Alma Mater Studiorum · Università di Bologna

SCUOLA DI SCIENZE
Corso di Laurea in Informatica

Generazione di dataset RDF su articoli scientifici e affiliazioni: un approccio modulare basato su DBPedia

Relatore:
Chiar.mo Dott.
Angelo Di Iorio

Presentata da:
Roberto Rossiello

Sessione I
Anno Accademico 2014 / 2015

Dedicato alla mia
famiglia ...

Introduzione

L'obiettivo di questa tesi è creare un tool che estrae informazioni sulle affiliazioni da articoli riguardanti pubblicazioni scientifiche, generando in output un dataset RDF contenente tali informazioni.
Al giorno d'oggi la più imponente fonte di informazioni è rappresentata dal Wolrd Wide Web.
Se ci si fermasse a pensare come questa tecnologia esista da nemmeno una generazione, sarebbe evidente come la sua influenza nel nostro quotidiano sia di incredibile entità e come le evoluzioni che si sono susseguite dalla creazione della prima pagina Web, ad opera di sir Tim Barners Lee il 6 agosto del 1991, siano stati continui.
Al giorno d'oggi ogni tipo di informazione è nel palmo delle mani di ognuno di noi, ad un istante di distanza, e i risultati che in questo brevissimo periodo sono stati conseguiti sono senza dubbio sorprendenti. Ma la natura dell'uomo di scienza deve essere quella di non fermarsi mai a contemplare i propri successi ma a guardare costantemente ad essi come punti di partenza per nuove ricerche ed innovazioni.
Ad impersonificare questa definizione di ricercatore è ancora una volta sir Tim Barners Lee, che nel 2001[1] ha proposto una visione nuova del Web, in cui le già enormi quantità di informazioni attualmente disponibili potessero essere collegate tra loro in maniera tale da renderne più efficacie e coerente la fruizione.
Nasce quindi il termine "Web Semantico", che identifica un modello in cui ad ogni tipo di documento presente sul Web (pagine Html, pdf, immagini,

ecc...) siano associati metadati che aiutino a meglio definire il tipo di informazioni associate ad esso, rendendo quindi possibili collegamenti trasversali basati sul contesto semantico delle risorse.
Un'applicazione di particolare interesse di questo modello riguarda sicuramente il "Semantic Publishing"[6], il cui obiettivo è quello di associare ad ogni pubblicazione di stampo scientifico una serie di metadati che forniscano informazioni sul contesto degli argomenti trattati, sugli autori di tali documenti, sulle loro affiliazioni, oltre ad eventuali citazioni ad altri articoli precedentemente pubblicati.
L'accesso a questo tipo di dati permetterebbe di creare complesse reti di collegamenti tra le varie pubblicazioni, migliorando sensibilmente la fruizione di questo tipo di risorse.
Per arrivare a realizzare simili obiettivi, si renderà necessario operare sostanziali modifiche alla modalità in cui le informazioni vengono mantenute sul Web facendo nascere l'esigenza di nuovi linguaggi e tecnologie capaci di esprimere la natura semantica dei collegamenti tra le risorse.
Attualmente siamo ancora agli inizi, ma questo non vuol dire che non ci si stia muovendo nella giusta direzione; negli ultimi anni sono stati introdotti linguaggi come RDF[1] (Resource Description Framework), uno standard utilizzato nella descrizione di risorse e modellazioni di informazioni, ed SPARQL[2] (SPARQL Protocol and RDF Query Language), un linguaggio preposto alla fruizione di risorse espresse in RDF.
Molti progetti sono stati realizzati sulla base di questi presupposti primo fra tutti va menzionato Dbpedia[3], in cui una comunità formata da informatici, ricercatori e semplici appasionati ha realizzato un dataset RDF interrogabile tramite query SPARQL contenente una moltitudine di informazioni sulla base di quelle contenute nella popolare enciclopedia online Wikipedia[4].
Come detto in apertura di questa introduzione il lavoro di questa tesi è in-

[1] http://www.w3.org/RDF/
[2] http://www.w3.org/TR/rdf-sparql-query/
[3] http://wiki.dbpedia.org/
[4] https://www.wikipedia.org/

centrato nel recupero e il trattamento di informazioni sulle affilizioni e tali informazioni sono importanti perchè costituiscono la maniera per valutare i risultati e i contributi che ogni persona, ente o organizzazione produce nell'ambito della ricerca scientifica.
Il rendimento di un'organizzazione è rappresentato dall'insieme dei contributi forniti dai ricercatori ad essa associati e dall'impatto che essi producono nel panorama scientifico di loro competenza.
Tale impatto viene misurato in rapporto alle pubblicazioni che ognuno di questi ricercatori produce, durante il loro rapporto di affiliazione presso una data organizzazione, sui journal che si occupano di rendere disponibili i risultati esposti nei vari articoli alla critica del resto della comunità scientifica.
Appare evidente come, per stimare il rendimento di un ente, ci sia la necessità di recuperare le affiliazioni che ogni autore dichiara di avere al momento della stesura delle proprie pubblicazioni.
Tali informazioni sono largamente presenti sul web ma, non sempre in un formato compatibile con gli standard proposti dal Semantic Publishing.
Esistono, infatti, diversi dataset RDF contenenti moltissime informazioni riguardanti pubblicazioni di stampo scientifico come DLBP++ di DBLP, il dataset di ACM Pubblications, Sematic Web Dog Food, il dataset di Semantic Web journal e ancora altri che verranno riportati nel prossimo capitolo.
Ognuno di questi dataset presenta informazioni parziali o addirittura assenti per quanto riguarda le affiliazioni, e quando queste informazioni sono effettivamente presenti, l'affidabilità di questi dati non risulta sempre assicurata a causa dei diversi problemi nell'estrazione ed analisi di questo particolare tipo di informazioni che verranno analizzati in dettaglio nel capitolo 2 di questa tesi.
Anche quando non espressi in RDF, i dati sulle affiliazioni sono comunque presenti nel testo degli articoli mantenuti nei repository dei journal o trascrizioni di conferenze dove sono stati pubblicati. Spesso infatti l'unica maniera per risalire all'affiliazione di un dato autore consiste nell'analizzare i repository dei journals in cui i suoi articoli sono stati pubblicati e cercare nel testo

dell'articolo stesso le informazioni necessarie.
Ovviamente questo non è sempre possibile vista l'enorme quantità di tempo che sarebbe richiesta per completare un'operazione del genere, facendo quindi nascere la necessità di una maniera automatica per estrarre tali informazioni e renderle disponibili in un forma facilmente fruibile, come la rappresentazione di tali dati in RDF.
Per rendere possibile ciò, un ostacolo si pone da tempo: l'assenza di una maniera standardizzata per identificare e correttamente riportare i nominativi delle suddette organizzazioni, in modo da potervi facilmente risalire per analizzarne il rendimento.
Non vi è garanzia che due o più ricercatori, anche facenti parte dello stesso dipartimento, riportino nella stesso modo la propria affiliazione nelle loro pubblicazioni, il che rende estremamente difficile riuscirne a tracciare carriere e contributi scientifici.
Diversi progetti sono in atto per raccogliere informazioni sulle affiliazioni e renderle disponibili in datasets RDF in modo da poter collegare semanticamente le entità coinvolte nella pubblicazione di ricerche scientifiche in accordo con le specifiche descritte nel precedente capitolo relative al Semantic Publishing.
Il lavoro di questa tesi fa parte di "Semantic Lancet"[12], progetto dell'Università di Bologna il cui proposito è quello di rendere disponibile un dataset RDF contenente diverse informazioni semantiche, riguardanti pubblicazioni scientifiche, oltre ad una rete di servizi sulla base di queste informazioni.
Nello specifico, il contributo fornito da questa tesi è quello di rendere disponibili informazioni riguardanti le affiliazioni di autori di articoli scientifici, generando un dataset RDF interrogabile tramite query SPARQL, partendo da documenti scritti in XML facenti parte del journal "Web Semantics: Science, Services and Agents on the World Wide Web" di Elsevier[5].
Gli ostacoli incontrati durante lo sviluppo del tool nominato "AffiliationExtractor" sono stati diversi, spaziando dal problema dell'estrazione dei dati

[5]http://www.elsevier.com/

riguardanti affiliazioni, dagli articoli costituenti il dataset di partenza, a quello dell'analisi delle informazioni estratte per riconoscerne il significato, alla difficoltà posta dalla modellazione di questi dati in una maniera che li renda facilmente fruibili.
Questi problemi sono stati affrontati proponendo un approccio modulare allo sviluppo, in cui ogni modulo si occupa di risolvere una diversa difficoltà, garantendo la possibilità di migliorare il tool in futuro o di riutilizzare un particolare modulo in altri progetti, vista l'indipendenza tra gli stessi.
Un ruolo di cardinale importanza è stato ricoperto proprio da Dbpedia, che rappresenta il modo principale per identificare i diversi significati delle varie parti del testo riguardante le affiliazioni, garantendo la possibilità di accedere ad informazioni già contestualizzate semanticamente.
Sono state inoltre definiti alcuni metodi per trattare i casi che esulano da quelli risolvibili tramite Dbpedia, basati su euristiche construite sui diversi modi in cui un'affiliazione può essere espressa da un autore su un proprio paper.
Infine il dataset RDF generato dalle esecuzioni del tool, oltre a rendere disponibili all'esterno le informazioni prodotte, viene riutilizzato per gestire più rapidametente i casi già risolti in iterazioni precedenti.
Questo elaborato fornirà, nel primo capitolo, informazioni riguardanti il contesto scientifico in cui il lavoro svolto si colloca, fornendo esempi di progetti correlati, sottolineando le soluzioni già esistenti ai problemi esposti.
Il secondo capitolo si soffermerà, in dettaglio, sull'esposizione delle diverse problematiche a cui si è provato a fornire una soluzione, proponendo un numero di esempi per meglio definire detti problemi.
Nel terzo capitolo si fornirá una spiegazione ad alto livello del funzionamento del tool, spiegandone i passi che vengono compiuti durante un'esecuzione di AffiliationExtractor, proponendo una panoramica di tutte le tecnologie impiegate nello sviluppo, dando risalto alle motivazioni delle scelte effettuate.
Il quarto capitolo sarà dedicato ai dettagli implementativi, in cui verrà fornita una spiegazione a basso livello del tool, concentrando l'attenzione sui

diversi moduli che lo compongono.
Si prosegue poi con il capitolo sull'evaluation del lavoro svolto, in cui verranno dati i numeri di precision e recall raccolti durante i test di valutazione del rendimento del tool, soffermandosi sul metodo scelto per tale valutazione e sull'analisi di punti di forza e limiti attuali del tool stesso.
L'ultimo capitolo riguarderà le conclusioni in cui si fornirà uno quadro finale del lavoro svolto, esponendo spunti per sviluppi futuri basati sulle analisi fornite durante l'evaluation.

Indice

Elenco delle figure

Elenco delle tabelle

Capitolo 1

Semantic Publishing e gestione delle affiliazioni

In questo capitolo si fornirà il contesto scientifico in cui gli argomenti di questa tesi si collocano, parlando dapprima del Semantic Publishing come mezzo per fornire una maniera più versatile per esprimere i metadati riguardanti pubblicazioni scientifiche nel Web, consentendo di migliorare i collegamenti tra più risorse correlate semanticamente. Verranno infine menzionati progetti correlati, sia per quanto riguarda la disambiguazione delle informazioni sulle affiliazioni, sia riguardanti impieghi simili del database semantico "Dbpedia" per analizzare linguaggi naturali.

1.1 Semantic Publishing

Le versioni online di articoli scientifici sono generalmente presenti in rete nella forma di documenti Html/XML o come documenti PDF statici, con il web usato esclusivamente come mezzo di distribuzione per gli articoli stessi. La natura statica di formati come il PDF, nonostante l'innegabile familiarità e facilità di lettura per gli utenti umani, costituisce un limite che è antitetico con lo spirito del Web, mancando di possibilità di interazione da parte degli utenti ed essendo complicato per le macchine da analizzare.

Queste necessità hanno dato vita al moviemento denominato "Semantic Publishing"[6], il cui obiettivo è proporre l'aggiunta di metadati riguardanti le diverse informazioni contenute e correlate alle pubblicazioni, per fornire un'infrastruttura che colleghi appunto, semanticamente, le risorse nel Web.
Informazioni sui contenuti di una pubblicazione, sui suoi autori e sulle sue affiliazioni sono estremamente preziose per "linkare" risorse diverse ma semanticamente correlate, rendendo la fruizione dei contenuti decisamente più intuitiva e pertinente.
I principali metodi per fornire questo genere di informazioni sono sicuramente rappresentate dai linguaggi XML ed RDF (Resource Description Framework) ma, come sottolineato nella pubblicazione "The Semantic Web: The Roles of XML and RDF"[5], XML a lungo andare risulterà inefficace per descrivere pienamente i dati necessari per realizzare le complesse connessioni di risorse previste nel Sematic Web e Semantic Publishing.
XML può fa riferimento solo alla struttura del documento di cui è linguaggio, mentre RDF ha nella rappresentazione di dati tramite ontologie estendibili a piacimento, la carta vincente per adattarsi al meglio ad ogni tipo di informazione che si avrà bisogno di rappresentare.
Molti sono i progetti attivi per realizzare dataset semantici basati su RDF per esprimere metadati relativi a pubblicazioni di stampo scientifico e la trattazione di questa tesi fa parte del progetto "Sematic Lancet" dell'Università di Bologna.
I datasets più completi e rilevanti sono stati creati per il dominio della biomedica. Uno dei primi fu il "NPG Linked Data Platform 2".
Esso includeva dati relativi ad articoli pubblicati dalla rivista "Nature"[1] a partire dal 1845 e può contare circa 400 milioni di triple RDF, strutturate seguento diverse ontologie come Dublin Core[2], FOAF [3], PRISM[4] e BiBO[5].

[1] http://www.nature.com/
[2] https://www.cs.umd.edu/projects/plus/SHOE/onts/dublin.html
[3] http://xmlns.com/foaf/spec/
[4] http://www.prismstandard.org/
[5] http://bibliontology.com/

Le informazioni sono di altissima qualità e coprono tutti gli aspetti esposti nel movimento del Semantic Publishing, ma la qualità dei dati non è costante per tutti gli articoli considerati; per esempio solo parte degli abstracts sono disponibili e i criteri di classificazioni non sono sempre gli stessi per ogni paper considerato, con alcuni articoli che non sono classificati affatto.
La rete di citazioni è coperta solo parzialmente in questo dataset mentre invece rappresenta una parte di estrema importanza nel progetto "JISC OpenCitation" che mette a disposizione gratutamente informazioni riguardanti articoli pubblicati nel "Open Access PubMed Central"[6].
Questo dataset usa primariamente le ontologie SPAR[7], PRISM e Dublin Core. Particolarmente interessante è il suo utilizzo dell'ontologia PRO[8] per descrivere i ruoli e per modellare le informazioni sugli autori.
Il dataset contiene anche divesi abstracts e informazioni sulle affiliazioni, ma tali informazioni non sono sempre disponibili per ogni paper.
In generali, questo dataset e molto ben strutturato e i dati sono di ottima qualità ma al momento non è attivo.
Un tipo di ricerca molto simile a quella di Semantic Lancet è portata avanti da BioTea. L'obiettivo di questo progetto è di rendere disponibili in RDF la letteratura relativa alla biomedica, attingendo ancora una volta ad articoli presenti nei repository di PubMed Central.
Il dataset di BioTea descive circa 270 mila articoli, pubblicati in 2400 journals, utilizzando divesi modelli ontoligici (BiBO, DublinCore, FOAF). Il fatto che il dataset sia generato automaticamente fornendo in input documenti in formato XML rende questo progetto molto pertinente al caso di studio portato avanti in questa tesi.
Diversi datasets sono stati resi disponibili direttamente da editori di articoli riguartandi l'Informatica.
Uno di questi è rappresentato da DBLP++[9], che rende disponibili informa-

[6]http://www.ncbi.nlm.nih.gov/pmc/tools/openftlist/
[7]http://purl.org/spar/
[8]http://purl.org/spar/pro
[9]http://dblp.l3s.de/dblp++.php

zioni in RDF corrispondenti ad articoli in DBLP.
Questo dataset usa l'ontologia SWRC[10] e il suo obiettivo originario era quello di tenere traccia della produzione bibliografica dei ricercatori; questo ha comportato a non considerare informazioni su affliazioni o citazioni.
Sono invece presenti alcune informazioni riguardanti abstracts e parole chiave riguardanti il contenuto delle pubblicazioni.
Gli stessi problemi sono apprezzabili, addirittura in maniera più evidente, nel datasets di "ACM Pubblications"[11]. A parte per le informazioni bibliografiche che sono molto complete, il dataset contiene solo pochi dettagli sugli autori, citazioni e poco altro.
È composto da circa 12 milioni di triple RDF ma non è stato aggiornato dal 2006.
Al contrario, un dataset costantemente sottoposto ad aggiornamenti, è rappresentato da "Semantic Web Dog Food", il quale contiene informazioni prese da conferenze sul Semantic Web.
Ad oggi può contare su circa 250 mila triple riguardanti circa 5 mila articoli e 10 mila autori.
Questo dataset è esplicitamente preposto a contenere informazioni rigurdanti eventi scientifici e record bibliografici riferiti agli articoli, includendo dati riguardanti autori e le loro affiliazioni.
Il corpo del dataset è composto da differenti dataset, creati e aggiornati separatamente per ogni conferenza comportando un'ovvia oscillazione nella qualità dei dati presenti a seconda delle conference a cui si riferiscono.
La situazione è più semplice da controllare su dataset di dimensioni più modeste, associati a singoli journal.
Un esempio è rappresentato dal dataset "Semantic Web Journal"[12], in cui sono presenti record bibliografici e dati molto ricchi riguardanti le pubblicazioni dell'omonimo journal.
Esso contiene circa 21 mila triple, strutturate secondo un'ontologia proprieta-

[10] http://ontoware.org/swrc/
[11] http://acm.rkbexplorer.com/
[12] http://semantic-web-journal.com:3030/

ria, che risponde al nome di "Semantic Web journal ontology"[13], e ontologie già nominate quali FOAF, Dublin Core e Bibo.
La peculiarità di questo dataset, che è direttamente derivata dal processo di review pubblico del journal da cui prende il nome, è di fornire informazioni riguardanti il processo di review degli articoli, includendone gli step intemedi, informazioni circa i recensori e in in generale sulla review stessa.
Ognuno dei dataset RDF menzionati contiene informazioni eterogenee per qualità e quantità e, nello specifico per quanto riguarda le informazioni sulle affiliazioni, le informazioni sono spesso non presenti e, quando presenti, sono caratterizzate da alcuni problemi che verranno esposti in dettaglio nel successivo capitolo.

