetterà di tradurre anche modi di dire.

Realizzazione del progetto

Introduzione

In questo capitolo è riportata la metodologia seguita nella realizzazione dell'applicazione. Di seguito viene introdotto inizialmente il modulo necessario a portare i file del dizionario in condizione da essere letti agevolmente dall'applicazione di traduzione. A seguire è descritta tutta la struttura del traduttore.

Il dizionario

Prima di tutto è necessario introdurre la procedura utilizzata per manipolare i file del dizionario per renderli idonei all'utilizzo da parte del traduttore. I file di testo in particolare sono tre: un file che contiene le accezioni, un file che contiene le locuzioni di accezione e un file che contiene le locuzioni di lemma. È stato richiesto di utilizzare i file per il dizionario, evitando di mappare le query all'interno del progetto di traduzione, in modo da non rallentarlo ulteriormente.

La procedura di manipolazione dei file è indipendente dal traduttore, e quindi sarà necessario sistemare i file del dizionario nel modo richiesto prima di mettere in funzione il traduttore vero e proprio.

La procedura è molto semplice: per quanto riguarda i file delle accezioni, è necessario pulirli da caratteri speciali, accenti, e simboli legati alla visualizzazione; per quanto riguarda invece i file delle locuzioni di accezione e delle locuzioni di lemma, è necessario sostituire l'accezione all'interno del suo contesto, al posto del "-", eliminando anche ulteriori campi che servono per precisare, nel dizionario, la natura del termine (come ad esempio il tipo grammaticale).

Linguaggio di programmazione

Il linguaggio con il quale viene realizzata questa procedura è Java 7.

Struttura

La struttura possiede una classe dedicata alla lettura di un file di configurazione, in modo tale da non cablare nel codice la posizione dei file del dizionario da tradurre. Esiste quindi un file di configurazione dove è possibile cambiare la posizione dei file sia di input che la cartella di destinazione in output.

Siccome il contenuto dei file da modificare è differente, da un file all'altro, è necessario personalizzare la struttura delle classi che li modificano.

Interazioni

La procedura è stand alone, non ha quindi la necessità di comunicare con altri componenti esterni.

Implementazione

All'avvio del main la procedura inizia a prendere in input un file per volta in modo sequenziale, partendo dal file delle accezioni, la cui locazione viene letta dal file di configurazione; una volta finita la manipolazione del primo documento, passerà ai successivi e così via fino a che l'elaborazione non è stata completata per tutti.

File delle accezioni

Per quanto riguarda il file delle accezioni, la classe *AccentModifier* si occupa della sua modifica.

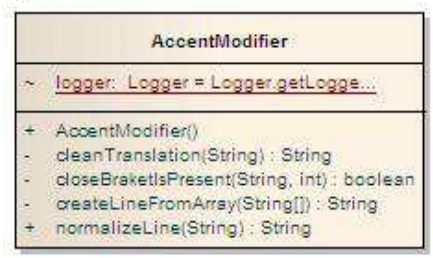


Figura 24 - UML classe AccentModifier

Le frasi lette dal file delle accezioni e prese in input dalla classe sono del tipo:

“mài avv. NULL jamais”

dove la prima parola è quella italiana, seguita da una tabulazione, poi il tipo grammaticale della parola, una tabulazione, un campo a NULL (si tratta di un campo di definizione che però, rimane tale in tutte le entry del dizionario, ma è necessario alla query di estrazione), una tabulazione ed infine la traduzione francese.

Con uno *String Tokenizer* isolo la prima parola, che è quella italiana; successivamente salvo in una variabile l’ultimo carattere della parola che è l’unico che deve mantenere il suo accento (l’accento su l’ultima lettera infatti è un accento tonico e non fonico come quelli che stiamo cercando di eliminare), per poi aggiungerlo nuovamente alla fine dell’elaborazione. Attraverso la funzione *normalizer*, con parametro *NFD* (decomposizione canonica) è possibile isolare dai singoli caratteri, i marker degli accenti, e attraverso una *replaceAll*, li elimino tutti sostituendogli “vuoto”. Di seguito viene riportato il codice di questo passaggio:

```
String itaWordNew =
Normalizer.normalize(itaWordOld,Normalizer.Form.NFD);
itaWordNew=itaWordNew.replaceAll("[^\\p{ASCII}]", "");
```

Da notare che la regular expression individua tutti i caratteri che non siano caratteri ASCII, quindi sostanzialmente tutti gli accenti isolati dai caratteri vengono eliminati dalla replace.

Infine aggiungo nuovamente alla parola modificata l’ultimo carattere.

Prima di ricostruire la stringa e richiedere la scrittura su un nuovo file, la procedura elimina tutti i caratteri speciali ed eventualmente i simboli inseriti per la corretta visualizzazione delle stringhe nel programma proprietario del dizionario. Questo avviene attraverso la funzione *cleanTranslation*.

File delle locuzioni di lemma e delle locuzioni di accezione

La struttura di classi che modifica il file delle locuzioni di lemma e quello delle locuzioni di accezione, è composta da un'interfaccia *ILocuzioniModifiers*, che espone un metodo *modifyLine*.

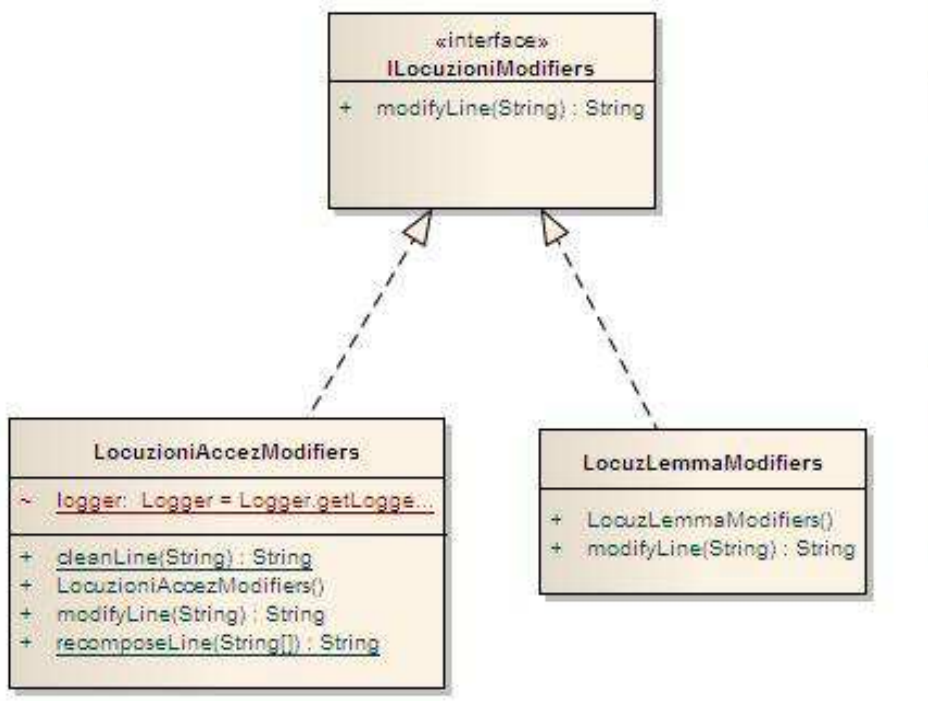


Figura 25 – UML delle classi di modifica dei file delle locuzioni

Le classi *LocuzioniAccezModifiers* e *LocuzLemmaModifiers*, implementano l'interfaccia *ILocuzioniModifiers*, andando a definire il contenuto della funzione *modifyLine*.

In particolare la classe *LocuzioniAccezModifiers* legge le stringhe del file "TradLocuzioniAccezioneITAFRA.txt" prendendo in input linee come questa:

```
"abbandono n.m. NULL – di nave  abandon de navire"
```

La linea viene modificata eliminando il NULL e il tipo grammaticale, e sostituendo la parola italiana al posto del "-" quindi la stringa manipolata risulterà:

“abbandono abbandono di nave abandon de navire”

Capita però che la locuzione italiana possieda anche l’articolo davanti al termine e che quindi risulti difficile ricercarla in modo automatico. È necessario quindi eliminarlo e, a questo scopo, la classe si serve di una classe pubblica statica *Article* che definisce quali sono gli articoli italiani.

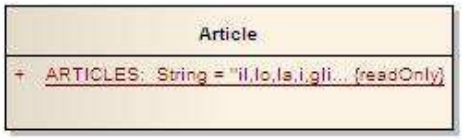


Figura 26 – UML della classe statica Article che definisce gli articoli della lingua italiana.

La classe *LocuzLemmaModifiers* si comporta allo stesso modo con le stringhe del file “TradLocuzioniLemmaITAFRA.txt”. La necessità di separare le procedure deriva dal fatto che il file presenta un campo in meno (il campo del tipo grammaticale), infatti le frasi che prende in input sono del tipo:

“acqua NULL essere con l'– alla gola avoir le couteau sous la gorge”

Il progetto di traduzione

L'approccio risolutivo parte innanzitutto, dalla definizione di un **parser** che dovrà scorrere i sorgenti di Cogito® e isolare il contenuto delle regole che incontra, elaborarle, passandole ad un **traduttore** che le trasforma, ed infine ricomporle così come si presentavano al momento della lettura.

Il parser ed il traduttore sono quindi due elementi fondamentali di questo progetto.

Si ha la necessità, inoltre, di tenere traccia della ***recall***, per capire a quante parole si riesce a dare una traduzione. Per quanto riguarda invece la ***precision***, non è facile in questo caso specifico valutare automaticamente la precisione della traduzione. È il caso quindi di tenere traccia della metodologia utilizzata per tradurre i termini e della quantità di termini tradotta con la stessa metodologia in modo da avere accesso ad una serie di dati statistici: per questo motivo, un altro elemento importante della struttura del progetto è un'entità che riesce a seguire la traduzione e a memorizzare come viene effettuata.

Per effettuare la traduzione vera e propria sono stati sperimentati e combinati tra loro diversi metodi:

- Dizionario bilingue italiano – francese;
- Traduzione con contestualizzazione attraverso corpora;
- Utilizzo del traduttore automatico Bing;
- Sintesi di regole grammaticali per tradurre con più precisione casi particolari derivanti dall'italiano;
- Distanza di Levenshtein;
- Coefficiente di Dice.

Struttura

La struttura del progetto presenta quindi tre entità principali che si occupano dell'esecuzione e risoluzione del problema: il parser, il traduttore e l'analizzatore.

Osservando le regole, come spiegato in precedenza, possiamo suddividerle due categorie principali: quelle che contengono termini e quelle che contengono syncon id. Le regole quindi andranno strutturate secondo gerarchia, partendo da una regola generica ad alto livello e poi scendendo a livello più basso specificando in modo sempre maggiore il tipo di regola, e trattando allo stesso modo quelle simili.

I metodi esposti sopra per la traduzione, saranno moduli indipendenti tra loro, alcuni dei quali potranno essere attivati o disattivati secondo necessità.

Il progetto presenta anche un file di configurazione, dove poter modificare percorsi specifici dei file, nomi dei file, id e porte delle risorse esterne con le quali è necessario interagire.

Interazioni

Il progetto si deve interfacciare esternamente con alcune entità quali Trek, Bing e Gsl Essex. In particolare necessita di Trek per ricevere informazioni sulla disambiguazione basata sui corpus; da Bing per la traduzione di termini e da Gsl Essex per la ricerca di termini all'interno del Sensigrafo®.

Per quanto riguarda **Trek**, è necessario creare il client attraverso il quale è possibile comunicare con il servizio. La procedura che è stata seguita comprende l'utilizzo di un corpora di oltre 200 mila articoli di giornale in lingua italiana, acquisiti dagli archivi delle principali testate giornalistiche, indicizzati tramite Trek e classificati nelle categorie definite dalla tassonomia IPTC. Per reperire i termini di cui si necessita nella traduzione, è necessario interrogare il motore di indicizzazione, il quale controllerà se il termine è presente nelle sue fonti che ha già indicizzato e disambiguato in precedenza. La creazione del client che comunica con Trek avviene sulla base di un file wsdl, che specifica la struttura e le funzioni che il client deve contenere e le funzioni che deve esporre. Il wsdl in particolare è piuttosto agevole quando si realizzano progetti C#, ma non per Java: infatti la creazione del client richiede di essere completata manualmente

dal programmatore. Nell'ambito del progetto, è stata utilizzata una libreria già creata e messa a disposizione dall'azienda.

Il traduttore automatico **Bing** invece è necessario come supporto alla traduzione. Viene utilizzato quando non si riesce a dare una traduzione possibile dal dizionario bilingue. Per interagire con Bing, utilizzeremo una libreria open source, trovata in rete, chiamata "*microsoft-translator-java-api*" [20]. La libreria fornisce un wrapper in Java per l'utilizzo delle API del traduttore Microsoft, ed è stata creata per ovviare alla mancanza delle API di Google translate che non sono più disponibili gratuitamente. Utilizza l'interfaccia di chiamata AJAX (v2) messa a disposizione da Microsoft.

