Pós-Graduação em Ciência da Computação

ONTOILPER: AN ONTOLOGY- AND INDUCTIVE LOGIC PROGRAMMING-BASED METHOD TO EXTRACT INSTANCES OF ENTITIES AND RELATIONS FROM TEXTS

By

RINALDO JOSÉ DE LIMA

Doctoral Thesis

Universidade Federal de Pernambuco
posgraduacao@cin.ufpe.br
www.cin.ufpe.br/~posgraduacao

RECIFE

2014

Universidade Federal de Pernambuco

CENTRO DE INFORMÁTICA

PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

RINALDO JOSÉ DE LIMA

ONTOILPER: AN ONTOLOGY- AND INDUCTIVE LOGIC PROGRAMMING-BASED METHOD TO EXTRACT INSTANCES OF ENTITIES AND RELATIONS FROM TEXTS

Este trabalho foi apresentado à Pós-Graduação em Ciência da Computação do Centro de Informática da Universidade Federal de Pernambuco como requisito parcial para obtenção do grau de Doutor em Ciência da Computação.

ORIENTADOR: ***Prof. Frederico Luiz Gonçalves de Freitas***

RECIFE

2014

Catalogação na fonte
Bibliotecária Jane Souto Maior, CRB4-571

L732o Lima, Rinaldo José de
Ontoilper: an ontology- and inductive logic programming-based method to extract instances of entities and relations from texts / Rinaldo José de Lima. – Recife: O Autor, 2014.
240 p.: il., fig., tab.

Orientador: Frederico Luiz Gonçalves de Freitas.
Tese (Doutorado) – Universidade Federal de Pernambuco. CIn. Ciência da Computação, 2014.
Inclui referências e apêndices.

1. Inteligência artificial. 2. Processamento de linguagem natural. 3. Ontologia. 4. Aprendizado de máquina. I. Freitas, Frederico Luiz Gonçalves de (orientador). I. Título.

006.3 CDD (22. ed.) UFPE- MEI 2014-154

Tese de Doutorado apresentada por **Rinaldo José de Lima** à Pós-Graduação em Ciência da Computação do Centro de Informática da Universidade Federal de Pernambuco, sob o título "**OntoILPER: an Ontology- and Inductive Logic Programming-based Method to Extract Instances of Entities and Relations from Text**" orientada pelo **Prof. Frederico Luiz Gonçalves de Freitas** e aprovada pela Banca Examinadora formada pelos professores:

_____

Prof. Rafael Dueire Lins
Centro de Informática / UFPE

_____

Prof. Bernard Espinasse
Aix-Marseille Université (AMU)
Laboratoire des Sciences de l'Information et des Systèmes

_____

Prof. Bruno Tenório Ávila
Departamento de Ciência da Informação / UFPE

_____

Profa. Solange Oliveira Rezende
Departamento de Ciência da Computação e Estatística/UFSP

_____

Profa. Vera Lúcia Strube de Lima
Departamento de Fundamentos da Computação / PUCRS

Visto e permitida a impressão.
Recife, 21 de agosto de 2014.

_____

Profa. Edna Natividade da Silva Barros
Coordenadora da Pós-Graduação em Ciência da Computação do
Centro de Informática da Universidade Federal de Pernambuco.

I dedicate this thesis to my mom Maria José, my sisters Taciana and Micelanea, my daughter Rebeca, and my wife Jamille Ribeiro.

Acknowledgements

This PhD thesis has been very challenging, hard but at the same time very interesting and enjoyable. The experience of participating in a PhD program has certainly enriched me both from a scientific and human point of view. I would like to thank all the people who directly or indirectly have allowed me to reach this aspired goal.

First of all, I am deeply grateful to my friend and thesis advisor Fred Freitas for his guidance since the first steps of my academic life that began in 2008, when he invited me to the Master's program. For more than 6 years that we have been working together. I am very grateful to him for the essential support during this time, and his indispensable advice.

I wish to express my deepest gratitude to my friend and research partner Bernard Espinasse for our discussions, encouragement, continuous support, and patience throughout my graduate studies. Since 2009, it has been a very pleasant and fruitful cooperation. Working with him has been my honor and also very productive, and I am looking forward to continuing our collaboration. "À vous, Bernard, un très grand merci!".

Thanks to my friends and research colleagues from the Informatics Center (CIn/UFPE) . I am especially indebted to Hilario Oliveira, Dimas Filho, and Jamilson Batista for their great assistance, particularly in the implementation of the OntoILPER framework.

I wish to thank Laura Pentagrossa, Fernando Lins, Luciano Cabral, Renê Gadelha, and Rafael Ferreira for their participation in the papers related to this thesis.

My sincere thanks to José Santos for clarifying some key points about the GILPS system.

I would like to thank Patrice Bellot and William Domingues for allowing me utilizing the computational infrastructure of the LSIS.

This research work has been partially supported by both CNPq which provided me a doctoral scholarship; and CAPES, which allowed me the opportunity to enrich my experience in an international doctoral program with Bernard Espinasse at the LSIS in Marseille, France.

I wish to thank all my colleagues, friends and relatives who encouraged me during my work on this thesis.

I am deeply grateful to my best friend and like a dad to me, Gerson Henrique, who provides the encouragement and support that I can always count on.

A final word of gratitude is dedicated to my wife, Jamille Ribeiro, for her love, care, support, and patience.

I do not know what I may appear to the world, but to myself, I seem to have been only like a boy playing on the seashore and diverting myself in now and then, finding a smoother pebble or a prettier shell than ordinary, whilst the great ocean of truth lay all undiscovered before me.

—SIR ISAAC NEWTON

Resumo

A área de Extração de Informação (IE) visa descobrir e estruturar informações dispostas em documentos semi-estruturados ou desestruturados. O Reconhecimento de Entidades Nomeadas (REN) e a Extração de Relações (ER) são duas subtarefas importantes em EI. A primeira visa encontrar entidades nomeadas, incluindo nome de pessoas e lugares, entre outros; enquanto que a segunda, consiste na detecção e caracterização de relações que envolvem as entidades nomeadas presentes no texto. Como a tarefa de criar manualmente as regras de extração para realizar REN e ER é muito trabalhosa e onerosa, pesquisadores têm voltado suas atenções na investigação de como as técnicas de aprendizado de máquina podem ser aplicadas à EI a fim de tornar os sistemas de ER mais adaptáveis às mudanças de domínios. Como resultado, muitos métodos do estado-da-arte em REN e ER, baseados em técnicas estatísticas de aprendizado de máquina, têm sido propostos na literatura. Tais sistemas normalmente empregam um espaço de hipóteses com expressividade propositional para representar os exemplos, ou seja, eles são baseado na tradicional representação atributo-valor. Em aprendizado de máquina, a representação proposicional apresenta algums fatores limitantes, principalmente na extração de relações binárias que exigem não somente informações contextuais e estruturais (relacionais) sobre as instâncias, mas também outras formas de como adicionar conhecimento prévio do problema durante o processo de aprendizado. Esta tese visa atenuar as limitações acima mencionadas, tendo como hipótese de trabalho que, para ser eficiente e mais facilmente adaptável às mudanças de domínio, os sistemas de EI devem explorar ontologias e recursos semânticos no contexto de um arcabouço para EI que permita a indução automática de regras de extração de informação através do emprego de técnicas de aprendizado de máquina. Neste contexto, a presente tese propõe um método supervisionado capaz de extrair instâncias de entidades (ou classes de ontologias) e de relações a partir de textos apoiando-se na Programação em Lógica Indutiva (PLI), uma técnica de aprendizado de máquina supervisionada capaz de induzir regras simbólicas de classificação. O método proposto, chamado OntoILPER, não só se beneficia de ontologias e recursos semânticos, mas também se baseia em um expressivo espaço de hipóteses, sob a forma de predicados lógicos, capaz de representar exemplos cuja estrutura é relevante para a tarefa de EI consideradas nesta tese. OntoILPER automaticamente induz regras simbólicas para classificar exemplos de instâncias de entidades e relações a partir de um modelo de representação de frases baseado em grafos. Tal modelo de representação é uma das constribuições desta tese. Além disso, o modelo baseado em grafos para representação de frases e exemplos (instâncias de classes e relações) favorece a integração de conhecimento prévio do problema na forma de um conjunto reduzido de atributos léxicos, sintáticos, semânticos e estruturais. Diferentemente da maioria dos métodos de EI (uma pesquisa abrangente é apresentada nesta tese, incluindo aqueles que também se aplicam a PLI), OntoILPER faz uso de várias subtarefas do Processamento de Linguagem

Natural(PLN), tais como a análise de dependência, resolução de correferência, desambiguação de sentido de palavras e anotação de papéis semânticos. A etapa de PLN em OntoILPER também considera diversos mapeamentos semânticos entre substantivos e verbos para recursos semânticos, tais como o WordNet, WordNet Domains, e o VerbNet. Outro mapeamento considerado em OntoILPER relaciona substantivos do texto de entrada a classes da ontologia de topo SUMO. OntoILPER foi implemento como um arcabouço para EI que foi avaliado experimentalmente nas tarefas de NER e ER. Esta tese relata os resultados experimentais das várias avaliações conduzidas em seis *corpora* padrão de avaliação pertencentes aos domínios de notícias (news) e de biomedicina. Os resultados alcançados demonstram a eficácia do método proposto nas tarefas de REN e ER. De fato, OntoILPER superou alguns dos sistemas de EI do estado-da-arte comparados nesta tese.

Palavras-chave: Reconhecimento de Entidades Nomeadas. Extração de Relação. Povoamento de Ontologias. Extração de Informação baseada em Ontologias. Programação em Lógica Indutiva.