1.2 Approcci alla disambiguazione di Affiliazioni

Il problema della disambiguazione dei dati sulle affiliazioni è stato affrontato molte volte in passato anche con risultati molto promettenti.
Nel 2011[4] Yong Jiang, Hai-Tao Zheng, Xinmin Wang, Binggan Lu e Kaihua Wu, ricercatori presso la Tsinghua University di Shenzhen in Cina, hanno proposto un metodo basato su NCD (Normalized Compression Distance), concentrandosi su algoritmi di compressione per disambiguare abbreviazioni presenti nelle trascrizioni dei nominativi delle affiliazioni.
Nel 2012[3] Fernanda Morillo, Javier Aparicio, Borja Gonzalez-Albo e Luz Moreno hanno realizzato un sistema per riconoscere le affiliazioni degli autori di documenti recuperati dal database di Web of Science, che opera in due fasi: la prima "supervised", in cui insiemi di indirizzi unici sono stati selezionati per estrarre keywords per istruire il sistema e la seconda "semi-supervised" in cui tali keywords sono state usate per disambiguare i restanti indirizzi.
Nel 2013[2] Shuiqing Huang, Bo Yang, Sulan Yan e Ronald Rousseau han-

[13]http://semantic-web-journal.com/ontology

no presentato un metodo che fa uso solo di euristiche basate su statistiche calcolate tramite una serie di assunzioni sulla natura delle traduzioni e abbreviazioni che sono possibili nriportando i dati sulle affiliazioni, classificando quelle recuperate a seconda del campo in cui tali organizzazioni operano.
Tutte le ricerche menzionate sottolineano l'importanza della questione sollevata, ma anche che nessuno dei metodi proposti fornisce una soluzione definitiva al problema che rimane aperto nella comunità scientifica.

1.3 Dbpedia per analisi di Linguaggio naturale

L'uso di Dbpedia per analizzare linguaggi naturali non è una novità di questa tesi e, sebbene la natura dei lavori considerati in questa sezione non sempre sia direttamente correlata con il problema della disambiguazione di informazioni relative ad affiliazioni, è importante considerarli, per fornire le opportunità e i limiti che una soluzione del genere comporta.
Il primo progetto che va menzionato riguarda "NERD"[11], introdotto da Giuseppe Rizzo e Raphael Troncy, che affronta tematiche molto simili per quanto riguarda la disambiguazine delle affiliazioni tramite utilizzo di database semantici, tra cui anche Dbpedia. Il punto di interesse di NERD è fornire un framework interrogabile tramite APIs REST, che combini le risorse di diversi dataset in modo da rendere fruibile una maggiore quantità di informazioni attraverso un canale unico.
Il problema del collegamento di risorse correlate semanticamente tramite i principi del Semantic Publishing è stato affrontato anche dall'emittente televisiva britannica BBC[8], tramite una combinazione nell'uso di Dbpedia come fornitore di "Linked Data" generici e "MusicBrainz"[14], un'enciclopedia semantica con informazioni derivate dal mondo della musica, per ottenere dati relativi ad autori e le loro produzioni.

[14]https://musicbrainz.org/

In questo caso Dbpedia viene usato per collegare semanticamente le risorse presenti su MusicBrainz a contesti più ampi.

Capitolo 2

Difficoltà nell'estrazione e modellazione dei dati su Affiliazioni

Sebbene quindi dati relativi alle affiliazioni siano largamente presenti in diverse forme nel web, vi sono ancora evidenti problemi nella rappresentazione degli stessi, il che compromette l'affidabilità di detti dati.

2.1 Rappresentazione dei dati

In questa sezione si analizzeranno i problemi riguardanti la rappresentazione dei dati sulle affiliazioni, proponendo esempi riguardanti alcuni tra i maggiori dataset RDF che contengano informazioni relative a pubblicazioni scientifiche.

2.1.1 Ambiguità nei Dataset

Un problema costante riguardante il corretto riconoscimento di affiliazioni riguarda la disambiguazione di riferimenti all'apparenza diversi, ma che in

realtà identificano la stessa entità.
I dataset RDF utilizzano come identificativi delle proprie risorse gli URI (Uniform Resource Identifier); l'URI individua univocamente la risorsa associata ed è possibile collegare le diverse istanze presenti nei dataset (tutte identificate da URI) tramite le proprietà descritte nelle ontologie implementate nel dataset di riferimento.

<http://data.semanticweb.org/organization/universita-di-bologna>

Tabella 2.1: esempio di URI dell'Università di Bologna sul dataset RDF Semantic Web Dog Food

La mancata disambiguazione delle affiliazioni risulta problematica in quanto, all'interno di un dataset RDF, una stessa risorsa potrebbe essere serializzata in maniere diverse, creando quindi più URI ridondanti e rendendo confusi e meno affidabili i dati riguardanti le organizzazioni.
Si consideri, per esempio, uno dei più completi dataset semantici ad oggi attivi, rappresentato dal progetto Semantic Web Dog Food, il quale contiene dati riguardanti conferenze sul Semantic Web.
La quantità di informazioni contenute in tale dataset è di sicuro interessante ma la qualità di tali informazioni non è sempre alta[12].
L'Università di Bologna su questo dataset è presente sotto diversi URI (tabella 2.2) Nelle figure 2.3 e 3.3 vediamo come sia stato compiuto un passo per

<http://data.semanticweb.org/organization/universita-di-bologna>
<http://data.semanticweb.org/organization/university-of-bologna>
<http://data.semanticweb.org/organization/bologna-university>

Tabella 2.2: esempio dei diversi URI dell'Università di Bologna sul dataset RDF Semantic Web Dog Food

Description of http://data.semanticweb.org/organization/universita-di-bologna:

property	hasValue	isValueOf
rdf:type	foaf:Organization	-
rdfs:label	"Univ. of Bologna"	-
rdfs:label	"Università di Bologna"	-
rdfs:seeAlso	<http://ontoworld.org/wiki/Special:ExportRDF/Universita_di_Bologna>	-
owl:sameAs	<http://ontoworld.org/wiki/Special:URIResolver/Universita_di_Bologna>	-
foaf:based_near	_:node19kl74jcgx2531	-
foaf:homepage	<http://www.eng.unibo.it/>	-
foaf:homepage	<http://www.unibo.it/>	-
foaf:member	<http://data.semanticweb.org/person/alessandra-toninelli>	-
foaf:member	<http://data.semanticweb.org/person/giorgia-lodi>	-
foaf:member	<http://data.semanticweb.org/person/oezalp-babaoglu>	-
foaf:member	<http://data.semanticweb.org/person/paolo-ciancarini>	-
foaf:member	<http://data.semanticweb.org/person/rebecca-montanari>	-
foaf:name	"Univ. of Bologna"	-
foaf:name	"Università di Bologna"	-
foaf:page	<http://data.semanticweb.org/organization/universita-di-bologna/html>	-
swrc:affiliation	-	<http://data.semanticweb.org/person/alessandra-toninelli>
swrc:affiliation	-	<http://data.semanticweb.org/person/giorgia-lodi>
swrc:affiliation	-	<http://data.semanticweb.org/person/oezalp-babaoglu>
swrc:affiliation	-	<http://data.semanticweb.org/person/paolo-ciancarini>
swrc:affiliation	-	<http://data.semanticweb.org/person/rebecca-montanari>

Figura 2.1: esempio proprietà associate ad Università di Bologna

accomunare più URI che si riferiscono alla stessa risorsa, cercando di legare detti URI tramite la proprietà "sameAs", la quale è definita nell'ontologia del modello di OWL[1] e serve appunto a dichiarare come risorse seriliazzate sotto due diversi URI si riferiscano in realtà alla stessa entità.
Ne traspare che University of Bologna e Università di Bologna sono la stessa cosa, ma questo non vale per ogni riferimento a questa organizzazione.
Si può infatti notare, sia nelle precedenti figure che nella 3.7 che non vi è nessuna property che leghi Bologna University alle precedenti istanze; questo per una macchina sta a significare che quest'ultima risorsa sia effettivamente diversa dalle altre quando all'occhio umano è chiaro il contrario.
Questo problema è risultato della mancanza di uno standard per riportare le proprie affiliazioni da parte dei ricercatori nelle loro pubblicazioni, facendo si che tali informazioni siano difficili da collegare tra loro, considerando le possibilità diverse con cui un affiliazione può essere espressa.
Come risultato di questo problema i deversi ricercatori presso la stessa organizzazione saranno considerati come affiliati ad enti diversi generando ingenti difficoltà in fase di valutazione dei contributi scientifici che le organizzazione forniscono nel corso degli anni.

[1] http://www.w3.org/2001/sw/wiki/OWL

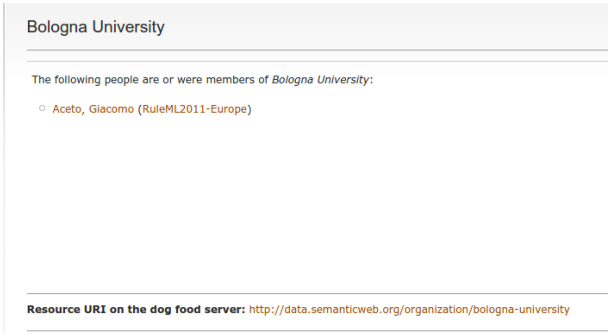

Figura 2.2: Semantic Web Dog Food: Bologna University

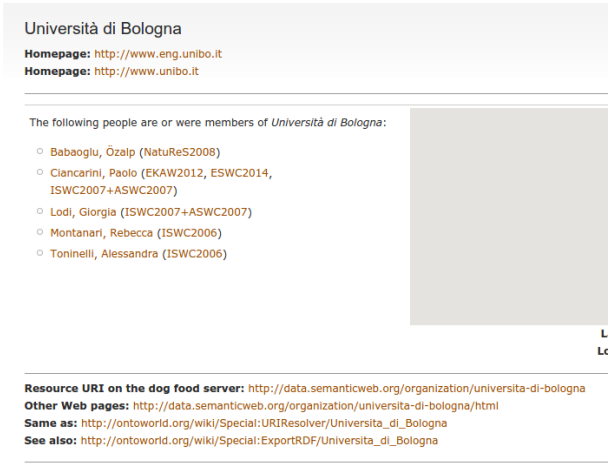

Figura 2.3: Semantic Web Dog Food: Università di Bologna

2.1.2 Mancanza di dati sulle Affiliazioni

A volte i dati sulle affiliazioni sono semplicemente non presenti in RDF come nel caso del dataset DBLP++ del progetto di DBLP[12], il quale contiene rilevanti informazioni su autori e loro pubblicazioni ma non dati riguardanti le loro affiliazioni.
Analogo è il caso del dataset di ACM pubblications[12] il quale non riporta

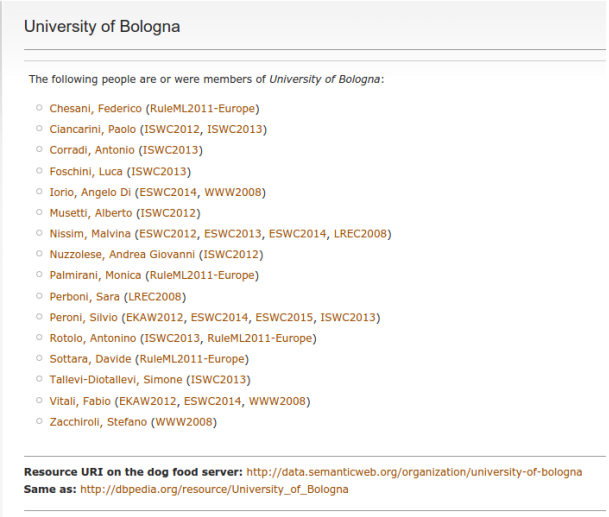

Figura 2.4: Semantic Web Dog Food: University of Bologna

che informazioni su autori e citazioni dei loro lavori.
In questi dataset le informazioni sulle affiliazioni sono presenti solo nel testo degli articoli contenuti nei loro repository.

2.1.3 Mancanza di riferimenti al tempo

Un'altra grave mancanza di tutti i dataset descritti riguarda l'inclusione di informazioni relative al periodo di tempo durante il quale un ricercatore è stato affiliato presso una particolare organizzazione.
Prendendo ancora spunto dal dataset di Semantic Web Dogfood si consideri, la situazione in figura 3.5.
Possiamo notare come una propretà denominata "swrc:affiliation" sia stata introdotta nell'ontologia implementata nel dataset per indicare l'associazione di affiliazione tra ricercatore (dominio della proprietà) e l'organizzazione a cui è affiliato (range della proprietà).
In questo tipo di associazione, però, non sono presenti riferimenti al periodo in cui il ricercatore Paolo Ciancarini (URI: http://data.semanticweb.org/person/paolo-ciancarini) sia stato rilevato come affiliato all'Università di Bologna.

La proprietà descritta è quindi statica e associa indefinitamente una persona all'organizzazione presso cui era collaboratore al momento della pubblicazione su cui tale informazione è stata fornita; ne traspare che un'associazione definita in questo modo non rispecchia la realtà, in quanto, nel caso in cui un ricercatore cambi affiliazione, questo non sarebbe esprimibile.
Tale dato sarebbe prezioso per poter fornire statistiche molto più precise riguardanti sia il rendimento di una particolare organizzazione che per tenere traccia della carriera dei diversi autori impegnati nella ricerca scientifica.

2.2 Estrazione dei dati

La difficoltà della corretta rappresentazione dei dati riguardanti le affiliazioni è direttamente correlata al problema dell'estrazione di tali informazioni dalle risorse in cui sono presenti, data la diversità con cui gli autori di pubblicazioni scientifiche riportano le proprie affiliazioni.
Come detto nel precedente capitolo i formati standard, in cui le pubblicazioni scientifiche sono realizzate, sono Html/XML e RDF. Le difficoltà nell'estrarre informazioni da tali formati sono diverse a seconda di quello considerato. Esistono alcuni vocabolari in rete, definiti dai diversi enti, che riservano di specifici tag XML per riportare i diversi tipi di informazioni, risultando quindi, generalmente più semplici da analizzare rispetto ad un PDF che necessita di un passo intermedio per essere convertito in un formato più facilmente leggibile (come appunto XML).
Una volta estratti i dati che si ritengono rappresentare le informazioni relative alle affiliazioni, si presentano i problemi sull'analisi di detti dati; tali difficoltà derivano dalle maniere diverse in cui un autore può esprimere la propria affiliazione a seconda della lingua, dettaglio di informazioni fornite e posizioni in cui tali informazioni vengono riportate.
Nelle prossime sezioni tali problemi verranno analizzati in dettaglio per dare un quadro più completo degli ostacoli che si devono affrontare nell'approccio

all'estrazione di informazioni sulle affiliazioni.

2.2.1 Differenze Linguistiche

VU University Amsterdam, Amsterdam, The Netherlands.
Vrije Universiteit Amsterdam, Amsterdam, Netherlands.

Tabella 2.3: esempi di rappresentazioni diverse di stessa entità: lingua

In questo esempio vediamo una differenza sostanziale nell'esprimersi alla stessa entità; i ricercatori possono riportare le proprie affiliazioni nella propria lingua o in inglese, a seconda del tipo di pubblicazione e delle abitudini personali; questo rende molto difficile riportare sotto un unico dominio questo genere di affiliazioni.

2.2.2 Differenze sul Livello di Dettaglio delle Informazioni riportate

Non sempre il livello di dettaglio di espressione della propria affiliazione è il medesimo, sia per scelte personali degli autori che per richieste avanzate dagli editori stessi.
Ci sono casi in cui il solo riferimento all'organizzazione di appartenenza viene ritenuto sufficiente come nel caso espresso nel primo item della tabella 2.4 altri in cui sia richiesta l'aggiunta di informazioni riguardanti il dipartimento interno all'organizzazione che genera un problema che va oltre il semplice rilevamento di un'affiliazione ma che consiste nella crazione proprietà che colleghino le unità alla loro organizzazioni principali.
Possono essere riportati anche molteplici dipartimenti, come nel caso del terzo item nella stessa tabella, facendo nascere la necessità di gestire tale

Vrije Universiteit Amsterdam, Amsterdam, Netherlands.
Department of Computer Science, Vrije Universiteit Amsterdam, Amsterdam, The Netherlands.
AI Department, Department of Computer Science, Vrije Universiteit Amsterdam, Amsterdam, The Netherlands.
Department of Computer Science, Vrije Universiteit Amsterdam, De Boelelaan 1081, 1081HV Amsterdam, The Netherlands.
Department of Computer Science, Vrije Universiteit Amsterdam (VUA), De Boelelaan 1081, 1081HV Amsterdam, The Netherlands.

Tabella 2.4: esempi di rappresentazioni diverse di stessa entità: quantità informazioni

situazione in maniera ancora diversa per non generare dati ambigui che sarebbero poi poco affidabili.
Un altro caso di interesse riguarda l'aggiunta di informazioni geografiche più particolareggiate rispetto alla sola coppia città - paese, prevedendo quindi la presenza di indirizzi a cui un'organizzazione fa capo complicando ulteriormente l'estrazione di informazioni corrette sull'affiliazione in questione.
In questa categoria rientrano casi particolari, che vanno menzionati, riguardanti enti presenti in stati federali (Stati Uniti d'America) o con strutture complesse (Creta).
Facendo riferimento alla tabella 2.5 possiamo notare come la presenza di stati interni a quello principale, come accade negli Stati Uniti, costituisca un ulteriore problema alla corretta analisi dell'affiliazione riportata. Tale caso si distingue in altri due sotto casi, in quanto ad ogni stato interno agli USA è associato un diminuitivo (es. Georgia - GA), quindi, a discrezione del ricercatore, questa informazione può essere riportata in entrambi i modi.
Un altro caso particolare è presente in tabella 2.6 riguardante l'isola di Creta,

che viene spesso riportata nei riferimenti alle organizzazione presenti in essa. La natura di questo problema riguarda la complessità maggiore nella strutturazione della label che aggiunge alle classiche informazioni geografiche (città, stato) anche una regione; questo tipo di informazioni aggiuntive vanno a complicare ulteriormente la corretta classificazione delle diverse parti delle label. Anche in questa situazione il livello di dettaglio con cui tali informazioni si possono presentare è arbitrario e quindi anche tale caso va fatto presente e gestito in maniera specifica.