La libreria espone un metodo statico *execute* che richiede in ingresso come parametri la stringa da tradurre, la lingua di partenza e quella di destinazione; dopo aver opportunamente settato i parametri riguardanti il client id e il client secret. Il metodo *execute* crea prima di tutto una stringa che contiene i parametri necessari al traduttore, poi inserisce tutto in un URL che spedisce attendendo la risposta contenente il testo tradotto.

```
public static String execute(final String text,
final Language from, final Language to) throws
Exception {

 final String params = (apiKey != null ?
 PARAM_APP_ID
 + URLEncoder.encode(apiKey,ENCODING) : "")
 + PARAM_FROM_LANG
 + URLEncoder.encode(from.toString(),ENCODING)
 + PARAM_TO_LANG
 + URLEncoder.encode(to.toString(),ENCODING)
 + PARAM_TEXT_SINGLE
 + URLEncoder.encode(text,ENCODING);

 final URL url = new URL(SERVICE_URL + params);
 final String response = retrieveString(url);
 return response;

}
```

L’interazione con **Essex** invece è importante per reperire i termini dal Sensigrafo®, sia italiano che francese. L’entità che si occupa di mappare l’interazione con Essex, dovrà esporre le query che sono possibili verso il Sensigrafo®, in modo che possano essere invocate secondo necessità.

Per l’italiano è necessario, dato un lemma, capire qual è la sua forma base, attraverso la query *getFormaBase*. Questa query è necessaria per permettere di tradurre i termini che non sono presenti nel dizionario nella forma in cui si trovano: si traduce infatti la forma base, e attraverso la sintesi di alcune regole grammaticali, si riportano alla loro forma originale. Per quanto riguarda il francese invece, è necessario risalire ai lemmi collegati ai syncon id tradotti: in questo caso si utilizza la query *getSyncon*.

Un breve approfondimento su Microsoft Bing Translator

Microsoft Bing Translator è il traduttore automatico di Microsoft. È un sistema di traduzione statistica allo stato dell’arte, e attualmente supporta la traduzione in 44 lingue differenti. Microsoft mette a disposizione diversi strumenti di traduzione, tra cui una serie di API per l’interrogazione del servizio, da integrare nelle proprie applicazioni o servizi.

All’interno di questo progetto, il traduttore Microsoft, è stato utilizzato per supportare la traduzione dal dizionario, in quanto il termine ricercato non sempre risulta essere disponibile: questo capita tipicamente quando la parola è formata da più token, oppure quando non è presente all’interno delle regole nella sua forma base (tipicamente succede con le keyword). È stato scelto Bing come traduttore in quanto le sue API sono disponibili gratuitamente, con il limite di utilizzo di due milioni di caratteri mensili;

sarebbe stato preferibile utilizzare Google translate, in quanto offre una traduzione più precisa, ma purtroppo non permette l'utilizzo delle proprie API gratuitamente. Il servizio di traduzione Bing [18] invece, è disponibile attraverso il web oppure, come viene utilizzato nell'ambito di questo progetto, attraverso un'applicazione client che ne richiama le API. Le API messe a disposizione da Microsoft presentano diverse interfacce: il traduttore può infatti essere utilizzato come web widget, oppure attraverso chiamate AJAX, http, SOAP oppure OData.

Per utilizzare il traduttore è necessario possedere un account Microsoft e registrarsi per l'accesso all'Azure Marketplace. In questo modo si ha la possibilità di acquistare qualsiasi applicazione presente nel market, ma in questo caso sottoscriveremo la licenza di utilizzo gratuito, che ci permette l'interrogazione del servizio fino ad un numero di caratteri pari a 2 milioni. A questo punto si ottengono due parametri fondamentali che dovranno essere sempre espressi per l'invocazione delle funzioni messe a disposizione dalle interfacce: il Client ID e il Client Secret. Entrambi sono reperibili all'interno dell'account del marketplace nella registrazione all'applicazione.

Implementazione

Considerazioni generali

Il progetto è stato realizzato con linguaggio di programmazione Java 7 e attraverso l'ide Eclipse Juno Enterprise Edition.

Il progetto possiede il sistema di logging **log4j** che tiene traccia di tutte le traduzioni che non sono possibili e del motivo per cui non sono possibili. Il file di configurazione di log4j è contenuto all'interno della cartella *Config* del progetto.

Insieme al file di configurazione di log4j, ne esiste un altro, **config.properties**, un file di proprietà utilizzato per evitare di cablare nel

codice la posizione di numerosi file di log particolari, flag, parametri di connessione con gli strumenti esterni, etc.

Gli elementi principali di questo progetto, come abbiamo già accennato nel capitolo precedente, sono il parser, il traduttore e l'analizzatore.

Parser

Non si tratta di un vero e proprio parser che effettua anche la validazione del file, ma più precisamente di uno scanner, che legge le regole e le riporta su un file destinazione una volta tradotte. Si basa su una procedura sequenziale che legge un file per volta scansionando regola dopo regola, riconoscendo il tipo delle regole in base a regular expression che le descrivono.

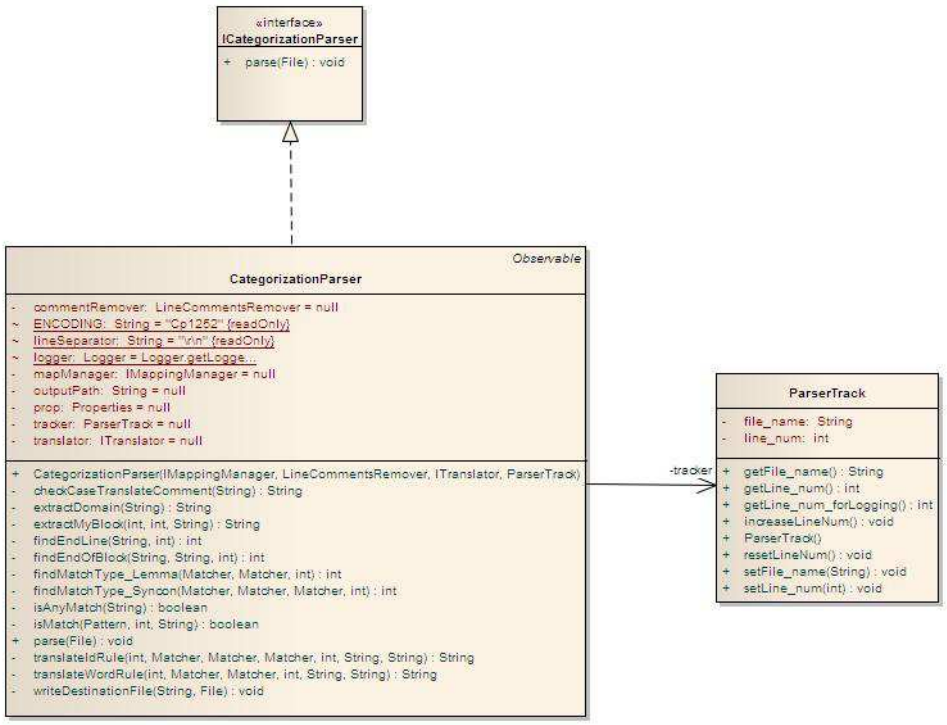


Figura 27 – Struttura del parser dei file di categorizzazione.

Le regular expression sono contenute all'interno di una classe che le raccoglie tutte e sono accessibili staticamente da qualsiasi classe.

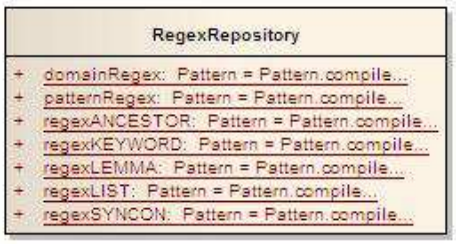


Figura 28 – Classe che raccoglie tutte le regular expression utilizzate dal parser.

Dapprima il file di categorizzazione viene letto ed immagazzinato completamente e mantenuto all'interno di una variabile. Dopodiché la stringa viene letta, cercando di isolare ed elaborare un blocco per volta: per blocco intendiamo l'insieme di regole che si trovano all'interno di DOMAIN.

Prima di effettuare questa operazione è necessario verificare se all'interno del blocco si trova una regola di tipo PATTERN: infatti la regola può contenere, essendo una regola che permette il riconoscimento di sequenze di simboli predefinite, anche un simbolo del tipo "}" che potrebbe quindi essere confuso banalmente come la fine del blocco, contrassegnata dallo stesso simbolo. Per questo motivo è necessario controllare se esiste una regola di questo tipo all'interno del blocco e in questo caso avvertire il parser di includerla verificando dove si trova la fine del blocco dopo la regola. Tipicamente, i pattern vengono utilizzati per il riconoscimento di sequenze alfanumeriche utili; all'interno dell'espressione regolare che descrive il pattern possono essere usate sequenze di lettere o parole con significato che possono essere anch'esse tradotte. Per questo motivo la classe pattern traduce le parole che trova all'interno dell'espressione regolare, ricomponendola poi com'era prima della traduzione, e aggiungendola nuovamente nella sua posizione all'interno del blocco.

Per la regola di tipo PATTERN è stato richiesto di creare un file contenente tutte le regole di questo tipo, con la specifica di dove sono state trovate: file e numero di riga all'interno del file.

A causa della necessità di tenere traccia di problemi di vario genere nel file di logging, è stato necessario creare una classe chiamata ***Tracker*** che tiene traccia di dove si trova il parser ad effettuare la sua scansione e che viene passata alle classi che necessitano di registrare su file questa informazione.

Una volta isolato il blocco della regola e definita la regione sulla quale deve lavorare il *Matcher* di java, inizializzato con le regular expression definite, si effettua il riconoscimento degli attributi, suddividendoli in due parti fondamentali: quelli contenenti syncon id e quelli contenenti le parole. Dapprima si cercano tutte quelle contenenti le parole, traducendole sempre in ordine di posizione: si procede in questo modo per evitare di tradurre due volte la stessa regola, quindi si traducono sempre tutte quelle identificate dallo stesso attributo all'interno del blocco in analisi, riportando le altre regole così come sono. Una volta tradotta, la regola, viene sostituita all'interno del blocco e il parsing prosegue finché non esiste più un matching possibile tra le regular expression. Infine il blocco viene riportato nel nuovo file di destinazione della traduzione (che deve avere lo stesso nome del file sorgente, ma essere situato in una cartella apposita che raccoglie i file tradotti) aggiornando la posizione in cui deve essere riscritto il prossimo blocco di regole.

Ogni volta che un attributo viene riconosciuto, viene creata una nuova entità corrispondente al tipo di attributo individuato. Invocando il *toString* della regola viene ricostruito il blocco della regola con il contenuto dell'attributo tradotto: sfruttando il polimorfismo è stata creata una struttura di classi gerarchica per rappresentare gli attributi.

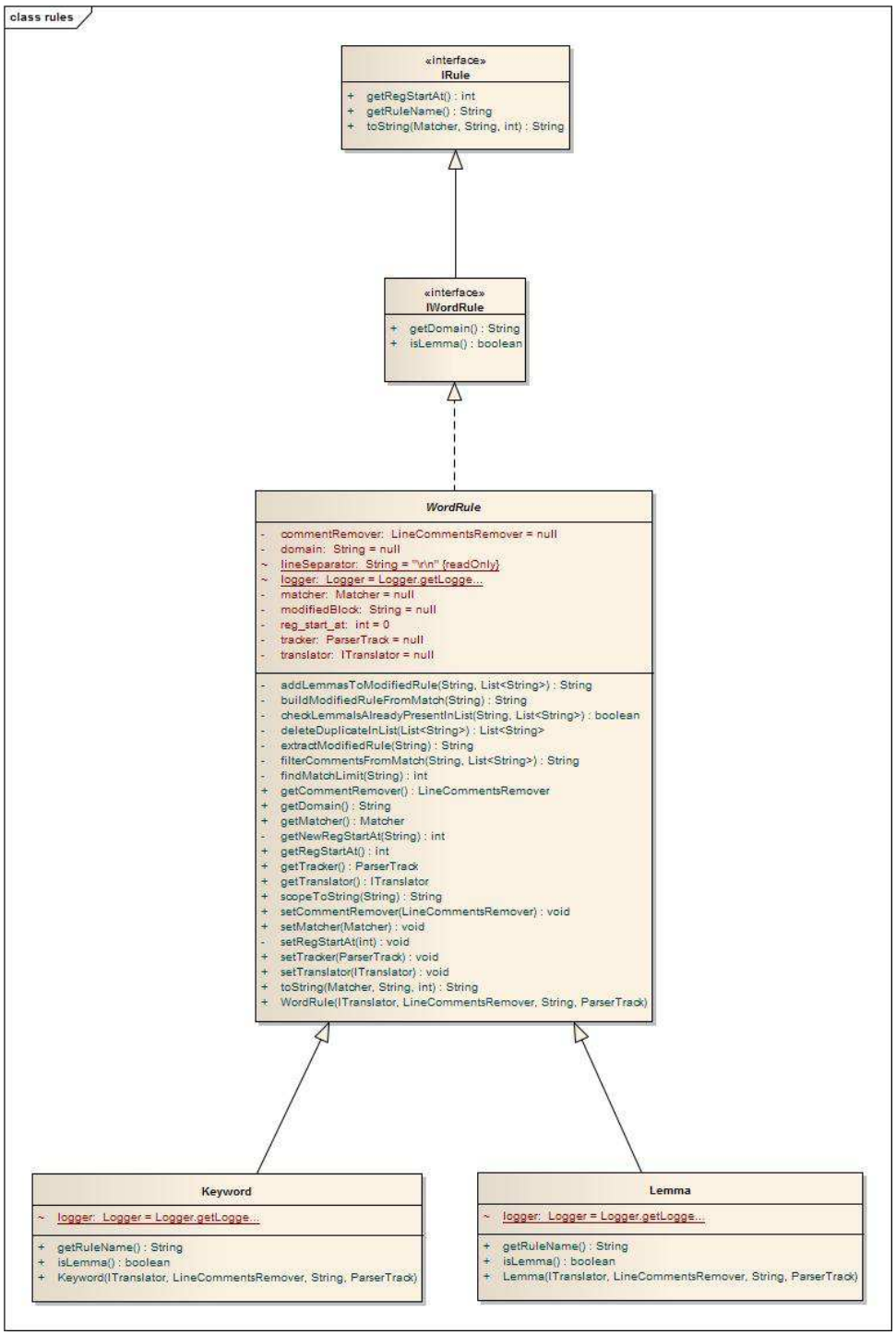


Figura 29 – Il diagramma delle classi in figura rappresenta la struttura gerarchica delle entità che rappresentano le regole. In particolare l'immagine riporta il ramo riguardante le regole di tipo "word".

