Consiglio Nazionale delle Ricerche
Istituto di Calcolo e Reti ad Alte Prestazioni

XONTO-LING:
uno Strumento per l'Acquisizione di Proprietà Linguistiche da Documenti Testuali

Cristian D'Andrea
Massimo Ruffolo

ICAR-CNR - Istituto di Calcolo e Reti ad Alte Prestazioni
del Consiglio Nazionale delle Ricerche -
Via P. Bucci 41/c, c/o Università della Calabria - 87030 Rende (CS) -
Tel: 0984-831711, Fax: 0984-839054,
Email: cristian.dandrea@gmail.com
ruffolo@icar.cnr.it

Rapporto Tecnico ICAR/CS/2009/??
Maggio 2009

XONTO-LING:
uno Strumento per l'Acquisizione di
Proprietà Linguistiche da Documenti Testuali

SOMMARIO

ICAR-CNR (*Istituto di Calcolo e Reti ad Alte Prestazioni del Consiglio Nazionale delle Ricerche*)
Via Pietro Bucci 41C – 87030 Rende (CS) – Italy - tel. +39 0984 831720 – fax. +39 0984 839054

1. Scopo

Lo scopo del presente documento è quello di descrivere un aspetto dell'*Information Extraction* (IE), ed in particolare della *Named Entity Recognition* (NER) che riveste un ruolo di rilievo nelle moderne tecniche di processamento del linguaggio naturale (*Natural Language Processing*, NLP). Oltre ad una panoramica sulle problematiche e sugli approcci a questi metodi si descriverà l'architettura e le funzionalità di uno strumento software realizzato allo scopo di estrarre contenuti semantici e lessicali da documenti di testo non strutturati (XONTO-LING). Nello specifico l'applicazione si propone come una soluzione per l'estrazione di *Named Entity*, lemmi, *Part of Speech tag* e relazioni semantiche (sinonimie, iperonimie, iponimie ed altro ancora).

2. Introduzione

La crescita vertiginosa del Web mette a disposizione una quantità potenzialmente infinita di contenuti testuali. Tali documenti, pur essendo capaci di fornire un'enorme mole di informazioni da utilizzare in applicazioni reali, risultano essere fortemente non strutturati e, quindi, difficili da manipolare in maniera automatica per mezzo di elaboratori elettronici. Negli ultimi anni l'accesso intelligente, diffuso ed automatico all'ingente quantitativo di dati disponibili sulla Rete è comunque divenuto uno dei principali bisogni degli utenti di Internet. Tutto ciò stimola studi riguardanti il trattamento automatico delle informazioni che danno origine ad approcci e sistemi destinati ad esempio ad estrarre informazioni da pagine Web e/o annotarne automaticamente i contenuti [15]. E' possibile inoltre abilitare funzionalità di ricerca avanzata (basata non solo su parole chiave ma anche su concetti) che sfruttino non solo la struttura sintattica delle informazioni ma anche il loro contenuto semantico [9]. Approcci e sistemi questi basati a loro volta su risultati provenienti dal settore del *Natural Language Processing* (NLP), considerato un sottocampo dell'intelligenza artificiale. Lo scopo degli studi in questa area è trovare una mediazione tra il linguaggio umano e la capacità di comprensione dei calcolatori, limitata alla descrizione delle informazioni mediante le sole regole formali. NLP si occupa principalmente della conversione del linguaggio umano in una rappresentazione formale che sia semplice da manipolare per un calcolatore. Le tecniche di NLP, infatti, permettono di acquisire una serie di proprietà linguistiche di base di un testo espresso in linguaggio naturale. Queste informazioni possono, poi, essere utilizzate per la modellazione di *pattern* utili alla definizione di approcci più complessi destinati al trattamento di informazioni non strutturate (estrazione, annotazione, classificazione, ecc.). In particolare, in NLP esistono alcuni sotto settori quali *PoS-Tagging* e *Lemmatization* che consentono di effettuare l'analisi grammaticale automatica di un testo e, quindi, di acquisire le tipologie delle parti del discorso presenti in una frase ed i lemmi delle parole coinvolte. Un altro settore importante è quello che riguarda la *Named Entity Recognition* (NER) che consente di riconoscere all'interno di una frase la presenza di n-grammi (gruppi di parole) aventi uno specifico significato (nomi di luoghi, animali, piante, medicinali, ecc.). Ed ancora, mediante i database lessicali è possibile risalire ai diversi significati di un vocabolo (sinonimie), oppure alle parole più generali o specifiche (casi di iperonimia ed iponimia) ed altro ancora.

Questo lavoro di tesi concerne lo studio delle tecniche, degli approcci e dei sistemi, provenienti dalle aree del NLP appena citate con particolare riferimento al settore del NER. Riguardo quest'ultimo è stato realizzato un completo stato dell'arte che traccia l'evoluzione di tali tecniche dalle loro origini fino ad oggi. Il presente documento, inoltre, mostra lo sviluppo di un modulo software, chiamato XONTO-LING, per l'acquisizione di proprietà linguistiche da documenti testuali sfruttando i sistemi più promettenti tra quelli analizzati. L'architettura ideata ed implementata permette quindi di: riconoscere ed estrarre le così dette *Named Entity*; effettuare l'analisi grammaticale (*PoS-Tagging* e *Lemmatization*) di frasi ed acquisire informazioni sulla loro struttura sintattica; ed ancora ottenere da database lessicali (ad esempio *WordNet*) informazioni sulla semantica e sul lessico delle parole che compongono una frase (sinonimi, meronimi, olonimi, ecc.). Tale modulo, dotato di una opportuna API, costituisce uno strumento utilizzabile nel contesto di applicazioni più ampie finalizzate all'estrazione di informazioni dal Web e all'annotazione semantica.

Il restante contenuto del documento è strutturato come segue: il capitolo 3 delinea il problema dell'estrazione delle Named Entity. Dalle esigenze che hanno portato all'identificazione di queste entità, proseguendo con una descrizione di cosa rappresentano all'interno di uno specifico contesto. Si introdurranno gli studi condotti in questo campo dell'*Information Extraction* e degli approcci per la soluzione al problema. Si descriveranno le *competitions* ed i progetti che nel tempo hanno sostenuto ed incoraggiato questi studi, e si tratteranno le tecniche di apprendimento automatico utilizzate allo scopo di rendere queste operazioni più o meno indipendenti dalla presenza dell'uomo. Si sono isolati i fattori che caratterizzano l'estrazione delle entità e le *features* che hanno lo scopo di individuare i *pattern* per il loro riconoscimento. Il riassunto di uno studio che copre ben diciassette anni di lavoro nel settore e che metterà in risalto l'impegno dei ricercatori nel campo, ed i risultati raggiunti nel corso degli anni. Nel quarto ed ultimo capitolo si descrive l'architettura di XONTO-LING, un modulo destinato ad essere integrato in XONTO, un sistema per l'estrazione di informazioni da documenti non strutturati [39]. XONTO-LING si propone come un valido strumento per l'individuazione di contenuti lessicali e semantici presenti in documenti testuali. Se ne descriveranno i singoli moduli operativi che prevedono l'estrazione di *Named Entity*, lemmi, *PoS-tag* e relazioni semantiche, di cui si fornirà una breve introduzione per ciascun argomento. Si descriveranno le API esportate da XONTO-LING spiegando le strutture su cui si basa l'idea che si è voluta realizzare.

3. *Named Entity Recognition*

Gli ultimi decenni hanno visto la comunità scientifica impegnata nel campo del *Language Engineering*. Molti sono stati gli studi e le tecniche sviluppate a fronte delle nuove e sempre più pressanti esigenze riguardanti lo sviluppo di sistemi per l'elaborazione del linguaggio naturale (*Natural Language Processing*). Quest'ultimo riguarda quei processi di estrazione di informazioni semantiche da espressioni del linguaggio umano, scritte o parlate, tramite l'elaborazione da parte di un computer. Parole e termini assumono un significato ben preciso in relazione al contesto in cui appaiono, quindi in base all'argomento della discussione e ai termini posti accanto. In pratica lo stesso termine può assumere differenti significati e per contro lo stesso significato può essere espresso con vocaboli differenti. L'*Information Retrieval* (IR) è quindi la scienza che si occupa di ricercare frammenti di contenuti informativi all'interno di documenti o basi di dati, siano questi stand-alone o facenti parte di reti dai contenuti ipertestuali, come il World Wide Web, apportando una notevole riduzione dei tempi di lavoro ed una diminuzione del quantitativo di overload informativo. Il suo utilizzo è previsto in quasi tutte le discipline scientifiche e non solo, diventando un elemento irrinunciabile per tutti gli ambiti di studio e di lavoro. Una semplice procedura di *Information Retrieval* ha inizio con l'immissione di una query nel sistema: questa rappresenta una formale richiesta di una specifica informazione, allo scopo di ottenere una risposta ad una domanda formulata in linguaggio naturale. Nell'ambito dell'IR essa però solitamente non identifica un unico oggetto della collezione, ma restituisce molte corrispondenze (match multipli), ciascuno con un grado di rilevanza (*score*) associato. Sarà compito del sistema determinare quale sia il dominio di pertinenza e da questo selezionare l'entità adeguata. Nell'ambito del *Natural Language Processing*, l'*Information Extraction* (IE) è una forma evoluta di *Information Retrieval* il cui obiettivo è estrarre automaticamente informazioni strutturate da una sorgente di dati *machine-readable* dalla forma non organizzata. L'*Information Extraction* è quindi un processo che, preso in input un testo mai analizzato, produce un output in un formato stabilito e non ambiguo. Questi dati possono essere direttamente mostrati all'utente, oppure salvati in database o fogli di calcolo per ulteriori analisi. Il passo successivo solitamente prevede l'applicazione del ragionamento logico allo scopo di derivare contenuti informativi basati su regole di inferenza a partire da una collezione di dati di input. La maggior parte delle sorgenti informative sono conservate in basi di dati non relazionali oppure sono documenti privi di metadati (Internet ne è l'esempio per eccellenza), con la conseguente difficoltà nel poterne ricavare la conoscenza in essa contenuta. Tra le tipiche applicazioni basate su *Information Extraction* troviamo la *Named Entity Recognition* che si occupa di trovare e classificare le entità presenti in un documento testuale non strutturato in nomi, luoghi, ecc.