`LSDIS Lab, Department of Computer Science, University of Georgia, Athens, GA 30602, USA`
`Department of Computer and Information Science, University of Pennsylvania, 200 South 33rd Street, Pennsylvania, PA 19104-6989, USA`

Tabella 2.5: esempi di rappresentazioni affiliazioni in USA

`Institute of Computer Science, FORTH, Vassilika Vouton, P.O. Box 1385, GR-711 10 Heraklion, Crete, Greece`

Tabella 2.6: esempi di rappresentazioni affiliazioni in Creta

In ultimo, una particolare menzione va fatta per gli acronimi delle organizzazioni, che spesso possono essere forniti in combinazione con il nome esteso della stessa (Vrije Universiteit Amsterdam (VUA)) o in sostituzione di quest'ultimo, creando un ulteriore problema di disambiguazione delle risorse. Per una macchina preposta all'estrazione ed analisi delle informazioni sulle affiliazioni, disambiguare labels che possone essere riportate in questa molteplicità di modi, risulta arduo.

2.2.3 Differenze Posizionali

`Vrije Universiteit Amsterdam (VUA), Department of Computer Science, De Boelelaan 1081, 1081 HV Amsterdam, Netherlands.`
`Department of Computer Science, Vrije Universiteit Amsterdam (VUA), De Boelelaan 1081, 1081HV Amsterdam, The Netherlands.`

Tabella 2.7: esempi di rappresentazioni diverse di stessa entità: posizione

Generalmente le informazioni sulle affiliazioni sono riportate dall'unitpiù interna alla più esterna, seguite da informazioni su città e paese in cui si trovano, ma come detto questa struttura non costituisce uno standard.
Anche una differenza all'apparenza marginale, come postporre le informazioni su un dipartimento rispetto all'organizzazione principale (tabella **??**d), costituisce un problema sostanziale per il corretto riconoscimento come unica entità di quella che all'occhio umano rappresenta chiaramente la stessa affiliazione.

Capitolo 3

AffiliationExtractor: un tool modulare per l'estrazione e serializzazione di informazioni sulle affiliazioni

Nei precendenti capitoli sono stati descritti i principali dataset RDF e i problemi che ancora sono presenti nel modo in cui le informazioni sulle affiazioni sono attualmente disponibili sul web.
Dato l'insieme di difficoltà descritte nel capitolo precedente, la soluzione implementata nel tool si è dovuta servire di diverse risorse per coprire il maggior numero di casi.
In questo capitolo si tratteranno, dopo un cenno al progetto "Semantic Lancet"[12] di cui il lavoro di questa tesi fa parte, le tecnologie impiegate nella realizzazione della soluzione proposta ai problemi descritti nel precedente capitolo. Questo sarà diviso in sezioni riguardanti la modellazione dei dati analizzati e le tecnologie usate nell'analisi vera e propria, soffermandosi sul motivo per cui sono state ritenute valide per il compito preposto e successivamente si passerà ad esporre il funzionamento ad alto livello di "AffiliationExtractor", il tool che implementa detta soluzione.

3.1 Semantic Lancet

Semantic Lancet è un progetto dell'Università di Bologna il cui obiettivo è quello di rendere disponibili una grande quandità di dati semantici riguardanti le pubblicazioni scientifiche e di fornire un insieme di sofisticati servizi sulla base dei dati mantenuti.
Le aree semantiche che questo progetto supporta spaziano dalle infomazioni sugli autori di un paper alle informazioni su citazioni che un dato articolo contiene, o citazioni dello stesso articolo in altre pubblicazioni, alle informazioni su abstract degli articoli.
L'obiettivo di questa tesi è quello di rendere diponibili anche informazioni riguardanti le affiliazioni degli autori e di conseguenza delle ricerche da loro pubblicate.

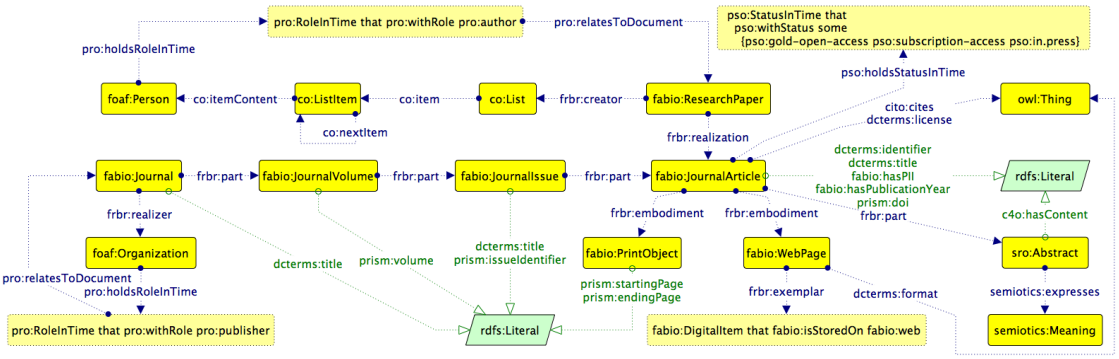

Figura 3.1: Semantic Lancet: modello dei dati

3.2 Modellazione dei Dati

In questa sezione verrà trattato il modello impiegato nella serializzazione dei dati estratti, descrivendo le ontologie implementate nella soluzione presentata.

3.2.1 Ontologie utilizzate

Per serializzare le informazioni raccolte si è scelto di usare l'ontologia PRO [10](Publishing Role Ontology) la quale fa parte di una più ampia suite di ontologie denominata SPAR (Semantic Publishing and Referencing Ontologies). La caratteristica principale di PRO è la possibilità di associare ad un Agent un ruolo, tramite la property ":hasRoleInTime" a cui è possibile a sua volta associare informazioni realtive al documento realizzato (:refersToDocument), all'affiliazione (:refersToOrganization) e al periodo durante il quale detto Agent ha ricoperto tale ruolo (RoleInTime).
Associare il ruolo, piuttosto che l'autore stesso, all'articolo pubblicato e all'organizzazione, fornisce un'informazione circa il momento nel tempo in cui una data posizione è stata ricoperta in una specifica organizzazione dalla persona considerata. Questo ci aiuta a stabilire una timeline che risulta molto utile per affrontare il problema della mancanza di riferimenti temporali di cui si é parlato nel precedente capitolo.
Un esempio di questa proprietà è apprezzabile nella figura 3.4, prestando attenzione a come, tramite la proprietà ":relatesToOrganizazion", non sia direttamente la persona Ronny Siebes ad essere stata collegata all'Univerisità Vrije di Amsterdam ma il suo ruolo ("RoleInTime") come autore di un determinato articolo ":relatedToDocument". L'ontologia PRO fa uso della più ampia Foaf Ontology includendo le classi già definite in essa quali Agent, Document e Organization.
Per completare la panoramica delle ontologie usate nella serializzazione dei dati, dopo l'esecuzione del tool, va citata Organization Ontology; Essa forni-

sce una classe di partenza (org:Organization) che è definita come equivalente all'omonima presente in Foaf Ontology.
La ragione per cui si è scelto di includere anche questa ontologia è motivata dalla completezza di proprietà e classi utili a definire un organizzazione, come le proprietà ":unitOf" e "hasUnit" utilizzate per definire le associazioni bidirezionali tra le organizzazioni e i loro dipartimenti (figura 3.5).
Una considerazione va fatta sulla creazione degli URI relativi ai dipartimenti; i loro nominativi, soprattutto per quanto riguarda i dipartimenti delle università, sono spesso identici (es un "Department of Computer Science" sarà presente in ogni università considerata), quindi la modellazione di un URI del tipo "http://www.semanticlancet.eu/resource/department_of_computer_science" avrebbe poco senso e genererebbe problemi di ambiguità in ogni riferimento a tale risorsa.
Si è scelto di aggiungere all'URI dei dipartimenti il nome dell'organizzazione di cui fanno parte, quindi il Department of Computer Science dell'Univeristà Vrije di Amsterdam sarà serilizzato sotto il dominio "http://www.semanticlancet.eu/resource/department_of_computer_science_vu_university_amsterdam rendendolo univoco e disambiguandolo da tutti gli altri dipartimenti dello stesso tipo presenti in altre organizzazioni.

3.3 Tecnologie della Soluzione

In questa sezione verranno descritte le tecnologie impiegate nell'analisi dei dati, per renderli disponibili al modulo serializzatore secondo il modello presentato nella precedente sezione.
AffiliationExtractor prende in input un documento XML e, dopo aver individuato le labels riguardanti le affiliazioni, tali labels vengono divise in parti per associare ad ognuna di esse un significato (informazioni riguardanti paesi, città e organizzazioni contenute nelle labels). Tali informazioni vengo ricavate tramite la consultazione di un dataset RDF locale e, nel caso quest'ultimo

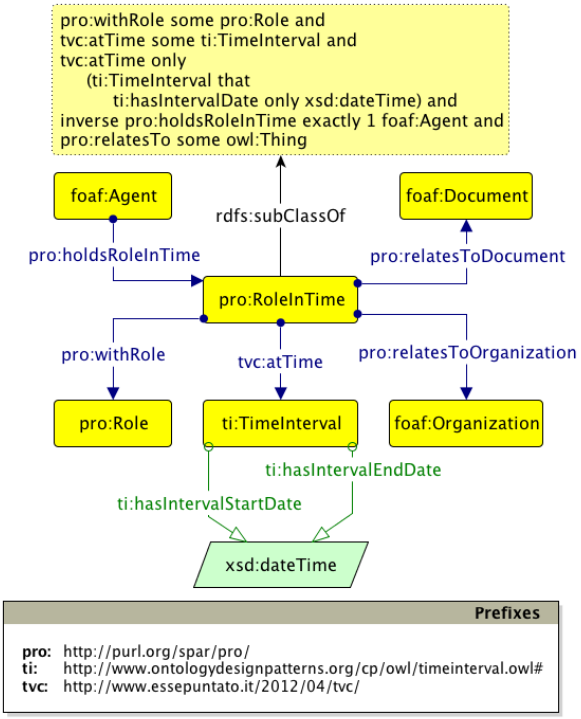

Figura 3.2: diagramma PRO ontology

non contenga le risorse desiderate, tramite l'interrogazioni a Dbpedia.
Non tutti i casi sono però gestibili utilizzando i datasets, per questo sono state definite regole di fallback per coprire anche queste situazioni.
I dati analizzati vengono poi serializzati nel dataset locale implementando un processo di machine learning per rendere sempre meno dipendente il tool dall'utilizzo di Dbpedia.

3.3.1 Perchè DBpedia?

DBpedia è un progetto portato avanti dalla community per rendere disponibili l'enorme quantità di informazioni presenti in Wikipedia, ad oggi la più ampia riserva di informazioni della storia, nei formati definiti dalle ontologie facenti parte del Semantic Web.

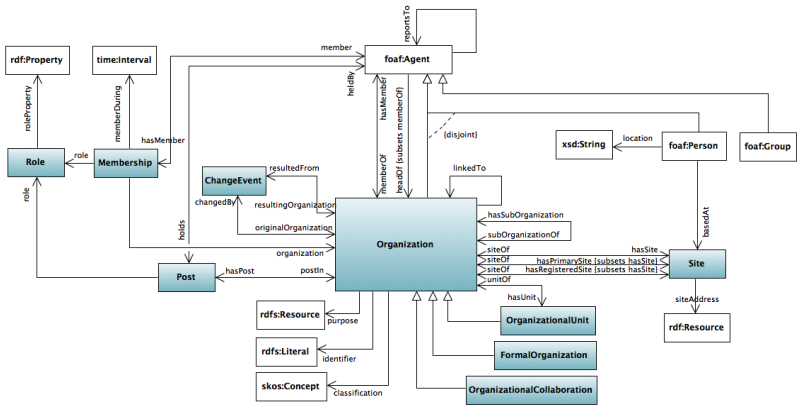

Figura 3.3: diagramma Organization Ontology

```
<rdf:Description rdf:about="http://www.semanticlancet.eu/resource/person/ronny_siebes">
    <pro:holdsRoleInTime>
        <pro:RoleInTime rdf:about="http://www.semanticlancet.eu/resource/1-s2.0-S1570826803000064/author/ronny_siebes_as_author">
            <pro:withRole rdf:resource="http://purl.org/spar/pro/author"/>
            <pro:isRoleHeldBy rdf:resource="http://www.semanticlancet.eu/resource/person/ronny_siebes"/>
            <pro:relatesToDocument rdf:resource="http://www.semanticlancet.eu/resource/1-s2.0-S1570826803000064"/>
            <pro:relatesToOrganization rdf:resource="http://www.semanticlancet.eu/resource/vu_university_amsterdam"/>
        </pro:RoleInTime>
    </pro:holdsRoleInTime>
</rdf:Description>
```

Figura 3.4: esempio serializzazione usando PRO

Una simile sorgente di informazioni risulta utilissima nell'implementazione della soluzione proposta in questa tesi, in quanto fornisce un metodo per verificare la natura dei dati che vengono analizzati per facilitarne la classificazione.

Le interrogazioni a tale servizio sono eseguite tramite domande ("query") scritte in un linguaggio specifico ("SPARQL") all'endpoint messo a disposizione all'indirizzo "http://dbpedia.org/sparql", rendendo possibile, per esempio, domandare se una data stringa identifica una particolare Organizzazione piuttosto che una città o un paese, permettendo quindi di associare con buona sicurezza un significato che sarebbe di difficile interpretazione altrimenti. In più la natura comunitaria nella rappresentazione delle informazioni presenti su tale dataset fornisce l'accesso ad una serie di traduzioni in più lingue della stessa risorsa, consentendo di affrontare il problema dei differenti idiomi

```
<rdf:Description rdf:about="http://www.semanticlancet.eu/resource/vu_university_amsterdam">
    <org:hasUnit>
        <org:OrganizationalUnit rdf:about="http://www.semanticlancet.eu/resource/
        department_of_computer_science_vu_university_amsterdam">
            <rdfs:label xml:lang="en">Department of Computer Science</rdfs:label>
            <semprop:country rdf:resource="http://www.semanticlancet.eu/property/country"/>
    </org:hasUnit>

</rdf:Description>
```

Figura 3.5: esempio serializzazione usando PRO

in cui un'affiliazione può essere espressa (figura 3.6).

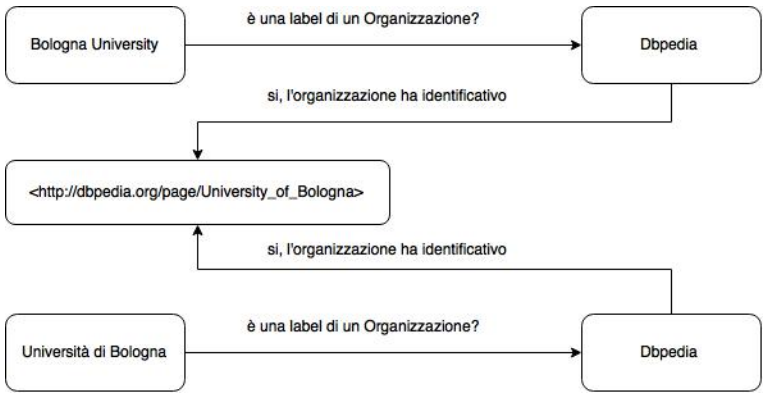

Figura 3.6: esempio di uso dbpedia per disambiguare sulla lingua

3.3.2 Machine Learning: dataset locale

I dati collezionati vengono serializzati in un dataset locale secondo le specifiche definite nelle ontologie esposte nella sezione precedente.
Tali informazioni, oltre ad essere destinate ad una fruizione esterna all'esecuzione del tool (es. nella realizzazione delle statistiche utili alla valutazione del rendimento di organizzazioni e ricercatori), sono anche consultate durante l'analisi nelle successive iterazioni del tool rendendolo progressivamente sempre meno dipendente dalle consultazioni del dataset di Dbpedia.

Questo consente, inoltre, di diminuire i tempi delle esecuzioni di AffiliationExtractor in quanto, essendo Dbpedia una risorsa comune all'intera comunità, essa è spesso interrogata da migliaia di servizi nello stesso momento, allungando i tempi di risposta per le singole richieste.

3.3.3 Regole di Fallback

Il confronto con i dataset non è l'unica maniera prevista per riconoscere e disambigure le informazioni analizzate.
Sono state previste alcune regole di fallback per garantire una gestione più ampia dei casi possibili, fornendo anche una valutazione della confidence sulla validità delle informazioni recuperate.

Liste Json di Risorse

Nel caso la ricerca di città e paesi non venga completata tramite interrogazioni ai dataset, sono mantenuti in locale diversi oggetti json contententi liste di stati, città principali indicizzate per paese e una lista specifica per riconoscere stati Americani.
La lista di paesi fornisce anche un buon numero di traduzioni note delle risorse contenute, in modo da poter ottenere risultati molto simili alla ricerca degli stessi su Dbpedia.
L'oggetto json contenente informazioni sulle principali città per ogni stato è meno completo ma costituisce comunque un valido metodo per riuscire ad individuare città che sfuggono alle query su Dbpedia che ha uno dei suoi limiti nella varietà in cui le risorse sono serializzate su di esso; non tutte le città sono istanze della classe dbpedia-owl:City (Bologna per fare un esempio non lo è).
Infine la lista degli Stati Americani, comprensiva dei diminuitivi associati, è stata introdotta per raffinare ulteriormente le analisi del tool.