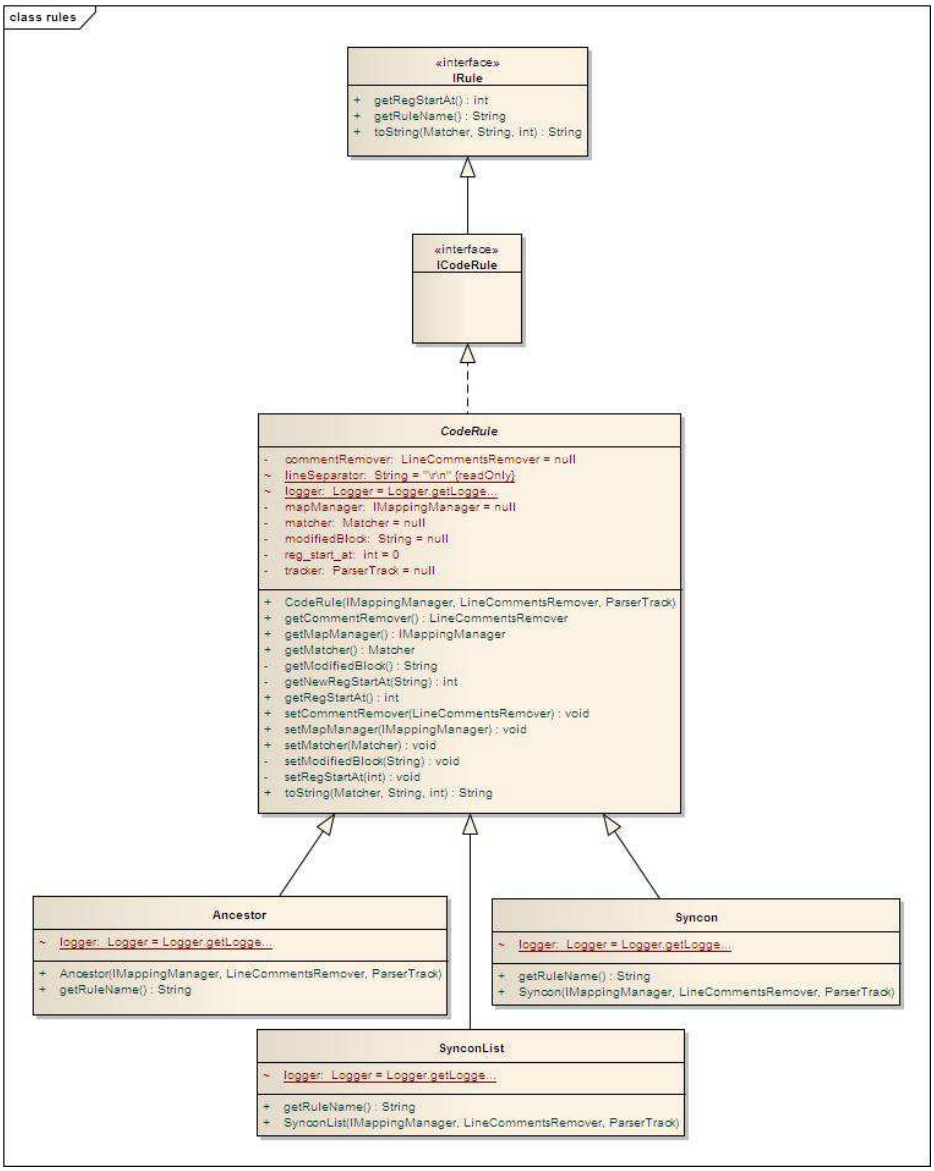


Figura 30 – L'immagine riporta il secondo ramo gerarchico della struttura delle entità che rappresentano le regole. Questo ramo riguarda le regole di tipo "code".

Al vertice della gerarchia c'è l'interfaccia *IRule* dalla quale deriva l'interfaccia *IWordRule*, che rappresenta gli attributi il cui contenuto sono stringhe; e l'interfaccia *ICodeRule*, che invece rappresenta gli attributi il cui contenuto sono i syncon id. L'interfaccia *IRule* espone i metodi *getRuleName()*, che restituisce il nome dell'attributo che stiamo traducendo; *toString()*, che come dicevamo sopra restituisce un nuovo blocco contenente l'attributo tradotto; e *getRegStartAt()* necessario per reperire dove inizia la regular expression riconosciuta. *IWordRule* invece

espone i metodi *isLemma()*, per capire se l'oggetto che implementa l'interfaccia in un determinato momento è di tipo lemma o keyword e *getDomain()*, che invece restituisce il dominio dell'attributo. Da queste due interfacce derivano due classi astratte *WordRule* e *CodeRule*, che implementano la funzione *toString()*, lasciando alle classi ultime concrete derivate, il compito di implementare la funzione *isLemma()* e *getRuleName()*.

Di particolare importanza risulta, all'interno delle classi astratte, il controllo riguardante il commento. Non ci sono problemi derivanti dai commenti single line, perché possono essere salvati e reinseriti tranquillamente a fianco alla regola, nella posizione in cui si trovavano in precedenza. I commenti a blocco, invece, per loro natura presentano diversi problemi, rilevabili anche osservando i file sorgenti: un blocco deve sempre essere aperto da "`/*`" e chiuso da "`*/`". Capita spesso però che all'interno di commenti a blocco ci siano altri commenti dello stesso tipo che non hanno i tag di apertura e chiusura incompleti. Ad esempio:

```
LEMMA("casa",/"scuola","asilo"*/);
```

oppure

```
LEMMA("casa",/*"scuola","asilo"/);
```

Questo succede solitamente perché all'interno dei commenti, non per forza quello che viene scritto deve essere assolutamente ben formato, inoltre se esisteva già un commento a blocco che viene incluso in quello più esteso, il tag di chiusura del blocco, se è ben formato, impedisce di terminare il commento dove necessario. Per questo motivo il tag di chiusura del commento non è quasi mai ben formato.

Quindi è stato deciso che i commenti a blocco ben formati che si trovano all'interno di un attributo, vengono salvati e copiati in fondo al blocco, in questo modo non creano problemi alla traduzione e non vengono eliminati, permettendo la loro consultazione anche all'interno dei file tradotti.

Gli altri, che invece non sono ben formati, dal momento che sicuramente si trovano all'interno di un commento più ampio, vengono cancellati. Nel caso, quindi, di un attributo così formato:

```
LEMMA("casa",/*"scuola", "asilo"*);
```

la cancellazione del commento porterà l'attributo a diventare di questo tipo:

```
LEMMA("casa",
```

La traduzione avverrà ugualmente come per gli altri attributi che non contengono commenti, ma solamente per il lemma "casa". È quindi necessario gestire anche la mancanza di parti dell'attributo al momento della traduzione.

Traduttore

La struttura del traduttore si compone di due elementi fondamentali: il traduttore vero e proprio che si occupa della traduzione dei termini che sono contenuti negli attributi, e della traduzione di tutto ciò che necessita di disambiguazione e di un'accurata elaborazione della traduzione; e di un modulo che invece si occupa della gestione del file di mapping tra i Sensigrafi® e quindi della traduzione dei syncon id. Sono due entità a sé stanti che non interagiscono tra loro in quanto la natura degli attributi è univoca.

Traduzione dei syncon id

Come già accennato in precedenza, la traduzione dei syncon id è molto semplice: siccome la creazione del Sensigrafo® francese avviene partendo da quello italiano, esiste anche un file dove vengono riportati tutti i mapping effettuati. Ogni volta che viene riconosciuto un syncon id, è sufficiente leggere il file di mapping e trovare il suo corrispondente o i suoi corrispondenti.

La gestione del file di mapping è delegata alla classe *MappingManager*.

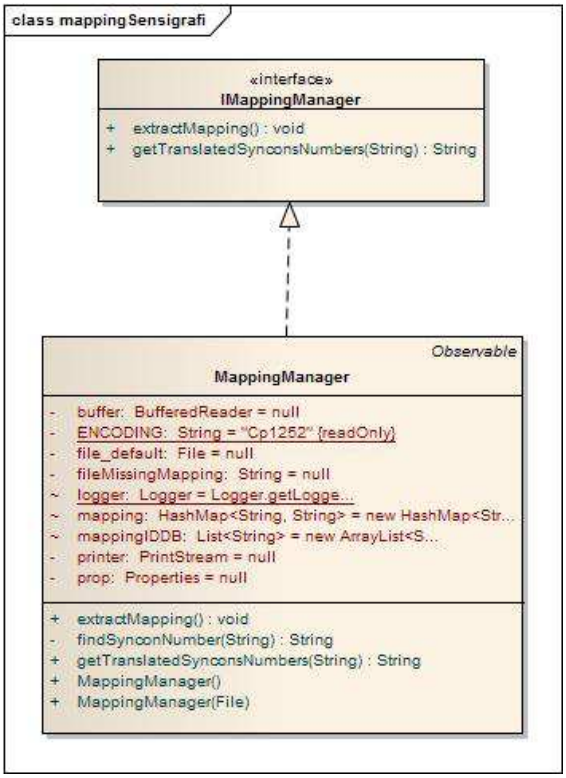


Figura 31 – UML della classe MappingManager

Alla creazione, il manager, inizializza una struttura di tipo *HashMap* in cui carica tutto il contenuto del file.

```
HashMap <String, String> mapping = new HashMap
<String, String>();
```

Attraverso poi il metodo *getTranslatedSynconNumber()* viene effettuata la conversione con i syncon id francesi. Da notare che la stringa che prende in input il metodo sopracitato, è la stringa così come si presenta all'interno dell'attributo. La stringa viene quindi ripulita e suddivisa nei singoli id, ognuno dei quali può presentare una profondità a seguito del numero di id: alcuni id sono quindi del tipo "12345:4". Il suffisso ":4" indica la profondità e siccome il mapping avviene sulla base del Sensigrafo® italiano, i concetti figli di un concetto in una lingua, sono lo stesso in un'altra lingua, per questo motivo la profondità che riportiamo è la stessa.

Una volta recuperati gli id francesi, è sufficiente appendere all'id,se esiste, la profondità trovata nell'italiano.

Traduzione di lemmi e keyword

Il modulo che si occupa della traduzione dei lemmi e delle keyword è piuttosto complicato: per effettuare al meglio la traduzione è stato necessario implementare una serie di funzionalità, che però non sono state integrate tutte all'interno del traduttore, ma sono state delegate a classi esterne, quindi il funzionamento del traduttore si basa sull'utilizzo di diversi moduli esterni alla classe stessa che implementano le diverse funzioni della traduzione.

Il traduttore presenta un'interfaccia che espone il metodo *getTranslatedLemmaOrKeyword()*.

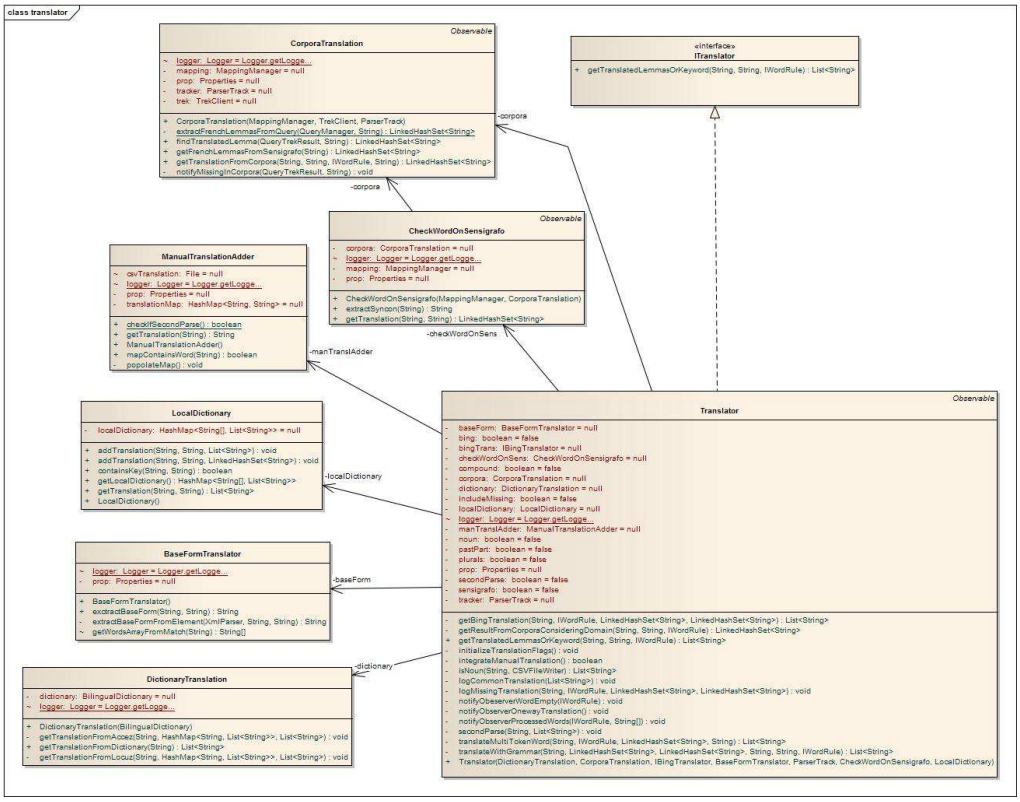


Figura 32- L'immagine riporta la struttura modulare del traduttore.