3.1. *Named Entity*: una definizione

Il termine "*Named Entity*", largamente usato nelle discipline dell'*Information Extraction*, del *Question Asnswering*, e negli altri settori del *Natural Language Preocessing*, fu coniato nel Novembre del 1995 in occasione del MUC-6, la sesta edizione della *Message Understanding Conferences*, che influenzò positivamente la ricerca nel campo dell'IE. Uno degli obiettivi proposti nel corso dell'evento prevedeva lo studio di nuovi approcci per l'elaborazione di informazioni appartenenti a società ed istituti estratte da fonti testuali non

strutturate; ad esempio nella forma di articoli di giornale. Il compito vedeva quindi l'indiscussa utilità del rilevare le singole unità informative (ad esempio i nomi intesi come di persona, di società o di località; ed espressioni numeriche come date, ore, percentuali o importi). Identificare i riferimenti a queste entità racchiuse in testi ed in altre fonti informative identifica un fondamentale task dell'IE noto come *"Named Entity Recognition and Classification"* (NERC). Nel corso del tempo sono nati e si sono evoluti molti progetti basati su questa disciplina; ad esempio il *Multilingual Entity Tracking* che si preoccupa di studiare ed ampliare il dominio del NERC in più lingue (ad esempio giapponese, cinese e spagnolo oltre ovviamente all'inglese in cui si sono avuti i primi esperimenti nel settore), conservando un approccio al problema del tutto indipendente le une dalle altre. In breve gli studi si sono allargati anche al di fuori degli Stati Uniti, luoghi che hanno visto la nascita del fenomeno: si cita tra tutti il progetto IREX (*Information Retrieval and Extraction Exercise*) in Giappone [40], ed il progetto condiviso CONLL [44] [45] per quattro lingue: inglese, tedesco, olandese e spagnolo. Purtroppo le *Named Entities* sono spesso parole sconosciute che non appartengono al dizionario di riferimento. Questa è una situazione molto difficile da affrontare soprattutto nei linguaggi senza delimitatori di parola, come ad esempio nel giapponese o nel cinese. Molti analizzatori morfologici infatti tendono a confondersi quando incontrano parole sconosciute o segmenti inappropriati Di base tutte i sistemi tendono ad applicare delle regole di riconoscimento ed alcuni di essi sono forniti di funzionalità di apprendimento. Le informazioni da utilizzare sono raccolte in diverse forme: i *gazetteers* elencano i nomi e le informazioni di determinate entità (ad esempio nomi geografici con le rispettive coordinate), mentre i *lexicon* sono rappresentati dalla coppia *<name, type>* e possono rappresentare conoscenza del tipo: *"John Smith"* è una entità di tipo *"Person"*. I *pattern* sono invece regole espresse in linguaggio naturale, ad esempio la frase *"cities such as London, New York"*, può essere ricondotta al *pattern* generico *"<type> such as <list of instances>"* espresso secondo la grammatica inglese. *Pattern*s, *lexicon* e *gazetteers* sono fortemente dipendenti dallo scenario dal quale vengono estratti; ad esempio le pagine web di uno stesso dominio hanno spesso delle similarità sulle voci dei lexicon e dei gazetteers. I sinonimi sono fonti di altri problemi che hanno come oggetto uno stesso nome con significati diversi su domini differenti. *Named Entity Extraction* e *Named Entity Recognition* sono quindi due aspetti per la risoluzione di una certa problematica affrontata in due tempi differenti: prima la *Recognition* mediante la quale istruiamo il sistema, e poi l'*Extraction* mediante il quale facciamo solo del *pattern* matching.

3.2. Il fattore linguaggio

Buona parte dei lavori di ricerca in questo campo sono orientati agli studi in inglese. Da qualche tempo però è crescente il numero degli studi in altre lingue, ad esempio in Tedesco (CONLL-2003) ed in Spagnolo (CONLL-2002). Il giapponese è stato oggetto di lavoro durante il MUC-6, l'IREX conference ed altre attività minori. Il cinese è stato abbondantemente studiato in letteratura [47] [8] [50] ed anche il francese [33] [35], il greco [5] e l'italiano [3] [11]. Molti altri linguaggi meno diffusi hanno meritato la dovuta attenzione, quali la lingua basca [48], il bulgaro [18], il catalano [7], il cebuano [27], il danese [2], l'hindi [12] [27], il coreano [48], il polacco [34], la lingua istroromena [12], il russo [37], lo svedese [24] ed il turco [12]. Il portoghese è stato esaminato da [32] e nel

corso della HAREM conference è stato da poco rivisitato. Giungono infine l'arabo [23] che ha iniziato a ricevere l'attenzione che merita grazie anche al progetto *Global Autonomous Language Exploitation* (GALE).

3.3. Tipologia del testo e dominio di appartenenza

L'impatto del genere del testo (giornalistico, scientifico, informale, ecc) e del dominio (giardinaggio, sport, business, ecc) è stato trascurato in NERC. Infatti sono pochi i lavori che hanno indirizzato i propri studi verso domini multipli. [28] ha progettato un sistema per l'analisi di e-mail e testi scientifici e religiosi; mentre [29] ha creato un sistema specificamente per l'esame del contenuto testuale dei messaggi di posta elettronica. Il fatto di trasportare un sistema da un dominio all'altro resta la sfida più allettante per gli studiosi del campo. Ad esempio, [36] hanno provato alcuni sistemi su una collezione di dati proposti durante MUC-6 composta da notiziari e da un insieme di trascrizioni di conversazioni telefoniche ed e-mail tecniche. Ciascun sistema ha evidenziato sostanziali variazioni di performance, con stime degli indici di precision e recall variabili tra il 20% ed il 40%.

3.4. I tipi di entità

Le categorie di NE hanno subito nel corso del tempo numerose variazioni, dovute soprattutto all'incremento delle applicazioni di IE ed alla comparsa del *Question Answering* (QA). Questo nuovo campo nel settore ha fornito una forte spinta nei lavori di classificazione. Ad esempio, in *Information Extraction* le categorie dipendono fortemente dal dominio di appartenenza. Se lo scenario di riferimento è *"corporate merger"* le sette categorie di base sono più che sufficienti ad elaborare i contenuti dei documenti riguardanti l'argomento. Ma se lo scenario cambia in *"rocket launch"* è necessaria l'aggiunta di ulteriori *entity class*, come ad esempio il tipo *"name of roket"*. La ricerca nel campo del QA invece si propone di costruire sistemi che possono produrre risposte secche ed esaurienti a domande specifiche. NE gioca un ruolo fondamentale nella creazione di applicazioni che supportano tali richieste. Il sistema analizza la tipologia di risposta ed in base al tipo individuato effettua una ricerca mirata nel campo di quella particolare richiesta. E' facile immaginare che per realizzare un sistema con un alto grado di versatilità è necessario un elevato numero di categorie. Nell'espressione *"Named Entity"*, la parola *"Named"* restringe il campo alle sole entità per le quali sono state definite delle rigide condizioni che ne permettono l'identificazione all'interno di un documento. Ad esempio con la frase *"the auto motive company created by Henry Ford in 1903"* è possibile riferirsi sia al signor Ford che alla "Ford Motor Company". I nomi propri sono stati inclusi come tipi naturali di termini come anche ad esempio le specie biologiche e le sostanze chimiche. C'è un'accettazione generale all'interno della comunità di NERC circa l'inclusione di espressioni temporali e numeriche come le somme di denaro e altri tipi di unità. Mentre alcune istanze di questi tipi sono dei buoni esempi di un rigido design (ad esempio, l'anno 2001 è il 2001th anno secondo il calendario Gregoriano) ce ne sono molte che invece non sono ritenute altrettanto valide e complete (ad esempio il mese Giugno non indica l'anno specifico di riferimento ma

resta piuttosto generico). E‘ lecito sostenere quindi che le definizioni di NE sono in molti casi povere per situazioni pratiche.

Lavori recenti formulano il NERC problem come l‘individuazione generale di “nomi propri” (ad esempio [9] [43]. In generale, i tipi più interessanti per i casi di studio sono tre specializzazioni di “nomi propri”: nomi di persona, di località e di organizzazioni. Questi tre tipi sono noti come *“enamex”* sin dal MUC-6. Il tipo località può essere a sua volta diviso in sottotipi che ne specificano sempre più il dettaglio di interesse: *“city”*, *“state”*, *“country”*, e così via [21] [25]. In modo del tutto simile, anche per le persone possiamo determinare delle sottocategorie quali ad esempio: *“politician”*, *“artist”* ed altro, come descritto nel lavoro di [22]. Il tipo *“person”* è abbastanza comune ed utilizzato nella sua forma originale da [4] che li combina in campo medico con altre richieste per ricavarne nomi su malattie e cure. Nel programma ACE il tipo *“facility”* sussume entità del tipo località ed organizzazione. Il tipo *―GPE”* è usato per rappresentare località dotate di una forma di governo, come comuni e paesi. La classe *“miscellaneous”* è usato nella CONLL conference ed include i nomi propri che non ricadono nella categoria *enamex*. Questa può essere incrementata mediante il tipo *“product”* [2]. Il *“timex”* racchiude date e ore mentre *“numex”* indica valute e percentuali; questi sono termini coniati in sede di MUC. Dal 2003 la comunità TIMEX2 [20] ha proposto uno standard studiato proprio per l‘annotazione e la normalizzazione di espressioni temporali. Infine è possibile individuare alcuni tipi meno comuni che sono solitamente utilizzati e descritti per specifici interessi: ad esempio *“film”* e *“scientist”*[17], *“e-mail address”* e *“phone number”* [49] [28], *“research area”* e *“project name”* [51], *“book title”* [6] [49], *“job title”* [10] e *“brand”* [2]. Il recente interesse nel campo della bioinformatica ed alla partecipazione del GENIA [59] vede molti studi dedicati a tipi particolari quali *“protein”*, *“DNA”*, *“RNA”*, *“cell line”* e *“cell type”* [42] oppure solo all‘individuazione delle sole proteine [46]. Altri lavori includono il rilevamento di nomi appartenenti alle classi *“drug”* [38] e *“chemical”* [30].