Analisi tramite Keywords

Uno dei principali problemi esposti nel capitolo precedente riguarda il riconoscimento di organizzazioni e dipartimenti, indipendentemente dalla posizione occupata all'interno della label.
Per rendere possibile ciò sono state introdotte parole chiave riguardanti unità di affiliazioni (department, faculty, school, institute, ecc..) ricavate dall'analisi delle labels stesse che aiutano a distinguere con maggiore sicurezza la natura dei dati analizzati.

Regola sulla Posizione

L'ultima risorsa per cercare di distingure e correttamente riconoscere le affiliazioni in un paper riguarda un vincolo imposto sulla posizione in cui le diverse parti delle labels si trovano all'interno della label stessa.
Detto vincolo, a cui si ricorre solo nel caso in cui ogni altro controllo fallisca, assume che le informazioni sulle affiliazioni siano riportate dal livello più basso fino a quello più alto

- Dipartimento, Organizzazione, Città, Stato

Valore di Confidence

Questo vincolo serve a valutare la qualità dei risultati prodotti in base alla maniera in cui sono stati ottenuti.
Il valore massimo viene assegnato a risorse verificate correttamente tramite uso dei datasets descritti, mentre un valore più basso viene assegnato all'applicazione delle regole descritte in precedenza.
Questo valore, oltre ad esprimere la sicurezza circa la correttezza dei risultati forniti, apre anche la possibilità ad estensioni future del tool andando a raffinare i dati con un valore di confidence non soddisfacente.

3.4 Dal paper all'RDF

Questa sezione si concentra nell'illustrare i passi che AffiliationExtractor compie nel suo cammino da un paper ad un insieme di triple RDF che ne riassumono le affiliazioni.
La struttura del tool è fortemente modulare facendo si che la parte di estrazione di dati, quella di analisi e quella di serializzazione siano indipendenti tra loro garantendo la possibilità di estensioni aggiungento ulteriori moduli e adattamenti sosituendone uno.
Per esempio nel caso di uso di questa tesi i dati sulle affiliazioni vengono estratti dal raw text XML degli articoli passati in input ma per supportare più formati basterebbe modificare l'estrattore di label in modo da prevedere la gestione di più tipi di input, lasciando inalterati i moduli di analisi e serializzazione.
Cominciamo includendo un grafico (figura 3.7)che ne riassume i punti salienti che saranno analizzati estensivamente nel proseguio del capitolo.

3.4.1 Estrazione delle label delle Affiliazioni

Il primo passo dell'attività di esecuzione di AffiliationExtractor è quella di estrazione delle labels di affiliazioni dal dataset fornito.

`Department of Computer Science, Vrije Universiteit Amsterdam (VUA), De Boelelaan 1081, 1081HV Amsterdam, The Netherlands`

Tabella 3.1: esempio label

Il caso di utilizzo preso in considerazione riguarda l'estrazione dal raw text in XML di articoli facenti parte del Journal "Web Semantics: Science, Services and Agents on the World Wide Web" numero 15708268.

Dato un articolo il tool cerca in particolari tag i dati relativi alle affiliazioni seguendo lo standard fornito da Elsevier restituendo messaggi in output circa la presenza o meno di questi dati.
In caso tali informazioni vengano trovate si passa all'analisi vera e propria della label recuperata.

3.4.2 Analisi delle label delle Affiliazioni

Successivamente all'identificazione di una label riguardante affiliazioni si passa all'analisi della stessa.
Il primo step è rappresentato dalla suddivisione in parti della label in presenza di "," o "/"(tabella 3.2).

Department of Computer Science
Vrije Universiteit Amsterdam (VUA)
De Boelelaan 1081
1081HV Amsterdam
The Netherlands

Tabella 3.2: esempio divisione label

Il primo punto di interesse è rappresentato dalla ricerca del Paese; assumendo che la struttura delle labels veda generalmente i riferimenti geografici situati all'estrema sinistra delle stesse, la suddetta ricerca comincia proprio dall'ultima parte generata dalla divisione descritta in precedenza, procedendo poi iterando sulle altre porzioni.
Ogni parte presa in considerazione viene epurata da articoli (The Netherlands - Netherlands) e confrontata preventivamente con la collezione di dati presenti nel database semantico locale, servendosi di particolari parametri nelle interrogazioni per restringere la ricerca ai soli dati riguardanti Paesi.

In caso le informazioni mantenute nel dataset locale non siano sufficienti AffiliationExtractor interroga il database semantico Dbpedia.
Il ruolo che Dbpedia ricopre all'interno dell'analisi delle labels è fondamentale in quanto è il mezzo usato per identificare con certezza la natura dei dati estratti e per disambiguare con altrettanta sicurezza dati diversi ma che si riferiscono alla stessa entità. Il vantaggio che Dbpedia fornisce consiste nella possibilità di consultare una serie di traduzioni note per le risorse richieste in modo da risolvere uno dei problemi illustrati in precedenza (figura 3.8) oltre al fatto di potersi riferire a risorse semanticamente correlate in successive richieste (come nel caso della ricerca della città).

Se la ricerca su entrambi i dataset forniti non produce risultati si interroga una lista di Paesi mantenuta localmente.
Conclusa la ricerca del Paese se tale informazione è stata trovata si elimina dalla lista di parti della label la porzione che ha prodotto il risultato in modo da non essere analizzata successivamente durante la ricerca di altre informazioni (tabella 3.3) e i dati recuperati vengono mantenuti in una struttura locale all'esecuzione del tool, in caso contrario si prosegue nell'analisi senza informazioni sul Paese.

Il passo successivo riguarda la ricerca della città, che ha molti punti in co-

Department of Computer Science
Vrije Universiteit Amsterdam (VUA)
De Boelelaan 1081
1081HV Amsterdam

Tabella 3.3: esempio divisioni label dopo ricerca Paese

mune con quella del Paese ma che si serve dei dati precedentemente raccolti per ottimizzare i tempi e la qualità delle ricerche. Le richieste a Dbpedia sono dispendiose in termini di tempo, se troppo generiche, il che rende prezioso il riutilizzo delle informazioni nelle successive interrogazioni a tale servizio.
Le parti interessate da questo momento dell'analisi vengono epurate da nu-

meri e stringhe interamente in upper case, in quanto tali informazioni fanno spesso riferimento a codici di avviamento postale o acronimi di quartieri (1081HV Amsterdam - Amsterdam).
Anche in questo caso la ricerca può usufruire di una lista mantenuta localmente serializzata nella forma di un dizionario la cui chiave è il Paese a cui sono associate le principali città.
Una volta raccolte tali informazioni, analogamente a quanto fatto dopo la ricerca del Paese, si elimina la parte di label che ha prodotto il rusultato, si salvano i dati nella struttura descritta in precedenza e si passa alla ricerca delle organizzazioni (tabella tab 3.6).

Il processo consiste nell'interrogare ancora una volta il dataset locale e suc-

Department of Computer Science
Vrije Universiteit Amsterdam (VUA)
De Boelelaan 1081

Tabella 3.4: esempio divisioni label dopo ricerca Città

cessivamente, in caso di informazioni non conclusive, interpellare nuovamente le risorse di Dbpedia per trovare la più corretta corrispondenza.
In questo caso dalla stringa da passare vengono eliminate parti in parentesi, le quali in questo caso di studio hanno generalmente riportato acronimi dell'organizzazione che non sono solitamente presenti nelle labels di Dbpedia (Vrije Universiteit Amsterdam (VUA) - Vrije Universiteit Amsterdam).
Molto spesso i risultati che tale servizio restituisce sono molteplici il che richiede un lavoro di scrematura sfruttando la vicinanza tra stringhe tramite un implementazione dell'algoritmo di Levenshtein.
Ancora una volta di conseguenza alla corretta verifica dei dati sulle organizzazioni si escludono le parti della label presa in considerazione (tabella 3.5).
Completata quest'ultima analisi ci si trova di fronte a diverse possibilità:

1. Organizzazione trovata e label completamente processata.

2. Organizzazione trovata ma parti di label ancora da analizzare.

Department of Computer Science
De Boelelaan 1081

Tabella 3.5: esempio divisioni label dopo ricerca Organizzazioni

3. Organizzazione non trovata ma parti di label ancora da analizzare.

4. Organizzazione non trovata e label completamente processata.

Analizziamo prima i casi 2 e 3. Se parti di label sono ancora presenti si passa ad un'analisi che segue le regole di fallback descritte nelle sezioni precedenti riguardanti parole chiave nel testo della label e sulla posizione che tali parti occupano nella label stessa.
Nel caso in cui un'organizzazione sia stata correttamente trovata, se parti di label ancora sono rimaste non assegnate, potremmo essere in presenza di dati riguardanti dipartimenti di quella specifica affliazione (è questo il caso proposto come esempio durante questa dissertazione).
Prima di poter procedere oltre vengono cercate ed eliminate parti relative ad indirizzi o luoghi servendosi sia dell'aiuto di Dbpedia che di una ricerca per keywords (street, road, ecc...) o ancora della presenza di numeri in stringhe in cui non compaiono parole chiave che indicherebbero la presenza di un'organizzazione (university, department). Ognuna delle analisi appena descritte vengono effettuate su traduzioni in lingua inglese delle stringhe interessate in modo da limitare al minimo il numero di keywords da dover immagazzinare in locale.

Department of Computer Science

Tabella 3.6: esempio divisioni label dopo analisi indirizzi

Questo passaggio di rimozione di dati non interessanti all'obiettivo preposto è in comune con tutti i casi in cui parti di labels restassero non analizzate.

Ora le parti ancora da analizzare vengono sottoposte ad un'ulteriore analisi riguardante l'assegnazione dei dipartimenti all'organizzazione. In questo caso vengono cercate parole chiave come institute, department, faculty, ecc... e nel caso vengano trovate le parti di label corrispondenti vengono marcate come unità dell'organizzazione principale.
In caso di organizzazione non trovata in precedenza, successivamente alla ricerca per keywords appena illustrata, si assume che l'organizzazione principale si trovi alla destra delle sue unità basandosi sulla maniera in cui generalmente vengono riportate le affiliazioni nei documenti considerati.
Processata l'intera label vengono assegnati valori numerici di confidence all'intera collezione di dati estratti per valutare la bontà delle informazioni estratte, privilegiando le ricerche che hanno avuto riscontro alla verfica tramite Dbpedia.
Una volta completata quest'ultima fase si passa alla serializzazione dei dati nel dataset locale in modo da poter riutilizzare le informazioni trovate nelle successive analisi.

3.4.3 Serializzazione delle informazioni

Questa sezione rappresenta l'ultimo passo dell'esecuzione di AffiliationExtractor il quale consiste nel salvataggio in un database semantico delle informazioni recuperate durante l'analisi descritta nella precedente sezione.
Tali informazioni vengono incapsulate in un oggetto che rende estremamente semplice il processo di serializzazione, che viene effettuato solo se un'organizzazione sia stata effettivamente recuperata durante l'analisi.
Vengono salvate prima le informazioni riguardanti il Paese poi, dopo aver recuperato l'autore affiliato all'organizzazione, tramite informazioni reperibili in specifici tag del sorgente dell'articolo preso in considerazione, si collegano le informazioni seguendo le specifiche delle ontologie PRO e Organization Ontology.
Particolare menzione va fatta per illustrare il caso in cui più unità siano sta-

te trovate per la stessa organizzazione. In tal caso ognuno dei dipartimenti recuperati viene collegato singolarmente allòrganizzazione principale facendo risultare l'autore dell'articolo (e di conseguenza l'articolo stesso) affiliato ad ogni unità trovata.

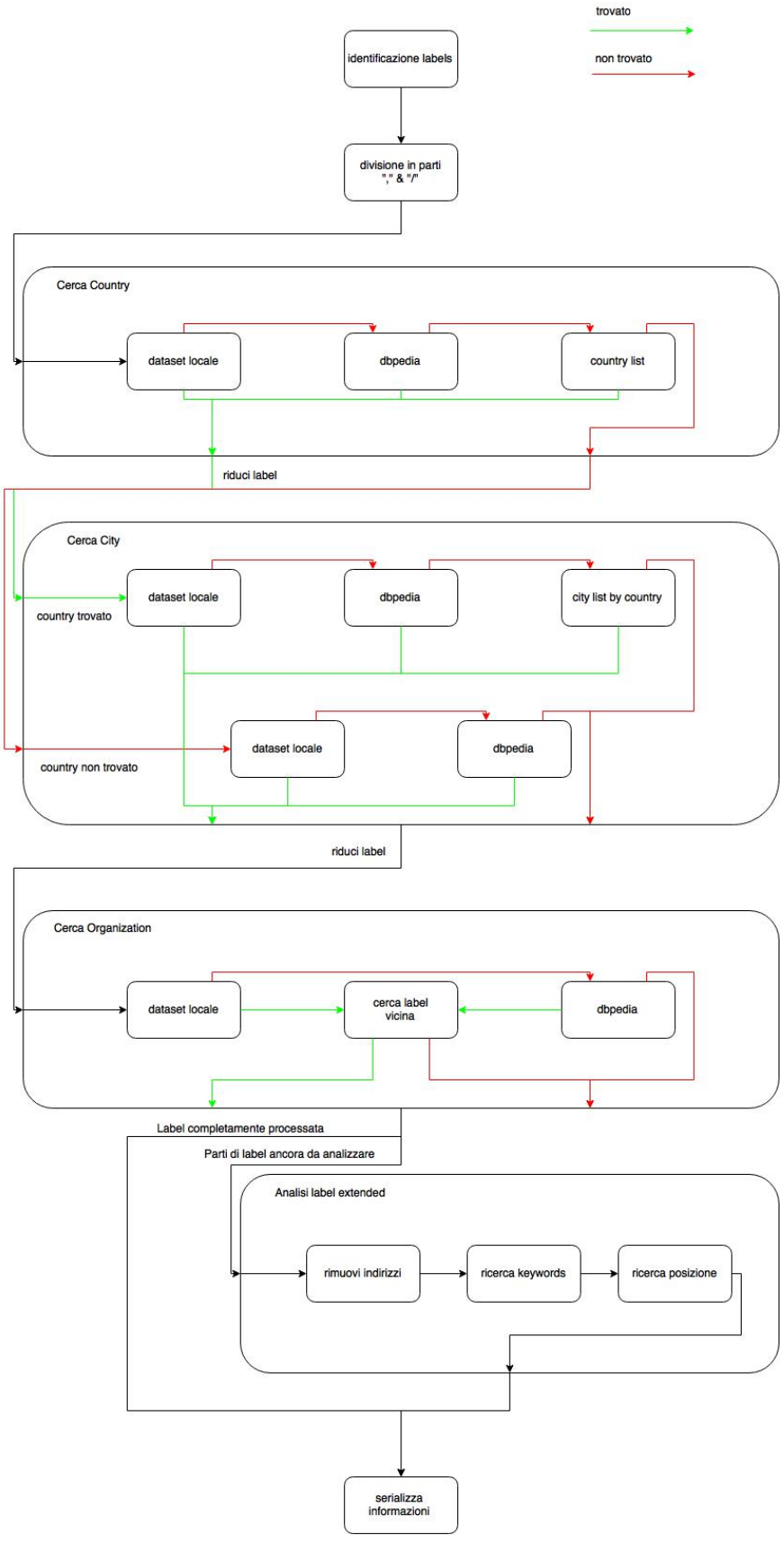

Figura 3.7: grafico del funzionamento di AffiliationExtractor

label
"Netherlands"@en
"هولندا"@ar
"Niederlande"@de
"Países Bajos"
"Pays-Bas"@fr
"Paesi Bassi"
"Nederland"@nl
"オランダ"@ja
"Holandia"
"Países Baixos"
"Нидерланды"@ru
"荷兰"

Figura 3.8: esempio traduzioni Netherlands su Dbpedia

```
<rdf:Description rdf:about="http://www.semanticlancet.eu/resource/person/ronny_siebes">
    <pro:holdsRoleInTime>
        <pro:RoleInTime rdf:about="http://www.semanticlancet.eu/resource/1-s2.0-S1570826803000064/author/ronny_siebes_as_author">
            <pro:withRole rdf:resource="http://purl.org/spar/pro/author"/>
            <pro:isRoleHeldBy rdf:resource="http://www.semanticlancet.eu/resource/person/ronny_siebes"/>
            <pro:relatesToDocument rdf:resource="http://www.semanticlancet.eu/resource/1-s2.0-S1570826803000064"/>
            <pro:relatesToOrganization rdf:resource="http://www.semanticlancet.eu/resource/
            department_of_computer_science_vu_university_amsterdam"/>
        </pro:RoleInTime>
    </pro:holdsRoleInTime>
</rdf:Description>

<rdf:Description rdf:about="http://www.semanticlancet.eu/resource/vu_university_amsterdam">
    <rdfs:label xml:lang="en">VU Amsterdam University</rdfs:label>
    <rdfs:label xml:lang="nl">Vrije Universiteit Amsterdam</rdfs:label>
    <rdfs:label xml:lang="fr">Université libre d'Amsterdam</rdfs:label>
    <rdfs:label xml:lang="de">Freie Universität Amsterdam</rdfs:label>
    <rdfs:label xml:lang="ru">Амстердамский свободный университет</rdfs:label>

    <semprop:country rdf:resource="http://www.semanticlancet.eu/property/country"/>

    <org:hasUnit>
        <org:OrganizationalUnit rdf:about="http://www.semanticlancet.eu/resource/
        department_of_computer_science_vu_university_amsterdam">
            <rdfs:label xml:lang="en">Department of Computer Science</rdfs:label>
            <semprop:country rdf:resource="http://www.semanticlancet.eu/property/country"/>
    </org:hasUnit>

</rdf:Description>
```

Figura 3.9: esempio risultati in RDF

Capitolo 4

AffiliationExtractor: Dettagli implementativi

In questo capitolo verrà esposto nel dettaglio il funzionamento di AffiliationExtractor descrivendone i diversi moduli, i compiti a loro assegnati e le implementazioni che permettono loro di svolgere detti compiti.
Il tool si compone di 5 moduli principali; "affiliationExtractor.py" è lo script preposto all'estrazione delle label iterando tra i file XML contenenti il testo degli articoli del dataset di partenza forniti in input. Le informazioni estratte vengono passate ad "analyzer.pt" che si occupa dell'analisi delle label sevendosi di "sets.py" come tramite per comunicare con i dataset (locale e Dbpdedia) che a sua volta attinge alle varie query SPARQL contenute in "queries.py". Completata l'analisi il controllo ritorna ad affiliationExtractor.py il quale si occupa di passare le informazioni processate allo script "serializer.py" il cui compito è quello di salvare i dati che riceve nel dataset locale secondo il modello espresso nel precedente capitolo.
Il tool si serve anche di un modulo di utilità chiamato "debug.py" il quale presispone diversi tipi di messaggi da stampare in console per scandire i diversi passi dell'esecuzione.
Vi sono infine una serie di librerie python aggiuntive utili per aggiungere funzionalità ad AffiliationExtractor che verranno enunciate nella descrizione

dei moduli che le utilizzano.
La scelta di sviluppare il tool in un linguaggio come Python è stata operata sulla base della predisposizione di questo linguaggio allo sviluppo di applicazioni modulari che, come esposto nei precendenti capitoli, costituisce un vantaggio tangibile per permettere sviluppi futuri o il semplice riutilizzo di alcune parti dello strumento descritto in altri progetti.