Una volta riconosciuto il matching, l'attributo, sotto forma di stringa generica, viene passato al traduttore per effettuare la traduzione del suo contenuto.

Come spiegato all'interno del capitolo riguardante l'approccio risolutivo, quando abbiamo a che fare con un lemma oppure con una keyword, dobbiamo distinguere tra quelle che hanno soltanto un significato, e quelle che invece è necessario capire in quale contesto si trovano per decidere la traduzione appropriata.

Per quanto riguarda i lemmi e le keyword che hanno solamente una traduzione, questa si può facilmente estrarre dal dizionario bilingue. In alternativa lo stesso principio del dizionario è stato applicato al Sensigrafo®: se il lemma ha soltanto un id associato, non occorre disambiguare tra diversi significati, quindi si estrae l'id e si controlla la corrispondenza nel file di mapping. Se anche il mapping ha soltanto un id francese associato allora a quella viene considerata come sua traduzione.

Per capire invece, a fronte di più traduzioni, quale sia quella giusta, si ricorre all'utilizzo di Trek e di un corpus di grosse dimensioni, che possiamo interrogare ogni volta che dobbiamo effettuare una traduzione.

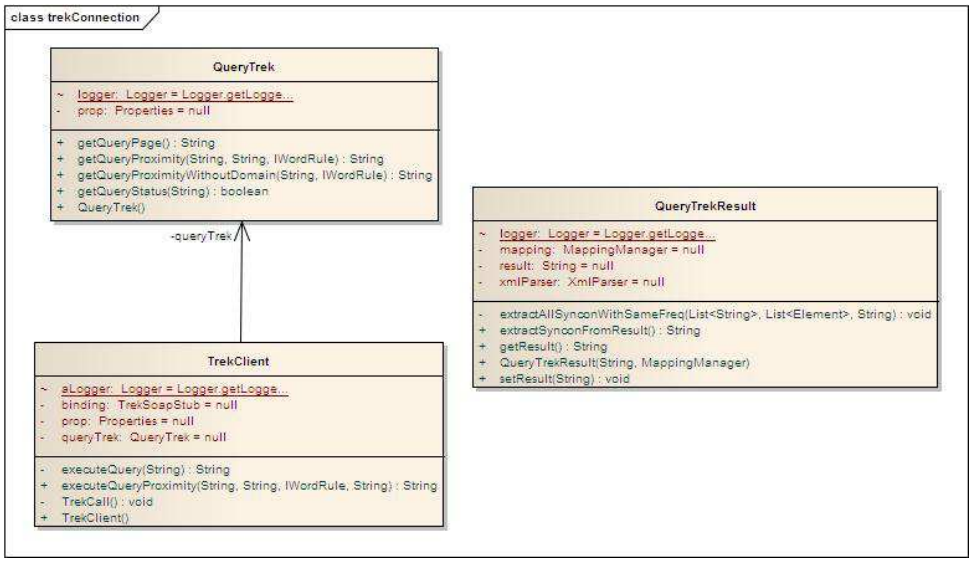


Figura 33 - L'immagine mostra la struttura di connessione verso Trek e la classe *QueryTrekResult* che estrae dal risultato dell'interrogazione i dati necessari.

In questo modo si ha la possibilità, data una regola di tipo lemma o keyword, appartenente ad un certo dominio, di ricercare il lemma o la keyword italiana all'interno degli articoli facenti parte della stessa categoria di cui fa parte la regola da tradurre ed in seguito estrarre il syncon id più frequente associato a quel termine. Una volta ottenuto il syncon id, si estrae dal file di mapping la sua corrispondenza con i syncon id francesi, e infine interrogando il Sensigrafo® si può risalire ai termini in lingua francese. Di questo di occupa la classe *CorporaTranslation*, attraverso il metodo *getTranslationFromCorpora()*.

I termini che vengono sostituiti all'interno della regola sono quelli in comune tra le traduzioni estratte dal dizionario e quelle estratte dai corpora con la procedura appena descritta.

Se non esistono termini in comune tra l'estrazione del dizionario e dei corpora i motivi possono essere:

- Il termine non esiste nel dizionario;
- Il termine non esiste nei corpora;
- Il termine è presente nei corpora ma non ha ancora una corrispondenza all'interno del Sensigrafo® francese, quindi il syncon id italiano che gli corrisponde non è presente all'interno del file di mapping.

Per questo motivo, quando non esiste una traduzione si ricorre all'utilizzo del traduttore Bing. La traduzione estratta da Bing viene comunque incrociata con le traduzioni presenti (a seconda che siano quelle del dizionario o quelle dei corpora) per evitare anche che il traduttore automatico inserisca delle parole inesatte.

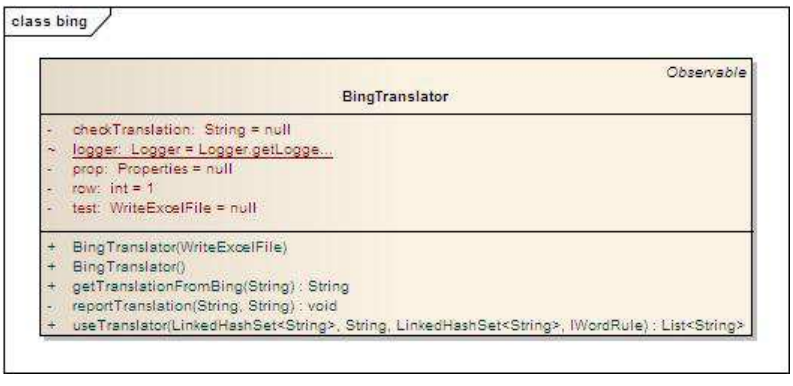


Figura 34 – In figura la classe utilizzata per interrogare Bing.

Tendenzialmente questa è la procedura seguita per quasi tutti i termini. Esistono però delle eccezioni derivanti anche dalla natura dei testi che devono essere categorizzati attraverso queste regole. Sono infatti presenti, all'interno degli attributi eccezioni particolari, quali: nomi propri, termini composti da più token, termini facenti parte di regole di domini virtuali, termini plurali e verbi al participio passato. Di seguito vengono introdotte le eccezioni e spiegate le procedure di traduzione utilizzate.

Nomi propri

I **nomi propri** rappresentano un'eccezione in quanto sono termini che in un'elaborazione complessiva risulterebbero non tradotti, ma che invece è impossibile tradurre e devono essere esenti da ogni traduzione. Quelli che vengono riconosciuti devono essere salvati (per essere eventualmente controllati), e vengono quindi riportati all'interno del file "noun.csv" in modo da separarli e avere un file di riferimento per vedere quelli che il programma ha effettivamente riconosciuto. Purtroppo non tutti i nomi propri vengono riconosciuti in quanto tali, e finiscono per essere conteggiati tra i termini non tradotti. La funzione utilizzata per riconoscerli, si limita a guardare se questi sono scritti con lettera iniziale maiuscola.

Termini composti da più token

Per i **termini composti da più token** la procedura di traduzione è uguale a quelle delle parole semplici con la sola differenza che vengono ricercati all'interno di tutto il corpora, senza distinguere la categoria di cui fanno parte, in quanto si suppone che per termini composti non sia necessario disambiguare tra significati diversi.

Termini appartenenti a domini virtuali

Esistono dei domini particolari, che chiamiamo **domini virtuali**, utilizzati nel lavoro di sintesi delle regole come ulteriore supporto all'analisi. Questi domini non vengono riconosciuti dalla tecnologia perché vengono in realtà modellati attraverso il linguaggio D e non fanno parte dell'albero dei domini. I termini facenti parte di domini virtuali vengono tradotti nel

medesimo modo dei termini normali ma al momento della ricerca all'interno dei corpora, non si distingue la categoria di cui fanno parte.

Termini plurali

Esistono diversi termini plurali, soprattutto all'interno degli attributi keyword, che non riescono ad essere tradotti ricercandoli semplicemente nel dizionario. Quindi i termini plurali, che vengono riconosciuti come tali, sono riportati dal Sensigrafo® alla loro forma base e tradotti nella loro forma base. Una volta ottenuta la traduzione francese singolare, attraverso un'opportuna classe che sintetizza la trasformazione da singolare a plurale nella grammatica francese, viene ricreata la parola plurale. I termini vengono in ogni caso salvati su un file a parte ("Plurals.txt") insieme alla traduzione proposta, per facilitare il lavoro del linguista che dovrà validare i file tradotti.

La traduzione avviene per mezzo della classe *BaseForm* che recupera con una query Essex la forma base della parola. Una volta recuperata la forma base, la traduzione viene effettuata nel medesimo modo, incrociando le traduzioni del dizionario e dei corpora. Infine, attraverso la classe *FrenchGrammarPlural*, applicando la grammatica francese sintetizzata nella classe, si trasforma la parola singolare al plurale. La classe contiene le irregolarità grammaticali più frequenti per il plurale in francese, in modo da riuscire a tradurre comunque anche alcune delle parole la cui forma plurale è irregolare. La classe presenta l'interfaccia *IFrenchGrammar* che è condivisa con la classe *FrenchGrammarPastParticiple* che unisce le funzioni comuni delle due classi. La classe *FrenchGrammarPastParticiple* funziona secondo lo stesso principio, trasformando però i verbi dalla forma infinita al participio passato.

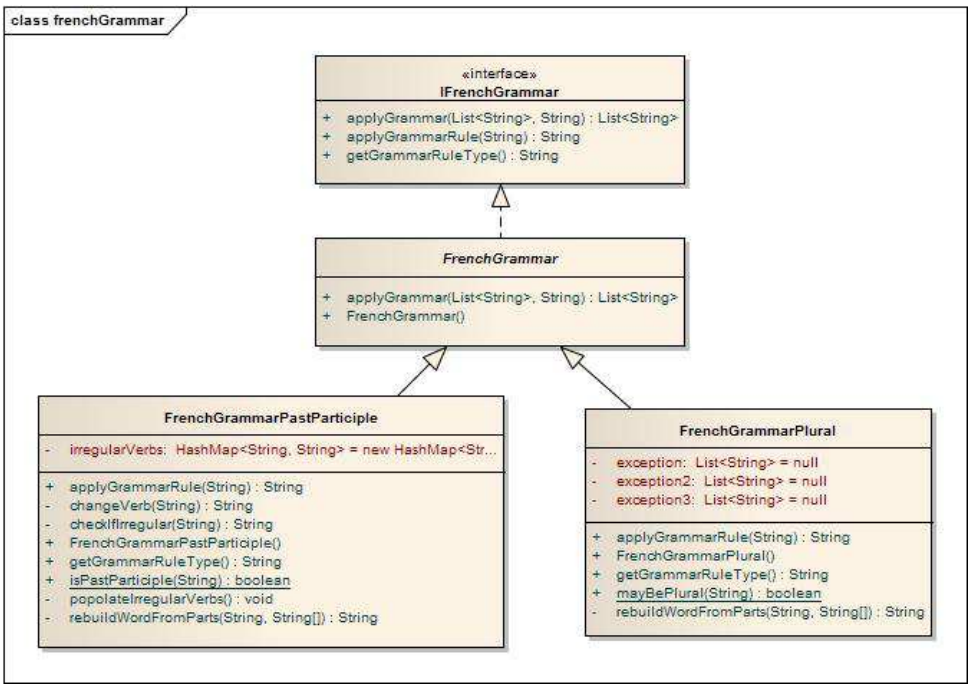


Figura 35 – Classi che sintetizzano la grammatica francese.

Participio passato dei verbi

Lo stesso procedimento utilizzato per le parole plurali viene applicato anche al **participio passato** dei verbi. La grammatica francese per creare il participio passato, è stata sintetizzata all'interno della classe *FrenchGrammarPastParticiple*. Come per i plurali è stata gestita tutta la parte regolare della grammatica che descrive come trasformare il verbo dall'infinito (recuperato sempre attraverso la query *getFormaBase* di Essex) nel suo participio passato, inserendo però all'interno della classe anche i principali verbi irregolari, quelli che vengono usati più di frequente. Anche in questo caso viene prodotto dalla procedura un file di log specifico ("PastParticiple.txt") che contiene la parola identificata come participio passato e la traduzione proposta.

Le classi che rappresentano la grammatica francese sono le uniche ad essere dipendenti dalla lingua in questione, e che devono essere cambiate nel caso di traduzione in una lingua che non sia francese. Tutta la struttura è indipendente dalla lingua di traduzione, cambiando infatti le connessioni

con le risorse esterne, come i GSL Essex o i file di mapping, è possibile tradurre i file anche in un'altra lingua.

Durante l'elaborazione viene mantenuta traccia di diversi elementi (oltre a quelli già citati) in appositi file di log, in modo da fornire la possibilità di verificare puntualmente il risultato di alcune traduzioni: i pattern, salvati sul file "pattern.txt"; i mapping mancanti nel Sensigrafo® francese, nel file "missingMapping.txt"; i termini che non riescono ad essere tradotti in nessun modo nel file "MissingTranslation.csv".