Alcuni studi recenti non pongono limiti ai possibili tipi da estrarre ma si riferiscono agli *open domain* NERC [1] [18]. Sulla stessa linea di ricerca, [41] definiscono una gerarchia delle *Named Entities* che include molte categorie a grana fine e quindi a dettagli sempre maggiori, come ad esempio *“museum”,”river”* ed *”airport”*. A questi è possibile aggiungere un vasto insieme di categorie come *“product”* ed *“event”* oppure ancora sottotipi di categorie appartenenti ad esempio ad *“animal”*, *”religion”* o *“color”*. Lo scopo del progetto vedeva il tentativo di individuare i più frequenti tipi di nomi e designatori che appaiono in articoli di quotidiani. Il numero di categorie raccolte in questo esperimento è stato di ben 200, ed è stato inoltre possibile definirne i più comuni attributi in modo da poterle organizzare in ontologie. E‘ da tener presente però che l‘indiscriminata introduzione di nuove categorie di NE può creare alcuni problemi. La definizione di categorie in uno specifico scenario di lavoro non è di per se un compito facile, anche quando si ha a che fare con un dominio che prevede un ristretto numero di classi. Inoltre sono molti i casi in cui diventa difficoltoso identificare la categoria adatta tra quelle a disposizione, creando in molti casi delle forti ambiguità. Questo è l‘annoso problema di classificare il mondo reale in categorie semantiche per il quale non esiste alcuna tecnica specifica che permetta di risolvere la questione che può essere affrontata solo con l‘esperienza ed uno studio attento, allo scopo di limitare le occasioni di incorrere in errore.

3.5. I metodi di apprendimento

Oltre al problema del determinare il numero e gli attributi adatti a rappresentare un dominio di riferimento, è necessario affrontare anche ciò che riguarda l'annotazione (*tagging*) del testo. Soprattutto per le tecniche che prevedono l'approccio orientato al *learning* supervisionato è necessario un ingente quantitativo di dati di training adeguatamente annotato. Purtroppo con l'aumentare del numero di categorie, mantenere consistenti le informazioni risulta essere un lavoro molto oneroso. Prendendo come esempio un esperimento già citato alla fine del paragrafo precedente, secondo un lavoro prodotto da [41] negli scorsi anni, l'annotazione di articoli di un quotidiano giapponese raccolti in 30 giorni ha portato ad identificare ben 200 classi. E' facile rendersi conto come un così esteso insieme di tipi di classi sia complicato ed oneroso da rendere consistente ed aggiornato nel tempo. Un'altra strategia che prevede l'uso di regole e dizionari redatti a mano ha consentito il popolamento del dizionario dei termini prelevando le informazioni dal Web; purtroppo questa lista non copriva in modo esaustivo le espressioni contenute sui documenti da analizzare rendendo in alcuni casi vana anche questa strategia. Una possibile soluzione a questi problemi prevede la riduzione del peso dovuto alle annotazioni dei dati di training, che per l'approccio supervisionato risulta eccessivo. Molte delle attuali tecniche per l'individuazione delle *Named Entity* sono basate sul riconoscimento di *pattern* ben definiti. Sfortunatamente anche semplici regole del tipo

<proper noun>+ <corporate designator> = <corporation>

non sono adatte al riconoscimento delle complesse entità presenti negli attuali documenti, né alla corretta identificazione di nuove convenzioni; ed inoltre non rispetta i canoni di performance oggi necessari per un moderno sistema di NERC. Si è reso quindi fondamentale fornire a queste applicazioni l'abilità di individuare preventivamente entità sconosciute e di imparare da ciò che non è noto. Un simile approccio risulta utile anche nei casi in cui si volesse rendere il sistema in grado di lavorare su più linguaggi. Infatti i *pattern* validi per l'inglese non sono sicuramente adatti a mappare entità espresse in cinese. Queste capacità sono il cardine sul quale le regole di individuazione e classificazione sono attivate ed utilizzate grazie alle caratteristiche distintive associate a dati di esempio positivi o negativi forniti in una fase preliminare di istruzione del sistema. Mentre i primi lavori si basavano su regole scritte a mano, gli studi recenti adottano un approccio all'apprendimento di tipo supervisionato (*Supervised Learning*), attraverso il quale è possibile istruire sistemi basati su regole o su algoritmi di *sequence labeling* a partire da sequenze di dati di training di esempio. Il cambio di tendenza è stato approvato dalla comunità dei ricercatori in riferimento a quanto esposto al MUC-7, poiché tra i lavori presentati cinque sistemi su otto sono stati di tipo *rule-based*. Ben sedici ne sono stati invece presentati nel corso del CONLL-2003. Quando i dati di training non sono disponibili, le regole *handcrafted* restano il metodo preferito, come mostrato da [41]. L'idea su cui si basa l'apprendimento supervisionato è studiare le caratteristiche di esempi positivi e negativi di NE su estese collezioni di documenti annotati, facendo riferimento a delle regole che catturano le istanze di un dato tipo. Il maggior difetto di SL è la richiesta di un ingente quantitativo di annotazioni di partenza. La non disponibilità di queste risorse e il costo proibitivo per la

loro creazione ha fatto si che nascessero due alternative a questo approccio: il *learning semi-supervisionato* (SSL) e quello *non supervisionato* (UL).

3.6. Le *features* nel campo del NERC

Le *features* sono i descrittori degli attributi delle parole da analizzare mediante gli algoritmi di NERC. Il vettore delle *features* è una astrazione in cui ciascun *token* è rappresentato mediante valori Booleani, numerici o nominali. Un ipotetico NERC system potrebbe rappresentare ciascuna parola mediante tre attributi:

- un attributo Booleano con valore true se la parola è maiuscola, false altrimenti;
- un attributo numerico corrispondente alla lunghezza, in caratteri, della parola;
- un attributo nominale corrispondente alla versione in caratteri minuscoli della parola.

In uno scenario simile, la frase *"The president of Apple eats an apple."*, escludendo la punteggiatura, può venir rappresentata con il seguente vettore di *features*:

<true, 3, "the">

<false, 9, "president">

<false, 2, "of">

<true, 5, "apple">

<false, 4, "eats">

<false, 2, "an">

<false, 5, "apple">

Solitamente, per individuare le NE si utilizzano dei sistemi a regole basati su *features* di questo tipo. Ad esempio, si potrebbe avere un insieme composto da due regole, una per il riconoscimento: *"capitalized words are candidate entities"*, ed una regola di classificazione: *"the type of candidate entities of length greater than 3 words is organization"*. Queste regole sono in grado di lavorare su frasi simili a quelle considerate nell'esempio precedente. Comunque un sistema reale è sicuramente più complesso e le sue regole sono spesso create in modo automatico mediante degli algoritmi di *learning*. Gli assi su cui si articolano l'identificazione delle *features* sono tre: *Word-level features*, *List lookup features* e *Document and corpus features*. Le *Word-level features* descrivono casi, punteggiatura, valori numerici e caratteri speciali. Per *"list"* in NERC si intendono delle *feature* privilegiate. I termini *"gazetteer"*, *"lexicon"* e *"dictionary"* sono usati spesso in modo intercambiabile con l'espressione *"list"*. L'inclusione in una *list* è un modo per esprimere una relazione di tipo *"is a"*. Le features all'interno dei documenti sono definite sia sulla sua strutture che in riferimento al contenuto. Grandi collezioni di documenti sono solitamente ottime fonti per estrarre *features* attendibili. Si procede solitamente analizzando le espressioni composte da una o più parole e le meta-informazioni circa il singolo documento e si termina con le statistiche dell'intero corpus.

4. Lo strumento XONTO-LING

Nelle pagine che seguono si illustrerà l'aspetto implementativo del lavoro svolto in questa tesi: XONTO-LING è il modulo di processamento linguistico pensato per XONTO [39], un progetto con uno scenario di utilizzo più ampio, basato su ontologie e che consente l'estrazione di informazioni semantiche da documenti; nello specifico da file in formato PDF. L'idea alla base di XONTO sono le ontologie auto-descriventi (*Self-Describing Ontology*) in cui gli oggetti e le classi possono essere affiancate da un insieme di regole chiamate *"descriptors"*. Queste regole rappresentano i *pattern* che permettono il riconoscimento e l'estrazione automatica degli oggetti contenuti all'interno dei documenti anche in formato tabellare. Il documento è analizzato al fine di costruire una rappresentazione interna adatta al sistema. Nel dettaglio questa analisi consta dei seguenti passi:

1. l'identificazione della struttura bidimensionale del documento;
2. il metching delle stringhe contenute con i *dictionary pattern* a disposizione;
3. l'estrazione delle *features* acquisite dai *metadata* del documento PDF;
4. la tokenizzazione delle stringhe e l'acquisizione per ciascun *token* delle proprie caratteristiche linguistiche (PoS-tag, lemma, ed altro ancora).

Il supporto a quest'ultimo passo è fornito proprio dal modulo XONTO-LING, che utilizza le proprietà di potersi interfacciare a strumenti di NLP, ai sistemi per NEE ed ai moduli che trattano l'individuazione delle relazioni semantiche tra le entità grammaticali.

4.1. Le proprietà linguistiche acquisite da documenti testuali

L'idea alla base di XONTO-LING è quella di racchiudere in un unico modulo gli strumenti per riconoscere ed estrarre le entità all'interno di documenti di testo, l'acquisizione di *features* e *patterns* linguistici, e l'individuazione delle relazioni semantiche. Del primo aspetto si è ampiamente discusso all'interno del capitolo precedente, indicando oltre alle evoluzioni degli studi nel settore, anche alcune tra le tecniche proposte. Per le restanti funzionalità si introdurrà in questo paragrafo lo studio agli approcci ai processi linguistici per l'identificazione delle parti di un discorso (*Part of Speech*) allo scopo di individuare all'interno di un documento il ruolo grammaticale svolto da ciascuna entità. Si parlerà inoltre dell'uso dei lessici semantici (ad esempio *WordNet*) che si preoccupano di risolvere i problemi di organizzazione, definizione e descrizione dei concetti espressi dai vocaboli. L'insieme di questi due moduli, oltre all'identificazione delle *Named Entities*, sono i tre aspetti dell'architettura alla base della nostra idea di gestire i contenuti linguistici di documenti testuali non strutturati. Le applicazioni pratiche di tali interrogazioni in campo informatico sono molteplici e ricadono ad esempio nel campo dell'intelligenza artificiale e della rappresentazione della conoscenza. Le Ontologie sono un tipico esempio di come differenti schemi vengono combinati in strutture in cui sono rappresentate le entità rilevanti e le loro relazioni in uno specifico dominio, utili per approcci di ragionamento induttivo o per la risoluzione di problemi di classificazione.