Abstract

Information Extraction (IE) consists in the task of discovering and structuring information found in a semi-structured or unstructured textual corpus. *Named Entity Recognition* (NER) and *Relation Extraction* (RE) are two important subtasks in IE. The former aims at finding named entities, including the name of people, locations, among others, whereas the latter consists in detecting and characterizing relations involving such named entities in text. Since the approach of manually creating extraction rules for performing NER and RE is an intensive and time-consuming task, researchers have turned their attention to how machine learning techniques can be applied to IE in order to make IE systems more adaptive to domain changes. As a result, a myriad of state-of-the-art methods for NER and RE relying on statistical machine learning techniques have been proposed in the literature. Such systems typically use a propositional hypothesis space for representing examples, i.e., an attribute-value representation. In machine learning, the propositional representation of examples presents some limitations, particularly in the extraction of binary relations, which mainly demands not only contextual and relational information about the involving instances, but also more expressive semantic resources as background knowledge. This thesis attempts to mitigate the aforementioned limitations based on the hypothesis that, to be efficient and more adaptable to domain changes, an IE system should exploit ontologies and semantic resources in a framework for IE that enables the automatic induction of extraction rules by employing machine learning techniques. In this context, this thesis proposes a supervised method to extract both entity and relation instances from textual corpora based on *Inductive Logic Programming*, a symbolic machine learning technique. The proposed method, called *OntoILPER*, benefits not only from ontologies and semantic resources, but also relies on a highly expressive relational hypothesis space, in the form of logical predicates, for representing examples whose structure is relevant to the information extraction task. OntoILPER automatically induces symbolic extraction rules that subsume examples of entity and relation instances from a tailored graph-based model of sentence representation, another contribution of this thesis. Moreover, this graph-based model for representing sentences also enables the exploitation of domain ontologies and additional background knowledge in the form of a condensed set of features including lexical, syntactic, semantic, and relational ones. Differently from most of the IE methods (a comprehensive survey is presented in this thesis, including the ones that also apply ILP), OntoILPER takes advantage of a rich text preprocessing stage which encompasses various shallow and deep natural language processing subtasks, including dependency parsing, coreference resolution, word sense disambiguation, and semantic role labeling. Further mappings of nouns and verbs to (formal) semantic resources are also considered. *OntoILPER Framework*, the OntoILPER implementation, was experimentally evaluated on both NER and RE tasks. This thesis reports the results of several assessments conducted using six standard evaluation

corpora from two distinct domains: news and biomedical. The obtained results demonstrated the effectiveness of OntoILPER on both NER and RE tasks. Actually, the proposed framework outperforms some of the state-of-the-art IE systems compared in this thesis.

Keywords: Named Entity Recognition. Relation Extraction. Ontology Population. Ontology-based Information Extraction. Inductive Logic Programming.

List of Figures

List of Tables

Table of Contents

Chapter

Chapter 1

Introduction

A large amount of information on many different topics in various digital formats is posted in the Web continuously. As a result, the Web has become a major source of information, bearing the potential of being the largest encyclopedic source of all the news, data, etc. This information load is mainly due to the advances in computer technology, which simplifies the creation, storage, and distribution of information on the Web. Most of such information is fragmented and stored in an unstructured form. Thus, this lack of structure greatly limits its use. Besides that, current technologies for searching web pages based on keywords are not mature enough to provide the user with the specific information the user needs [Freitas, 2002].

In this context, the interesting idea of converting this sheer volume of unstructured textual data into useful information by means of automatic information extraction is certainly appealing. Such automation could, for instance, greatly increase the efficiency of searching for information, facilitate the creation of large-scale models of the relationships among relevant entities from a given domain, and allow for automated inference of new information, e.g., extracting implicit information from extracted facts mentioned in texts.

The research field of *Information Extraction* (IE) holds the key promise of addressing the information overload problem by automating the process of "understanding" the relevant parts of textual data sources [Pyysalo,2008]. IE is mainly grounded on computational linguistics and machine learning, both applied to unlock access to the knowledge generated by humans in the form of natural language documents. However, accurate information extraction from web pages by means of automatic tools capable of processing human language is an intensive and time-consuming task [Sarawagi, 2008]. Thus, the development of efficient and robust information extraction systems constitutes a big challenge.

1.1. Problem Description and Motivation

Information Extraction is an important task in Text Mining and has been extensively studied in various research communities including Natural Language Processing (NLP), Web Mining, Information Retrieval (IR), among others. The goal of IE consists in discovering and structuring information found in semi-structured or unstructured documents, leaving out irrelevant information [Jiang, 2012]. IE can also be regarded as the task of automatically identifying and recovering certain entities, relations or events from textual sources.

In IE, two of the most important subtasks are *Named Entity Recognition* (NER) and *Relation Extraction* (RE).

Named Entity Recognition. The sixth Message Understanding Conference formally introduced NER as a subtask in IE in 1995. The aim of NER is to identify named entities from text and to classify them into a set of predefined types such as person, organization, location, among others. These types are the most useful for many application domains [Turmo *et al.*, 2006].

The output of a NER system is illustrated with the following input sentence:

"Paul bought 300 shares of Acme Corp. in 2012."

Then, an NER system can output the annotated version of the above sentence in which the identified named entities are marked by brackets together with their corresponding entity types:

[Paul] PERSON bought 300 shares of [Acme Corp.] ORGANIZATION in [2012] TIME.

NER consists of the most fundamental task in IE, since the extraction of more complex structures, such as relations and events, depends upon accurate NER as a previous step. NER cannot be simply accomplished by string matching against pre-compiled lists of entities because instances of a given entity type usually do not form a closed set and, therefore, any list of this kind would be incomplete. In addition, the type of a named entity usually is context or domain-dependent.

Other application domains have also been attracted to NER, such as the biomedical domain [Pyysalo, 2008]. In such a domain, named entities refer to biological or medical terms. Such entities include gene and protein names, medical problems and treatment, drug names, just to name a few.

Particularly in the biomedical domain, NER has been notably challenging for two reasons [Yeh *et al.*, 2005]: the dynamic nature of scientific discovery that constantly increases in this domain; and the abundant use of synonyms, acronyms/abbreviations that makes it difficult to identify the concepts related with terms.

Relation Extraction. RE consists of *detecting* and *characterizing* semantic relations among entities in text. The former task consists in only determining if a relation between two entities holds, whereas the latter, refers to the classification problem of assigning a relation type label to a particular relation instance. Much work on RE focuses on *binary relations*, i.e. relations between two entities, or arguments. Examples of such relations include *physical* (e.g. an entity is physically near another entity), and *employment/affiliation* (e.g. a person is employed by an organization). An example of a sentence comprising two *mentions* or two pairs of relation instances is given below, where the entities are highlighted in italics.

"*American* saxophonist *David Murray* recruited *Amidu Berry*".

The above sentence contains two relation mentions: "*Citizenship"* relation between the words 'David Murray' and 'American'; and "*Business*" between 'David Murray' and 'Amidu Berry'.

Supervised machine learning is widely employed to approach both NER and RE. However, research work has shown that extracting relations among entities is still a substantially harder task than NER [Jiang, 2012]. In fact, the performance results of the state-of-the-art NER systems employing machine learning techniques is around 90%, whereas RE systems exhibit considerably lower performance of around 70% on the ACE[1] datasets [Jiang, 2012].

IE has been of great importance to the biomedical domain [Pyysalo, 2008]. Indeed, due to both the exponential growth of bioinformatics literature and the infeasibility of processing all this information manually, IE systems have been heavily exploited as tools for populating bioinformatics databases.

[1] Automatic Content Extraction (ACE). http://www.itl.nist.gov/iad/mig//tests/ace

Despite the fact that many IE systems have already been proposed, as presented in Chap. 3, there are still in both NER and RE tasks many challenges that require new methods and tools.

Specifically, IE systems present the following shortcomings [Bui, 2012], as presented and discussed in more details in Chapter 3:

- Portability is a main issue in current IE systems. It concerns the capability of how easily an IE system can be applied to a new domain. Thus, the portability to a new domain is still considered hard to be achieved by IE systems [Jiang, 2012].
- long and complex sentences with plenty of vocabulary ambiguity can cause the performance of NLP tools to drop considerably [Ananiadou and John, 2006];
- the high degree of variation in natural language equally hampers the overall performance of IE systems [Miyao *et al.*, 2009].

The bottom line is that the performance of extraction systems, usually measured in terms of precision and recall [Baeza-Yates and Ribeiro-Neto, 1999], needs to be improved to satisfy the demands of NER and RE tasks, especially in the biomedical domain [Zhou and He, 2008].

1.2. Ontology-based Information Extraction

In the last two decades, there was considerable advance in NLP, and IE has been taking advantage of such advances, allowing for more sophisticated text analysis. Despite of that, efficient, scalable and reliable IE systems have not yet reached a point where these requirements could be fully realized [Hahn and Wermter, 2006].

As put before, the main problem that current IE systems have to face concerns system portability, i.e., adaptability problem, a serious bottleneck in the field.

Another important issue concerns the fact that automated IE systems are trained to explicitly extract stated information. Consequently, they have limitations in their ability to extract implicit facts, for many reasons [Raghavan *et al*., 2012]. First, being limited to the scope of a sentence at a time, state-of-the-art extraction systems are not suitable to discover implicit relations. Second, implicit relations exist in different sentences, paragraphs, or even across documents, and they require further relational knowledge to be inferred. Third, those systems have no access to commonsense knowledge and, thereby they are incapable of performing deeper inference.

Modern IE systems yield limited performance results on most difficult tasks, especially those related to semantic understanding of natural language texts which prompted the emergence of the IE subfield called *Ontology-Based Information Extraction* (OBIE).