Analizzatore

Nell'ambito di questo progetto è difficile capire con quale precision è stata eseguita la traduzione se non chiedendo a traduttori professionisti di analizzare tutti i file oggetto della traduzione. Questo tipo di test risulta impossibile però, data la dimensione dei file tradotti. Si può però capire con quale recall. Questa analisi viene effettuata dalla classe *CounterObserver* che mantiene traccia della quantità di regole e parole analizzate e tradotte, e anche della quantità di parole tradotta applicando le diverse tecniche.

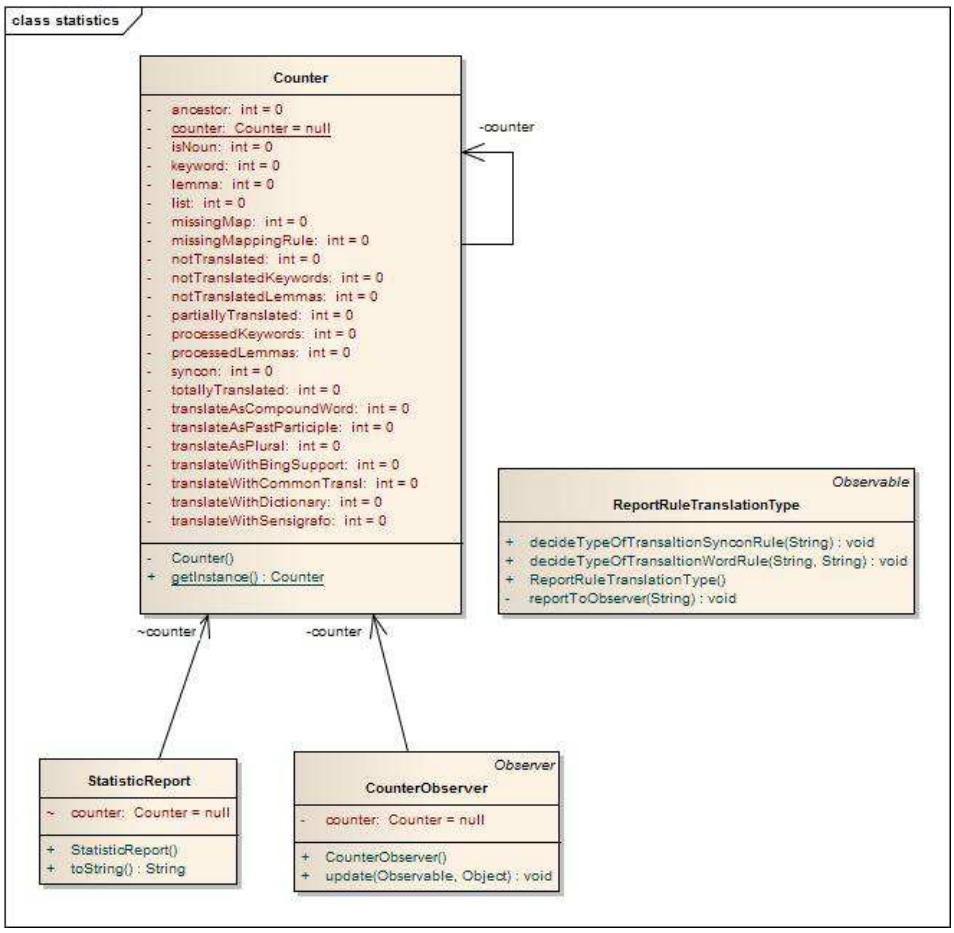


Figura 36 – In figura la struttura che si occupa delle statistiche.

La classe implementa il pattern *Observer*, le classi che realizzano la traduzione sono tutte implementazioni dell'interfaccia *Observable*: in questo modo nel momento in cui viene effettuata una traduzione di un certo tipo, la classe *Observable* avvisa che la traduzione è avvenuta, invocando il metodo *setChanged()*, e poi notifica il tipo di traduzione effettuata attraverso il metodo *notifyObservers()*. La stringa argomento di questo metodo, contiene delle parole chiave che corrispondono al nome delle variabili da incrementare all'interno della classe *Counter*.La classe *Counter* è un oggetto che fa solamente da contenitore per le variabili che mantengono i conteggi di traduzione.

Il metodo update della classe *CounterObserver*, ad esempio, nel caso dei lemmi processati, prende in input la keyword "*processedLemmas*" seguita da un intero che rappresenta il numero di parole processate:

```
notifyObservers("processedLemmas,"+words.length);
```

A seconda della stringa in input che riceve, invoca il metodo di *Counter* per incrementare la variabile corrispondente.

Esiste anche un'altra classe di supporto all'analisi, *StatisticReport* che invece si occupa solamente di formare una stringa per la stampa a video di un report contenente le statistiche di elaborazione.

I dati che vengono elaborati riguardano innanzitutto la quantità di termini processati e tradotti. In particolare il report fornisce il dettaglio di quanti termini, tra lemmi e keyword non è stato possibile tradurre. Inoltre tra i termini tradotti, viene anche specificato con quale sistema sono stati tradotti. Il report specifica anche, grazie alla classe *ReportRuleTranslationType*, quante regole vengono elaborate dalla procedura, riportando nel dettaglio quante regole sono state tradotte totalmente, parzialmente o non tradotte totalmente.

Coefficiente di Dice e Levenshtein

Il **coefficiente di Dice** e quello di **Levenshtein** sono stati introdotti per tentare di fornire una traduzione anche per quei termini in cui la traduzione è presente tra le scelte possibili, ma non viene individuata per mancanza di matching tra le proposte. Entrambi sono coefficienti che ragionano sulla similarità delle parole: in particolare mentre il coefficiente di Dice misura la similarità esistente delle parole, il coefficiente di Levenshtein ne misura la distanza: nel primo caso quindi è utile massimizzare il coefficiente, nel secondo caso invece minimizzarlo. Per motivi matematici non sono comunque da considerare l'uno l'opposto dell'altro. È stato deciso di effettuare anche questo tentativo di traduzione, in quanto esiste un alto grado di affinità tra l'italiano e il francese; infatti se la lingua di destinazione della traduzione avesse avuto un'origine

diversa, ad esempio se non fosse stata una lingua neolatina, il tentativo di applicazione di questo metodo sarebbe stato inutile.

In ogni caso questi metodi non hanno riportato risultati soddisfacenti pertanto non è stato possibile inserire il loro contributo all'interno della traduzione. Essi infatti, hanno il grosso limite di non avere a disposizione un contesto, ma di ragionare solamente sulle lettere che formano la parola. L'idea fondamentale seguita in questo lavoro di tesi, è stata quella di cercare una possibile traduzione per i termini contenuti nelle regole, senza cercare però di proporre una possibile traduzione ad ogni costo. Infatti, in letteratura, questi due indici vengono spesso applicati nell'ambito della traduzione di un testo, quando fornire una possibile traduzione ad una parola è molto più importante della precisione della traduzione stessa, perché in quel caso il contesto permetterebbe all'utilizzatore finale, cioè colui che richiede la traduzione del testo, di coglierne il giusto significato.

L'analisi dell'applicazione degli indici è stata effettuata su un campione di un centinaio di termini, e i risultati sono stati analizzati da un linguista, in modo da poter individuare e capire se la traduzione fornita fosse totalmente sbagliata, accettabile o corretta.

Coefficiente di Dice

Per quanto riguarda il calcolo del coefficiente di Dice, l'algoritmo implementato è abbastanza semplice in quanto è sufficiente effettuare il conteggio di quante sillabe sono presenti e di quante sono effettivamente condivise tra i due termini e applicare la formula[1]:

$$s = \frac{2 * |A \cap B|}{|A| + |B|}$$

Il test è stato effettuato su un campione di cento termini che sono risultati intraducibili dai moduli. Per effettuare il test sono state estratte le traduzioni possibili sia dal dizionario che dai corpora; dopodiché è stato calcolato il coefficiente tra il termine italiano e tutte le possibili traduzioni (sia quelle dei corpora che quelle del dizionario). La traduzione

[1] Come spiegato nel capitolo su "Utilizzo dei *cognates* nella costruzione del lessico di traduzione" a pagina 20.

considerata migliore, e quindi scelta, è quella con il coefficiente di similarità più alto. Il test ha evidenziato che non esiste una soglia sopra la quale nel campione di elementi non siano presenti errori di qualche tipo.

Coefficiente di Levenshtein

L'algoritmo implementato per il coefficiente di Levenshtein invece è un po' più complicato, infatti prevede di calcolare il numero minimo di modifiche elementari che consentono di trasformare la parola A nella parola B. Per modifica elementare si intende: la cancellazione di un carattere, la sostituzione di un carattere con un altro, oppure l'inserimento di un carattere. L'algoritmo si basa sull'utilizzo di una matrice (N+1) * (M+1), dove M è il numero di caratteri della parola A e N è il numero di caratteri della parola B.

Dopo aver inizializzato la matrice, la prima riga con i numeri da 0 a N e la prima colonna con i numeri da 0 a M, gli altri coefficienti vengono calcolati nel modo seguente:

```
for i1 from 0 to lenStr1
  d[ i1, 0 ] := i1
    for i2 from 0 to lenStr2
      d[ 0, i2 ] := i2
      for i1 from 1 to lenStr1
        for i2 from 1 to lenStr2
         if str1[i1-1] = str2[i2-1] then cost:= 0
          else cost := 1
          d[ i1, i2 ] := minimum(
                  d[i1-1,i2]+1,//inserimento
                  d[i1,i2-1]+1,//cancellazione
                  d[i1-1,i2-1]+cost//sostituzione
                  )
return d[lenStr1,lenStr2]
```

Il minimo viene calcolato tra le celle adiacenti quella indicata da i1 e i2.

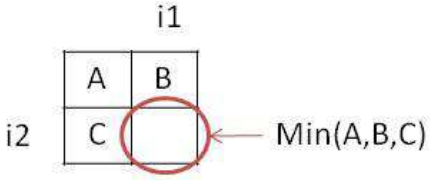


Figura 37

Quando tutta le celle della matrice sono state completate, viene restituita quella corrispondente all'ultima cella in fondo a destra (quindi in posizione strlen(A), strlen(B)), che rappresenta appunto la distanza di Levenshtein.

Il test effettuato su un campione di cento termini, e fatto analizzare ad un linguista, rivela che il 42% dei termini sono tradotti in modo scorretto: è importante sottolineare che restituisce lo stesso risultato del coefficiente di Dice. Affiancando invece le traduzioni, possiamo notare che le parole scelte come traduzione sono tendenzialmente le stesse scelte da Dice, ma non in tutti i casi. Anche in questo caso il coefficiente impedisce di stabilire un valore di soglia che porti un notevole vantaggio nella traduzione e che allo stesso tempo sia priva di errori.

Integrazione traduzione manuale

Tra i file che l'applicazione produce in output, c'è il file "MissingTranslation.csv". Questo file permette di effettuare una traduzione manuale dei termini che risultano impossibili da tradurre automaticamente ed integrarli nella traduzione appena effettuata. È sufficiente che il linguista che esegue la traduzione affianchi al termine italiano che trova nel file, la sua traduzione francese, oppure copi semplicemente il termine se pensa che la traduzione non possa essere effettuata correttamente o non esista un diretto corrispondente del temine nella lingua di destinazione. Una volta completato il file, è necessario posizionarlo all'interno della cartella "TranslationStore" e far girare nuovamente l'applicazione. La procedura, riconoscendo la presenza del file all'interno di questa cartella, andrà quindi a leggere le traduzioni ed integrarle all'interno dei file già precedentemente tradotti.

Deployment

Per permettere all'applicazione di funzionare, è necessario predisporre tutti gli strumenti esterni necessari al suo funzionamento. È necessario avere a disposizione anzitutto un GLS Essex IPTC per poter interrogare il Sensigrafo®; aver creato un corpus di articoli di giornale categorizzati secondo IPTC utilizzando Trek; disporre dei file di estrazione da un dizionario bilingue; disporre del file di mapping tra il Sensigrafo® italiano e quello della lingua destinazione della traduzione.