4.1 Lo standard XML di Elesevier

Gli articoli presi in considerazione durante lo sviluppo e la fase di test di AffiliationExtractor fanno parte del journal "Web Semantics: Science, Services and Agents on the World Wide Web" di Elsevier.
Questo ente ha sviluppato uno standard per quanto riguarda la forma di scrittura di articoli in XML fornendo specifiche precise su come e dove inserire determinate informazioni tramite la definizione di specifici tag da inserire nel raw text degli articoli che verranno pubblicati sul proprio journal.
L'intero standard è reperibile nella pubblicazione "The Elsevier DTD 5 Family of XML DTDs" che ne descrive tutti i tag nel dettaglio ma in questa sezione si farà riferimento al nodo in cui vengono inserite le informazioni utili per il tool.
Il nodo che è stato utilizzato per ricavare le informazioni sulle affliazioni è "ce:affilation" dove il prefisso "ce" sta ad indicare il namespaces proprio di Elsevier con cui l'ente ha definito tutti i nodi nel suo standard.
In figura si può apprezzare un esempio di affiliazioni espresse in questo standard in cui è possibile notare la presenza di un attributo "id", utile a collegare l'informazione di affiliazione all'autore a cui tale informazione si riferisce traminte l'attributo "refid" posto nel nodo "ce:author" del ricercatore corrispondente. In caso di una sola affiliazione l'id può essere omesso ad intendere che ogni autore della pubblicazione in questione condivide la stessa affiliazione.

Il testo effettivo che verrà recuperato per l'analisi si trova invece nell'elemento "ce:textfn"; questo tipo di struttura rende l'estrazione delle giuste informazioni molto semplice ed intuitivo.
Si proseguirà nella prossima sezione ad esporre i dettagli implementativi di AffliationExtractor.

XML

```
<ce:affiliation id="aff1">
  <ce:label>a</ce:label>
  <ce:textfn>Elsevier, Radarweg 29,
    1043 NX Amsterdam, The Netherlands</ce:textfn>
</ce:affiliation>
<ce:affiliation id="aff2">
  <ce:label>b</ce:label>
  <ce:textfn>Elsevier Inc., P.O. Box 945, New York,
    NY 10159-0945, USA</ce:textfn>
</ce:affiliation>
<ce:affiliation id="aff3">
  <ce:label>c</ce:label>
  <ce:textfn>Elsevier Ltd, The Boulevard, Langford Lane,
    Kidlington, Oxford OX5 1GB, UK</ce:textfn>
</ce:affiliation>
```

Figura 4.1: esempio un di affiliazioni espresse nello standard di Elsevier

4.2 Estrarre labels da XML: affiliationExtractor.py

4.2.1 Librerie Aggiuntive

etree from lxml

il modulo "etree" della libreria lxml viene usato nel parsind dell'XML fornito in input al tool e, grazie al metodo xpath permette di utilizzare il linguaggio Xpath per individuare i nodi di nostro interesse ed estrarre le giuste informazioni da essi.

sys

il modulo “sys” viene utilizzato per permettere di eseguiere controlli sui file input; il tool accetta in ingresso un unico parametro che può esser un singolo file o una lista di file in una cartella, e sul tipo di file in input, controllando l'estensione del file considerato e scartando i file che non sono XML.

4.2.2 Implementazione

affiliationExtractor.py si occupa di estrarre le informazioni da analizzare da un raw text XML scritto secondo lo standard DTD fornito da Elsevier per fornirli ad analyzer.py e di girare i dati processati a serilizer.py per il salvataggio di detti dati.
Svolge quindi l'attività di ricezione del file e mediazione tra i moduli preposti ad analisi e mantenimento delle informazioni contenute in tale file.
Nel momento in cui il tool viene lanciato da riga di comando le possibilità di input contemplate sono di due tipi: un singolo file (comando: “python affliationExtractor.py articolo.xml”) o una lista di file (comando: “python affiliationExtractor.py *”), nel caso nessun file venga passato un messaggio di errore viene stampato in console terminando immediatamente l'esecuzione del tool.
Successivamente comincia un iterazione tra tutti i file forniti in input e, per ognuno di essi, viene contrallato che l'estensione sia “.xml”; in caso questo controllo vada a buon fine, il file in questione viene parsato tramite la funzione “parse” del modulo etree per poi recuperare in nodi relativi alle affiliazioni come descritto nella precedente sezione.
Questa ricerca viene operata servendosi del linguaggio Xpath sempre supportato dalla libreria etree tramite la richiesta “//ce:affilation”, cercando quindi i nodi di quel tipo in ogni punto del documento analizzato.
In caso il documento non contenga tale tag un messaggio di warning viene stampato in console e viene ripresa l'esecuzione sul successivo file dato input.

In caso contrario per ogni nodo trovato si recupera il testo contenuto tramite il path "./ce:textfn/text()" estrando così una label da analizzare.
tale label viene passata al metodo "labelAnalyzer" definito nel modulo analyzer.py e, una volta completate le analisi previste, un oggetto comprensivo di tutti i dati estratti per quella affiliazione viene poi passato al metodo "serialize" in serializer.py per permetterne il salvataggio.
Al momento questo modulo garantisce il solo supporto ad articoli scritti in XML nello standard DTD rilasciato da Elsevier ma epandendolo con altre funzionalità sarebbe possibile permettere l'analisi di più formati vista l'indipendenza dell'estrattore di informazioni e i moduli di analisi e serializzazione.

4.3 Come avviene l'analisi: analyzer.py

4.3.1 Liberie Aggiuntive

distace from Levenshtein

Il metodo distance del modulo python Levenshtein è stato incluso per poter utilizzare il celebre metodo di confronto tra stringhe introdotto da Vladimir Levenshtein nel 1965 per misurare il numero minimo di modifiche elementari necessarie per trasformare una prima stringa in una seconda.
Questo metodo è stato implementato nella funzione "stringMatcher" che si occupa di restituire la percentuale di match tra le stringhe fornite in input.

4.3.2 Implementazione

Seguendo il grafico in figura 4.2 riassuntivo dei passi operati da analyzer.py si tratterà ora il punto centrale del funzionamento di AffiliationExtractor: l'analisi delle informazioni fornite.
Si noti che i metodi per comunicare con i datasets sono definiti nello script "sets.py" che verrà descritto nella sezione successiva quindi in quella corrente

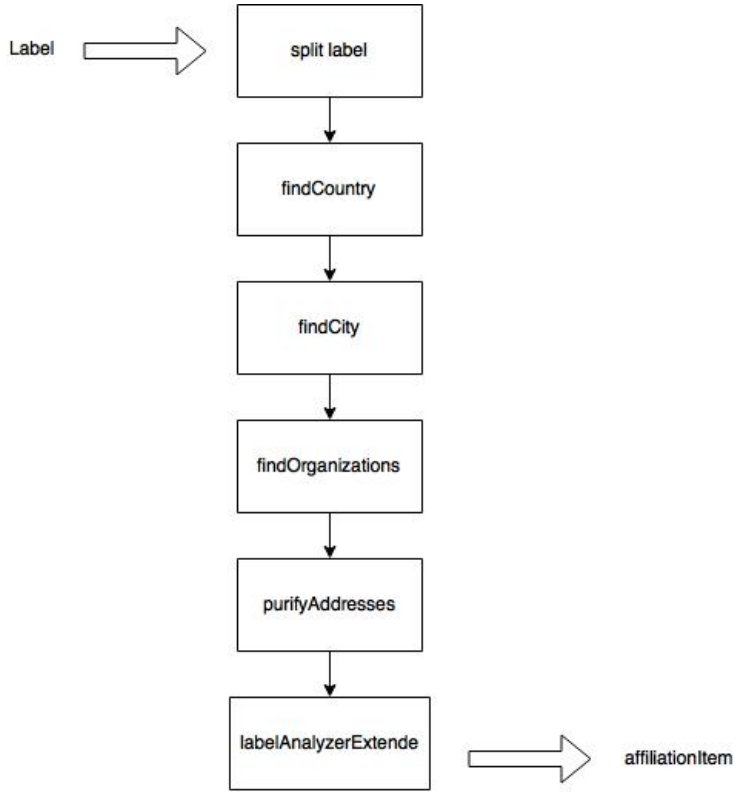

Figura 4.2: schema passi analyzer.py

non si esporrà in dettaglio la maniera in cui avvengono le interrogazioni ma solo come vengono gesititi i risultati.
La prima operazione eseguita è la divisione della label in parti, all'occorrenza di "," o "/", tramite la funzione split che restituisce un array i cui elementi sono le parti stesse della label.
Il risultato della split viene salvato in affiliationParts, varibile su cui tutte le prossime operazioni verranno realizzate.
Il passo successivo consiste nella ricerca del Paese; come sottolineato nel precendente capitolo, si assume che le informazioni geografiche siano da localizzarsi alla fine della label, quindi la ricerca partirà dall'ultimo elemento di affiliationParts.
Per ogni parte analizzata viene controllato che non comincino con spazi, cosa

molto probabile nel caso successivo alla parte iniziale della label in quanto generalmente dopo una virgola si aggiunge uno spazio prima di scrivere altro testo.
La parte correntemente in analisi viene passata alla funzione "findCountry".

findCountry

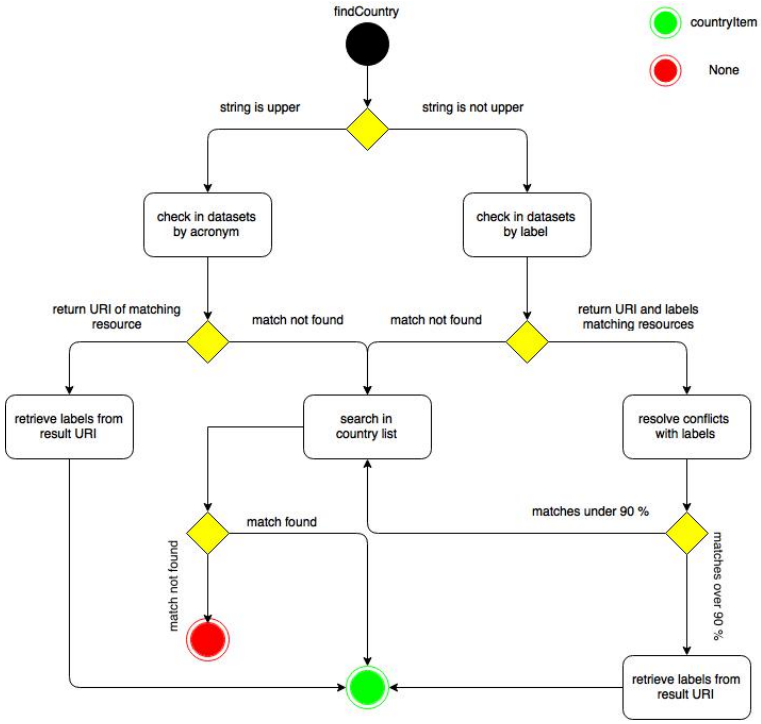

Figura 4.3: stati in findCounty

Tale funzione prende in input una stringa e restituisce un oggetto contenente URI del country (relativo a dbpedia o al dataset locale a seconda di dove è stata trovata la risorsa corrispondente) e un array di labels (sempre restituite da uno dei dataset interrogabili) se la stringa passata trova un riscontro sui dataset a disposizione, altrimenti restituisce "None". Seguendo lo schema in fugura 4.3 verranno esposti i metodi che si occupano di portare a termine ogni singolo passo.

La prima condizione riguarda il caso in cui la stringa passata in input sia espressa in maiuscolo (upper case); questa condizione è stata introdotta per gestiere diversamente i casi relativi a Stati generalmente espressi con i loro acronimi (UK, USA).
In caso un acronimo sia stato passato in input vengono interrogati i datasets in maniera mascherata attraverso i metodi definiti nel modulo sets.py (le firme di tali metodi da qui in avanti verranno precedute da "sets" per indicare dove siano definiti).
La funzione "sets.searchForGivenCountryAcronymFromLabel" restituisce un booleano per comunicare se vi siano match o meno sui dataset a disposizione; in caso affermativo, tramite il metodo "sets.retrieveCountryFromAcronym" viene recuperato l'URI della risorsa su cui c'è stato un riscontro ed infine, vengono recuperate le labels corrispondenti alla risorsa trovata ("sets.retrieveLabelsFromCountryUri") passando l'URI recuperato al passo precedente.
Tali informazioni vengono riposte nell'oggetto "countryItem" che viene ritornato a labelAnalyzer per continuare la sua esecuzione.
Passiamo ora ad analizzare l'altro caso di esecuzione di findCountry, ossia il caso in cui la stringa passata in input non sia in upper case.
In questa situazione il metodo per interrogare i datasets è "sets.searchForGivenCountryFromLabel"; analogamente a quanto scritto in precedenza, tale funzione restituisce un booleano a significare l'esistenza di possibili match tra la stringa in esame e le risorse nei datasets. Nel caso tale risposta sia affermativa vengono recuperate tutte le risorse che costuiscono un match tramite il metodo "sets.retrieveCountry", e l'intero insieme di risultati viene confrontato con la stringa di partenza per trovare quella con la percentuale di corrispondenza più alta tramite la funzione "conflictResolver". Questa funzione, il cui funzionamento verrà trattato più in dettaglio alla fine questa sezione ha il ruolo di trovare la risorsa più adatta ad identificare la stringa passata in input a findCountry iterando tra i risultati restituiti dalla ricerca sui datasets. In caso una risorsa venga ritenuta valida viene restituito il suo indice interno

all'array delle risorse risultato della ricerca suddetta, in caso contrario viene restituito None.
Se un indice è stato restituito da conflictResolver, viene recuperato l'URI della risorsa corrispondende e passato alla funzione "sets.retrieveLabelsFromCountryUri" che ritorna l'intero insieme di labels relativo a quella risorsa e viene restituito il controllo a labelAnalyzer con l'oggetto comprensivo di URI e labels come risultato.
In caso le ricerche sui datasets non producano risultati un ulteriore tentativo di recupero del Paese viene fatto cercando nella lista di country mantenuta in un file json consultabile tramite il metodo "checkForUnresolvedCountries" il quale si serve dei metodi "checkForState" e "retrieveCountryInformations", definiti nel modulo "countryChecker.py", per verificare la presenza nella lista di un item che corrisponda a quello cercato e ritornare un insieme di traduzioni note per tale elemento.
Questo passaggio termina l'esecuzione di findCountry.
Ritornato il controllo a labelAnalzer se una parte della label in input ha prodotto un risultato positivo, l'oggetto risultante viene inserito nella struttura "affiliationItem" e la parte di label che ha prodotto il risultato viene eliminata da affiliationParts in modo da non venire considerata nelle future analisi.

findCity

Questa funzione prende in input una stringa (epurata da numeri o sigle in uppercase tramite il metodo "findAddress") corrispondete ad una parte di label da analizzare e l'URI del Paese, nel caso sia stato trovato nel passo precendete, altrimenti None.
Il primo passo è quindi distinguere se un URI di un country sia stato effettivamente passato; in caso affermativo vengono interrogati i datasets per cercare una città fornendo anche lo Stato in cui si trova ("sets.cityKnowingCountry"), in modo da restringere la ricerca ed ottimizzare i tempi e la qualità della ricerca stessa.

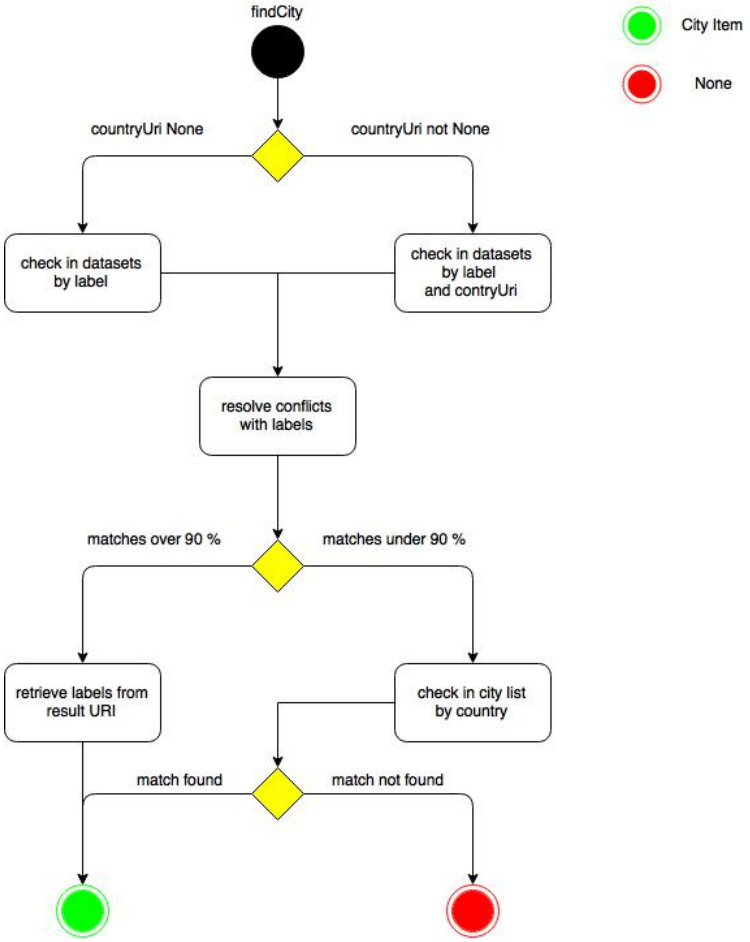

Figura 4.4: stati in findCity

Se il parametro countryURI è None viene fatta una ricerca estesa sui datasets nel tentativo di trovare una città in un qualsiasi Stato che dia un match con la stringa fornita in input ("sets.city").
In entrambi i casi, se la ricerca nei datasets ha prodotto risultati, essi vengono valutati tramite conflictResolver e, analogamente a quanto esposto in findCountry, se un match valido viene trovato, l'indice della risorsa nell'array dei risultati dei datasets viene usato per recuperare l'URI da passare al metodo "sets.retrieveCityLabels" per restituire l'oggetto contenente URI e labels a labelAnalyzer.