An OBIE system can be defined as a system that processes unstructured or semi-structured texts through a mechanism guided by ontologies to extract certain types of information and link such information to its semantic description in an ontology [Wimalasuriya and Dou, 2010]. In its turn, ontologies are explicit specifications of conceptualizations [Gruber, 1993]. They serve as explicit models of conceptual knowledge of a given domain. In practical terms, ontologies encompass definitions of concepts, properties, relations, constraints, axioms, and instances about a certain domain or universe of discourse. The backbone of ontologies consists of a generalization/specialization hierarchy of concepts, i.e., a taxonomy of classes. They also enable the reuse of domain knowledge, which makes domain assumptions explicit, separating domain knowledge from the operational one. Particularly to IE, ontologies can offer formal and computer-understandable representations of relevant information [Karkaletsis *et al.*, 2011].

OBIE normally takes place by exploiting domain ontologies as well as IE techniques to discover individuals for classes and values for their properties.

Several authors [Wu and Weld, 2008], [Cimiano, 2006], [Kietz *et al.*, 2000] agree that, even being a relatively new field of study, OBIE presents lot of potential. Besides the automatic processing of the information contained in natural language texts mentioned above, the potential for fully exploiting OBIE is two-fold [Wimalasuriya and Dou, 2010]:

- *Creating semantic contents for the Semantic Web*. The Semantic Web aims at providing semantic content to the current World Wide Web, in a way that it can be processed by software agents [Berners-Lee *et al.*, 2001]. On the other hand, it is quite hard to imagine that such content has to be manually annotated, given the prohibitive size of the Web. As a result, a massive metadata generation is required in order to make the Semantic Web a reality [Popov *et al.*, 2004]. In this context, OBIE has the great potential as a means for automatic generation of semantic contents by converting the information contained in existing web pages into ontologies.
- *Improving overall ontology quality*. Interestingly, OBIE can be used in the assessment of the quality of an ontology [Kietz *et al.*, 2000] [Maynard *et al.*, 2006. If one assumes that a given domain ontology can be successfully used by an OBIE system to extract the semantic contents from a set of documents related to that domain, then it can be deduced that the ontology itself serves as a good representation of that domain. Consequently, errors in the ontology can be identified by analyzing the types of semantic elements that the OBIE system has failed to extract.

According to [Petasis *et al.*, 2011], the first potential application of OBIE systems mentioned above is strictly related to the *Ontology Population* task. Ontology Population (OP) consists of the process of inserting new instances of classes, properties and/or relations in an existing ontology [Petasis *et al.*, 2011]. Therefore, an OP system does not alter the structure of the ontology, i.e., no change in the hierarchy of both classes and relationships is carried out. The updating task is restricted to the set of instances of concepts, relationships, and properties of an input ontology. Instantiating ontologies with new factual knowledge is a relevant step towards the provision of valuable ontology-based knowledge services [Cimiano, 2006].

OBIE systems are closely related to OP systems, as pointed out by Petasis *et al.* (2011) because they provide mechanisms to associate pieces of the information with concepts and relationships of an ontology. Indeed, every OBIE system can be considered as an OP system, as it can be extended to assimilate extracted instances into the ontology [Wimalasuriya and Dou, 2010]. In addition, a populated ontology can be employed in several applications such as information retrieval, text mining, automatic reasoning, among others.

The investigation of the state-of-the-art methodologies for OBIE has identified some shortcomings (see Chapter 3):

- Very few IE systems actually perform automatic construction or induction of extraction rules in symbolic form. In reality, all OBIE systems surveyed in Chapter 3 prefer to manually create extraction rules. This can be explained by the fact that it is easier and more straightforward to handle such kind of extraction rules than integrating statistical extraction models into the IE process.
- Most work on ontology population relies on shallow natural language processing techniques (including NER and chunking), and WordNet as a semantic lexicon

resource. Other deeper natural language processing subtasks, such as semantic role labeling has been much less exploited.

- Most of the current OBIE systems use a single ontology for guiding their IE process [Wimalasuriya and Dou, 2010].
- Very few OBIE systems are able to extract implicit information.

The motivational hypothesis in this thesis is that the aforementioned shortcomings should be addressed in their specific aspects in order to improve IE. Thus, it is necessary to develop methods and tools for effective IE and, to some extent, to enable domain portability. For achieving that, this thesis investigates to what extent ontologies are suitable to guide IE processes, and how extracted instances can be integrated into ontologies for later use in more sophisticated inference tasks.

The present thesis focuses on the investigation of approaches to IE based on ontologies. More precisely, it introduces an OBIE method called *OntoILPER*, and its implementation as a framework suitable for extracting instances of classes (entities) and relations (properties) from natural language texts.

In the proposed methodology, a class denotes a set of objects sharing the same characteristics, whereas a binary relation denotes the relationship of two instances of distinct classes. The interest on binary relations, in special non-taxonomic binary relations, is explained by the fact that the Semantic Web has proposed many ways of formalizing knowledge representation based on classes of individuals and binary relations among them, such as OWL/DL ontologies [Hiztler *et al.*, 2009]. Actually, OWL/DL ontologies can model complex domains by means of basic axioms defining several binary relations.

In this respect, this thesis proposes and evaluates a *unified relational model* for representing both entities and relations found in documents. This model, one of the contributions of this thesis, consists of an expressive graph-based model for sentence representation that comprises four types of features (or dimensions) that describe the examples in the solution, including lexical, syntactic, semantic, and relational features. Such group of features defines a very rich hypothesis space to be explored by a machine learning technique suitable for inducing symbolic extraction rules.

In OntoILPER implementation, the Inductive Logic Programming (ILP) technique provides mechanisms for both the supervised learning of symbolic models from examples and declarative background knowledge. The final set of induced rules is expressed as Horn clauses. ILP combines machine learning with logic programming in its learning paradigm [Lavrac and Dzeroski, 1994].

The decision of adopting a symbolic rule induction technique, such as ILP, is motivated by following key points:

- a significant amount of time and effort can be reduced by employing a machine learning technique instead of the manual development of the extraction rules;
- the user has the interesting options of providing additional a priori knowledge (background knowledge) about the problem at hand, and defining some restrictions, or biases, which drives the learning process. Moreover, both aforementioned options are usually defined by employing the same logic-based formalism, which considerably facilitates the learning task [Muggleton and Raedt, 1994].
- Given that the rules are expressed as Horn clauses, a knowledge engineer can easily intervene in the IE process by, for example, validating the extraction rules, optimizing the rules, or using them as components to form other rules or axioms in the domain ontology.

Another important part of study conducted in this thesis concerns the investigation of several types of background knowledge integrated into the IE process. Indeed, OntoILPER incorporates two ontologies: the first one representing the domain elements (classes and relations), whereas the second, the *annotation ontology*, integrates and formalizes background knowledge in terms of a rich model which defines lexical, syntactic, semantic, and relational features used by the ILP-based learning component in OntoILPER. The annotation ontology is domain-independent, i.e., it can be used for supporting a broad range of domains. Therefore, this rational use of ontologies in OntoILPER consists of an attempt to make the proposed method flexible, extensible, and portable to new domains. Furthermore, OntoILPER encompasses what it is considered the right elements to enable inference and, consequently, the derivation of implicit information as well, due to its potential for using background knowledge in a declarative form.

One can conclude that, although many approaches and techniques have been developed and evaluated for ontology-based information extraction, this field still needs further research, once there is plenty of room for improving the quality of the extracted instances resulting from the (semi)-automatic methods for ontology population already proposed. The present thesis takes a step in that direction by offering a specific novel solution based on induction learning techniques able to generate expressive extraction rules, which are more intelligible to the human expert, allowing him to know the particular reason for the choices made by the OBIE method proposed here.

1.3. Thesis Objectives and Research Questions

> *The goal of this thesis is to propose, implement, and evaluate an ontology- and ILP-based method that, besides inducting symbolic extraction rules suitable for classifying instances of classes and relations in text sources, also takes into account the structural aspects of the examples through a rich and expressive relational representation model which integrates linguistics and ontological annotations as background knowledge, and finally feeds back the extracted instances into the domain ontology.*

In other words, this thesis proposes an OBIE system that takes into account substantial sources of domain-independent background knowledge, such as semantic taxonomies, and multiple ontologies. This substantial background knowledge consists of a first class element in the proposed methodology for achieving an effective, extensible, and portable OBIE system.

The extraction process in the proposed OBIE method is ontology-based in the sense that ontologies provide the background knowledge in the form of two ontologies: *domain* and *annotation*. The latter comprises a formal relational hypothesis space implemented as an OWL/DL ontology encompassing four dimensions of features: lexical, syntactic, semantic and structural.

Although initially motivated by ontology population, the proposed solution for IE was design as an OBIE framework since it is more general that OP systems. Nevertheless, it is also suitable for OP indeed, as it is shown in Section 5.7.

The proposed method also relies on ILP, which consists of a relational learning formalism allowing both the representation of expressive relational knowledge, and the exploitation of this knowledge as background knowledge.

In ILP, it is possible to integrate background knowledge, including relational knowledge, given that this knowledge can be expressed in a declarative way. This is

possible because the representation language used for describing the example and BK has the expressiveness close to the one of the first-order logic, allowing the representation of n-ary predicates and variables, which enables capture contextual relationships between the examples.

Furthermore, the proposed OBIE method encompasses the following key elements:

- a comprehensive NLP component that not only considers the lexical and syntactic features typically used in related work, but also enables the integration of several semantic resources and ontologies. More precisely, several types of annotations derived from text preprocessing using shallow and deep NLP tools, semantic resources (WordNet [Fellbaum, 1998], WordNet Domains [Bentivogli *et al.*, 2004], VerbNet [Kipper-Schuler, 2005]), and semantic mapping to the top-level SUMO ontology [Nile and Pease, 2001] composes a rich set of features that are used in the final induction of extraction rules.
- a graph-based model for sentence representation which is suitable for representing all the feature types mentioned above.

The working hypothesis assumed in this thesis is that given an automatic acquisition of a substantial body of linguistic knowledge from textual data, and its formalization using ontologies, in combination with an expressive inductive learning technique, it is possible to automatically generate effective information extraction models (in terms of precision and recall). In this hypothesis, the inductive learning component should allow prior knowledge about the domain to be integrated in the construction of the classification model, in such a way that, the classification of examples is performed by reasoning involving their syntactic, semantic, and structural features formalized by a rich relational representation model.