Oltre a voler riconoscere le *Named Entities* all'interno dei documenti testuali, potrebbe essere estremamente utile in molti ambiti etichettare le singole entità secondo il ruolo grammaticale svolto all'interno di una frase (PoS) e le relazioni semantiche con le altre entità. Tutti aspetti che potrebbero tra loro essere collegati ed elaborati in sequenza, oppure incrociati per dedurne conoscenza aggiuntiva. Come si vedrà in seguito quando parleremo dell'architettura di XONTO-LING, ciascun modulo di analisi viene specializzato secondo uno o più engine di annotazione ed estrazione: per l'aspetto riguardante le *Named Entities* si è scelto di adottare l'utilizzo di GATE, per il *POS-tagging* invece *Freeling*, mentre per l'utilizzo di ontologie lessicali si è specializzato il modulo per poter sfruttare le potenzialità offerte da *WordNet*. Si discuterà nelle pagine seguenti di questi ultimi due aspetti introducendo l'utilizzo degli specifici sistemi adottati.

4.2. Proprietà grammaticali

Il *Part of Speech tagging* è il processo attraverso il quale è possibile identificare il contenuto morfologico di una entità. La morfologia è la parte della linguistica che si preoccupa dello studio della struttura grammaticale delle parole e ne stabilisce la classificazione e l'appartenenza a determinate categorie quali nomi, pronomi, verbi ed aggettivi, riconoscendo inoltre le coniugazioni dei verbi e le declinazioni per i nomi. Senza dilungarci molto nella teoria degli aspetti linguistici per i quali servirebbe una tesi ad hoc, ci limiteremo a dire che una parola è una sequenza di morfemi caratterizzata da diverse accezioni. Un morfema è la minima unità grammaticale isolabile di significato proprio. E' composto di fonemi ed è portatore di un significato proprio e preciso, non indipendente da altri morfemi. I morfemi possono essere tra loro combinati in sequenze lineari per dar luogo a termini complessi. Un semplice esempio chiarificatore potrebbe essere il seguente: la parola *"pesca"* è composta dai morfemi *"pesc"* + *"a"*, quest'ultimo indicante l'appartenenza ad un sostantivo femminile singolare. Per indicare il plurale basterebbe sostituire *"a"* con *"e"* ed apporre il carattere *"h"* come detta la specifica regola della grammatica italiana: *"pesc(h)e"*. In questo modo il morfema non modifica la parte del discorso ma il concetto di numero. Sostituendo invece il morfema *"a"* con *"are"* otteniamo *"pescare"* che individua l'appartenenza della parola alla categoria dei verbi. La cosa si complica ulteriormente se andiamo a considerare aspetti come le flessioni, le derivazioni e le composizioni. Le prime individuano l'insieme delle regole che determinano la funzione logica di una parola. Per i verbi si parla di coniugazioni, mentre per gli elementi nominali sono noti con il termine declinazione. Le derivazioni consentono di soddisfare le esigenze lessicali mediante un insieme di mutamenti allo scopo di fornire un nuovo significato rispetto a quello di partenza. Le composizioni invece sono dei processi morfologici che generano nuove parole partendo da quelle esistenti senza l'utilizzo di prefissi e suffissi. Questo tipo di analisi basa le sue deduzioni sul concetto di *Lemma*. In linguistica un *Lemma* è la forma di citazione di una parola: una forma canonica per intenderci. Il rapporto fra lemmi e parole è di fondamentale importanza per le lingue dotate di inflessione. Per i verbi si utilizza in genere il modo infinito al tempo presente, mentre il lemma degli aggettivi è il maschile singolare. Sostanzialmente il lemma è la forma che rappresenta tutte le altre flesse che la parola specifica può avere. L'operazione che porta dalla forma flessa al corrispettivo lemma si chiama lemmatizzazione.

Anche solo dopo questa breve panoramica è evidente la difficoltà in cui si incorre nel dover procedere ad una analisi di questo tipo con uno strumento informatico poco istruito o approssimativo. L'analisi morfologica considera una parola alla volta, il che non è sempre sufficiente per decidere la categoria grammaticale di appartenenza. E' necessario infatti considerare il contesto di cui l'entità fa parte. Questo è il lavoro svolto dai *PoS tagger*. Per tener conto del contesto è necessario un modello della lingua, oggi realizzati secondo varie tecniche: basati su regole, *Hidden Markov Model*, *Decision Tree*, *Entropy Maximization* ed altre metodologie. Lo strumento usato per equipaggiare XONTO-LING di tale capacità è *Freeling* [15] [16]: una libreria *open-source* che fornisce i servizi di base per operazioni di NLP sotto licenza LGPL. Mediante questo package è possibile svolgere *text tokenization*, *sentence splitting*, *morfological analysis*, *date recognition*, *PoS tagging*, *shallow parsing*, e tra le altre funzionalità l'ultima release è dotata anche di un motore per *NE classification* ed un annotatore basato su *WordNet* (di cui parleremo nel paragrafo successivo). La localizzazione di *Freeling* prevede il supporto allo spagnolo, al catalano, al galiziano, all'inglese ed all'italiano. Tutte dotate di un dizionario attraverso i quali estrarre i lemmi, i PoS tag e tutte le altre informazioni utili. Freeling si propone come la soluzione alle due cruciali problematiche dovute all'integrazione di analizzatori del linguaggio all'interno di applicazioni basate su NLP: la riusabilità e l'efficienza. Essa inoltre è facilmente estendibile ed è portabile su altri linguaggi. Questa facilità di uso e di integrazione è possibile grazie alla sua architettura ben strutturata e fortemente orientata agli oggetti. Questa si presenta come un'architettura client-server a due livelli: il primo contenente i servizi di analisi di base ed il secondo puramente applicativo che, operando come un client, richiede i servizi per mezzo degli analizzatori. In questo scenario, integrare gli analizzatori in una nuova applicazione di NLP si riduce a tre semplici passi: convertire i dati in un formato comprensibile dalle API di *Freeling*; chiamare il servizio ed attendere la restituzione dei risultati; ed in ultimo convertire i risultati in modo comprensibile all'applicazione client. Una simile architettura consente l'uso dell'analizzatore mediante una semplice chiamata ad un metodo e non come un package a se stante. Le richieste del client includono anche moduli che non riguardano strettamente l'ambito dell'NLP, la cui combinazione con le funzioni di base arricchisce le analisi a disposizione. Diviene inoltre non necessario definire dei formati intermedi per lo scambio dei dati tra gli analizzatori. Ciascuna applicazione può infatti scegliere la propria rappresentazione a patto che sappia come mappare le strutture dati ed i parametri passati durante l'invocazione del metodo. In questo modo si riducono gli overhead dovuti alla lettura, al parsing, alla scrittura ed alla trasmissione di formati intermedi. Questo non significa che il client debba adattarsi alle rappresentazioni proprie di *Freeling*, bensì saranno le API a disposizione a gestire i dati ed il loro formato. I processori linguistici a disposizione non necessitano di continua inizializzazione per ciascuna analisi, ed è l'applicazione inoltre a decidere come e quando invocare ciascun analizzatore. L'approccio client-server permette le interazioni tra gli oggetti attraverso i più comuni standard per la comunicazione distribuita (ad esempio CORBA), che rendono possibile la distribuzione delle applicazioni anche in rete.

Tra le varie funzionalità esportate, quella che più interessa il questa parte di lavoro è il modulo per il *Part-of-Speech tagging* per il quale si prevedono due algoritmi. Il *PoS tagger* di base utilizza un classico algoritmo basato su *trigram HMM* istruito mediante i dizionari di cui *Freeling* è fornito. Il tagger si attesta con una precisione di quasi il 97% sulla maggior parte dei linguaggi che localizzano il sistema. Il secondo algoritmo si basa su un *relaxation*

labelling model. Nonostante le performance di entrambi i tagger siano pressoché simili, si è scelto di dotare *Freeling* di due alternative allo scopo di permettere l'uso sia di regole scritte a mano che di modelli statistici.

L'architettura interna del sistema si fonda su due tipologie di oggetti: *linguistic data object* e *processing object*. Le classi base della libreria sono utilizzate per contenere le informazioni linguistiche (una parola, un PoS tag, una frase, un intero documento). Ciascuna applicazione deve poter conoscere queste classi allo scopo di poter fornire ai moduli per le analisi le giuste informazioni ed il corretto modulo per l'interpretazione dei risultati estratti. Le *Processing Classes* invece contengono le librerie di trasformazione dei dati; quali ad esempio:

- *tokenizer*: riceve del testo piano e restituisce una lista di *word object*;
- *splitter*: riceve una lista di *word object* e restituisce una lista di *sentence object*;
- *morfo*: riceve una lista di *sentece object* e annota secondo i risultati dell'analisi orfologica ciascun *word object*. Si applicano a cascata una serie processori specializzati e ciascuno di essi restituisce una specifica analisi;
- *tagger*: riceve una lista di *sentece object* e disambigua i PoS di ciascun *word object* nelle frasi date.
- *NE classifier*: riceve una lista di sentece object e classifica ciascuna *word object* rispetto ad un tipo.

e vari parser. Il client è quindi libero di decidere in quale formato ricevere i dati di input, l'output da restituire oppure il salvataggio delle informazioni linguistiche.