In caso findCity restituisca None, se un country era stato trovato al passo precedente, un ulteriore tentativo di riconoscere la città viene operato cercando in una lista di città indicizzate per Paese consultabile tramite la funzione "checkForUnresolvedCitiesGivingCountry"; tale metodo, definito nel modulo "cityChecker" prende in input una stringa e il nome di un Paese in inglese e restuisce la città trovata o None.
Il controllo viene quindi restituito a labelAnalyzer e, se una parte ha prodotto risultati, essa viene eliminata da affiliationParts in modo da non venire considerata nei successivi passi.

findOrganization

Si passa ora alla ricerca di organizzazioni fornendo una stringa in ingresso alla funzione findOrganization.
Il primo passo operato corrisponde alla rimozione di elementi in parentesi per eliminare possibili informazioni che potrebbero rendere meno preciso il match con labels di risorse nei dataset. Questa operazione viene effetuata dal metodo "removeParentheses".
Successivamente si interrogano i datasets ("sets.searchForOrganizationFromLabel") circa la presenza di risorse che matchino con la stringa data in input; in caso affermativo si passano le risorse risultanti a conflictResolver e viene ritornata la risorsa ritenuta valida o None analogamente a quanto descritto nelle precedenti sezioni.
In caso la sets.searchForOrganizationFromLabel non abbia restituito risorse, la stringa passata in input viene "tokenizzata" eliminando atricoli traminte il metodo "tokenizeString" e la lista di token risultante viene passata alla funzione "sets.searchForOrganizationFromLabelExtended" che effettua una ricerca più estesa rispetto alla precedente, cercando risorse le cui labels contengano tutte i token passati senza tenere conto dell'ordine.
In caso vengano trovati riscontri si segue la solita routine passando da conflictResolver per trovare una risorsa valida.

Il risultato di findOrganization consiste in un oggetto contente URI della risorsa trovata, labels e il tipo di risorsa; Per tipo si intende se tale risorsa identifica un'organizzazione o un'unità.
Viene inoltre assegnato un valore di confidence pari ad 1 in quanto per risorse verificate tramite i datasets si ha una ragionevole sicurezza che le informazioni rilevate siano corrette.
Il controllo viene quindi restituito a labelAnalyzer elimanindo l'eventuale parte che ha prodotto risultati da affialiationParts.

purifyAddesses

Il compito di questo passo è quello di eliminare da affiliationParts le parti relative ad indirizzi fisici dove le organizzazioni si trovano.
La funzione purifyAddress prende in input le parti rimaste della label e lascia inalterate le parti in cui vi siano riscontri con le parole chiave definite per organizzazioni e unità mantenute nelle liste di stringhe "ORG_NAMES" e "SUBORG_NAMES".
Vengono invece eliminate le parti in cui vi siano numeri e riscontri con parole chiave riguardanti strade o edifici salvate nell'array "ADDRESSES_NAMES".
Sono anche previsti controlli per eliminare parti in cui vi siano riscontri con suffissi giapponesi per luoghi, mantenuti in "PLACES_SUFFIX".
Un'ulteriore richiesta ai datasets viene operata per eliminare qualsiasi luogo noto tramite la funzione "sets.places".

labelAnalyzerExtended

Nel caso la label in ingresso di labelAnalyzer non sia stata completamente processata e parti siano ancora rimaste non assegnate il controllo passa a "labelAnalyzerExtended".
In questa funzione vengono messe in pratica le regole di fallback esposte nel predente capitolo.

Questo metodo prende in input un array contentente le parti rimaste e restituisce un oggetto contenente i dati definitivi dell'analisi.
La prima distinzione nell'esecuzione di questo metodo viene fatta in caso un'organizzazione sia stata trovata nei passi precendenti. In tal caso il ruolo di labelAnalyzerExtended si riduce ad iterare tra le parti rimaste creando oggetti contenenti la traduzione in inglese della parte considerata (usando le funzioni del modulo "translator.py") un insieme di label contenente la label in lingua originale più quella tradotta in inglese (in caso la label sia già in inglese viene mantenuta inalterata) e il tipo di organizzazione che viene settato a "dep" (indicando che la parte di label rappresenta un dipartimento dell'organizzazione trovata nei passi precenti).
Nel caso opposto si itera tra le parti rimaste gestendo nel modo appena descritto tutte le parti della label che abbiano dei riscontri con le keywords relative a dipartimenti, contenute nell'array "SUBORG_NAMES".
Anche le parti rimaste vengono gestite allo stesso modo fatta eccezione per l'ultima il cui tipo viene settato a "org", indicando quindi che quella parte si riferisce all'organizzazione principale, secondo la regola di fallback che definisce la parte della labels più a destra come quella relativa a questo tipo di informazione.
Il valore di confidence assegnato alle risorse recuperate tramite queste analisi è di 0.5 che andrà a fare media con lo stesso valore per organizzazioni trovate nelle fasi precedenti al momento della serializzazione delle informazioni.
Terminata quest'ultima fase l'analisi si ritiene conclusa e l'oggetto comprensivo dei dati estratti viene restituito ad affiliationExtractor per poter procedere alla serializzazione.
Verranno esposti alcuni metodi di utilità di cui ci si serve per completare le analisi in modo corretto.

conflictResolver

Questa funzione si occupa di iterare tra un array di labels delle risorse restituite dalle ricerche sui datasets ed usare la distanza di Levenshtein per calcolare la percentuale di matching con la stringa in analisi. Si noti come solo match più alti del 90% siano considerati validi.
Il metodo restituisce None nel caso nessun match superiore al 90% sia stato trovato e nel caso in cui l'array di risorse da confrontare abbia un numero di elementi superiori alla constante "LIMIT_OF_MATCHES"; tale costante è stata introdotta per definire un limite al numero di riscontri ritenuti accettabili in fase di matching, in quanto in caso di un numero molto alto di risultati dai datasets la stringa potrebbe essere troppo generica per essere associata correttamente ad una risorsa contenuta nei datasets.
In caso contrario restituisce l'indice della risorsa ritenuta essere più vicina alla stinga passata in input.
Per calcolare la percentuale di matching tra le labels restituite dalle ricerche sui datasets e la stringa in analisi ci si serve del metodo stringMatcher.

stringMatcher

Questo metodo implementa la distanza di Levenshtein per determinare la percentuale di match tra le due stringhe passate in input.
Ogni parola della prima stringa in ingresso viene confrontata con ogni parola contenuta nella seconda stringa e quest'ultima viene riordinata a per essere il più simile possibile alla prima.
Completato questo passo viene calcolata la distanza tra le due stringhe non tenendo conto di eventuali spazi e tale numero viene tradotto nella percentuale che viene ritornata a conflictResolver.

4.4 Come comunicare con i dataset: sets.py

In questa sezione verrà descritto il modulo "sets.py" il quale è predisposto a mascherare le comunicazioni tra analyzer.py e i datasets interrogabili.
La logica di questo modulo consiste in un insieme di funzioni che prendono in input una stringa o una lista di stringhe e generano le query sparql adeguate per completare e richieste.
Le query sparql sono definite nel modulo "queries.py" che definisci una serie di metodi che ritornano una query sparql inserendone nel testo la stringa passata in input.
Come mostrato in figura 4.7 funzioni chiamate da analzer.py controllano prima sul dataset mantenuto localmente la presenza della risorsa cercata e, in caso essa venga trovata essa viene restituita immediatamente come risultato. In caso contrario si interroga dbpedia per provare a completare la richiesta.
Questo tipo di approccio rende analyzer.py indipendente dal dataset usato per completare la ricerca rendendo quindi possibile aggiungerne di nuovi nella pipeline restituendo un risultato compatibile con le necessità di utilizzo di analyzer.py.
Passiamo ora ad elencare le librerie aggiuntive utili al funzionamento di sets.py e la sua implementazione.

4.4.1 Liberie Aggiuntive

SPARQLWrapper

La libreria SPARQLWrapper definisce metodi per interrogare datasets RDF che definiscano un endpoint SPARQL.
I datasets interrogati da AffiliationExtractor sono un dataset locale che va definito sulla propria macchina e il dataset di Dbpedia interrogabile all'indirizzo "http://dbpedia.org/sparql". I risultati delle interrogazioni sono codificate in JSON e vengono convertite dal metodo "convert" che trasforma il JSON

in risultante in un dizionare consultabile dalle funzioni Python.

4.4.2 Implementazione

Le funzioni in sets.py si dividono in tre tipi principali: funzioni di richiesta di esistenza di un tipo di risorse, funzioni che recuperano tutte le risorse che fanno match con quella richiesta (divise a loro volta in semplici ed "extended") e funzioni che ritornano informazioni specifiche di una risorsa di cui si conosce l'URI.
A loro volta queste funzioni si distinguono in metodi rivolti a cercare nel dataset locale (preceduti dal termine "local" nella firma) e metodi che si occupano di fare richieste a Dbpedia, ma l'unica differenza tra queste due categorie di metodi sta nelle diverse query che devono essere generate in quanto i due datasets non condividono le stesse ontologie. Si noti come i metodi "local" vengano chiamati all'interno delle loro controparti principali per cercare le risorse sempre prima in locale (figura 4.7).
Di seguito verranno trattate queste diverse categorie fornendo un esempio per ognuna e verrà infine descritto il modulo queries.py per fornire esempi delle query SPARQL utilizzate nelle interrogazioni ai datasets.

Richieste di Esistenza

Fanno parte di questa categoria tutte le funzioni che restituiscono booleani come risultato. Il loro compito è verificare l'esistenza di risorse sui datasets le cui labels costituiscano un match con le stringhe passate in input.
Un esempio di tali funzioni è "searchForOrganizationFromLabel" che chiede l'esistenza di un'organizzazione che tra le sue labels abbia una corrispondenza con la stringa passata in input.

Richiesta di Tutte le risorse che matchino con la stringa in input

Fanno parte di questa categoria le funzioni che ritornano un oggetto comprensivo di tutti i possibili match con la stringa data in input.
Un esempio è rappresentato da "retrieveOrganization" che viene usata per ritornare tutte le organizzazioni le cui labels costituiscano un match con la stringa in input.

Richiesta di labels con URI in input

Fanno parte di questa categoria le funzioni che ritornano l'insieme di labels associate alla risorsa identificata dall'URI dato in input.
Un esempio di tali funzioni è rappresentato da "retrieveOrganizationLabels".

Versioni Extended

Le versioni delle funzioni di ricerca con suffisso "Extended" ricevono in input una lista di tokens invece di una semplice stringa e effettuano la ricerca richiedendo la versione extended della query utilizzata nella versione semplice della stessa funzione.

queries.py

Questo modulo si occupa di generare le query SPARQL utili ad interrogare oppurtunamente i datasets. Le query sono restituite da funzioni che prendono in input stringhe o liste di torken (versioni "Extended") e ritornano le query compilate opportunamente.
Ogni metodo ha firma uguale alla funzione in sets.py che lo chiama con l'aggiunta del suffisso "Query".
Un esempio è dato dal metodo "organizationQuery" (figura 4.8), chiamato dalla funzione in sets searchForOrganizationFromLabel, che verifica l'esisten-

za di una risorsa tramite il comando SPARQL "ask" e sostituice la stringa passata in input nella regex del comando filter.
Per quanto riguarda le ricerche extended l'esempio in figura 4.9, relativo al metodo "organizationQueryExtended",si può notare come questo tipo di ricerche siano completate inserendo un numero di condizioni contains come fitri pari ai token passati in input.

4.5 Serializzare i risultati: serializer.py

Questo modulo si occupa della serializzazione dei dati ricavati dall'analisi nel dataset locale secondo il modello esposto nel capitolo precedente.
Le diverse informazioni su paesi e organizzazioni vengono serializzate i grafi diversi per poi effettuare un merge tra gli stessi compiendo l'update che aggiorna il dataset locale con nuovi dati.

4.5.1 Liberie Aggiuntive

rdflib

Questa libreria definisce una serie di metodi utili per seriliazzare le informazioni seguendo gli standard imposti nel linguaggio RDF.
Fanno parte di questa libreria le funzioni "Graph", utile a creare grafi per collegare le informazioni, "URIRef", che serve a restituire URI validi e "Namespace", per definire i prefissi delle ontologie usate.

SPARQLWrapper

Questa libreria, già descritta nella precedente sezione, viene utilizzata in questo modulo per inviare le informazioni serializzate al dataset locale tra-

mite il metodo update.

4.5.2 Implementazione

Il grafico in figura 4.10 mostra come gli stati previsti nella serializzazione siano essenzialmente due: “serializeCountry” e “serializeOrganizations”.
Di questi due passi solo il secondo è da ritenersi obbligatorio in quanto la serilize viene invocata da affiliationExtractor.py solo nel caso in cui informazioni su organizzazioni siano state effettivamente trovate.

serializeCountry

La serializeCountry prende in input un oggetto contenente l’URI della risorsa ottenuta dai datasets in fase di analisi e l’insieme delle sue labels. Dall’URI viene separata la parte finale (la parte dopo l’ultimo “/”) in modo da ricavare il nome della risorsa da concatenare al namespace relativo al modello del dataset locale.
Nel caso il country fosse stato trovato tramite la lista di paesi di cui si è parlato nella precedente sezione, nel campo relativo all’URI si troverà il nome del paese in inglese, il quale verrà convertito in URI valido tramite la funzione “normalizeURI”, per cui ci sarà una menzione alla fine di questa sezione per spiegare le normalizzazioni operata in fase di creazione degli URI.
Successivamente alla creazione dell’URI ad esso viene assegnata la classe “semtype:Country” usando la property “rdf:type”.
Si prosegue associando le labels tramite la proprietà “rdfs:label” ed infine, se il paese era stato riconosciuto tramite Dbpedia, viene associata l’informazione relativa all’URI della stessa risorsa su tale dataset tramite la proprietà “semprop:dbpediaReferenceTO”.
Completata questa operazione viene restituito l’URI del country appena serializzato alla funzione serialize che lo passerà alla serializeOrganization in

modo da poter associare tale informazione alle organizzazioni da serilizzare.

serializeOrganizations

la serializeOrganization viene chiamata per serializzare le informazioni sulle organizzazioni trovate, occupandosi anche di collegarle al ricercatore corrispondente, già presente nel dataset del progetto Semantic Lancet.
I dati in input sono quindi le informazioni sulle organizzazioni recuperate in fase di analisi, l'URI del paese serializzato al passo precente e l'URI del RoleInTime dell'autore, a cui l'affiliazione deve essere collegata nel rispetto del modello proposto nell'ontologia PRO, recuperato passando il nome trovato sul paper al servizio "retrieveAuthorRoleInTimeURI".
Il primo passo consiste nel separare le informazioni sull'organizzazione principale da quelle su eventuali dipartimenti, ponedo la prima nella variabile "organizzazion" e i secondi in un array chiamato "units".
Successivamente si passa alla creazione dell'URI dell'organizzazione nelle modalità del tutto simili a quelle esposte nella sezione precedente.
Si passa poi all'associazione della classe "org:Organization", definita nell'Organization Ontology, tramite la property "rdf:type", proseguendo ad associare le labels trovate tramite la proprietà "rdfs:label"; successivamente l'organizzazione viene collegata al proprio paese tramite la proprietà "semprop:country", dove il range consiste nell'URI relativo al country passato in input.
Il passo successivo consiste nel ripetere le operazioni per gli eventuali dipartimenti, iterando tra gli item nella variabile "units"; particolari menzioni vanno fatte per la creazione dell'URI di un dipartimento, a cui viene concatenato il nome dell'organizzazione di cui fa parte per evitare ambiguità nel dataset, e per l'associazione tra un'unità e l'organizzazione di cui fa parte tramite le proprietà "org:unitOf" (da dipartimento a organizzazione) e "org:hasUnit" (da organizzazione a dipartimento), entrambe definite in Organization Ontology.

Viene infine collegata l'organizzazione al RoleInTime passato (o il dipartimento se presente) attraverso la proprietà "pro:relatesToOrganization" definita nell'ontologia PRO.
Le informazioni così serializzate vengono quindi inviate al dataset locale tramite l'update definita nel modulo SPARQLWrapper, completando un ciclo di esecuzione di AffiliationExtractor.