This hypothesis resulted from a detailed study focused on the open challenges and issues in both IE and OBIE fields. This working hypothesis is supported by the following evidences:

- *the performance increase when ILP-based models are utilized*. Specia *et al.*, (2006) reported experimental results in Word Sense Disambiguation showing that ILP-constructed models obtained better performance than those obtained using a SVM-based classification model equipped with shallow syntactic features [Isozaki and Kazawa, 2002]. These results suggested that the use of ILP with diverse sources of background knowledge could provide a way for making substantial progress in the WSD.
- *the effectiveness of integrating ontologies in IE*. Liu *et al.* (2011) showed that ontologies are of great importance for effective IE, since the reliable relation extraction cannot be accomplished without knowledge of instances and their relationships to the corresponding ontology classes.
- *the utilization of ontologies as a means for enhancing portability of IE systems.* As put before, traditional IE systems suffer from being specialized on a single IE template of a domain of concern. The utilization of ontologies as a means of formal information exchange increases flexibility in adapting IE systems to new domains [Adrian *et al.*, 2009].
- *different perspectives on a domain using multiples ontologies*. In [Wimalasuriya and Dou, 2010], the authors showed that the utilization of multiple related but distinct ontologies could bring several benefits to IE, as ontologies can offer different

perspectives on a domain, with the potential of giving more accurate answers to queries related to distinct users' perspectives.

Research Questions. The following research questions will be addressed in this thesis:

1. How can ontologies and semantic resources be explored to address one of the main challenges for OBIE systems: achieving state-of-the art performance?
2. What is the influence of syntactic, semantic, and structural features on RE?
3. What is the role of the ILP-based learning component to *both NER and RE tasks*?
4. What are the benefits of the graph-based model of sentence representation in the final induced classification models?
5. Concerning the graph-based model of sentence representation mentioned above, would it be useful to adopt a simplification strategy before the learning phase?
6. To what extent is OntoILPER portable to other domains?

The answers to each of these research questions are given in Chapter 7 (*Conclusions*), whereas some experimental questions derived from the above-mentioned research questions are answered in Chapter 6.

1.4. Thesis Contributions

The research activities performed in this work resulted in the following contributions to the OBIE area:

(1) **OntoILPER**. An OBIE method that automatically extracts instances of classes and relations from text sources. This OBIE method, called OntoILPER, takes advantage of several linguistic-related knowledge sources and additional ontologies (in OWL/DL) that formalize the background knowledge for achieving state-of-the-art performance. Its distinguishing features, compared to related work, consist in its higher expressiveness of the extraction rules, and the rich set of features that are utilized by an ILP-based component for inducing symbolic extraction rules. Another OntoILPER advantage is that it can extract implicit information. In OntoILPER, the requirements regarding the use of ontologies for IE are analyzed and discussed. In particular, the role of ontologies as means for formalizing BK in the logic-based induction of extraction rules is addressed.

(2) **OntoILPER Framework**. The implementation of OntoILPER as a framework for ontology population, called OntoILPER Framework. This framework was implemented as a modular, pipelined architecture that integrates all of the models proposed in this thesis.

(3) **A Complete Ontological Environment for ILP-based IE.** As this environment converts the input text into ontological elements, uses the domain ontology during the extraction process, translates the extraction rules into SWRL, and populates the domain ontology with the extracted instances, this means that the whole process is completely ontology-based. Moreover, because of that, the proposed OBIE framework could be easily integrated to ontology engineering tools such as Protégé[2] or Neon[3].

[2] Protégé Ontology Editor. http://protege.stanford.edu

[3] NeOn Toolkit. http://neon-toolkit.org/wiki/Main_Page

(4) The integration and implementation of various NLP subtasks/tools as a unique component able to perform a comprehensive natural language preprocessing of textual corpora. The current implementation of this component is modular, and extensible to new languages and additional language analysis.

(5) A hybrid XML-based model for linguistic annotation representation which combines the advantageous aspects of two standard linguistic annotation formats: inline and stand-off annotation.

(6) **Annotation Ontology**. The design and implementation of a domain-independent and expressive annotation ontology in OWL/DL. The annotation ontology formalizes the resultant analysis carried out in the preprocessing stage, in which various kinds of describing features were mapped to formal structures. Such structures are defined by an expressive knowledge representation formalism that defines all types of background knowledge used by the rule learning component in OntoILPER. The annotation ontology also provides the formalization of IE results in OWL/DL.

(7) A graph-based model for sentence representation which encompasses several types of features, including morphological, syntactic, semantic, and relational ones. This model also integrates mapping to linguistic semantic resources and ontologies in a rich and unified model.

(8) The design and implementation of a method for transforming and simplifying graph-based representations of sentences. This simplification method allows improving the overall extraction performance results in terms of precision and recall.

Fig. 1.1, at the end of this chapter, provides a schematic view of the main contributions of the present thesis with indication of the published papers related to this thesis.

In summary, it can be argued that the main contributions of this thesis represent a novel specific solution for information extraction and ontology population from textual data. Indeed, the particular combination of several elements that comprise the proposed solution (as seen in the list of contributions above) distinguishes the present research study from previous related work.

1.5. List of Publications

Parts of this thesis have been published in several international conferences:

(1) **R. Lima**, B. Espinasse, H. Oliveira, F. Freitas (2014). Ontology Population from the Web: an Inductive Logic Programming-Based Approach. In *Proceedings of the 11th International Conference on Information Technology: New Generations, ITNG 2014*, Las Vegas, Nevada, USA.

(2) **R. Lima**, B. Espinasse, H. Oliveira, L. Pentagrossa, F. Freitas (2013). Information Extraction from the Web: An Ontology–Based Method using Inductive Logic Programming. In *Proceedings of the IEEE International Conference on Tools with Artificial Intelligence, ICTAI 2013*, Washington DC, USA.

(3) **R. Lima**, B. Espinasse, H. Oliveira, R. Ferreira, L. Cabral, F. Freitas, R. Gadelha (2013). An Inductive Logic Programming-Based Approach for Ontology Population from the Web. In *Proceedings of the 24th International*

Conference on Database and Expert Systems Applications, DEXA 2013, Prague, Czech Republic.

(4) **R. Lima**, J. Batista, R. Ferreira, F. Freitas, R. Lins, S. Simske and M. Riss (2014). *Transforming Graph-based Sentence Representations to Alleviate Overfitting in Relation Extraction*. In Proceedings of the 14th ACM Symposium on Document Engineering (DocEng 2014), September, Fort Collins, Colorado, USA. (*in printing*)

More recently, two other papers were submitted:

(5) **R. Lima**, B. Espinasse, F. Freitas, H. Oliveira, R. Ferreira (2014). OntoILPER: an Ontology- and Inductive Logic Programming-based Method to Extract Instances of Entities and Relations from Text. *Information Sciences Journal*. (*submitted in July, 2014*)

(6) **R. Lima**, B. Espinasse, H. Oliveira, F. Freitas (2014). Ontology-based Information Extraction with OntoILPER. *IEEE International Conference on Tools with Artificial Intelligence, ICTAI 2014*, Limassol, Cyprus. (*submitted in July, 2014*)

1.6. Thesis Outline

The rest of this thesis is structured into the following chapters:

- *Chapter 2* introduces the necessary fundamental concepts for a better understanding of this thesis. First, it is presented the concept of ontologies as a means for formalizing knowledge bases, followed by a brief description of commonly used ontology representation languages as well as ontology queries languages. Next, a myriad of subtasks in NLP which aims at analyzing natural language by automatic means are presented. The focus here is on morphologic, syntactic, and semantic preprocessing tasks commonly used by IE systems. In addition, semantic resources, and semantic role labeling are also explored. Finally, Inductive Logic Programming, the machine learning technique adopted in this thesis, is described in detail. This machine learning paradigm is explored by presenting an illustrative example of a relational learning task in ILP.
- *Chapter 3* gives a brief introduction to Information Extraction. Starting by some definitions and a historical overview, this chapter also discusses the general architecture of IE systems. A comprehensive review of the literature on Information Extraction (classical IE and OBIE) is presented. For the latter, it is emphasized how ontologies can be exploited to guide the IE process. A taxonomic classification of specific implementations of the main approaches in both research fields is also proposed, focusing on their advantages and drawbacks. Another goal of Chapter 3 is to highlight the trends and open problems in IE.
- *Chapter 4* presents the proposed method for OBIE, OntoILPER, the main contribution of this thesis. It explains the main benefits that the use of ontologies and a richer text preprocessing can bring to OntoILPER, comparing it to other close related IE/OBIE systems. Furthermore, Chapter 4 presents the main design decisions made in OntoILPER which is grounded on the Inductive Logic Programming, a machine learning technique that is suitable to induce symbolic classification rules. It is shown how expressive relational representations of sentences provided by formal ontologies may open up new opportunities for an

effective and adaptive IE process. Then, a second contribution of this thesis, the transformation rules for reducing the graph-based representation of sentences are described. The aim of these rules is to reduce considerably the size of the graph that represents a given sentence in the proposed sentence representation model. This constitutes one of the working hypotheses that is tested in this thesis. Finally, the roles of domain and annotation ontologies as background knowledge sources are introduced. The main goal of Chapter 4 is to discuss the main hypothesis in this work that the integration of the two aforementioned ontologies combined with an expressive rule learning formalism, may improve the EI process.