4.3. Proprietà lessicali e semantiche

Un altro aspetto affrontato in questo lavoro è l'apporto alle analisi fornito dai lessici semantici, intesi come raccolte dei significati delle singole parole, e le relazioni che intercorrono tra esse. Queste sono collezioni ordinate di informazioni estremamente dettagliate che riguardante la definizione e la descrizione di concetti espressi da vocabolari. Ciascuna parola è organizzata in concetti, la maggior parte delle quali è connessa ad altri concetti tramite relazioni semantiche. Lo scopo è duplice: produrre una combinazione di dizionari e thesaurus che sia usabile ed intuitiva; e supportare le elaborazioni automatiche di testi per applicazioni ad esempio di intelligenza artificiale. Le relazioni semantiche variano in funzione del tipo di parola a cui si applicano: che siano verbi o sostantivi. Alcune delle relazioni semantiche più utilizzate sono riportate in tabella e tabella . Anche in questo caso, una analisi eseguita mediante strumenti informatici risulterebbe particolarmente utile in casi di interrogazioni complesse se alle spalle dell'applicazione è presente una struttura completa e ben organizzata come questa. In XONTO-LING l'implementazione del modulo per la ricerca delle relazioni semantiche è stata realizzata usando *WordNet* come analizzatore. *WordNet* [19] è un lessico semantico che si preoccupa di organizzare, definire e descrivere i concetti rilevanti in lingua inglese. Il progetto è stato iniziato più di dieci anni fa dal linguista George Miller presso l'Università di Princeton. La concettualizzazione del lessico

si avvale di raggruppamenti di termini con significato tra loro affine, chiamati *"synset"* (*synonym set*), e sul collegamento dei loro significati attraverso i diversi tipi di relazione visti nelle tabelle sottostanti.

esempi di Relazioni Semantiche per i sostantivi	
sinonimia:	X è sinonimo di Y se entrambi hanno lo stesso significato (casa ed edificio sono sinonimi)
iperonimia:	Y è iperonimo di X se ogni X è "una specie di" Y (canide è iperonimo di cane)
iponimia:	Y è un iponimo di X se ogni Y è "una specie di" X (cane è iponimo di canide)
olonimia:	Y è un olonimo di X se X è parte di Y (aereo è olonimo di elica)

tabella 1: esempi di relazioni semantiche per i sostantivi

esempi di Relazioni Semantiche per i verbi	
iperonimia:	il verbo Y è iperonimo di del verbo X se l'attività X è "una specie di" Y (viaggio è iperonimo di movimento)
toponimia:	il verbo Y è un troponimo del verbo X se nel fare l'attività Y si fa anche X (parlare e troponimo di mormorare)

tabella 2: esempi di relazioni semantiche per i verbi

Queste entità sono strutturate in nodi e sono collegati da relazioni di senso. I *synset* sono quindi gruppi di sensi sinonimi tra loro, secondo un concetto di sinonimia molto ampio: i sensi devono essere intercambiabili in almeno un contesto. In ciascun concetto, o *synset*, le differenze di senso (polisemie) sono distinte, numerate e definite mediante relazioni tassonomiche e associative. *WordNet* è considerato uno dei più importanti lessici standard per la lingua inglese ed è disponibile gratuitamente su internet, sia consultabile online che scaricabile. Attualmente giunto alla versione 3.0 per Unix, mentre è fermo alla 2.0 per Windows, consta di un totale di ben 147278 stringhe tra nomi, verbi, aggettivi e avverbi. Con il progetto *EuroWordNet* (EWN), finanziato dalla comunità europea dal 1996 al 1999, sono stati sviluppati lessici *WordNet* per vari linguaggi europei, collegati in un database multilingue attraverso un *Inter-Lingual Index* (ILI). Rispetto al progetto inglese, di cui adottano la struttura base, i lessici europei utilizzano una nozione allargata di equivalenza di significato, estesa anche a differenti categorie sintattiche, ed una più ampia

classe di relazioni di senso atte a trattare in modo più approfondito la polisemia. Si è basata la metodologia di sviluppo sul riutilizzo di risorse linguistiche esistenti e su procedure semiautomatiche di costruzione degli alberi lessicali. Ciascun *WordNet* europeo è un lessico autonomo, strutturato da relazioni semantiche interne, ed è in più collegato da una relazione di equivalenza al corrispondente *synset* della lingua inglese (dalla versione 1.5) contenuto nell'ILI. Ciò consente il collegamento di ogni lessico con tutti i lessici sviluppati con la stessa metodologia. Ad oggi sono moltissime le lingue che condividono la medesima struttura e metodologia di sviluppo. Partner di *EuroWordNet* e sviluppatore del *WordNet* per l'italiano (IWN) è l'Istituto di Linguistica Computazionale del C.N.R. di Pisa. Un'altra iniziativa parallela è ad esempio *Balkanet*: un database multilingue per le lingue balcaniche e lessici terminologici per domini specifici (economico, sociologico).

4.4. Architettura

XONTO-LING si propone come uno strumento semplice ed usabile per poter estrarre contenuti lessicali e semantici da documenti testuali. Tramite le tre componenti software di cui si compone, è possibile interrogare il sistema allo scopo di individuare *Named Entity*, oppure fare operazioni di NLP ricavando *pattern* lessicali, ed infine restituire relazioni semantiche tra entità contenute all'interno di un testo. L'esigenza di poter disporre di un unico strumento che includesse queste tre tipologie di analisi ha reso desiderabile la realizzazione di un framework facilmente integrabile ed espandibile. L'architettura è infatti estendibile con altri analizzatori grazie ad un insieme di interfacce che isolano il comportamento di ciascun modulo del framework.

Ma partiamo con ordine. Il comportamento di XONTO-LING è quello di una classica *black box*, in cui si sottomettono degli input e se ne attendono i risultati in un qualche formato comprensibile dall'applicazione client. In ingresso il sistema accetta semplici file di testo oppure in formato HTML o XML: *plain text* ad esempio in formato ASCII o Unicode. Ciò che ci si attende in uscita è una serie di informazioni riguardanti il lessico e la semantica dei documenti sottomessi. Una sestupla di dati nel formato:

< *Word, Type, Lemma, PoS-tag,* < *Position* >, < *Semantic Relation* > >

dove *Word* è un *token* estratto dai documenti in input; *Type* si riferisce alla tipologia della *Named Entity* individuata per quello specifico *token*; *Lemma* e *PoS-tag* sono i risultati dell'analizzatore per NLP e riferiscono rispettivamente al lemma ed al ruolo che l'elemento estratto riveste all'interno della frase in cui compare.

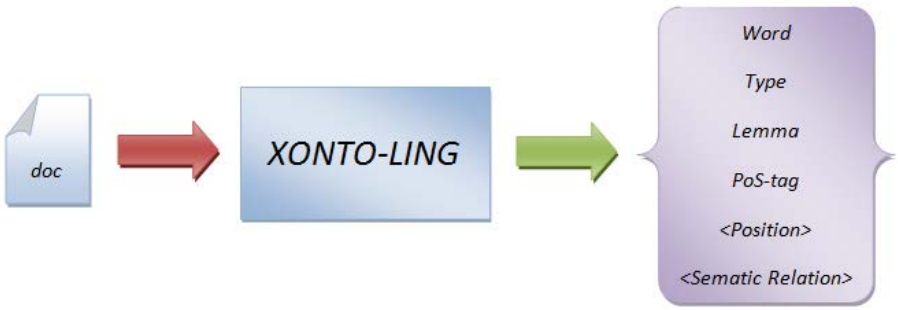

figura 1:XONTO-LING – black box

Una lista di posizioni mantiene le coordinate all'interno del documento in cui compare la *Word*, mentre le relazioni semantiche sono conservate all'interno di una lista apposita.

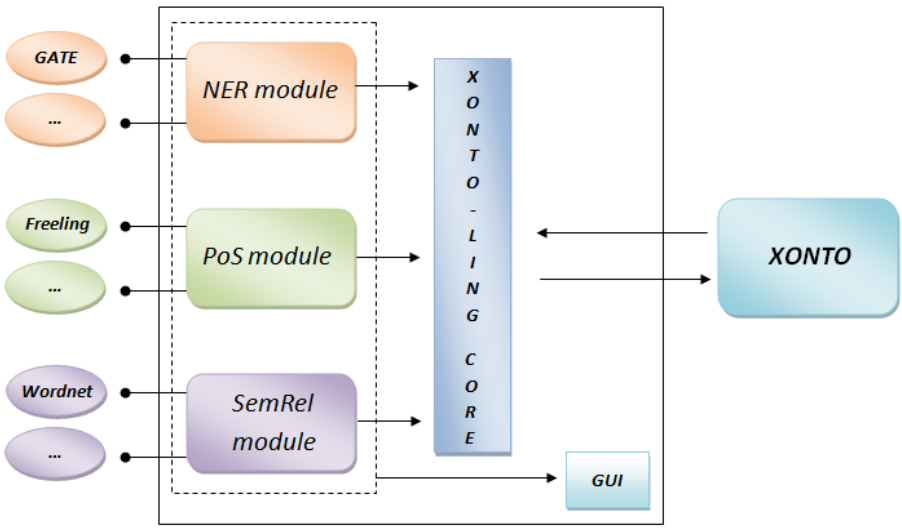

figura 2: architettura di XONTO-LING

In figura apriamo l'involucro di XONTO-LING e ne esploriamo il contenuto. Sono subito evidenti le tre componenti che hanno il compito di svolgere le analisi e di restituirne infine le informazioni richieste:

- *NER module*: si occupa del processo di identificazione e di estrazione delle *Named Entity*;
- *PoS module*: analizza il contenuto morfologico delle singole frasi e restituisce i lemmi ed i *PoS-tag*;
- *SemRel module*: individua le relazioni semantiche intercorrenti tra i *token* contenuti nei documenti in input.

Oltre alla architettura si è proposto anche un possibile scenario di utilizzo implementando le interfacce su tre engine per le annotazioni: la parte sulla NER è stata affidata a GATE, mentre come *PoS tagger* si è scelto *Freeling* e per il modulo sulle relazioni semantiche si è specializzata l'interfaccia per *WordNet*. L'idea di base è quella avere una interfaccia comune per le implementazioni specifiche di ciascun modulo. Si è cercato infatti di isolare le operazioni più utili da effettuare su ciascun tipo di estrazione in modo da avere un comportamento omogeneo su qualsiasi implementazione. Una applicazione client può facilmente interrogare i vari moduli tramite le API esportate. Questa operazione è uniformata indipendentemente da quale engine di estrazione viene usato, questo perché ciascuna implementazione rispecchia l'interfaccia del modulo di appartenenza. Ad esempio, l'utilizzo di due implementazioni appartenenti al *NER module* è identico. Le firme dei metodi di interrogazione sono le stesse, e le specifiche dei singoli approcci alle annotazioni e alle relative estrazioni dei dati sono del tutto trasparenti al client che non si curerà minimamente di questo particolare per lui del tutto irrilevante. In questo modo è possibile dotare l'ambiente di più metodi di estrazione e di scegliere di volta in volta quello preferito, oppure farne i confronti tra l'utilizzo dell'uno anziché dell'altro. Inoltre alcuni motori possono coprire più aree di utilizzo: un engine potrebbe avere funzionalità di NER e di PoS tagging (*Freeling* si comporta in questo modo) e nulla ci impedirebbe di specificare più interfacce su un unico analizzatore.