Dopo aver recuperato tutte le risorse necessarie, è indispensabile configurare opportunamente le risorse nel file di proprietà dell'applicazione. Il file si trova all'interno della cartella *Config*; modificandolo è possibile specificare i percorsi dei file del dizionario, e del file di mapping; e inserendo id e porte delle risorse da interrogare, installate. È possibile specificare anche dove devono essere prodotti file di log dei termini tradotti e i file di categorizzazione tradotti.

```
#CONNECTION TO ESSEX ITA
host_IT=localhost
resourceId_IT=3339
connType_IT=GSL ESSEX IPTC ITA

#CONNECTION TO ESSEX FRA
host_FR=es-testw1
resourceId_FR=1143
connType_FR=GSL ESSEX FR

#TREK CONNECTION
trekHost=localhost
trekPort=9190

#TREK CORPUS
corpus1=/archiviocorriere_201303151700
corpus2=/webnews_201303151232
corpus3=/test_iptc_201304150922

#DICTIONARY FILE LOCATION
tradAccezione=TraduzioniRevised/TradDiAccezioneITAFRA.
txt
```

```
tradLocuzAccez=TraduzioniRevised/TradLocuzioniAccezion
eITAFRA.txt
tradLocuzLemma=TraduzioniRevised/TradLocuzioniLemmaITA
FRA.txt

#FLAG PER TRADUZIONE
compoundFlag=false
nounFlag=false
bingFlag=false
pluralsFlag=false
pastPartFlag=false

#MAPPING FILE LOCATION
mappingFile=MappingFile/fr_it_attr.txt

#MISSING MAPPINGS FILE LOCATION
fileMissingMapping=MappingFile/MissingMappings.txt

#PATTERN FILE LOCATION
patternfileName=Pattern/Pattern.txt

#PLURALS TRANSLATION FILE LOCATION
plurals=Translation/Plurals.txt

#PAST PARTICIPLE TRANSLATION FILE LOCATION
pastParticiple=Translation/Participle.txt

#MISSING TRANSLATION FILE LOCATION
translationFile=Translation/MissingTranslation.csv

#NOUN FILE LOCATION
nounFile=Translation/Noun.csv

#SOURCE FOLDER
inputDirPath=C://Users//stage02//Documents//Documenti/
/CartellaProvaTraduzioni//source

#DESTINATION FOLDER
outputDirPath=C://Users//stage02//Documents//Documenti
//CartellaProvaTraduzioni//destination

#COMMENTS
translateComments=true

#ENABLE USE BING TRANSLATOR, checkBing riporta la
traduzione su file per controllarla
checkBing=true
client=
clientSecret=
```

```
#PATH FOR MANUAL TRANSLATION FILE
manTranslation=TranslationStore/Translation.csv


#TEST

#PATH FOR PLURALS TEST
provaPluraliTest=Translation/provaPluraliTest.csv

#PATH FOR PAST PARTICIPLE TEST
provaPastParticipleTest=Translation/provaPastParticipl
eTest.csv

#FILE TEST LEVENSHTEIN
levFile=C://Users//stage02//Desktop//Levenshtein_test.
csv

#FILE TEST DICE - il file di test di Dice arricchisce
quello di Lev
diceFile=C://Users//stage02//Desktop//Levenshtein_Dice
_test.csv

commonTranslation=FileDiProva//Base.csv
commonTranslationNoun=FileDiProva//Noun.csv
```

Testing

Per effettuare il testing dell'applicazione è stato innanzitutto necessario creare delle classi ad hoc per verificare il funzionamento dei singoli moduli che cooperano alla traduzione. Il primo test effettuato è stato quindi di tipo funzionale, per controllare la presenza di malfunzionamenti o problemi derivanti dalla logica di programmazione.

Per quanto riguarda invece la traduzione delle regole vera e propria, è utile riportare alcuni dati significativi riguardanti la traduzione stessa. I dati sono stati registrati dall'analizzatore, ossia dal modulo creato appositamente per tenere traccia delle statistiche di traduzione.

Le regole tradotte sono state suddivise in tre sottoinsiemi specifici che evidenziano in quale quantità sono stati tradotti i termini che sono presenti nelle regole: nel grafico seguente, quindi, compaiono quante regole sono state totalmente tradotte, quante parzialmente tradotte (e per parziale si intende se è stata effettuata la traduzione anche di una parola soltanto) e infine quante non sono state tradotte completamente. Nel conteggio non si effettua distinzione tra regole di tipo lemma o keyword o regole il cui argomento è un id.

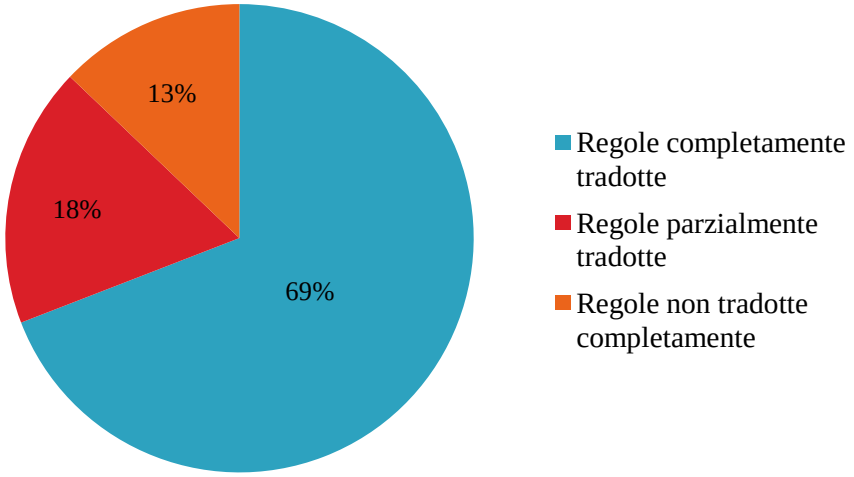


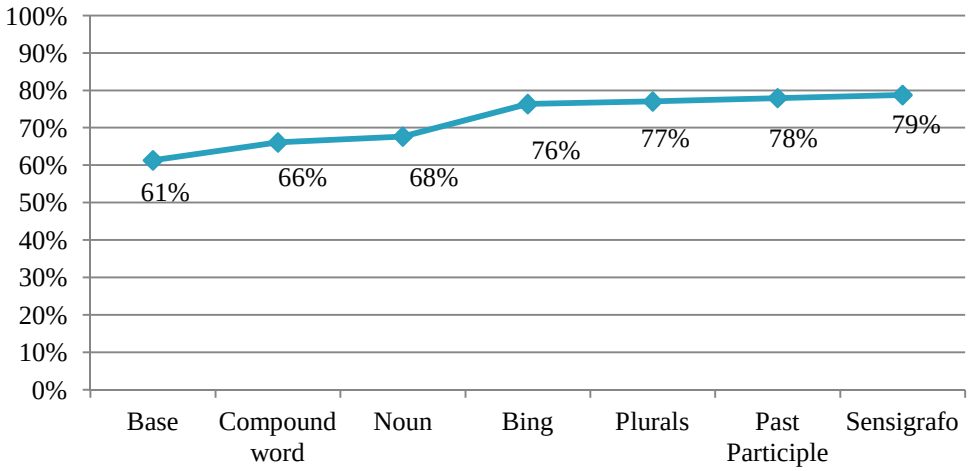
Figura 38 Il grafico riporta come sono state tradotte le regole.

Testing della recall

Ogni singolo modulo che contribuisce alla traduzione è stato reso indipendente dal modulo di traduzione base, in modo da evidenziare l'apporto positivo in termini di recall della traduzione. È possibile quindi abilitare e disabilitare ogni singolo modulo con un flag all'interno del file di configurazione.

```
#FLAG PER TRADUZIONE
compoundFlag=true
nounFlag=true
bingFlag=true
pluralsFlag=true
pastPartFlag=true
sensigrafo=true
```

Il grafico riportato di seguito mostra l'aumento di recall apportato dai vari moduli di traduzione.



Possiamo notare come il modulo base, considerato come traduzione con dizionario e utilizzo di corpora per lemmi con più significati, traduca il 61% dei termini che gli vengono sottoposti per la traduzione. La traduzione delle parole composte, come parole indipendenti dal contesto, incrementa la recall di un 5% mentre l'identificazione dei nomi solo di un 2%. Bing è il modulo che porta il miglioramento più sostanzioso: grazie a Bing viene tradotto un ulteriore 8% di termini. La traduzione dei plurali e del participio passato incrementano, in numero di termini tradotti, di una piccola percentuale 2%. Ed infine un piccolo contributo dell'1% viene apportato dalla traduzione con Sensigrafo.

Analizzando il file dove vengono salvati i termini che non è possibile tradurre, possiamo notare che molti di questi sono termini che dipendono direttamente dall'italiano e che quindi non possono essere tradotti, non avendo un corrispondente nella realtà francese. Molti sono anche nomi di associazioni o sigle che non vengono riconosciute come tali durante l'elaborazione e che quindi vengono conteggiati come termini intraducibili.

Il grafico seguente mostra la composizione totale della traduzione, evidenziando che il modulo base, rappresentato nel grafico tramite

Dictionary e CommonTranslation, traduce la maggioranza dei termini, mentre i moduli aggiuntivi contribuiscono in modo inferiore, apportando però nel complesso un aumento di recall rilevante.

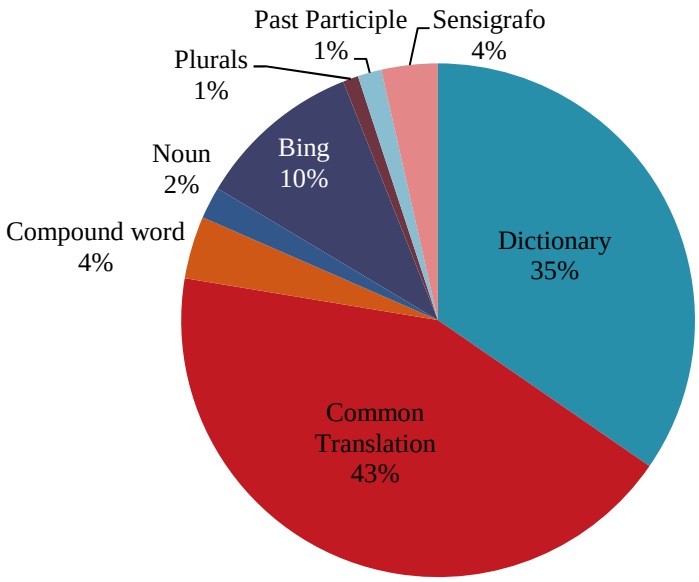


Figura 39 – Il grafico rappresenta la modalità di traduzione dei termini.

Testing della precision

Per quanto riguarda la precision, il test da effettuare è più complesso. Si tratterebbe infatti di far valutare ad un esperto di lingua francese la qualità e la fedeltà dei termini tradotti, rispetto ai corrispondenti italiani.

Data la quantità di termini e l'onere che comporterebbe validare tutti i termini manualmente, per effettuare questo test è stato deciso di provare a categorizzare un campione di articoli francesi, e valutare la qualità della categorizzazione.

Per il test è stato necessario creare un Sensigrafo® francese modificato, aggiungendo i termini non ancora presenti, in modo automatico senza collocarli esattamente all'interno della rete semantica. Questo permette la compilazione del progetto Cogito, e quindi di conseguenza la creazione di uno nuovo GSL Essex che comprenda il progetto IPTC francese tradotto, insieme al suo Sensigrafo®.

È importante specificare che il Sensigrafo® francese essendo ancora in via di sviluppo, contiene solo una parte dei termini francesi totali che vi andrebbero mappati, quindi oltre ad aggiungere automaticamente diversi termini francesi utili alla compilazione del progetto Cogito® Studio, è stato necessario commentare anche alcune regole all'interno dei file sorgente di categorizzazione. In particolare sono stati esclusi dalla compilazione gli attributi necessari a determinare i domini più specifici, quelli che rappresentano le foglie dell'albero IPTC. Per riportare un esempio: ha presentato numerosi problemi l'attributo booleano *sottrazione*, che solitamente è utilizzato per evitare falsi positivi. Il compilatore di Cogito® Studio effettua un controllo di coerenza tra i membri che sono gli operatori della sottrazione: se si specifica un syncon come primo operatore, che non è in nessun modo collegato al syncon specificato come secondo operatore, il compilatore segnala l'errore.

Il test che è stato effettuato quindi mette a confronto la categorizzazione in lingua italiana e in lingua francese di un campione di cento articoli giornalistici. Si è cercato di selezionare articoli italiani e francesi che appartenessero alle stesse categorie, in modo da confrontare la categorizzazione effettuata dal progetto di partenza italiano e quella del progetto tradotto in francese. Il test è stato effettuato da un linguista esperto. I dati riportati a livello di categorizzazione sono i seguenti:

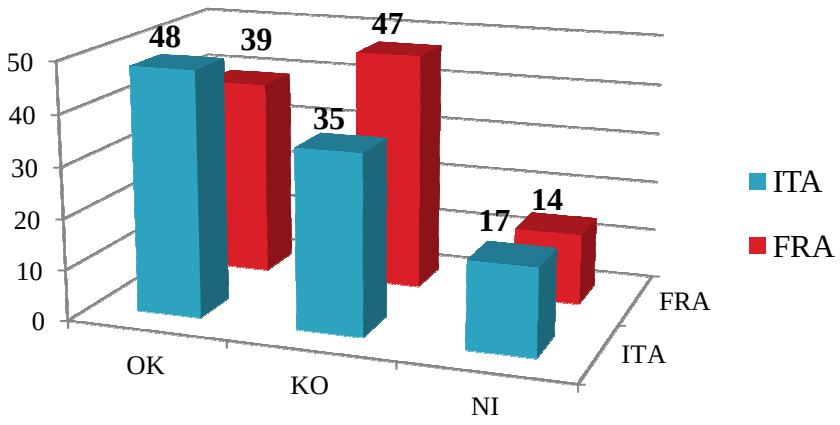


Figura 40 – Il grafico riporta il confronto tra la categorizzazione effettuata con il progetto di riferimento italiano e quella effettuata con il progetto tradotto in francese.

Come evidenziato nel grafico, circa il 10% in più (rispetto all'italiano) degli articoli considerati viene categorizzato in modo incorretto dal progetto francese tradotto (35 articoli vengono categorizzati in modo incorretto in italiano contro i 47 del francese). Questi dati sono stati estrapolati esaminando ogni articolo oggetto di test: considerate le categorie associate dal sistema a quell'articolo, il linguista ha controllato quali fossero corrette e quali incorrette. Se più del 50% dei domini che gli sono stati associati era corretto, la categorizzazione di quell'articolo viene considerata corretta; altrimenti incorretta. In caso di parità l'articolo risulta non identificato (sigla NI del grafico).