- *Chapter 5* gives an in-depth view of the implemented components of the proposed OBIE method introduced in Chapter 4. First, the main modules of the OntoILPER architecture, their functionalities, and the integration of supporting tools are described. Then, each component that constitutes the OntoILPER Framework is explained, in most cases, with illustrative examples. This chapter focuses on the aspects related to the most important design decisions made during the implementation of the proposed OBIE method. Different scenarios for applying OntoILPER are outlined.
- *Chapter 6* is dedicated to answer the research questions raised in the introductory part of this thesis through the discussion of experimental results on NER and RE using 6 distinct datasets from two domains: news (3 datasets) and biomedical domain (3 datasets). The achieved results are also compared with other state-of-the-art NER and RE systems.
- *Chapter 7* summarizes this thesis presenting an overall conclusion and the lessons learned. In addition, the major contributions introduced by this thesis, and future research lines of investigation are presented.

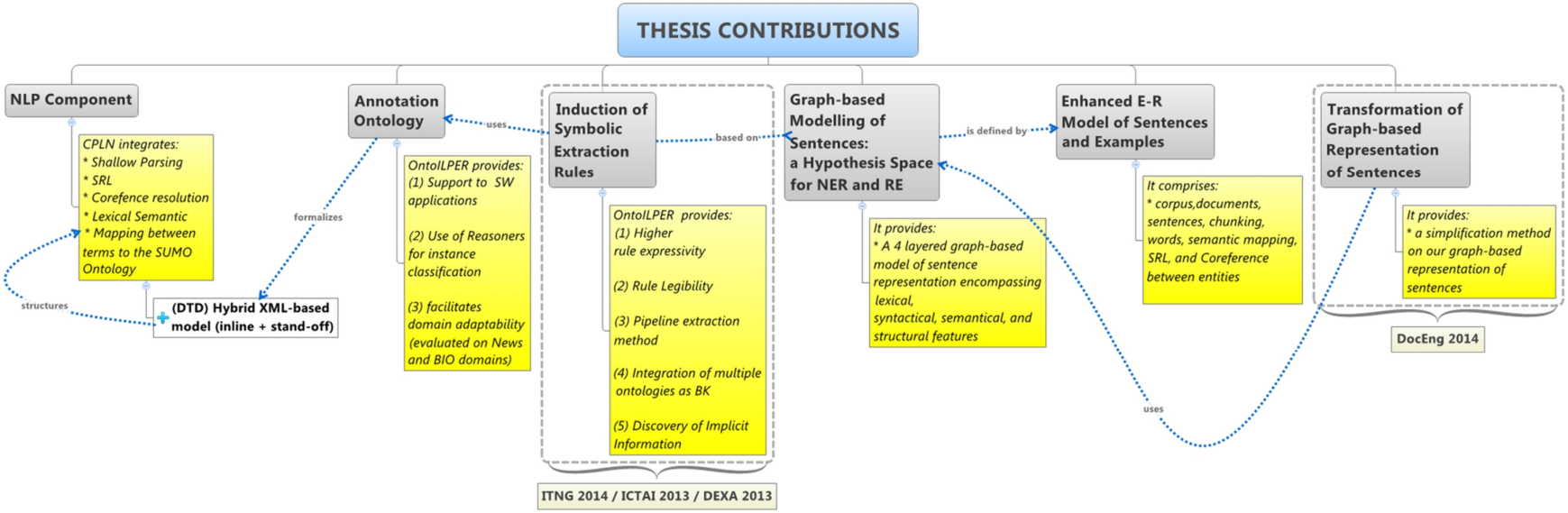


Figure 1.1. Schematic view of the main contributions of this thesis.
The papers that introduced the contributions are also indicated.

Chapter 2

Foundations

In this chapter, the necessary fundamental concepts for a better understanding of this thesis are covered. This includes an overview of *Ontologies*, the concepts and subtasks in *Natural Language Processing*, and *Inductive Logic Programming*, the machine learning technique adopted in this work.

2.1. Ontologies

Ontologies have become a prominent theme in Computer Science, where they serve as explicit models of conceptual knowledge of a given domain. They also play a key role in the vision of the Semantic Web (SW) [Berners-Lee *et al*., 2001] because they provide the semantic vocabulary used to annotate web sites in a non-ambiguous way for the interpretation of computers.

Ontologies, under a computational point of view, can be regarded as logical theories that describe some aspect of reality. Typically, such logical theories describe a specific domain, i.e., some part of the reality that is relevant to some applications. For instance, an ontology can be defined for the domains of Biomedicine [Grenon *et al*., 2004], and Biochemistry (The Gene Ontology Consortium, 2000), among others.

In one of the most cited definitions of ontologies, Gruber states that "an ontology is an explicit specification of a conceptualization" [Gruber, 1993]. In this definition, the term conceptualization means an abstract, simplified view of the world that one wishes to represent for some purpose. Later on, Borst (1997) defined an ontology as a "formal specification of a shared conceptualization". This second definition additionally required that the conceptualization should express a shared view between several parties, a consensus, rather than an individual view. In 1998, Studer *et al.* (1998) merged these two definitions stating that: "An ontology is a formal, explicit specification of a shared conceptualization." where "formal" means that such conceptualization should be expressed in a formal machine readable format.

In practical terms, ontologies encompass definitions of concepts, properties, relations, constraints, axioms, and instances about a certain domain or universe of discourse. The backbone of an ontology consists of a generalization/specialization hierarchy of concepts, i.e., a *taxonomy*. They also enable reuse of domain knowledge, which makes domain assumptions explicit, separating the domain knowledge from the operational one.

The rest of this section presents the main ontology-related elements used in this thesis, which includes *ontology types*, *ontology representation languages*, and *ontology querying languages*.

2.1.1. Ontology Types

Many researchers have tried to classify the ontologies into specific categories. Fig. 2.1, for example, displays the classification proposed by Guarino (1998). The arrows in the figure indicate a *specialization* relationship between the following types of ontologies:

- *Top-level ontologies*: they describe many general concepts related to time, space, matter, etc.
- *Domain ontologies*: they describe the vocabulary used in a specific domain.

- *Task ontologies*: they specify the vocabulary used by a generic task or activity.
- *Application ontologies*: they model the concepts that normally are specializations of other domain ontologies or tasks.

As a more concrete example, consider a domain ontology about books and an ontology about sales. A bookstore could then specialize both ontologies to describe a service of online book sales.

In what follows, an example of a top-level ontology is presented.

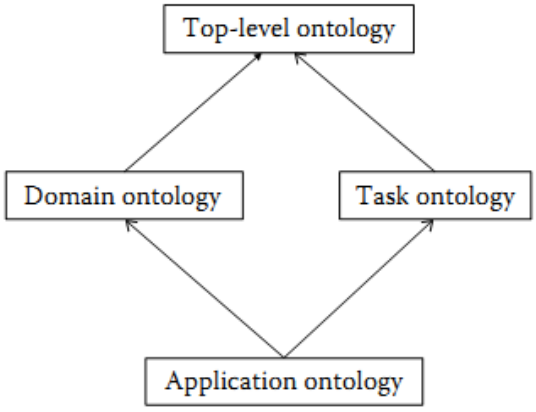


Figure 2.1. Types of ontologies, according to Guarino (1998).

SUMO Ontology[1]**.** The Suggested Upper Model Ontology (SUMO) [Niles and Pease, 2001] consists of the largest formal ontology currently available. SUMO was created by merging publicly available ontological content into a single, comprehensive, and cohesive structure.

This ontology is also a huge database which, together with its domain-specific extensions, contains more than 20,000 concepts and 60,000 axioms. The concepts include the most common geographic names, languages, financial terms, and even chemical elements.

A mapping between SUMO classes and WordNet synsets[2] was proposed by Niles and Pease (2003). Such a mapping enriches WordNet database files by tagging each synset with the corresponding SUMO concept. As a result, SUMO enhances WordNet by organizing them into a logical structure.

Another SUMO key feature concerns the well-defined and well-documented concepts that are interconnected into a semantic network and accompanied by several axioms. Such axioms intend to both constraint interpretation of concepts, and provide guidelines for automated reasoning systems.

The most general concepts in SUMO are illustrated in Fig. 2.2. This picture shows that in SUMO, even abstract entities like sets and propositions are elements of the domain.

[1] SUMO Ontology portal. http://www.ontologyportal.org

2 WordNet synsets are sets of cognitive synonyms, each expressing a distinct concept.

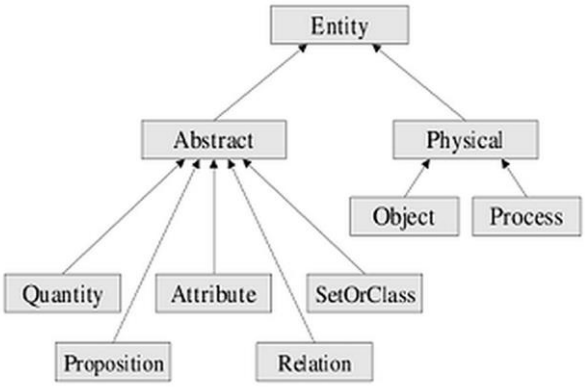


Figure 2.2. Top level classes in SUMO.

Below, a SUMO axiom is represented in simplified Knowledge Interchange Format (KIF) [Genesereth and Fikes, 1992]:

=>
(instance ?DRIVE Driving)
(exists (?VEHICLE)
(and
(instance ?VEHICLE Vehicle)
(patient ?DRIVE ?VEHICLE))))

This axioms means that "if there is an instance of *Driving*, there is a *Vehicle* that participates in that action".

2.1.2. Ontology Representation Languages

The main standard representation languages for ontologies are presented here. Such languages include *Resource Description Framework* (RDF), *Resource Description Framework Schema*(RDFS), and *Web Ontology Language* (OWL).

Resource Description Framework (Schema) [Brickley and Guha, 2000]. RDF defines the data model for the SW, as recommended by W3C. It has been developed to represent information about web resources. The main goal of RDF is to annotate web resources that are uniquely identified by a URI (*Unique Resource Identifier*).