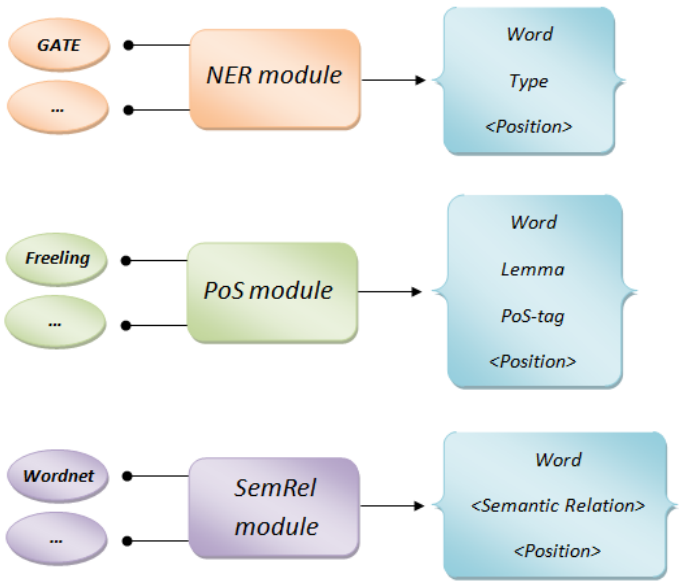

figura 3: le informazioni estratte da ciascun modulo di XONTO-LING

Il ruolo dello *XONTO-LING core* è quello di raccogliere le informazioni restituite dai tre moduli a monte e di aggregarle per comporre la sestupla di dati di cui alla figura . I tre moduli forniscono ciascuno solo la parte di informazioni di loro competenza (figura 3). Sarà compito di un'altra entità quella di combinare queste strutture e restituirle all'applicazione client (in questo caso l'applicazione XONTO) secondo le richieste da lui fatte. Al momento il modulo *core* non è stato implementato poiché si è pensato alla struttura di base del framework, ma sarà oggetto di sviluppi futuri integrare questa idea nel progetto. In questo

modo l'ambiente prodotto rispecchia una valida soluzione alle esigenze di analisi linguistica del sistema XONTO che incorporerà tale struttura tra i suoi moduli di processo delle informazioni.

4.5. Funzioni

Le API esportate da XONTO-LING sono di facile utilizzo e comprensione ed è inoltre possibile aggiungere nuove funzionalità a quelle già presenti. Per la loro spiccata generalità sono integrabili nei sistemi che ne richiedono i servizi e sono soprattutto espandibili con motori di analisi aggiuntivi. La struttura del framework realizzata prevede in primo luogo una generica interfaccia che rappresenta l'oggetto *"Annotation"* (figura 4). Questa classe serve a conservare i risultati individuati dai vari motori di annotazione ed estrazione. Permette di conservare il *token* corrente (*entity*) e la rispettiva annotazione (*value*). Si mantiene anche la lista delle posizioni (*postionList*) delle entità contenute nel documento.

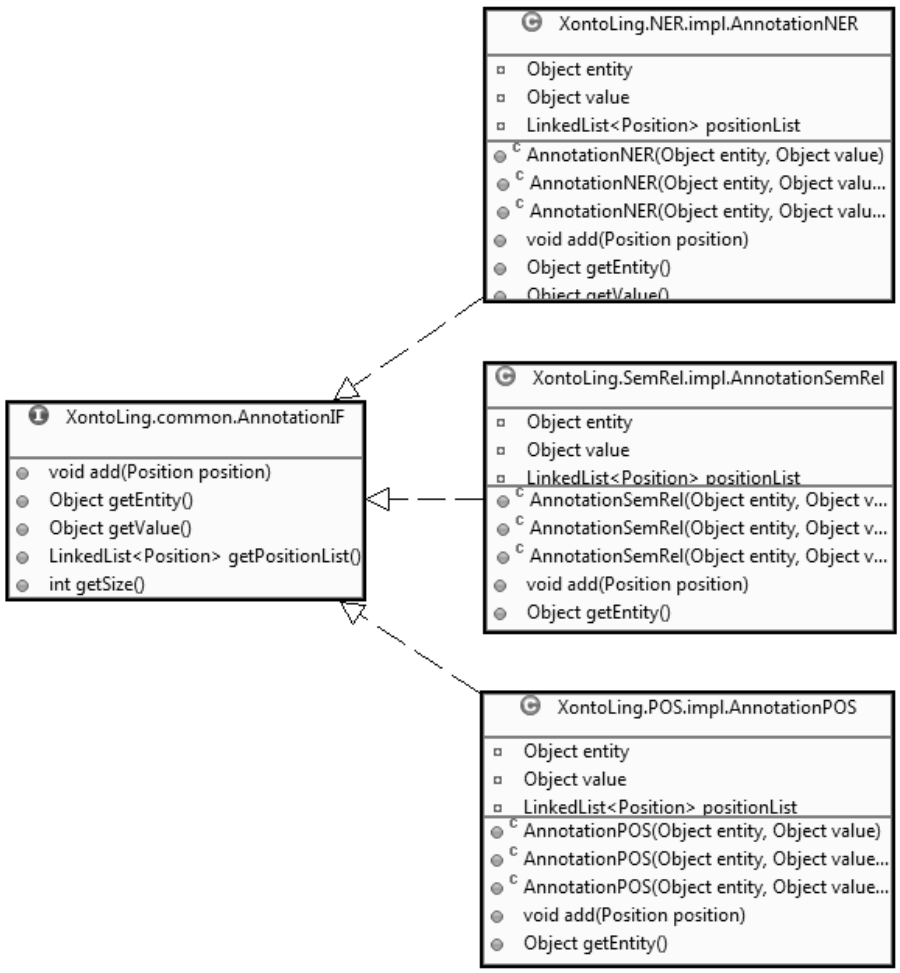

figura 4: class diagram – Annotation objects

L'interfaccia è specializzata per ciascun modulo e per ciascun engine in modo da rispondere esattamente a precise esigenze di utilizzo. In pratica quindi un'unica interfaccia che mappa le linee di comportamento delle classi di annotazioni di ciascuna parte del modello. Ciascun modulo espanderà questa unica classe adattandola a conservare le informazioni estratte dalle proprie specifiche operazioni. Le specializzazioni dei vari moduli implementano la rispettiva interfaccia di appartenenza, che esporta i metodi di estrazione delle annotazioni comuni a tutti i motori che si vorranno aggiungere al framework. Ciascuna parte ha i riferimenti alla specifica tipologia di annotazioni. In figura 5 si riportano le relazioni che intercorrono tra le classi del modulo che si occupa dell'estrazione delle annotazioni delle *Named Entity*. I metodi dell'interfaccia sono stati specializzati secondo le modalità di uso di GATE.

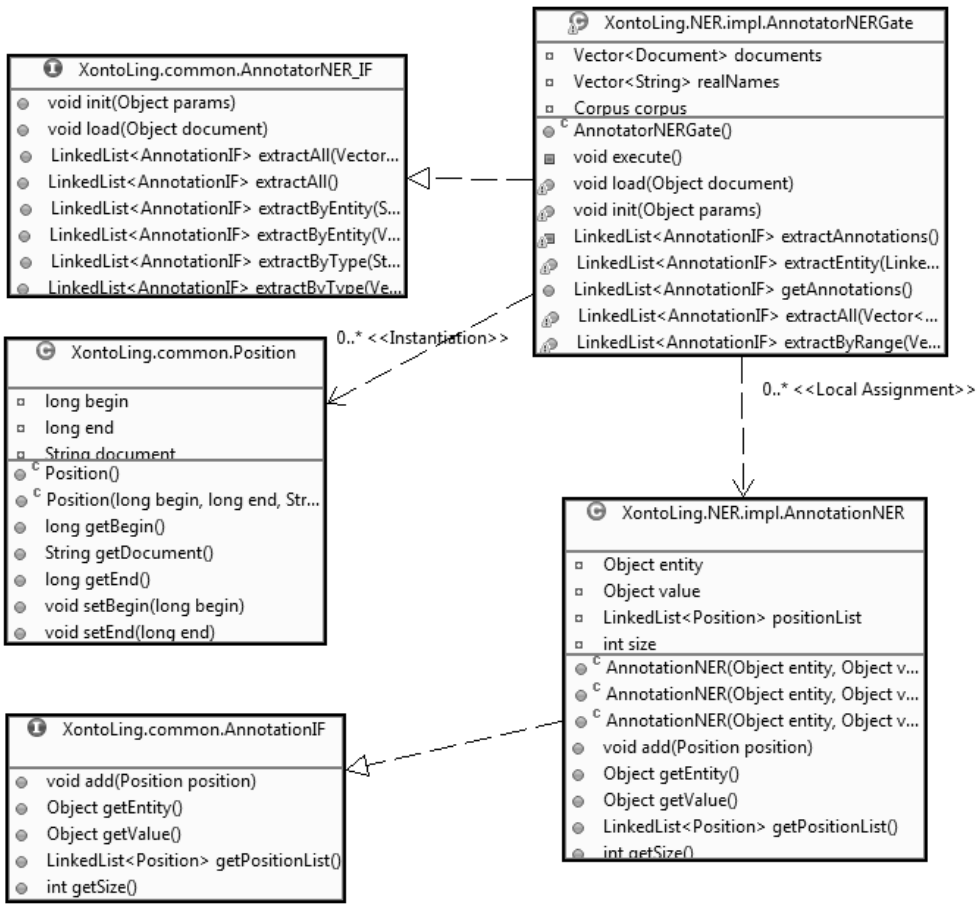

figura 5: class diagram – NER module

L'interfaccia *AnnotatorNER_IF* è implementata da *AnnotatorNERGate* che specifica l'interazione con le API di GATE per l'uso di questo strumento di annotazione. Valutando le esigenze che questo primo modulo richiede, è stato possibile delineare le funzioni generali da esportare per le operazioni di annotazione ed estrazione delle *Named Entity*. E' possibile estrarre le informazioni secondo una stringa che rappresenta l'entità cercata restituendo una lista di annotazioni riferite al corpus fornito in input:

LinkedList<AnnotationIF> extractByEntity(Vector<Object> corpus, String entity); *(1)*

oppure rispetto ad un tipo desiderato:

LinkedList<AnnotationIF> extractByType(Vector<Object> corpus, String type); *(2)*

Un'altra tipica modalità di estrazione è quella che prevede la specifica di un intervallo di posizioni in cui ricercare delle entità:

LinkedList<AnnotationIF> extractByFromToAndEntity (Vector<Object> corpus, long from, long to, String entity); *(3)*

oppure un tipo specifico, sempre all'interno di un dato range di posizioni all'interno dell'insieme dei documenti passati come parametro:

LinkedList<AnnotationIF> extractByFromToAndType (Vector<Object> corpus, long from, long to, String type); *(4)*

E' possibile inoltre annotare il testo secondo entità e tipo:

LinkedList<AnnotationIF> extractByEntityAndType(Vector<Object> corpus, String entity, String type); *(5)*