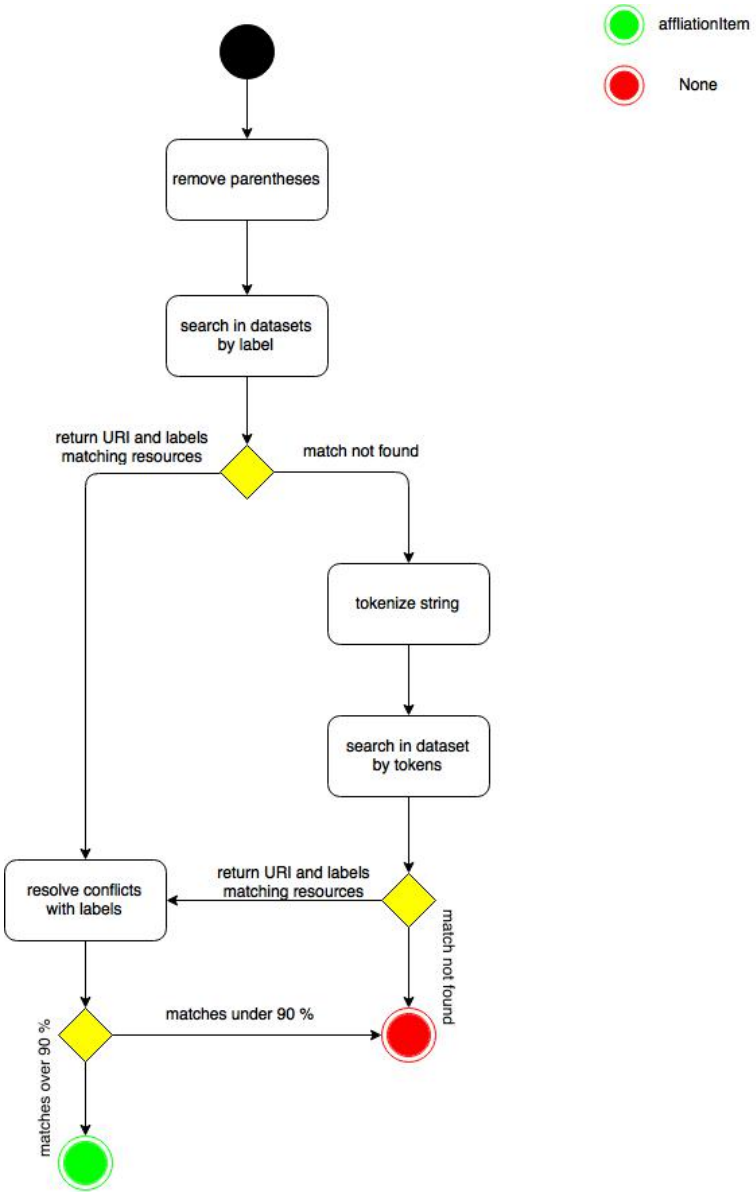

Figura 4.5: stati in findOrganization

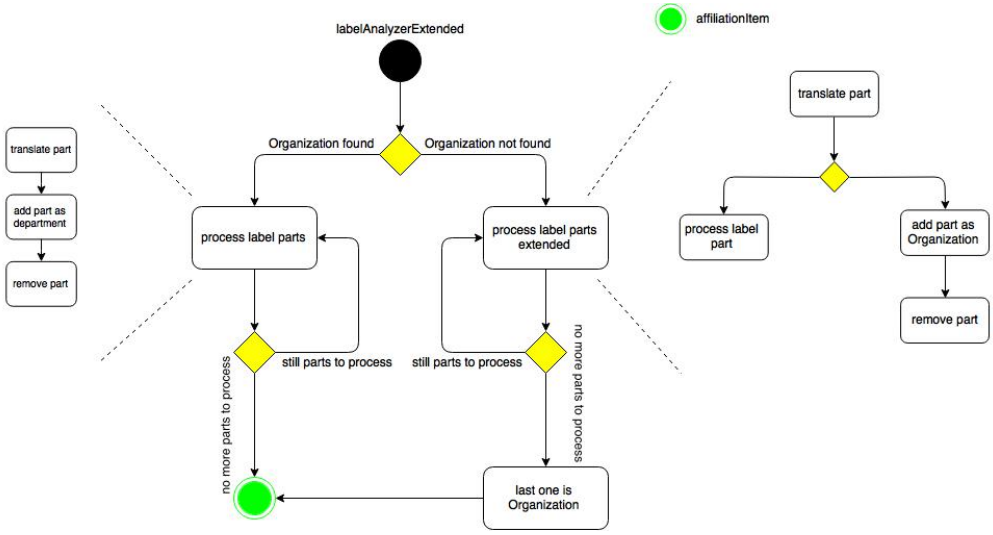

Figura 4.6: stati in labelAnalyzerExtended

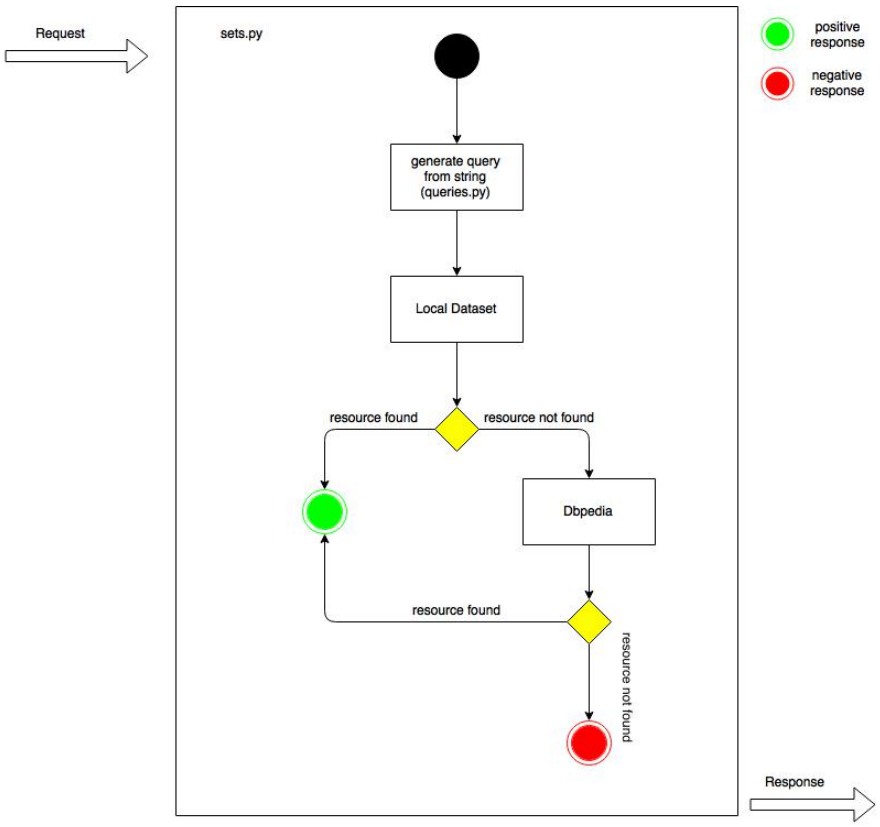

Figura 4.7: stati in sets.py

```
def organizationQuery(stringToMatch):
    return("""
            ask
            where {
            ?org a dbpedia-owl:Organisation;
                 rdfs:label ?label.

            FILTER regex(str(?label), \"""" + stringToMatch + """\", "i")
            }

            """)
```

Figura 4.8: esempio query SPARQL

```
def organizationExtendedQuery(stringsToMatch) :
    filterString = ''
    for string in stringsToMatch :
        filterString += "filter contains(?label, \"" + string + "\")."

    return ("""
            ask
            where {
                ?org a dbpedia-owl:Organisation;
                     rdfs:label ?label.

                """ + filterString + """
            }

            """)
```

Figura 4.9: esempio query extended SPARQL

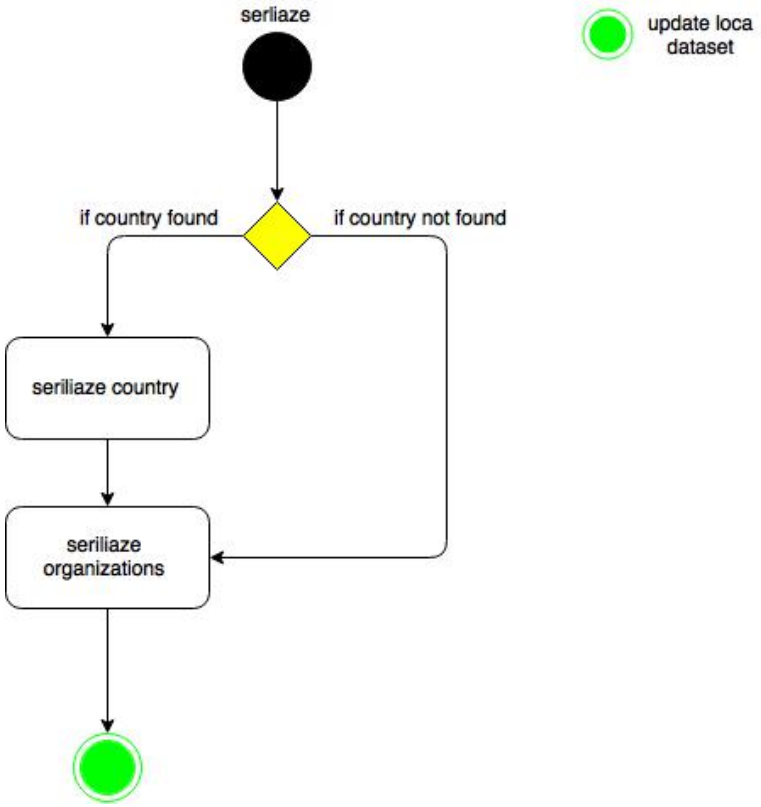

Figura 4.10: esempio stati in serialize

Capitolo 5

Risultati ottenuti: metodi di valutazione e problemi aperti

In questo capitolo verrà trattata la valutazione per misurare l'efficacia di AffiliationExtractor.
Il capitolo si distinguerà in due sezioni: Metodi di Evaluation e Analisi dei risultati.
La prima sezione si concentrerà nell'analisi dei mezzi usati per definire le prestazioni del tool sviluppato, concentrandosi anche sullo script preposto per calcolare le cifre che ne rappresentano l'efficacia.
Nella seconda sezione ci si soffermerà sull'analisi vera e propria dei risultati, spiegando perchè sono da ritenersi promettenti e sui punti di forza del tool ma anche sulle debolezze che ancora fanno parte della sua implementazione.

5.1 Metodo di Evaluation

Il metodo scelto per la valutazione delle prestazioni di AffiliationExtractor è basato sul calcolo dei valori di precision e recall ottenuti dal confronto

tra risultati costruiti a mano (Gold Standard) con i risultati effettivamente ottenuti dopo l'esecuzione del tool su tutto il dataset di partenza.

5.1.1 Precision & Recall

Precisione e recupero, o richiamo (in inglese precision e recall) sono due comuni classificazioni statistiche, utilizzate in diversi ambiti del sapere, come per es. l'information retrieval. La precisione può essere vista come una misura di esattezza o fedeltà, mentre il recupero à una misura di completezza. Nell'Information Retrieval, la precisione è definita come il numero di documenti attinenti recuperati da una ricerca diviso il numero totale di documenti recuperati dalla stessa ricerca, e il recupero è definito come il numero di documenti attinenti recuperati da una ricerca diviso il numero totale di documenti attinenti esistenti (che dovrebbe essere stato recuperato).
Altrimenti detto, la precisione è la frazione di documenti attinenti che sono stati trovati, mentre il recupero è la frazione di documenti trovati che sono attinenti. Dalla definizione, è possibile intuire che precisione e recupero sono grandezze inversamente proporzionali: maggiore è la precisione in una ricerca, minore sarà il recupero, e viceversa. Ne consegue dunque che motori di ricerca "perfetti", cioè che ritrovino tutti e soli documenti pertinenti ad una particolare ricerca, non sono possibili.
Nell'Information Retrieval, un valore di precisione di 1.0 significa che ogni risultato recuperato da una ricerca è attinente mentre un valore di recupero pari a 1.0 significa che tutti i documenti attinenti sono stati recuperati dalla ricerca.
In un processo di classificazione, un valore di precisione di 1.0 per la classe C significa che ogni oggetto che è stato etichettato come appartenente alla classe C vi appartiene davvero (ma non dice niente sul numero di elementi della classe C che non sono stati etichettati correttamente) mentre un valore di recupero pari ad 1.0 significa che ogni oggetto della classe C è stato

etichettato come appartenente ad essa.

$$precision = \frac{|\{documenti\ attinenti\} \cap \{documenti\ recuperati\}|}{|\{documenti\ recuperati\}|}$$

$$recall = \frac{|\{documenti\ attinenti\} \cap \{documenti\ recuperati\}|}{|\{documenti\ attinenti\}|}$$

5.1.2 Gold Standard

Per Gold Standard si intende l'insieme dei risultati attesi, creati a mano dall'utente come metro di paragone per i risultati effettivamente calcolati.
Nello sviluppo del tool il metro di paragone per misurare la validità dei risultati ottenuti è rappresentato da un json per ogni articolo, comprensivo delle informazioni relative alle organizzazioni, eventuali unità e del paese di appartenenza a cui detto articolo è affiliato.
Si ricordi che, nel rispetto dell'ontologia PRO, sono gli autori dei papers a figurare come affiliati ad una data organizzazione, quindi, per avere la possibilità di quantificare correttamente le informazioni ottenute, si è reso necessario passare attraverso una fase intermedia della serializzazione dei dati, definendo una proprietà diretta tra articolo e affiliazioni.
Presa una label sono stati incapsulati i dati in tre campi principali: organization, unit, country.
In caso un'organizzazione non avesse un'unità, questo implicherebbe la non inclusione della proprietà corrispondente nell'oggetto json relativo ad essa, similmente per il paese.
Nel caso in cui un'organizzazione avesse più dipartimenti ad essa affiliati, essi sono stati collegati singolarmente ad essa facendo risultare l'articolo, e conseguentemente l'autore del papers, affiliato ad ognuna delle unità presenti.
Definito tale gold standard si è passato al confronto dei risultati tramite lo script "evaluator.py" che si occupa di richiedere le risorse dal dataset locale compilato durante l'esecuzione di AffiliationExtractor tramite query sparql e di produrre strutture json paragonabili a quelle descritte in precedenza.

5.1.3 Metodo di valutazione

Il metodo descritto di seguito è stato applicato per ognuno dei 181 articoli costituenti il dataset di partenza che, grazie alla molteplicità di casi nei riferimenti alle loro affiliazioni, rappresentano un campione accettabile per misurare l'efficacia di AffiliationExtractor come soluzione alle problematiche esposte nei precedenti capitoli.
Preso un articolo le cui affiliazioni risultino essere quelle in tabella 5.1 si è prodotto il file json relativo al gold standard riconoscendo manualmente la natura delle informazioni proposte andando a compilare tale json come in figura A.1.

Department of Computer Science, University of Manchester, Oxford Road, Manchester M13 9PL, UK
Bell Labs Research, Lucent Technologies, 600 Mountain Avenue, Murray Hill, NJ 07974, USA
AI Department, Vrije Universiteit Amsterdam, de Boelelaan 1081a, 1081HV Amsterdam, The Netherlands

Tabella 5.1: esempio label di affiliazioni in un articolo

Fatto ciò si lancia AffiliationExtractor con in input tale articolo e, una volta completata l'esecuzione del tool, si lancia l'evaluator.py che genererà un json come quello in figura A.2 analogo a quello del gold standard ma con la differenza che l'oggetto relativo ai risultati comprenderà tutte le labels relative alle risorse disambiguate tramite Dbpedia.
Si confronteranno ora i campi uno per uno per trovare le corrispondenze con tra gold standard e results dando punteggio 1 a tutti gli item interni ai json che abbiano piena corrispondenza e 0 altrimenti.
Viene poi operata la media tra le somme dei punteggi fratto, il numero di

items nel gold standard per quanto riguarda il valore di recall e il numero di items nel json dei risultati per calcolare il valore di precision.
In ultimo si sommano tutti i valori di precision e recall e si dividono per il numero di articoli considerati per calcolare i valori globali che forniranno la misura di valutazione complessiva dell'efficienza del tool.

5.1.4 Normalizzazioni nei confronti

Normalizzazioni
Cancellazione Articoli (The, of)
Confronti fatti in lower case

Tabella 5.2: tabella normalizzazioni nei confronti

Particolare menzione va fatta per le normalizzazioni sulle labels (tabella 5.2) per rendere i confronti tra gli elementi del gold standard e le loro controparti prodotte dallo script di evaluation descritto nella sotto sezione di seguito
Si è scelto di non considerare articoli e di non eseguire match che tengano conto del "case" riportando tutto in minuscolo.
Si noti che queste modifiche non minano la validità dei test operati ma sono rilassamenti necessari per consentire all'evaluator di non scartare casi chiaramente validi (tabella 5.3).

Si è invece deciso di non normalizzare dati in cui siano presenti divisioni con " - " lasciando quindi fallire il confronto in quei casi nonostante i dati risultino comunque esatti ad una verifica manuale, in quanto non sempre tale separatore viene sostituito nello stesso modo(tabella 5.4).

Gold Standard	Results
The Open University	Open University
The Netherlands	Netherlands
FORTH	The Forth
Technical University Berlin	Technical University of Berlin

Tabella 5.3: casi di normalizzazione

Gold Standard	Results
University of California-Berkeley	University of California, Berkeley
University of Koblenz-Landau	University of Koblenz and Landau

Tabella 5.4: casi di normalizzazione

5.1.5 evaluator.py: script di costruzione dei risultati

Lo script "evaluator.py" distingue la sua esecuzione in due momenti principali: recupero delle informazioni e confronto delle stesse con quelle riportate nel gold standard.
La prima fase comincia con il recupero di tutti gli URI degli articoli serializzati nel triple store locale.
La funzione "articleCall" interroga il triple store locale con la query denominata "articleQuery" riportanto gli URI degli articoli analizzati nella variabile "articles".
Per ognuno di essi vengono richieste le risorse affiliate ritornando una struttura comprensiva degli URI delle organizzazioni, eventuali unità associate ad esse e Paesi di appartenena. Si occupa di ciò il metodo "affiliationCall" che usa la query "affiliationQuery" per interrogare il dataset.
L'intero item contenente le affiliazioni viene a sua volta passato alla funzione "createResultsGoldStandard" che si occupa sia di recuperare le labels associate alle risorse recuperate che di creare un oggetto json relativo alle

affiliazioni dell'articolo in questione da paragonare con la sua controparte rappresentata dal gold standard.
Per il recupero delle labels viene usato il metodo "constructResults" che si occupa anche di operare alcune normalizzazioni importanti per rendere possibile la corretta valutazione dei risultati. È fondamentale il recupero di tutte le labels associate ad una particolare risorsa in quanto nel caso tale risorsa sia stata correttamente disambiguata tramite i metodi descritti nei precedenti capitoli non vi è garanzia che la label presente nel gold standard sia nella stessa lingua della label principale della risorsa (solitamente presente in inglese).
Costruiti gli oggetti e salvati in file nominati secondo l'articolo a cui fanno riferimento la prima fase dell'evaluator si può ritenere conclusa.
Si passa ora al confronto dei json facenti parte del gold standard con quelli appena creati.
Iterando su ognuno dei file json costituenti il gold standard si recupera il corrispettivo file contente i risultati per lo stesso articolo e, una volta deserializzati gli oggetti json in dizionari python denominati rispettivamente "goldItem" e "resItem", si passano alla funzione "evaluate" che, come il nome lascia trasparire, costituisce il punto nevralgico dello script che si sta descrivendo.
Il metodo evaluate si occupa di confrontare campo per campo i dizionari passati dando un punteggio di 1 nel caso tutti i campi corrispondano e 0 nel caso vi sia una qualsiasi imprecisione.
Si parte recuperando il campo organization del golditem e confrontandolo con l'analogo campo presente nel resItem; organization è l'unica property che si è certi sia presente in entrambi gli item in quanto le risorse non vengono serializzate se un'organizzazione non sia stata trovata.
Passato il primo controllo si verifica che nel goldItem sia presente un campo "unit"; se esso è presente si controlla che non sia presente nel resItem in modo da non ultimare i controlli successivi vista l'incongruenza tra le strutture comparate.