In RDF, the basic statement is a triple of the form (*subject, property, property value*) or, equivalently, (*subject, predicate, object*). Thus, each triple expresses a *binary relation*. The subject of a triple is a resource, which is identified by a *URI*. The predicate is also denoted by a URI, and the object is either another resource or datatype value, also called *literal*. In case the property value is a resource, the property is called an *object property*, otherwise it is called *datatype property*.

While RDF is applied for specifying statements about instances, RDF-Schema (RDFS) defines schema and subclass hierarchies. The statements in RDF and RDFS can be represented as one combined directed graph, also called a *RDF graph*. A set of RDF statements constitutes a RDF graph because each triple (s, p, o) defines an edge which goes from *s* to *o* and has label *p* .

Most of the SW data currently available is defined using the lightweight formalism of RDF(S). Each resource may be associated with one or several concepts (e.g. classes) via the type-property. In addition, RDFS allows for the definition of restrictions on properties

and concepts. The most important of these restrictions are subclass relationships, the definition of *domains* and *ranges* of properties.

OWL/DL. Since 2004, *OWL* (*Ontology Web Language*) [Hitzler *et al*., 2009] has turned out to be the most expressive ontology language and accepted worldwide as an expressive formalism for modelling and building ontologies. OWL not only provides a powerful formalism for knowledge representation and reasoning, but also extends RDFS to formulate more expressive schemas and subclass hierarchies with additional logical constraints.

OWL is based on the *Description Logic* (DL) [Baader *et al*., 2008] that defines a family of knowledge representation formalisms and reasoning techniques. DL combines a rigorous semantic based on First-Order Logic (FOL) with a very expressive way to structure and code the conceptual knowledge of a domain [Baader *et al*., 2008]. DL models an application domain by defining its relevant concepts (or classes) and specifying the properties of the objects and the individuals contained in such a domain. In OWL, the formal definition of the interpretation of individuals, concepts, and relationships is given by a model-theoretic semantic allowing for many inference services that are concretely provided by reasoners.

OWL, in its variant OWL/DL, is a specific DL [Baader *et al*. 2008]. An OWL/DL ontology contains explicit and implicit information about the domain or world that one wishes to represent using a *terminological box* (*TBox*) and an *assertional box* (*ABox*). TBox is a set of (equivalence or inclusion) axioms that define concepts (sets of individuals) and roles (binary relationships) using primitive (atomic) concepts and role names and combining them through specific language constructors. ABox contains factual assertions concerning the individuals.

In what follows, further explanation about TBox and ABox elements are provided:

- *Instances* denote elements or individuals in an ontology.
- *Classes* (or *concepts*) denote sets of individuals. Classes are usually organized in taxonomies through which inheritance mechanisms can be applied. For example, in the *vertebrate* domain, *mammal* and vertebrate classes are related through a taxonomic relationship "is-a", stating that a mammal is a vertebrate.
- *Properties* represent either an association (relationship) between individuals or an association between individuals and values (data type values). OWL ontologies typically contain binary relations that are denoted by *R(x, y)*, where *x* and *y* represent individuals, and *R*, the relationship or property. The behaviour of object properties can be classified as *symmetric*, *transitive*, *functional* or *inverse functional*.

  Such properties can be of three types:

  i. *Object properties* relates individuals. Unlike hierarchical relationships, object properties represent a non-taxonomical relationship that is typically expressed by a verb relating a pair of concepts;
  ii. *Datatype properties* link individuals to data types which can be represented by a string or number, for instance.
  iii. *Axioms* represent sentences in FOL that are assumed to be true, that is, without proof. They specify constraints on the ontology and can be applied in the verification of its consistency as well as to infer new knowledge from an inference mechanism. For instance, OWL/DL enables stating the equivalence or disjointness of classes and the (non-)identity of properties respectively instances.

OWL/DL allows the user to construct classes by enumerating their content, through forming intersections, unions, and the complements of classes. Cardinality constraints can also be formulated and classes can be defined via property restrictions.

Reasoning Support. OWL/DL also provides reasoning support. This is important because it allows the user to check the consistency of the ontology and the knowledge, check for unintended relationships between classes, and automatically classifying instances in classes. Reasoning is based on the interpretations that satisfy the axioms/assertions in the knowledge base (i.e. their logic models). Hence, the standard notions of satisfiability, validity, and entailment defined for FOL naturally extend to DLs.

The following OWL/DL reasoning services are available:

- class membership: if *c* is an instance of a class *C*, and *C* is a subclass of *D*, then it can be inferred that *c* is an instance of *D*.
- equivalence of classes: if class *A* is equivalent to class *B*, and class *B* equivalent to class *C*, then *A* is equivalent to *C*.
- consistency checking: This ensures that an ontology does not contain any contradictory facts. For instance, consider the situation where *a* is an instance of the class *A*, being *A* a subclass of the class *B*, with the constraint that *A* and *B* are disjoint. Then, this reveals an inconsistency which points to a probable error in the ontology.
- class subsumption: It aims to prove or disprove for any given pair of classes *C* and *D* that *C e*ntails *D*.

2.1.3. Ontology Query Languages: SPARQL and SWRL

For the purpose of this thesis, it follows a give a brief overview of the basic concepts of the two standard Semantic Web query languages, namely *SPARQL* and *SWRL*.

SPARQL. The SPARQL Query Language[3] is a W3C recommendation for querying and manipulating data stored in RDF format. It became a standard by the RDF Data Access Working Group of the WWW Consortium, and it is recognized as one of the key technologies of the SW. SPARQL allows users for querying RDF ontologies consisting of triple patterns, conjunctions, disjunctions, and optional patterns. With SPARQL, users write queries against data that follows the RDF specification of the W3C, that is, the entire database is thus a set of "subject-predicate-object" triples.

The SPARQL query language is based on matching graph patterns. The simplest graph pattern is the triple pattern, which is like an RDF triple, but with the possibility of a variable instead of an RDF term in the subject, predicate, or object positions. Combining triple patterns gives a basic graph pattern, in which an exact match to a graph is needed to fulfill a pattern.

SWRL: Rules in OWL/DL. The *Semantic Web Rule Language* (SWRL) [Horrocks *et al*., 2010] is a standard language based on OWL/DL and on the *Rule Markup Language* (RuleML) which provides both OWL/DL expressivity and rules from RuleML [Horrocks *et al*., 2010]. This rule language is the unrestricted union of OWL/DL (*SHOIN(D))* and (binary) function-free Horn logic. The basic idea of SWRL is to extend OWL/DL with a form of rules while maintaining maximum backwards compatibility with OWL syntax and semantics.

[3] SPARQL Query Language for RDF. W3C Recommendation 15 January 2008. http://www.w3.org/TR/rdf-sparql-query

SWRL rules are expressed as an implication between an antecedent (body) and consequent (head). Hence, the syntax of SWRL is of the following form:

$$antecedent \rightarrow consequent \quad (1)$$

The informal meaning of a SWRL rule is: whenever the conditions specified in the antecedent hold, then the conditions specified in the consequent must hold. OWL expressions can occur in both antecedent (*body*) and consequent (*head*) which themselves are sets of SWRL atoms.

A SWRL atom may be of the following forms [Hitzler *et al.*, 2009]:

Unary atoms:
- *C(arg1)* , where *C* is an arbitrary OWL/DL class expression;
- *D(dataArg1)*, where *D* is a datatype URI or an enumerated value range.

Binary atoms:
- *P(arg1,arg2) where P is an object property;*
- *Q(arg1, dataArg1) where Q is a datatype property;*
- *arg1 = arg1 equality, or "sameAs";*
- *arg2 ≠ arg2 inequality, or "differentFrom"*

Where arguments are of the form:
- *arg1* | *arg2* these are either individuals denoting URIs or individual ranging variables;
- *dataArg1* these are data literals or data value ranging variables.

For instance, a rule in SWRL asserting the composition of *parent* and *brother* properties, which imply the *uncle* property, would be written:

$$parent(?a,?b) \wedge brother(?a,?c) \rightarrow \text{uncle}(?c,?b).$$

SWRL has several strengths, to name a few:
- arbitrary OWL classes can be used as predicates in rules;
- it consists of an extension to OWL, i.e., rules and axioms in OWL/DL can be freely mixed;
- it provides a human readable syntax, like the simple form of Horn-style rules.

Additionally, SWRL also allows for "built-in" atoms with a fixed, predefined interpretation [Hitzler *et al.*, 2009]. SWRL built-ins can be considered as a supplement to OWL's datatype facility. For instance, with the help of the Protégé ontology editor, one can utilise predefined built-ins such as *swrlb:greaterThan*, which enables a more expressive condition to be inferred. SWRL includes built-ins for mathematics, string, and date support, among others.

The combination of OWL/DL and RuleML makes the SWRL undecidable [Hitzler *et al.*, 2009]. However, decidability can be regained with the imposition of a safety condition on SWRL rules. The decidability of SWRL rules with the DL-Safety condition was established in [Motik *et al.*, 2004]: the possible values of (explicit) variables in SWRL rules are restricted to named individuals only, which restricts the effects of such rules to the ABox. This safety condition is known as "DL-Safety" and SWRL rules are called "DL-Safe rules". As a result, DL-Safe Rules are much more computationally reasonable, and the support for SWRL inference is already available by means of ontology reasoners, like Pellet[4].

[4] Pellet: OWL 2 Reasoner for Java. http://clarkparsia.com/pellet/.

2.2. Natural Language Processing

Natural Language Processing (NLP) is a subfield of computational linguistics which analyses natural language by automatic means. It provides the theoretical and practical foundations for many applications involving text mining. Applications of this technology include automatic translation, text retrieval, automatic summarization, ontology population, and notably information extraction.

NLP is particularly challenging for two reasons: natural language has plenty of ambiguity, and words can be combined into sentences in many possible ways, which makes it impossible, for computers, to simply list the possible contexts and meaning of a given word in a sentence.