Ed ovviamente si possono estrarre tutte le informazioni senza alcun filtro:

LinkedList<AnnotationIF> extractAll(Vector<Object> corpus); *(6)*

Tutte queste funzioni restituiscono una lista di annotazioni specifiche per il modulo che si occupa di NER che implementa l'interfaccia *AnnotationIF*. In questo scenario, la classe che si preoccupa di derivare questo comportamento è *AnnotationNER*, come mostrato in figura 4, e nella interazione con la classe *AnnotatorNERGate* in figura 5.
Si può avere inoltre l'esigenza di estrarre soltanto il tipo data una entità:

LinkedList<String> extractTypeByEntity(Vector<Object> corpus, String entity); *(7)*

Queste richieste possono essere invocate anche senza indicare nella chiamata il corpus su cui lavorare, ad esempio con il metodo:

LinkedList<AnnotationIF> extractByFromToAndType(long from, long to, String type); *(8)*

A patto aver già fissato il corpus mediante la routine:

void setCorpus(Vector<Object> corpus); *(9)*

questo perché solitamente i motori di estrazione per NE lavorano con più di un documento alla volta; e quindi si è reso utile fornire un modo per evitare di volta in volta il caricamento dello stesso corpus, leggendolo una sola volta all'inizio della sequenza delle operazioni di annotazione. Questo modulo lavora anche con i *Gazetteers* che vanno quindi fissati anch'essi prima dell'inizio delle operazioni di annotazione:

void setGazetteers(Vector <String> list); *(10)*

E' possibile inizializzare l'applicazione con un certo numero di parametri specifici secondo l'engine specifica tramite la chiamata al metodo:

void init(Object params); *(11)*

Il secondo modulo del framework è il *Part of Speech tagger*. La struttura è del tutto simile a quella appena illustrata. L'interfaccia del *POS module* esporta i metodi da specializzare nelle implementazioni dei nuovi annotatori. Come mostra l'immagine sottostante (figura 6), si è adattato il framework sfruttando *Freeling*. *AnnotationIF* è implementata attraverso *AnnotationPOS* ed utilizzata da *AnnotatorPOSFreeling* che ne specifica il comportamento secondo le API esportate da Freeling. Oltre ai meccanismi di inizializzazione comune a tutti i moduli e di restituzione dei linguaggi che permettono di localizzare i motori di estrazione:

void init(Object params); *(12)*
Vector<String> getLanguages(); *(13)*

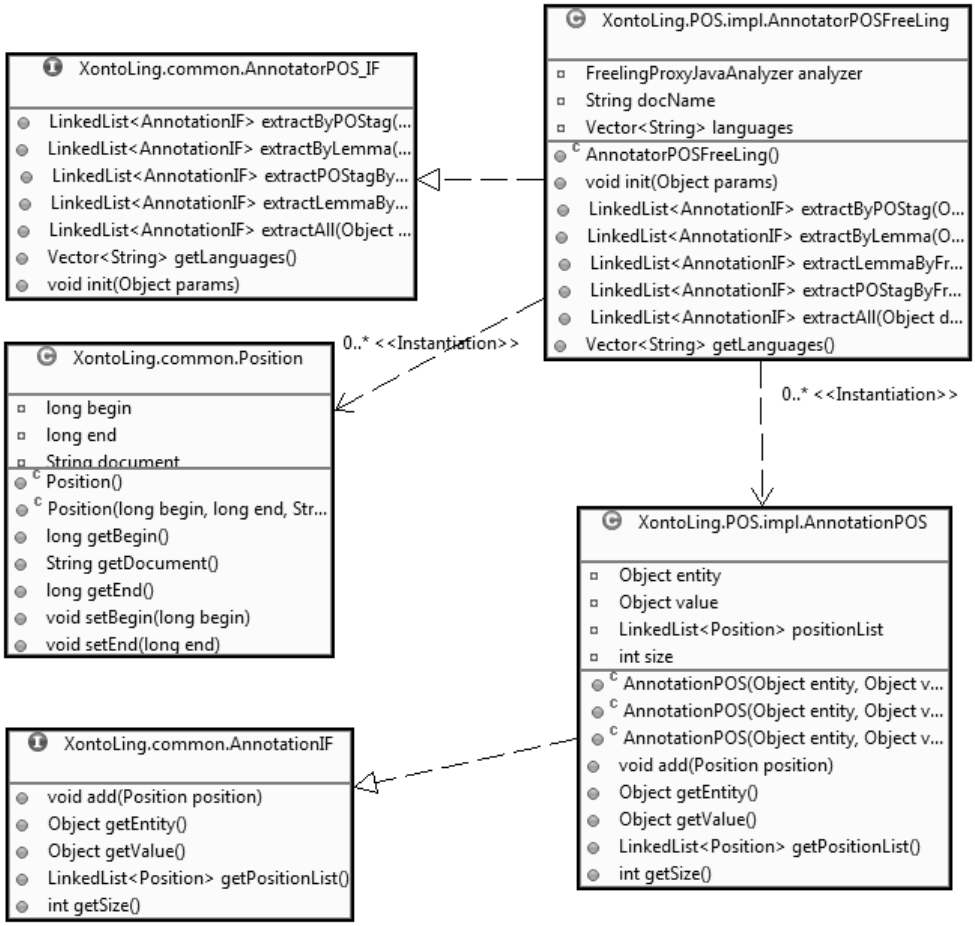

figura 6: class diagram – POS module

Le annotazioni vengono restituite attraverso i metodi:

LinkedList<AnnotationIF> extractByPOStag(Object doc, String POStag); *(14)*

utile se vogliamo estrarre le informazioni da un documento specifico data una etichetta.

LinkedList<AnnotationIF> extractByLemma(Object doc, String lemma); *(15)*

se invece siamo interessati alle informazioni estratte dato il lemma come discriminante della ricerca.

LinkedList<AnnotationIF> extractPOSTagByFromTo(Object document, long from, long to); *(16)*

adatto se vogliamo una lista dei PoS-tag e delle posizioni contenute all'interno di un dato intervallo nel documento specificato. Mentre il corrispondente duale per la restituzione dei lemmi è:

LinkedList<AnnotationIF> extractLemmaByFromTo(Object doc, long from, long to); *(17)*

Per ultimo:

LinkedList<AnnotationIF> extractAll(Object doc); *(18)*

individua tutti i *token* con il rispettivo lemma, *PoS-tag* e posizione nel documento specificato. E' importante notare come le annotazioni restituiscano in alcuni casi il PoS-tag o il lemma corrispondente ad una parola, oppure entrambi, sempre utilizzando gli stessi campi all'interno della classe *AnnotationPOS*; questo grazie ad una spiccata generalità dei tipi adottata per la sua implementazione, che la rende versatile e adatta ai vari utilizzi.

L'interfaccia per il modulo che si occupa di individuare le relazioni semantiche all'interno di un documento è mostrato in figura 7. L'interfaccia *AnnotationIF* è implementa dalla classe *AnnotationSemRel* che ne specifica il comportamento per questo modulo di XONTO-LING. *AnnotatorSemRel*_IF invece è implementata tramite le API esportate da *Wornet* mediante la classe *AnnotatorSemRelWordNet*.

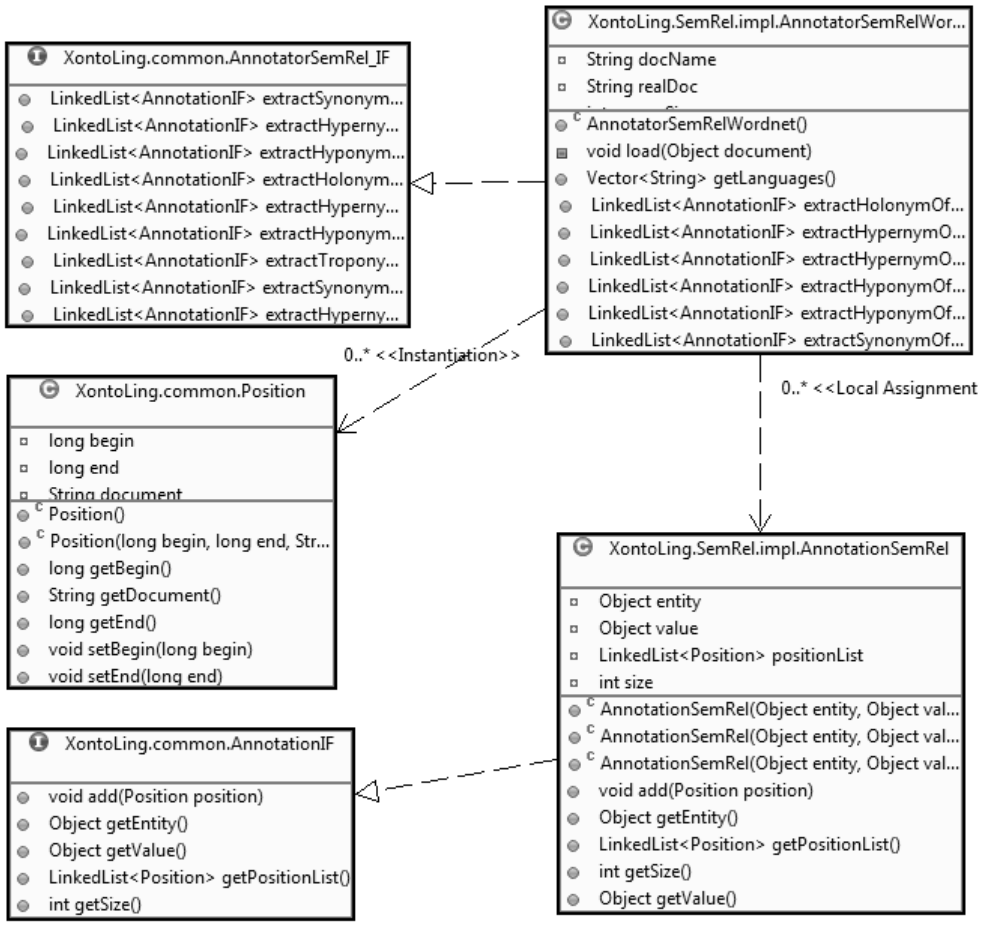

figura 7: class diagram – SemRel module

Per i sostantivi è possibile estrarre i sinonimi:

LinkedList<AnnotationIF> extractSynonymOfWord(Object doc, String word); *(19)*

gli iperonimi:

LinkedList<AnnotationIF> extractIpernymOfWord(Object doc, String word); *(20)*

gli iponimi:

LinkedList<AnnotationIF> extractIponymOfWord(Object doc, String word); *(21)*

e gli olonimi:

LinkedList<AnnotationIF> extractOlonymOfWord(Object doc, String word); *(22)*

Mentre per i verbi possiamo individuare gli iperonimi:

LinkedList<AnnotationIF> extractIpernymOfVerb(Object doc, String verb); *(23)*

gli iponimi:

LinkedList<AnnotationIF> extractIponymOfVerb(Object doc, String verb); *(24)*

ed i troponimi:

LinkedList<AnnotationIF> extractTroponymOfVerb(Object doc, String verb); *(25)*

E' possibile localizzare la ricerca solo in alcune posizioni del testo specificando gli estremi dell'intervallo di ricerca, come per esempio nel caso di seguito riportato di estrazione dei sinonimi di un sostantivo:

LinkedList<AnnotationIF> extractSynonymOfWord(Object doc, String word, long from, long to); *(26)*

Riferimenti Bibbligrafici

1. Alfonseca, E., & Manandhar, S. (2002). An Unsupervised Method for General Named Entity Recognition and Automated Concept Discovery. *In: Proc. International Conference on General WordNet.*

2. Bick, E. (2004). A Named Entity Recognizer for Danish. *In: Proc. Conference on Language Resources and Evaluation.*

3. Black, W. J., Rinaldi, F., & Mowatt, D. (1998). Facile: Description of the NE System used for Muc-7. *In: Proc. Message Understanding Conference.*

4. Bodenreider, O., & Zweigenbaum, P. (2000). Identifying Proper Names in Parallel Medical Terminologies. In *Stud Health Technol Inform* (p. 443-447). Amsterdam: IOS Press.

5. Boutsis, S., Demiros, I., Giouli, V., Liakata, M., Papageorgiou, H., & Piperidis, S. (2000). A System for Recognition of Named Entities in Greek. *In: Proc. International Conference on Natural Language Processing.*

6. Brin, S. (1998). Extracting *Pattern*s and Relations from the World Wide Web. *In: Proc. Conference of Extending Database Technology. Workshop on the Web and Databases.*

7. Carreras, X., Márques, L., & Padró, L. (2003). Named Entity Recognition for Catalan Using Spanish Resources. *In: Proc. Conference of the European Chapter of Association for Computational Linguistic.*

8. Chen, H. H., & Lee, J. C. (1996). Identification and Classification of Proper Nouns in Chinese Texts. *In: Proc. International Conference on Computational Linguistics.*

9. Coates-Stephens, S. (1992). The Analysis and Acquisition of Proper Names for the Understanding of Free Text. San Francisco: Morgan Kaufmann Publishers.

10. Cohen, W. W., & Sarawagi, S. (2004). Exploiting Dictionaries in Named Entity Extraction: Combining Semi-Markov Extraction Processes and Data Integration Methods. *In: Proc. Conference on Knowledge Discovery in Data.*

11. Cucchiarelli, A., & Velardi, P. (2001). Unsupervised Named Entity Recognition Using Syntactic and Semantic Contextual Evidence. In *Computational Linguistics* (p. 1.123-1.131). Cambridge: MIT Press.

12. Cucerzan, S., & Yarowsky, D. (1999). Language Independent Named Entity Recognition Combining Morphological and Contextual Evidence. *In: Proc. Joint Sigdat Conference on Empirical Methods in Natural Language Processing and Very Large Corpora.*

13. Da Silva, J. F., Kozareva, Z., & Lopes, G. P. (2004). Cluster Analysis and Classification of Named Entities. *In: Proc. Conference on Language Resources and Evaluation.*

14. Davies, J., Fensel, D., & van Harmelen, F. (2003). Towards the Semantic Web: Ontology-driven Knowledge Management.

15. Doxygen. (2008). FreeLing Refernce Manual.

16. Doxygen. (2008). Freeling User Manual.

17. Etzioni, O., Cafarella, M., Downey, D., Popescu, A.-M., Shaked, T., Soderland, S., et al. (2005). Unsupervised Named-Entity Extraction from the Web: An Experimental Study. In *Artificial Intelligence* (p. 165.91-134). Essex: Elsevier Science Publishers.

18. Evans, R. (2003). A Framework for Named Entity Recognition in the Open Domain. *In: Proc. Recent Advances in Natural Language Processing.*

19. Fellbaum, C. (1998). *WordNet an Electronic Lexical Database.*

20. Ferro, L., Gerber, L., Mani, I., Sundheim, B., & G., W. (2005). TIDES 2005 Standard for the Annotation of Temporal Expressions. The MITRE Corporation.

21. Fleischman, M. (2001). Automated Subcategorization of Named Entities. *In: Proc. Conference of the European Chapter of Association for Computational Linguistic.*

22. Fleischman, M., & E., H. (2002). Fine Grained Classification of Named Entities. *In Proc. Conference on Computational Linguistics.*

23. Huang, F. (2005). Multilingual Named Entity Extraction and Translation from Text and Speech.

24. Kokkinakis, D. (1998). AVENTINUS, GATE and Swedish Lingware. *In: Proc. of Nordic Computational Linguistics Conference.*

25. Lee, S., & Geunbae Lee, G. (2005). Heuristic Methods for Reducing Errors of Geographic Named Entities Learned by Bootstrapping. *In: Proc. International Joint Conference on Natural Language Processing.*

26. Li, Y., Bontcheva, K., Dowman, M., Roberts, I., & Cunningham, H. (2004). Ontology-Based Information Extraction.

27. May, J., Brunstein, A., Natarajan, P., & Weischedel, R. M. (2003). Surprise! What's in a Cebuano or Hindi Name? In *ACM Transactions on Asian Language Information Processing* (p. 3.169-3.180). New York: ACM Press.

28. Maynard, D., Tablan, V., Ursu, C., Cunningham, H., & Wilks, Y. (2001). Named Entity Recognition from Diverse Text Types. *In: Proc. Recent Advances in Natural Language Processing.*

29. Minkov, E., Wang, R., & Cohen, W. (2005). Extracting Personal Names from Email: Applying Named Entity Recognition to Informal Text. *In: Proc. Human Language Technology and Conference Conference on Empirical Methods in Natural Language Processing.*

30. Narayanaswamy, M., E., R. K., & K., V.-S. (2003). A Biological Named Entity Recognizer. *In: Proc. Pacific Symposium on Biocomputing.*

31. Ohta, T., Tateisi, Y., Kim, J., Mima, H., & Tsujii, J. (2002). The GENIA Corpus: An Annotated Research Abstract Corpus in Molecular Biology Domain. *In: Proc. Human Language Technology Conference.*

32. Palmer, D. D., & Day, D. S. (1997). A Statistical Profile of the Named Entity Task. *In: Proc. ACL Conference for Applied Natural Language Processing.*

33. Petasis, G., Vichot, F., Wolinski, F., Paliouras, G., Karkaletsis, V., & Spyropoulos, C. D. (2001). Using Machine *Learning* to Maintain Rule-based Named-Entity Recognition and Classification Systems. *In: Proc. Conference of Association for Computational Linguistics.*

34. Piskorski, J. (2004). Extraction of Polish Named-Entities. *In: Proc. Conference on Language Resources an Evaluation.*

35. Poibeau, T. (2003). The Multilingual Named Entity Recognition Framework. *In: Proc. Conference on European chapter of the Association for Computational Linguistics.*

36. Poibeau, T., & Kosseim, L. (2001). Proper Name Extraction from Non-Journalistic Texts. *In: Proc. Computational Linguistics in the Netherlands.*

37. Popov, B., Kirilov, A., Maynard, D., & Manov, D. (2004). Creation of reusable components and language resources for Named Entity Recognition in Russian. *In: Proc. Conference on Language Resources and Evaluation.*

38. Rindfleisch, T. C., Tanabe, L., & Weinstein, J. N. (2000). EDGAR: Extraction of Drugs, Genes and Relations from the Biomedical Literature. *In: Proc. Pacific Symposium on Biocomputing.*

39. Saccà, D., Ruffolo, M., & Oro, E. (2009). Ontology-Based Information Extraction from PDF Documents with XONTO.

40. Sekine, S., & Isahara, H. (2000). IREX: IR and IE Evaluation project in Japanese. *In: Proc. Conference on Language Resources and Evaluation.*

41. Sekine, S., & Nobata, C. (2004). Definition, Dictionaries and Tagger for Extended Named Entity Hierarchy. *In: Proc. Conference on Language Resources and Evaluation.*

42. Settles, B. (2004). Biomedical Named Entity Recognition Using Conditional Random Fields and Rich Feature Sets. *n: Proc. Conference on Computational Linguistics. Joint Workshop on Natural Language Processing in Biomedicine and its Applications.*

43. Thielen, C. (1995). An Approach to Proper Name Tagging for German. *In: Proc. Conference of European Chapter of the Association for Computational Linguistics. SIGDAT.*

44. Tjong Kim Sang, E. F. (2002). Introduction to the CoNLL-2002 Shared Task: Language-Independent Named Entity Recognition. *In: Proc. Conference on Natural Language Learning.*

45. Tjong Kim Sang, E. F., & De Meulder, F. (2003). Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition. *In: Proc. Conference on Natural Language Learning.*

46. Tsuruoka, Y., & Tsujii, J. (2003). Boosting Precision and Recall of Dictionary-Based Protein Name Recognition . *In: Proc. Conference of Association for Computational Linguistics. Natural Language Processing in Biomedicine.*

47. Wang, L.-J., Li, W.-C., & Chang, C.-H. (1992). Recognizing Unregistered Names for Mandarin Word Identification. *In: Proc. International Conference on Computational Linguistics.*

48. Whitelaw, C., & Patrick, J. (2003). Evaluating Corpora for Named Entity Recognition Using Character-Level Features. *In: Proc. Australian Conference on Artificial Intelligence.*

49. Witten, I. H., Bray, Z., Mahoui, M., & J., T. W. (1999). Using Language Models for Generic Entity Extraction. *In: Proc. International Conference on Machine Learning. Text Mining.*

50. Yu, S., S., B., & Wu, P. (1998). Description of the Kent Ridge Digital Labs System Used for MUC-7. *In: Proc. Message Understanding Conference.*

51. Zhu, J., Uren, V., & Motta, E. (2005). ESpotter: Adaptive Named Entity Recognition for Web Browsing. *In: Proc. Conference Professional Knowledge Management. Intelligent IT Tools for Knowledge Management Systems.*