Lo stesso avviene in caso una unit sia presente anche nel resItem ma le due units non siano uguali.
In caso una unit non sia presente nel goldItem si operano i controlli opposti per cercare di uscire dall'iterazione di confronto prima Analogamente si passa al controllo sul country che segue la stessa routine vista per le units.
Se tutti i controlli vengono eseguiti senza successo gli oggetti si ritengo equivalenti e viene aumentata la variabile "matchingItems" per che rappresenterà le intersezioni dagli insiemi rappresentati dal gold standard e dai risulati.
Una volta ultimati i controlli per tutti gli item contenuti nel gold standard di quel particolare articolo vengono calcolati i valori di precision e recall locali all'articolo preso in considerazione seguendo le formule esposte nelle sezioni precedenti e vengono aggiunti alle variabili di precision e recall globali ("globalPrecision" e "globalRecall") che verranno divise per il numeno totale degli articoli in modo da avere questi dati riferiti all'intero dataset di partenza.
Nello specifico la variabile matchingItems corrisponderà al numeratore di entrambe le formule e per "documenti attinenti" e "documenti recuperati" si intendono rispettivamente il numero di item nel gold standard di un articolo e il numero di item nell'oggetto risultato dell'esecuzione di AffiliationExtractor.

5.2 Analisi risultati

In questa sezione si passerà all'analisi dei risultati ottenuti fornendo in principio i valori di precision e recall complessivi dei 181 articoli considerati in questo caso di studio; successivamente verranno esposti valori locali relativi a specifiche porzioni del dataset di partenza ottenute dividendo gli articoli che lo compongono in diversi cluster in base alla decima cifra dell'identificativo dell'articolo stesso (es. 1-s2.0-S1570826805000041 fa parte della sezione 1-s2.0-S1570826805*, tale sezione verra chiamata con il solo numero identificativo, 5, per semplificare la lettura).
In questo modo è possibile soffermarsi su alcuni dei problemi noti che non hanno portato alla corretta classificazione delle affiliazioni, ottenendo una

più completa panoramica dell'efficacia di AffiliationExtractor.

Precision & Recall globali

Di seguito sono riportati i valori di precision e recall su un bacino di 181 articoli costituenti il dataset di partenza preso come input per i test di AffiliationExtractor.
A prima vista tali dati possono sembrare non pienamente soddisfacenti (tabella 5.5) ma è doveroso riportare come gran parte delle difficoltà nella valutazione del tool sono state riscontrate in una sezione critica (sezione 5) per via di dati di partenza compromessi; si può infatti apprezzare come mettendo da parte tale sezione i risultati risultino decisamente più promettenti (tabella 5.6).
I principali ostacoli verranno affrontati nella sezione seguente riguardante i cluster a cui si è fatto riferimento precendemente.

N° Articoli	Precision	Recall
181	0.79410755055	0.81687248493

Tabella 5.5: valori di precision & recall globali

N° Articoli	Precision	Recall
148	0.8741860817	0.89232766339

Tabella 5.6: valori di precision & recall senza sezione 5

5.2.1 Sezioni di Valutazione

Le seguenti sezioni forniranno un quadro rappresentativo dell'efficienza di AffiliationExtractor limitate ai cluster definiti come detto nell'introduzione.

Le sezioni partono dalla sezione 3, in quanto non vi sono articoli con identificativi 1-s2.0-S1570826801* o 1-s2.0-S1570826802* nel journal considerato.

Sezione 3

N° Articoli	Precision	Recall
16	0.925347222222	0.940972222222

Tabella 5.7: valori di precision & recall: sezione 3

I valori di precision e recall in tabella 5.7 sono ottimi in questa sezioni, l'unico neo è rappresentato dalla presenza di una label il cui confronto è fallito anche se la risorse sono state effettivamente riconosciute (University of California, Berkeley vs. University of California-Berkeley) in quanto, come detto si è scelto di non normalizzare i "-".

Sezione 4

N° Articoli	Precision	Recall
21	0.848412698413	0.857142857143

Tabella 5.8: valori di precision & recall: sezione 4

I valori di precision e recall sono ottimi in tabella 5.8 in questa sezione, l'unico neo è rappresentato dalla presenza di una label il cui confronto è fallito anche se la risorse sono state effettivamente riconosciute (University of Koblenz-Landau vs University of Koblenz and Landau) in quanto, come detto si è scelto di non normalizzare i "-".
In questo settore incontriamo un problema ricorrente in tutte le prossime sezioni; in presenza di affiliazioni molto strutturate, come ad esempio quella

in tabella 5.9, le regole di fallback, così come sono state pensate attualmente, risultano ancora inefficienti per correttamente riconoscere tutte le parti presenti.
Il termine "Vassilika Vouton" sfugge alla rimozione degli indirizzi in quanto non contiene parole chiave particolari e, nemmeno una richiesta a Dbpedia riesce ad identificarlo come luogo.
Per questa ragione viene listato come un dipartimento dell'organizzazione "FORTH" facendo fallire i confronti relativi ad essa.

`Institute of Computer Science, FORTH, Vassilika Vouton, P.O. Box 1385, GR 71110 Heraklion, Greece`

Tabella 5.9: label problematica

Sezione 5

N° Articoli	Precision	Recall
33	0.313636363636	0.364141414141

Tabella 5.10: valori di precision & recall: sezione 5

Questa sezione va analizzata a parte, visti i risultati in tabella 5.10 in forte contrasto con il resto dei cluster considerati.
La motivazione di tale discordanza è dovuta alla presenza di dati sulle affiliazioni errati nei sorgenti di 11 degli articoli presenti in tale sezione.
Il problema riguarda due labels in particolare, le quali fanno riferimento alle università di Toronto e Creta che sono riportate in questo modo:

- Department of Computer ScienceUniversity of TorontoToronto, Canada

- Department of Computer ScienceUniversity of CreteandInstitute of Computer ScienceFoundation for Research and Technology (FORTH) Crete, Greece

Si può notare come la premessa circa il funzionamento di AffiliationExtractor, riguardante la divisione delle labels in parti, risulti inefficace in casi del genere e sarebbe necessario implementare una routine di gestione specifica, basata per esempio sulla divisione in presenza di una lettera maiuscola preceduta da una minuscola (supporto al "calmel case"), la quale non è stata prevista nello sviluppo del tool.

Sezione 6

N° Articoli	Precision	Recall
19	0.913157894737	0.973684210526

Tabella 5.11: valori di precision & recall: sezione 6

In questa sezione i valori in tabella 5.11 di precision e recall sono ottimi; gli unici problemi incontrati in questa sezione sono stati i riferimenti alla Cina con la sigla PR (Repubblica Popolare), che ha causato il fallimento nell'identificazione tramite Dbpedia e lista di country.

`Department of Computer Science and Technology, 10-201, East Main Building, Tsinghua University, Beijing 100084, PR China`

Tabella 5.12: label problematica

N° Articoli	Precision	Recall
19	0.925347222222	0.940972222222

Tabella 5.13: valori di precision & recall: sezione 7

Sezione 7

In questa sezione i valori in tabella 5.13 di precision e recall sono ottimi; gli unici problemi sono dati dalla presenza di una strutturazione inprevista delle informazioni geografiche, nello specifico nell'aggiunta di informazioni sul contintente (America) in alcuni riferimenti ad università negli Stati Uniti (tabella 5.14), ricadendo nel problema, già visto in altre sezioni, dell'inefficacia delle attuali regole di fallback in presenza di una complessa strutturazione delle informazioni da analizzare.

`Vulcan Inc., 505 Fifth Avenue South, Suite 900, Seattle, WA 98104, United States, America`

Tabella 5.14: label problematica

Sezione 8

N° Articoli	Precision	Recall
27	0.882716049383	0.885802469136

Tabella 5.15: valori di precision & recall: sezione 8

In questa sezione i valori in tabella 5.15 di precision e recall sono molto buoni; i problemi ricorrenti sono sempre rigurdanti l'eccessiva complessità nella strutturazione di alcune labels, in cui sono state aggiunte informazioni

circa la regione all'interno dello stato considerato (item 1 in tabella 5.16 o in cui erano presenti informazioni su indirizzi senza parole chiave riconosciute (item 2 in tabella 5.16).

`Yahoo! Research, Avinguda Diagonal 177, 8th floor, Barcelona, 08018, Catalunya, Spain`
`Diari Segre Media Group, Del Riu, 6, E-25007 Lleida, Spain`

Tabella 5.16: labels problematiche

Sezione 9

N° Articoli	Precision	Recall
28	0.721428571429	0.736904761905

Tabella 5.17: valori di precision & recall: sezione 9

In questa sezioni abbiamo valori in tabella 5.17 di precision e recall leggermente sotto la media; questo è dovuto alla presenta di labels contenti più affiliazioni nella stessa label, caso non previsto nello sviluppo del tool (tabella 5.18), oltre ai problemi già descritti in precedenti sezioni.

5.2.2 Punti di Forza

Dai dati riportati nelle sezioni precedenti si possono scorgere alcuni risultati molto promettenti per quanto riguarda il riconoscimento di moltissimi casi diversi di rappresentazione di affiliazioni.
Grazie all'utilizzo di Dbpedia e la possibilità di poter avere accesso ad un insieme di traduzioni note si è reso possibile disambiguare con buona sicurezza

Semantic Computing Research Group (SeCo), Helsinki University of Technology (TKK), Department of Media Technology, University of Helsinki, Department of Computer Science, Finland

Tabella 5.18: label problematica

circa il 90% di casi riguardanti affiliazioni, città e paesi.
Le regole introdotte sulle parole chiave aiutano ad associare a stesse affiliazioni anche labels che riportano ordini diversi nella rappresentazione di organizzazioni e propri dipartimenti.
Le norme sulla posizione, anche se non sempre precise, forniscono un'ultima possibilità di fornire associazioni tra autori e loro affiliazioni e l'assegnamento di un valore di confidence basso in casi risolti con queste regole fornisce un punto di ingresso per moduli ad estensione di questo tool che vadano a verificare ulteriormente la validità dei risultati proposti.
È inoltre da sottolineare che la natura fortemente modulare di AffiliationExtractor garantisce la possibilità di ampliare il ventaglio di riconoscimento delle affiliazioni inserendo il supporto a soluzioni complementari a quella proposta durante questa dissertazione.

5.2.3 Debolezze

È doveroso far notare i limiti che caratterizzano il tool nella sua ricerca.
Come sottolineato nei capitoli precedenti, il processo di analisi consiste nell'identificazione delle diverse parti delle labels, eliminando gradualmente le informazioni verificate per poter restare con i soli dati rigurdanti affiliazioni al momento dell'applicazione delle regole di fallback.
Ma in presenza di labels di affiliazioni particolarmente strutturate, le eu-

ristiche previste nelle regole di fallback risultano poco efficienti, causando l'impropria seriliazzazione di alcune parti non riguardanti organizzazioni.
È questo il caso di "Vassilika Vouton", quartiere della città di Heraklion in Grecia, che viene spesso serializzato come dipartimento dell'Università di Creta a causa dell'errata applicazione delle regole di fallback.
Le affiliazioni possono essere riportate usando gli acronimi delle organizzazioni a cui fanno riferimento; AffiliationExtractor non riesce a disambiguare correttamente questo tipo di riferimenti, andando a serializzare sotto URI diversi la stessa organizzazione. Un esempio è dato dall'Università del Sud California a volte riportata con il proprio acronimo (USC).
AffiliationExtractor è lontano dall'essere perfetto ma le premesse proposte sono promettenti e le possibilità di sviluppi futuri molteplici.

Conclusioni

In questa tesi si è presentato AffiliationExtractor, un tool modulare che produce un datasets RDF contenente le informazioni sulle affiliazioni di autori di pubblicazioni scientifiche estratte direttamente dal raw text XML degli articoli passati in ingresso al tool.
Esso rappresenta una possibile soluzione al problema dell'estrazione delle affiliazioni da documenti relativi a pubblicazioni di stampo scientifico, fornendo un servizio di disambiguazione tramite sui dati che andranno a popolare il dataset risultato del tool, basato sull'utilizzo di Dbpedia.
Le valutazioni sull'efficacia di questo tool hanno riportato risultati molto promettenti e come primo proposito per sviluppi futuri, ci si riserverà di provare il tool su documenti relativi ad altri journals in modo da aumentare le casistiche previste e raffinare il processo di analisi.
Sebbene questi primi risultati siano incoraggianti, molti sono i campi in cui AffiliationExtractr può essere migliorato.
In questo senso una possibile estensione nello sviluppo potrebbe riguardare l'aumento di risorse consultabili per riconoscere e disambiguare ancor meglio le informazioni estratte, estendendo il modulo sets.py.
Altro miglioramento faciltmente implementabili consisterebbe nell'estendere il supporto a più formati in ingresso oltre al solo XML andando a modificare il modulo affiliationExtractor.py.
Partendo dal valore di confidence assegnato per misurare la qualità dell'analisi operata si potrebbe pensare ad un nuovo modulo che cerchi nel dataset prodotto tutti i casi in cui questo valore sia minore di una certa soglia per

andare a verificare le informazioni estratte con altri metodi, andando a raffinare maggiormente la qualità dei dati forniti.
Per quanto riguarda la disambiguazione delle informazioni sulle affiliazioni si potrebbe pensare ad inserire un campo nel modello di serializzazione utilizzato, che descriva gli acronimi delle organizzazioni per permettere di raffinare i dati forniti nel dataset RDF risultato dell'esecuzione del tool.

Appendice A

Prima Appendice

```
[
... ,
  {
    "organization": {
      "value": "University of Manchester"
    },
    "unit": {
      "value": "Department of Computer Science"
    },
    "country": {
      "value": "United Kingdom"
    }
  }
... ,
]
```

Tabella A.1: esempio json gold standard

81

```
[
... ,
  {
    "organization": {
      "value": [
        "universidad de manchester",
        "universiteit van manchester",
        "universita di manchester",
        "universite de manchester",
        "universidade de manchester",
        "university manchester"
      ]
    },
    "unit": {
      "value": [
        "department computer science"
      ]
    },
    "country": {
      "value": [
        "wielka brytania",
        "reino unido",
        "verenigd koninkrijk",
        "united kingdom",
        "regno unito",
        "royaume-uni",
        "vereinigtes konigreich"
      ]
    }
... ,
]
```


Tabella A.2: esempio label json di result

Bibliografia

[1] Berners-Lee, Tim, James Hendler, and Ora Lassila. *"The semantic web."*Scientific american 284.5 (2001): 28-37.

[2] Shuiqing Huang, Bo Yang, Sulan Yan, Ronald Rousseau, *Institution name disambiguation for research assessment*, Scientometrics, 2014, 99, 3, 823.

[3] Fernanda Morillo, Javier Aparicio, Borja GonzÃ¡lez-Albo, Luz Moreno, *Towards the automation of address identification*, Scientometrics, 2013, 94, 1, 207.

[4] Yong Jiang, Hai-Tao Zheng, Xinmin Wang, Binggan Lu e Kaihua Wu, *Affiliation disambiguation for constructing semantic digital libraries*, Journal of the American Society for Information Science and Technology, Volume 62, Issue 6, pages 1029â1041, June 2011.

[5] Decker, S.; Melnik, S.; van Harmelen, F.; Fensel, D.; Klein, M.; Broekstra, J.; Erdmann, M.; Horrocks, I., *"The Semantic Web: the roles of XML and RDF"*, Internet Computing, IEEE , vol.4, no.5, pp.63,73, Sep/Oct 2000.

[6] Shotton D, Portwin K, Klyne G, Miles A (2009) *Adventures in Semantic Publishing: Exemplar Semantic Enhancements of a Research Article.* PLoS Comput Biol 5(4): e1000361. doi:10.1371/journal.pcbi.1000361.

[7] de Waard, A., *"From Proteins to Fairytales: Directions in Semantic Publishing"*, Intelligent Systems, IEEE , vol.25, no.2, pp.83,88, March-April 2010.

[8] Georgi Kobilarov, Tom Scott, Yves Raimond, Silver Oliver, Chris Sizemore, Michael Smethurst, Christian Bizer, Robert Lee, *Media Meets Semantic Web â How the BBC Uses DBpedia and Linked Data to Make Connections*, The Semantic Web: Research and Applications, Lecture Notes in Computer Science Volume 5554, 2009, pp 723-737.

[9] Bill Bernickus, Jos Migchielsen, Simon Pepping and Rob Schrauwen, *The Elsevier DTD 5 Family of XML DTDs*, Version 1.1, February 2007.

[10] Peroni, Silvio, David Shotton, and Fabio Vitali. *"Scholarly publishing and linked data: describing roles, statuses, temporal and contextual extents."*, Proceedings of the 8th International Conference on Semantic Systems. ACM, 2012.

[11] Rizzo, Giuseppe, and Raphaël Troncy. *"NERD: a framework for unifying named entity recognition and disambiguation extraction tools."*, Proceedings of the Demonstrations at the 13th Conference of the European Chapter of the Association for Computational Linguistics. Association for Computational Linguistics, 2012..

[12] Bagnacani, A., Ciancarini, P., Di Iorio, A., Giovanni, A., Nuzzolese, S. P., & Vitali, F. *Towards a Linked Open Dataset for Scholarly Publishing: Semantic Lancet Project.*