This section provides a broad view of NLP techniques (in English) that will be explored in this thesis. After introducing the basic concepts of natural language: syntax, semantics, and pragmatics in Section 2.2.1; the common NLP subtasks used by most of text mining applications are presented in Section 2.2.2. Sections 2.2.3 and 2.2.4 discuss various aspects concerning lexical semantics, including semantic resources such as WordNet, and WordNet Domains. Other semantic-related subtasks such as Word Sense Disambiguation and Semantic Role Labelling are presented in Sections 2.2.5 and 2.2.6.

2.2.1. Linguistic Concepts

A language is a finite set of symbols with syntax and semantics, where *syntax* is a set of rules defining the structure of well-formed expressions; and *semantics*, which defines the meaning of words and expressions in a language.

The meaning of words or phrases is closely related with the difference between *meaning* and *reference*. This distinction can be represented by a triangle - the so-called *semiotic triangle* [Ogden and Richards, 1923]. In this triangle (Fig. 2.3), a symbol (a natural language word or phrase) is connected to its referent (an object in the real world), via some thought of reference (or concept). The branch of linguistics that investigates the meaning of words and their relationships is called *lexical semantics* (see Section 1.2.3).

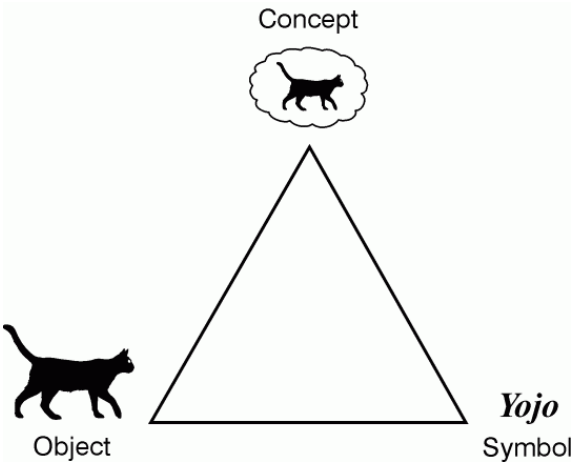


Figure 2.3. Semiotic triangle. Extracted from [Sowa, 2000].

Traditionally, the research work on NLP tends to see the language analysis process as decomposable into a series of stages, reflecting the different existing linguistic theories of *morphology*, *syntax*, *semantics*, and *pragmatics*. In this simplified view, the sentences are first analyzed in terms of its syntax, which provides order and structure that are more amenable to the later analysis in terms of literal meaning (semantics). The semantic

analysis, in turn, is followed by the pragmatic analysis which tries to determine the meaning of the text in its context. This last analysis also deals with the *discourse*, while the former two usually only consider the limits of a sentence at a time. In the following, each language analysis stage mentioned above is briefly presented in turn.

Morphology. Morphology concerns the internal structure of words, i.e., how words can be broken into meaningful pieces. Conversely, morphology comes up with rules for *inflection* and word formation that enable humans build a rich vocabulary from a basic inventory of morphemes. A morpheme is the smallest unit in natural language that carries meaning. Words that are built from more than one morpheme can be split into a *stem* and one or more *affix*, i.e., a morpheme attached to a stem like in "*book*" + "*s*", in which the plural "*s*" is an *inflectional* morpheme, as it alters the base form without changing its syntactic category.

Words are usually referred to as *lexical items*. Most language theories assign features to words, e.g., whether they are singular or plural, transitive or intransitive, or first or third person.

Syntax. Syntax refers to the way words are arranged together, forming legal structures of a language. First, words presenting similar behaviour are categorized into *syntactic* categories, or *parts of speech* (POS). Such categories include noun, verb, and adverb, for example. Syntactical knowledge is expressed by rules for combining such categories into phrases and the structural roles that these phrases can play in a sentence. For example, the sentence (2.1) is a syntactically correct sentence in English, whereas sentence (2.2) is not.

Diamond is the hardest material on earth. (2.1)

Diamond the is earth material hardest on. (2.2)

A *grammar* is a way of expressing the valid syntactic structures in a language. Another important syntactic notion is the notion of *grammatical dependency*. For example, in the sentence "*Mary reads an interesting book*.", "Mary" and "book" are dependents of a *reading* event. They are both the arguments of the verb *read*. The adjective "interesting" is a *dependent* of "book", and modifies "book".

Semantics. Semantics studies the meaning of a language unit: a word, a sentence, or the entire discourse. In other words, semantics focuses on the way how word meanings combine into the meaning of sentences.

For instance, sentence (2.3) below is an example of a meaningful sentence, while (2.4) is not.

I have read the book on Semantic Web. (2.3)

The book spoke to me about its large head (2.4)

One can notice that, even though the latter sentence is syntactically well formed, it does not make sense. Here, this kind of distinction is captured by *selectional restrictions*, which describe the semantic regularities associated with the possible complements of a verb. In this example, the verb mention "speak" suits for people as subject, instead of non-animate objects (book) as the subject.

Pragmatics. Pragmatics studies the language use in terms of how sentences relate to one another. The so-called, pragmatic knowledge includes information about sentences that might have meaning in a specific context, i.e., the way in which context clearly contributes to meaning.

Unlike semantics, which examines meaning that is conventional in a given language, pragmatics investigates how the transmission of meaning depends not only on structural and linguistic knowledge, but also on the context of the utterance, which includes any pre-existing knowledge about those involved, the inferred intent of the speaker, and other factors [Carston, 2002].

2.2.2. Preprocessing: Morphological and Syntactical Analysis

Natural language processing in most current text mining applications typically consists of the sequential application of several components performing the NLP subtasks as shown in Fig. 2.4. Usually, such subtasks are performed in *pipeline mode*, i.e., starting with simpler analysis like sentence splitting and tokenization, whose output results serve as input for the next complex subtasks like POS tagging and syntactic parsing. It follows a brief description of the NLP subtasks shown Fig. 2.4.

Figure 2.4. A common-used pipeline of NLP subtasks

Sentence Splitting. It consists of determining the sentence boundaries in a document.

Tokenization. It is the process of breaking a stream of text up into words, symbols, or other meaningful elements, usually referred to as *tokens*. For example, punctuation signs, such as periods, can be problematic because either they can denote the end of a sentence, the end of an abbreviation, or they can be used for specifying dates, telephone numbers, etc. An additional problem concerns the blanks that do not always indicate word boundaries as it is the case for many named entities like "New York", which actually denotes multi-word expressions formed by more than one token. Because of that, sometimes it seems more useful to apply named entity recognition before actually performing tokenization [Jurafsky and Martin, 2009].

Lemmatization. It consists in the process of reducing each word to its base form, or its *lemma*. For example, in English, words may appear in many inflected forms. Consider the verb 'to call' that can appear as “call”, “called”, “calls”, “calling”. Thus, the base form “call”' is the lemma for its inflected forms.

Lemmatisation is closely related to stemming. The difference is that a stemmer operates on a single word without knowledge of the context, and therefore cannot discriminate between words which have different meanings depending on part of speech.

Part-of-Speech (POS) tagging. Part-of-speech (POS) tagging is the task of assigning to each token its corresponding part-of-speech tag, i.e. its syntactic word category such as noun, adjective, verb, etc. Fig. 2.5 shows the result of POS tagging applied to the sentence "Mary is going to the Soccer World Cup in Brazil in 2014". In this figure, the tag (label) above each word denotes its syntactic word category.

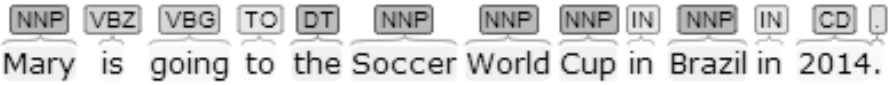


Figure 2.5. POS tagging results for the sentence: "Mary is going to the Soccer World Cup in Brazil in 2014" [5].

Table 2.1 shows the meaning of these POS tags. The complete set of POS tags, as proposed by the *Penn Treebank Project*[6], listed in alphabetical order, can be found in Appendix A.

Some of the state-of-the-art POS tagging systems are based on statistical/probabilistic approaches, such as Markov Models [Church, 1988] [Charniak *et al*., 1993].

Table 2.1. POS tags descriptions

Tag	Meaning
DT	Determiner
NNP	Proper noun, singular
VBZ	Verb, 3rd person singular present
VBG	Verb, gerund
IN	Preposition
CD	Cardinal number

Chunking. It consists of dividing a sentence into groups of syntactically correlated words, like nouns, verbs, and prepositional phrases specifying neither its internal structure nor its role in the main sentence. In other words, chunks are *non-overlapping* groups of words forming small syntactic units (or phrases).

Chunking is sometimes called *chunk parsing*, or *shallow partial parsing*. Typically, chunks are required to be non-recursive, i.e., no other chunk can be embedded within a chunk.

The main unit (*head*) in a noun phrase in English is commonly the rightmost noun. For instance, in the "the exciting modern art museum" noun phrase, "museum" denotes the main constituent unit, while other words are essentially modifying or restricting the meaning of the head noun.

Chunking analysis typically adopts a bottom-up approach, i.e., it starts detecting simpler units, and then it integrates such units in more complex units. This type of analysis does not discover syntactic relations such as subject or object. Moreover, it adopts a conservative strategy and tends to avoid producing errors, since it does not attempt to solve semantic or syntactic ambiguities. The big advantages of chunking are its robustness and efficiency.

Below the result of a chunking analysis performed on the sentence mentioned above, where *NP*, *VP*, *PP* stands for noun phrase, verb phrase, and prepositional phrase, respectively. The boundaries of each chunk are indicated by square brackets "[ ]".

[*NP* Mary] [*VP* is going] [*PP* to] [*NP* the Soccer World Cup] [*PP* in]
[*NP* Brazil] [*PP* in] [*NP* 2014] .