Esaminando infine il dettaglio dei domini riportati per ogni articolo si può evidenziare che alcuni di questi vengono associati numerose volte all'articolo in modo errato. Un esempio fra tutti è il dominio *"poesia"* che viene associato continuamente agli articoli, anche a quelli appartenenti ad un dominio completamente diverso. La causa di questo errore così frequente è dovuta al fatto che le regole del dominio *"poesia"* scattano più frequentemente di altre perché nella traduzione probabilmente sono diventate molto generiche e quindi scattano anche quando il dominio non è presente. Il fatto che siano diventate generiche può essere dovuto al fatto che molti termini contenuti nelle regole scritte per riconoscere il dominio *"poesia"* non sono stati tradotti, oppure non sono stati ancora mappati all'interno del Sensigrafo® francese.

Nell'analisi della categorizzazione, viene considerata corretta la categorizzazione degli articoli che hanno più del 50% dei domini assegnati, corretti. Il grafico seguente riporta nella totalità degli articoli categorizzati correttamente, in quale percentuale le categorie assegnate sono giuste.

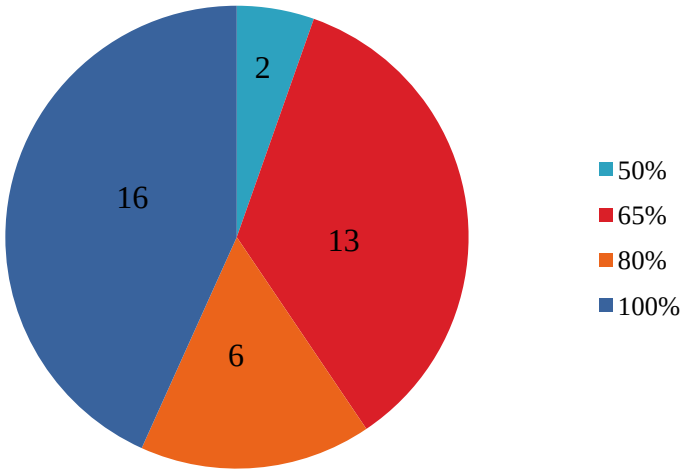


Figura 41 – Il grafico riporta quante categorie sono corrette tra gli articoli categorizzati correttamente.

Come si può notare dal grafico, tra gli articoli che sono stati classificati come categorizzati correttamente le categorie che gli sono assegnate sono quasi tutte corrette. La maggioranza del grafico è occupata dagli articoli che hanno il 100% dei domini classificati correttamente al quale gli possiamo aggiungere gli articoli che hanno il'80% dei domini classificati correttamente.

Dalle considerazioni derivanti dai test risulta quindi che la traduzione automatica compie già da sola una buona parte di lavoro, ma che sicuramente sarà necessario l'intervento di linguisti per poter correggere errori più frequenti. Il risultato deve essere valutato anche alla luce del fatto che lo sviluppo del Sensigrafo® francese non è terminato e che sarebbe necessario valutare nuovamente la traduzione con il Sensigrafo® completo.

Conclusione

Nell'ambito di questo progetto di tesi si è cercato di effettuare la traduzione in modo automatico del contenuto delle regole di categorizzazione di Cogito®. In particolare, le regole oggetto della traduzione sono quelle scritte all'interno di un progetto volto a categorizzare le news italiane sulla base dei domini definiti dall'IPTC. I problemi da affrontare sono stati molteplici, primo fra tutti, fulcro del progetto, la corretta interpretazione della semantica del lemma o della keyword inserita nella regola generica. Le strategie adottate per aggirare questo problema hanno influenzato l'approccio risolutivo; proprio per natura stessa delle regole, non è possibile infatti comprendere in modo automatico la semantica del termine contenuto se non affidandosi al dominio di cui fa parte la regola. Inoltre il problema più significativo è rappresentato dalle keyword: infatti queste non hanno una forma base nella quale vengono inserite, ma possono essere di qualsiasi genere, forma, tempo verbale o coniugazione. Per questo motivo sono stati di particolare importanza gli strumenti messi a disposizione da Expert System, in quanto già da solo, il Sensigrafo®, può supportare la ricerca dei termini e riportarli alla loro forma base permettendo così la ricerca su un dizionario. Oltre all'integrazione degli strumenti di Expert System, è stato inserito anche il supporto del traduttore automatico Bing, che permette di proporre una traduzione nel caso altre tecniche abbiano fallito. Analizzando i sorgenti, sono state osservate alcune caratteristiche che li accomunano: all'interno delle regole esiste una grande quantità di nomi propri, sigle e fatti dipendenti proprio dalla realtà italiana. Questi aspetti sono difficilmente traducibili e modificabili rispetto alla lingua di destinazione e alla sua realtà. Nonostante ciò, i test hanno evidenziato che il traduttore è in grado di tradurre circa l'80% dei termini totali del progetto. Inoltre un test effettuato da un linguista esperto ha dimostrato che i termini sono tradotti con una buona precisione, infatti preso un campione di articoli italiani e di articoli francesi appartenenti più o meno

alle stesse categorie, il progetto francese creato, sbaglia la categorizzazione solamente del 10% in più rispetto alla categorizzazione italiana di riferimento.

Il progetto in futuro, potrebbe diventare il sistema di base per uno strumento di supporto alla realizzazione delle regole per i linguisti in progetti applicabili a molte lingue e per cui esiste già una versione italiana corretta e funzionante. Inoltre, vista la sua struttura indipendente dalla lingua di destinazione della traduzione, il progetto può integrare anche altre lingue diverse dal francese, richiedendo soltanto la sintesi delle regole grammaticali che sono state identificate come importanti ai fini di questa traduzione.

Appendice A

Ulteriori approfondimenti su Trek

Trek Seacher

Il *Trek Seacher* ha la possibilità di effettuare le ricerche ed indicizzare basandosi su un dizionario locale. Il dizionario locale permette di avere un dizionario *ad hoc* a disposizione di ogni CFS, e quindi di accedere ai dati molto più velocemente nel caso di progetti di notevoli dimensioni. Essendo le operazioni di scrittura indipendenti non è necessario generare lock di attesa per attendere la risorsa e per mantenere il merge dei termini sul dizionario.

Il Searcher mette a disposizione due protocolli di comunicazione per i client e due ad uso interno per la comunicazione con gli altri Searcher. I primi due protocolli sono il SOAP ed il protocollo nativo di Trek: tutti e due viaggiano su standard TCP/IP. Il primo protocollo permette a qualsiasi client soap di chiamarne i metodi, una volta configurato il proxy con l'apposito wsdl.

I due protocolli interni, invece, sono il TrekBase (TB) ed il TrekData (TD). Il TrekBase viene utilizzato per distribuire le query splittate ai Searcher remoti e ricevere i risultati. Ogni Searcher esegue tre connessioni socket verso i restanti Searcher del sistema: le prime due connessioni vengono utilizzate come canale di comunicazione per le query standard (*boolean*, *ranked*) e le query di analisi (*proximity*, *relations*); la restante connessione viene utilizzata per il caricamento delle paginate di risultati delle query di analisi. Ogni Searcher può accedere in modo trasparente ai dati memorizzati nei CFS degli altri Searcher.

Tipicamente quando il client esegue una query, questa viene mandata ad un Searcher. Il Searcher prende in carico la query, la esegue accedendo dal proprio dizionario locale, direttamente agli indici inversi dei corpus ed esegue operazioni logiche espresse nella query. Allo stesso tempo la

inoltra agli altri Searcher esistenti. Infine, colleziona tutti i risultati, li compatta e li pagina per restituirli al client.

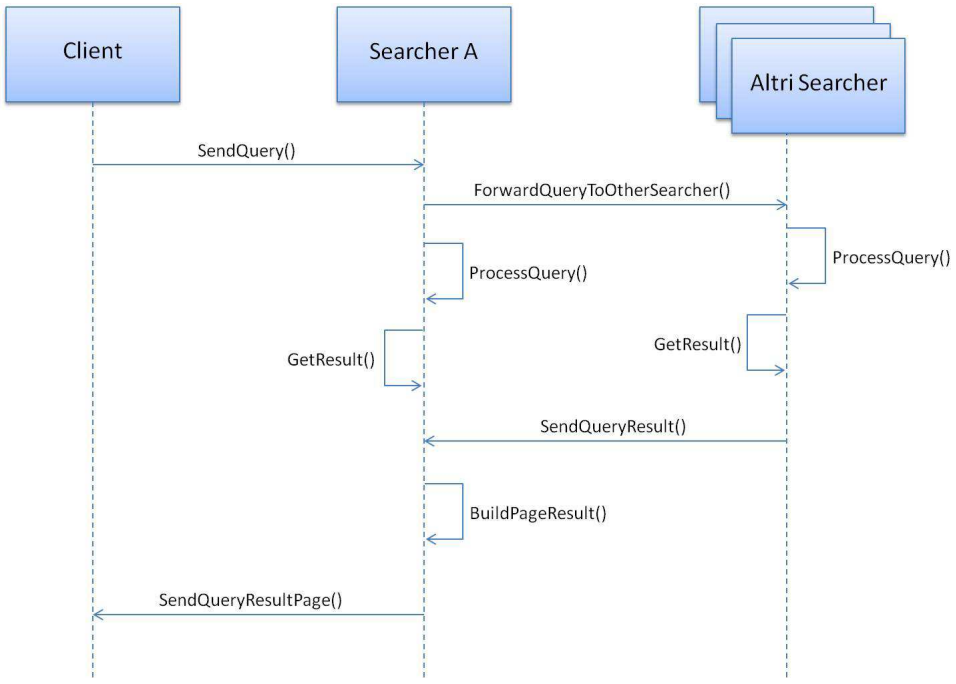


Figura 42 – Diagramma di interazione dei moduli del Searcher al momento dell'arrivo di una query. Il diagramma mostra il caso semplice, dove il server lavora con un modulo worker soltanto.

Query standard

Sono query di ricerca semantica classiche, quelle possibili in Trek sono la *boolean search* e la *ranked query*. La query ranked restituisce dei documenti ordinati per score, in base ai termini inseriti in ingresso e ai relativi pesi; mentre la boolean search effettua un semplice ricerca booleana.

La query ranked prende in ingresso una lista di termini "qualificati" (con tipo) per cui esiste un indice inverso in Trek. Questi termini possono quindi essere keyword, lemmi, syncon, extradati qualunque. Se ci sono altri vincoli che i risultati devono soddisfare si possono mettere nel filtro booleano della query stessa che viene applicato alla fine dell'elaborazione,

una volta trovati tutti i documenti rilevanti. Il risultato è una lista di documenti ordinati per “rilevanza”.

Appendice B

La tassonomia IPTC

Di seguito vengono riportate le categorie di primo livello dell'albero IPTC tratto dal sito internet dell'organizzazione [19], sono 17 categorie che rappresentano il livello più alto di classificazione delle notizie.

Level 1 - NewsCode	Name (it)	Definition (it)
subj:01000000	Arte, cultura, intrattenimento	Creazione e rappresentazione dell'opera d'arte, gli interessi intellettuali, il gusto e le emozioni umane
subj:02000000	Giustizia, Criminalità	Le regole e le leggi, la punizione delle trasgressioni, le forze di polizia, le organizzazioni e le persone coinvolte in attività criminali.
subj:03000000	Disastri, Incidenti	Disastri naturali o provocati dall'uomo che determinano la morte o il ferimento di esseri viventi e/o danni a oggetti o proprietà
subj:04000000	Economia, affari e finanza	La pianificazione, la produzione, la circolazione, lo scambio e il consumo della ricchezza
subj:05000000	Istruzione	Acquisizione della conoscenza e apprendimento tramite la formazione in tutte le sue caratteristiche
subj:06000000	Ambiente	Condizione dell'ecosistema del pianeta, protezione e danni
subj:07000000	Salute	Conseguimento e mantenimento del buono stato psicofisico della persona, ma anche degli altri esseri viventi
subj:08000000	Storie, Curiosità	Argomenti di natura insolita o bizzarra, temi leggeri.
subj:09000000	Lavoro	Aspetti sociali, organizzazioni, regole e condizioni riguardanti impiego e forme di occupazione e le forme di sostegno per chi non è occupato
subj:10000000	Tempo libero	Piaceri personali e svaghi, gastronomia e viaggi
subj:11000000	Politica	Attività di governo e legislativa a livello locale, regionale, nazionale e internazionale, i partiti, le relazioni tra enti governativi e tra gli stati
subj:12000000	Religioni, Fedi	Insieme dei dogmi, dei riti, delle credenze e delle tradizioni riguardanti la teologia, filosofia, etica e spiritualità.
subj:13000000	Scienza, Tecnologia	Comprensione e conoscenza della natura e del mondo fisico e lo sviluppo e applicazione di tale sapere
subj:14000000	Sociale	Aspetti del comportamento umano che influiscono sulla qualità della vita

subj:15000000	Sport	Esercizio competitivo che richiede uno sforzo fisico. Organizzazioni ed enti coinvolti in tali attività
subj:16000000	Agitazioni,Conflitti, Guerre	Atti di protesta e/ o violenza motivati da conflitti sociali o politici
subj:17000000	Meteo	Studio, previsione e bollettini riguardanti i fenomeni meteorologici

Per dimostrare concretamente quali temi sono coinvolti nella tassonomia e il livello di dettaglio della classificazione, si è voluto riportare un esempio dell'albero su tre livelli, espandendo uno dei macrotemi principali dell'albero di primo livello riportato sopra.

In particolare l'espansione riportata riguarda l'argomento "arte, cultura e intrattenimento".

Level 1/ NewsCode	Level 2/ NewsCode	Level 3/ NewsCode	Name (it)	Definition (it)
subj:01000000			Arte, cultura, intrattenimento	Creazione e rappresentazione dell'opera d'arte, gli interessi intellettuali, il gusto e le emozioni umane
	subj:01001000		Archeologia	Studio delle civiltà e delle culture del passato attraverso la raccolta e documentazione delle tracce e vestigia lasciate
	subj:01002000		Architettura	Ideazione e progettazione di edifici, monumenti e degli spazi loro circostanti
	subj:01003000		Corrida	Tradizionale combattimento fra uomo e toro
	subj:01004000		Feste, Carnevale	Parate, feste, celebrazioni ed eventi simili non necessariamente legati ad occasioni o date fisse
	subj:01005000		Cinema	Ideazione, produzione, realizzazione di storie attraverso la pellicola cinematografica e fruizione da parte del pubblico
		subj:01005001	Festival Cinema	Festival e rassegne cinematografiche, selezioni, giurie, candidature, premi, ecc.

	subj:01006000		Ballo	Comunicazione di un'emozione o messaggio attraverso il movimento.
	subj:01007000		Moda	Espressione di un gusto e di una cultura nel disegno e nella creazione di abiti e accessori, l'industria, le sfilate, il design
		subj:01007001	Gioielli	Accessori ornamentali da indossare, di solito realizzati con metalli o altri materiali nobili ma anche la bigiotteria
	subj:01008000		Linguaggio	Forme attraverso le quali le persone comunicano.
	subj:01009000		Biblioteche, musei	Collezioni di libri, musica, arte, o oggetti del passato e del presente per fruizione da parte del pubblico o per esposizione.
	subj:01010000		Letteratura	Opuscoli, libri o altro materiale stampato per comunicare al pubblico idee, storie o altri messaggi.
		subj:01010001	Narrativa	Racconto in forma di prosa di storie strutturate che generalmente non si basano su fatti reali e che sono frutto dell'immaginazione degli autori; romanzi, racconti, novelle
		subj:01010002	Poesia	Arte, struttura e forme dell'espressione poetica
	subj:01011000		Musica	Espressione di un'emozione o di un messaggio attraverso l'uso di strumenti e/o della voce per mezzo di differenti suoni, toni, armonie
		subj:01011001	Classica	Genere musicale che segue strutture classiche di ritmo e armonia, musica colta europea sviluppatasi fino al XIX secolo
		subj:01011002	Folk	Genere musicale sviluppato dalle culture popolari, e che trova le

				sue radici nelle tradizioni di un popolo o di una particolare zona geografica
		subj:01011003	Jazz	Genere musicale caratterizzato da armonie diverse, spesso improvvisate, nato fra fine ‘800 e primo ‘900 nel sud degli Usa
		subj:01011004	Pop	Genere musicale in cui rientrano in particolare le ultime novità discografiche, in contrapposizione ad altri generi considerati colti, destinato soprattutto alle giovani generazioni
		subj:01011005	Country	Genere musicale tipico degli Usa, con caratterizzazione analoga al Folk e con la presenza di forme musicali della tradizione americana come la ballata
		subj:01011006	Rock and roll	Popolare genere di musica da ballo nato negli anni '50 negli Usa e poi diffusosi nel mondo. Da esso sono derivati vari altri generi, di solito indicati con l'abbreviazione ‘Rock'
		subj:01011007		
	subj:01012000		Pittura	Espressione di un'emozione o di un messaggio attraverso l'uso di strumenti di raffigurazione ad olio, tempera, pastelli, matite, gesso, matite a pastello, ecc... su varie superfici.
	subj:01013000		Fotografia	Procedimento che permette di creare immagini e di fissarle su di una superficie utilizzando la luce e materiali ad essa sensibili con prodotti chimici, o per mezzo di strumenti e procedimenti digitali.
	subj:01014000		Radio	Strumento di comunicazione di massa,

				notizie collegate ad esso come arte e spettacolo
	subj:01015000		Scultura	Espressione di un'emozione o di un messaggio con rappresentazioni di forme in materiali di vario tipo (argilla, pietra, legno, metalli, o altro)
		subj:01015001	Arti plastiche	Forme d'arte create a mano o vari tipi di installazioni d'arte
	subj:01016000		Televisione	Strumento di comunicazione di massa, notizie collegate ad esso come arte e spettacolo
		subj:01016001	Soap Opera	Genere di dramma interpretato da attori inizialmente nato per la radio e poi diffusosi al mezzo televisivo, trasmesso in numerosi episodi o puntate; Serial Televisivo
	subj:01017000		Teatro	Rappresentazione artistica sulla scena per raccontare una storia o un'idea attraverso il dialogo e/o la musica
		subj:01017001	Musica a Teatro	Opera, operetta, varietà musicale, musical, rivista ecc.
	subj:01018000		Monumenti, Siti storici	Aree o opere che commemorano personaggi o eventi di rilevanza storica
	subj:01019000		Tradizioni	Usi e costumi di una comunità sviluppatisi nel corso del tempo
	subj:01020000		Arti (generico)	Espressione di un messaggio o un'emozione attraverso musica, letteratura, pittura, teatro o altri mezzi.
	subj:01021000		Intrattenimento (generico)	Spettacolo attraverso tv, radio, teatro, musica e altri mezzi per intrattenere un pubblico
		subj:01021001	Premi	Premi e cerimonie di premiazione nel mondo dello spettacolo e dell'industria dell'intrattenimento: Oscar, Grammy, ecc.

	subj:01022000		Cultura (generico)	Insieme delle idee, costumi, arti, attitudini di un determinato gruppo o di una determinata società.
		subj:01022001	Sviluppo culturale	Storia dello sviluppo artistico e culturale
	subj:01023000		Locali notturni	Esercizi aperti in orari notturni dove è possibile fruire di musica e/o di altre forme di intrattenimento e spettacolo
	subj:01024000		Fumetti	Illustrazioni editoriali, caricature e vignette
	subj:01025000		Animazione	Cartoni animati, lungo e cortometraggi, artisti e commercializzazione di oggetti collegati a personaggi di cartoni animati
	subj:01026000		Mass Media	Mezzi di comunicazione di massa: televisione, radio, riviste, giornali, ecc ...
		subj:01026001	Periodici	Pubblicazioni periodiche o riviste con cadenza settimanale, quindicinale, mensile o altro
		subj:01026002	Mezzi informazione	Televisione, servizi via cavo, radio e altro con la raccolta di notizie e informazioni e la presentazione di esse al pubblico
		subj:01026003	Giornali	Pubblicazioni quotidiane o settimanali di notizie o con servizi speciali
		subj:01026004	Recensioni	Articoli critici o di commento dedicati al lavoro artistico: un film, un'opera teatrale o uno scritto, un'opera musicale
	subj:01027000		Internet	Il web e le sue potenzialità nel mondo della comunicazione e nell'intrattenimento
	subj:01028000		Storia	Fatti connessi con il passato: scoperte, conservazione o scoperte di manufatti, anniversari di eventi di portata storica

Bibliografia

[1] I. D. Melamed, «Empirical methods for MT lexicon development,» Philadelphia.

[2] P. F. Brown, «A statistical approach to language translation,» Budapest, Hungary, 1990.

[3] F. J. Och e H. Ney, «The alignment template approach to statistical machine translation,» Association for computational linguistics, 2004.

[4] R. Rapp, «Automatic identification of word translation from unrelated English and German corpora».

[5] J. Tiedemann, Automatic construction of weighted string similarity measures., Uppsala University, 1999.

[6] Y. Al-Onaizan, J. Curin, M. Jahr, K. Knight, J. Lafferty, D. Melamed e F.-J. Och, Statistical machine translation, JHU Workshop, 1999.

[7] W. Heeringa, J. Nerbonne e P. Kleiweg, «Validating dialect comparison methods,» 2002.

[8] G. Kondrak, «Phonetic alignment and similarity,» University of Alberta.

[9] M. A. Convington, «An algorithm to align words for historical comparison,» 1996.

[10] H. L. Somers, «Similarity metrics for aligning children's articulation data,» 1998.

[11] D. Gildea e D. Jurafsky, «Learning bias and phonological-rule induction,» 1996.

[12] B. Kessler, «Computational dialectology in Irish Gaelic,» 1995.

[13] M. P. Oakes, «Computer estimation of vucabulary in protolanguage from word list in four daughter languages,» 2000.

[14] «Precision and recall,» [Online].

Available: http://it.wikipedia.org/wiki/Precisione_e_recupero.

[15] G. A. Miller, R. Beckwith, C. D. Fellbaum, D. Gross e K. Miller, «WordNet: database lessicale ad uso pubblico,» 1990. [Online].

[16] «ETL,» [Online].

Available: http://it.wikipedia.org/wiki/Extract,_transform,_load.

[17] «Proximity search,» [Online].

Available: http://en.wikipedia.org/wiki/Proximity_search_%28text%29.

[18] «Microsoft translation MSDN documentation,» [Online].

Available: http://msdn.microsoft.com/en-us/library/dd576287.aspx.

[19] «IPTC WEB SITE,» [Online].

Available: http://www.iptc.org/site/Home/.

[20] J. Griggs, «Microsoft translator Java API,» [Online]. Available: https://github.com/boatmeme/microsoft-translator-java-api.

[21] R. Nazar, «Parallel corpus alignment at the document, sentence, and vocabulary levels.,» Barcellona, 2011.

[22] A. Becca. [Online].

Available: http://andreabecca.nova100.ilsole24ore.com/2011/05/traduttori-automatici-come-funzionano-e-perch%C3%A9-sono-gratuiti.html.

[23] I. D. Melamed, «A word - to - word model of translation equivalence».

[24] J. Slocum, «A survey of machine translation: its history, current

status, and future prospects,» Austin, Texsas, 1985.

[25] «Levenshtein distance,» [Online].

Available: http://en.wikipedia.org/wiki/Levenshtein_distance.

[26] «Rete semantica,» [Online].

Available: http://it.wikipedia.org/wiki/Rete_semantica.

[27] «Locuzioni,» [Online].

Available: http://www.treccani.it/enciclopedia/locuzioni_%28Enciclopedia-dell%27Italiano%29/.

[28] P. Koehn, F. J. Och e D. Marcu, «Statistical phrase-based translation,» 2003.

[29] G. Kondrak, D. Marcu e F. Knight, «Cognates can improve statistical translation models,» Edmonton, Canada.

[30] F. J. Och e H. Ney, «A systematic comparison of various statistical alignment models,» 2003.

Ringraziamenti

Alla conclusione di questo importante traguardo, trovo doveroso ringraziare le persone che lo hanno reso possibile, ognuno a modo suo chi più e chi meno, per me sono stati comunque davvero importanti.

Ringrazio innanzitutto la professoressa Milano per la sua disponibilità nell'avermi guidato in questo progetto di tesi, per la professionalità e la passione con cui si dedica al suo lavoro. Nell'ambito universitario vorrei ringraziare anche il professor Denti, che oltre ad avermi guidato nel precedente lavoro di tesi, è riuscito a motivarmi ed indirizzarmi anche in quest'ultimo percorso.

Ringrazio Expert System Spa, nella figura di Marco Varone, che mi ha dato l'opportunità di svolgere questo lavoro di tesi all'interno dell'azienda e di entrare in contatto con le sue tecnologie specializzate.

Voglio ringraziare davvero **tutti** i colleghi di Expert System, perché sono davvero un bel gruppo, con il quale mi sono trovata troppo bene; li ringrazio per la pazienza e per il tempo che hanno dedicato a spiegarmi. Tra questi vorrei citare Sara, grazie per il supporto e per l'amicizia che si è creata.

Ringrazio la mia famiglia, mia mamma, mio papà e mia sorella, perché senza di loro tutto questo non sarebbe stato possibile. Grazie per tutta la sopportazione dei tanti momenti difficili che ci sono stati, per il sostegno e per la fiducia che mi avete regalato.

Ringrazio il mio fidanzato Filippo, perché è stato sempre presente e mi ha sostenuto in ogni momento, senza tirarsi mai indietro. Grazie per la pazienza con la quale hai ascoltato e stemperato le mie difficoltà.

È venuto il momento degli amici, tanti, e tutti davvero importanti per me. Ringrazio Giadina, perché è capace di strapparmi sempre un sorriso e mettermi di buon umore. Ringrazio Frency, la mitica compagna di banco che non devo perdere di vista. Ringrazio Erica, che ha condiviso con me

gioie e dolori universitari, perché si sa, il carico condiviso con qualcuno che ne conosce il peso e che ti sostiene sembra essere sempre un po' più leggero. Grazie a Silvan, perché mi ascolta, mi incoraggia e mi ricorda quanto sia appassionante l'arrampicata sportiva. Per questo ringrazio anche Massimo e Virgi per avermi permesso la domenica di portare la testa altrove, in giro per qualche falesia. Grazie ai miei compagni di corso Seby, Cristina, Baio e Luca DB con i quali ho condiviso estenuanti pomeriggi di studio e tanti progetti; spero di non perderli mai di vista e ritrovarmi con loro a rivivere qualche ricordo universitario divertente.

Spero di non aver dimenticato nessuno, in ogni caso grazie di cuore a tutti voi per essere con me oggi, alla fine di questo difficile traguardo.

Sara