Syntactic Parsing. Parsing, in contrast to chunking, aims at uncovering the full syntactic structure of a given sentence. In other words, parsing typically designates the analysis and

[5] This figure was generated with the online CoreNLP tools at http://nlp.stanford.edu:8080/corenlp.
[6] The Penn Treebank Project, 1999. http://www.cis.upenn.edu/~treebank/home.html.

building of syntactic structures according to a grammar, a formalism that describes the correct syntactic structures in a language.

A parsing algorithm (parser) is a procedure for searching through the possible ways of combining grammatical rules to find one (or more) structure(s) that matches a given sentence's structure.

Currently, two types of grammars have emerged as the most widespread means to analyze and generate syntactically well-formed parsing trees in natural language [Jurafsky and Martin, 2009]:

– *Phrase structure grammar.* [Jurafsky and Martin, 2009] defines *syntactic* structures in terms of phrasal categories, e.g., NP, VP, etc. For instance, Fig. 2.6 displays the resultant parse tree as produced by a phrase structure parser for the sentence "Economic News had little effect on financial markets". One can notice that the prepositional phrase "on financial markets" is correctly attached to the verb phrase (VP). The reader should also observe that the non-terminal nodes denote phrases, while the POS tags (above the terminal nodes) clearly identify their syntactic categories.

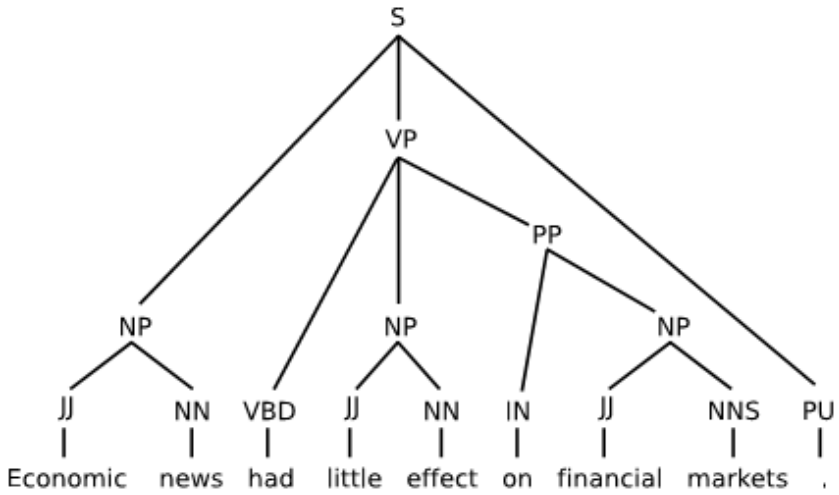


Figure 2.6. Phrase structure or constituent parsing of a sentence in English. Extracted from [Kübler, 2009].

– *Dependency Grammar*. The dependency grammar theory [Gerdes *et al.*, 2014] formalizes the construction of a dependency structure, or *dependency graph* between two linguistic units that immediately dominate each other in a syntax tree. Such relations between two tokens are expressed as a binary relation, where the first argument is a governor and the second one is a dependent. Dependency parsing performs a full syntactical analysis of sentences, according to the dependency grammar theory [Gerdes *et al.*, 2014].

The basic underlying assumption of a dependency grammar is the idea that the syntactic structure consists essentially of words or phrases connected by asymmetric binary relations, also called dependency relationships, hereafter referred to as dependencies. According to the dependency grammar, there exists a dependency relationship between a syntactically subordinate word called *dependent*, and another word, called *governor* or *head*. This is illustrated in Figure 2.7, which shows a dependency graph for a sentence in English, where the dependency relationships are represented by directed edges starting from the head and pointing to the dependent. Moreover, each edge has a label indicating the type of dependency. For example, the noun "news" is a dependent of the verb "had" to the type of dependence *subject*

(*nsubj*). Furthermore, the noun "effect" is dependent of the governor, the verb "had", characterizing the typed dependency *dobj* (*direct object*). One can also note that the noun “news”, in its turn, is itself a syntactic head with respect to the word “Economic", and so forth.

Such dependency structures are formally defined as labelled directed graphs, where the nodes correspond to words or lexical items, and labelled edges correspond to dependency relationships with their respective types.

A more detailed description of the Stanford typed dependencies can be found in Appendix D, and in [De Marneffe and Manning, 2008].

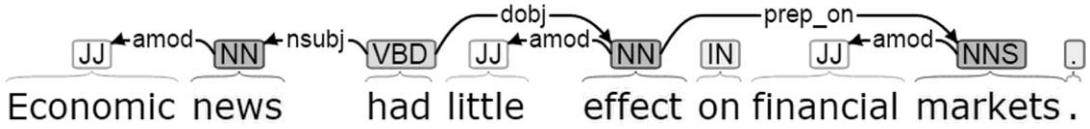


Figure 2.7. Dependency graph of the sentence “Economic News had little effect on financial markets”.

A final remark about the difference between the information encoded in a dependency graph and the information captured in a phrase structure representation. The latter consists of the most commonly used type of syntactic representation in Computational Linguistics. This can be better appreciated when one compares the two forms of representation to the same sentence as shown in Figures 2.6 and 2.7. Thus, while the dependency structure represents head-dependent relationships between lexical items, classified by functional categories, such as subject (*nsubj*) and direct object (*dobj*); the constituent structure of the same sentence represents the word grouping, ranked by structural categories as Noun Phrase (NP) and Verb Phrase (VP).

NER. Named Entity Recognition (NER) identifies named entities in texts and associates a semantic category to them. Usually, named entities include the names of people and organizations, date expressions, percentage, just to mention a few. In addition, NER relies on named entity dictionaries often tuned for a specific domain of interest.

In Fig. 2.8, the NER was able to find four named entities. The “Misc*”* label denotes a *miscellaneous* multi-word entity, while “Loc*”* denotes a *Location*.

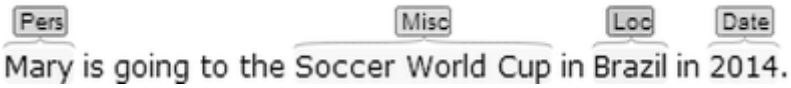


Figure 2.8. Named Entities found by the Stanford CoreNLP tools in the same sentence introduced earlier.

Coreference Resolution. In computational linguistics, a coreference occurs when two or more expressions in a text refer to the same person or thing. For example, the coreference resolution task can recognize that "John Adams", J. Adams' and ‘Adams’, in fact, refer to a single real-world entity, given a particular communication context.

In other words, coreference resolution can estimate the relative importance of various mentioned subjects, pronouns, and other referring expressions, and connected them to the right individuals.

Fig. 2.9 shows an example of coreference resolution where the word "Mary", with its first mention in the first sentence, is referenced in the second sentence by the word "her". This denotes a pronominal reference for the word "Mary".

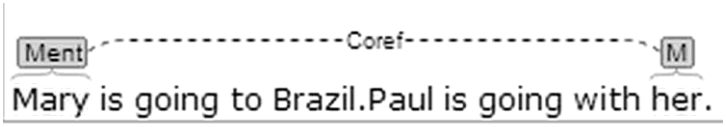


Figure 2.9. Example of two sentences containing a pronominal reference between them.

Coreference algorithms commonly try to find the nearest preceding individual that is compatible with the referring expression (see Fig. 1). Other types of coreference relations, discovered by anaphora resolution algorithms or complex discursive inferences [Cimiano *et al*., 2005] are not usually considered as a necessary preprocessing step in most current IE systems [Zouaq *et al.*, 2010].

2.2.3. Computational Lexical Semantics

The goal of this subsection is to introduce a richer model of word semantics, which focuses on the study of word meaning, a subfield in NLP called *Computational Lexical Semantics (CLS)* [Saint-Dizier and Viegas, 1994]. For the purposes of CLS, particularly for dictionaries and thesauri, a word is represented by its lemma.

A lemma is often the base form. For instance, *book* is the lemma for *books*. For a conjugated verb, the lemma is the infinitive form, e.g., the lemma form of *took* and *taken* is *take*.

In the remainder of this section, after introducing the fundamental concepts related to lexical semantics, the computational semantics resources that provide means for dealing with sense-related issues in CLS is presented.

Word Sense. The meaning of a lemma can vary greatly according to the context. For instance, consider these three uses of the lemma *bank*: meaning a financial institution, the land along the side of a river or lake, and a verb denoting the act of putting or keeping money in a bank, respectively:

I have to go to the bank at lunch time.

The banks of the river Seine.

Did you bank that check?

A sense (or word sense) is a discrete representation of one aspect of the meaning of a word, and when two or more senses are related semantically to this same word, this relationship is called *polysemy*.

Word Sense Disambiguation. In CLS, Word Sense Disambiguation (WSD) [Jurafsky and Martin, 2009] denotes the task of examining word tokens in context and determining which sense of each word has in a given sentence. This task has a long history in computational linguistics, and constitutes an open research field given that disambiguating word senses has the potential to improve many natural language processing tasks, including machine translation, question answering, information extraction, and text classification.

Given that sufficient hand-labelled data with correct word senses is available, one can apply a supervised learning approach to WSD. Such approach extracts features from the text that are helpful in predicting particular senses, and then a classifier is trained to assign the correct sense given such features. The training output is a classification model capable of assigning sense labels to words in context.

Graph-based approaches can also be employed to WSD. Sinha and Mihalcea (2007) proposed an algorithm for unsupervised WSD that annotates all the words in a text by

exploiting similarities identified among word senses, and using centrality algorithms applied on the graphs encoding these sense dependencies.

Relations between Words and Senses. Some relations that hold among word senses have received significant investigation in the past. Such relations are also of special interest for the objectives of the present thesis:

- *Synonymy*. When the meaning of two different words (or lemmas) is identical or nearly identical, it is said that the two are *synonyms*. For instance, the following pairs of words are synonyms: *cough/sofa*, *car/automobile*, *liberty/freedom*. It is worth noting that, at least, true synonymy is rarely found in natural language. In fact, the evolution of language has a tendency towards emphasizing meaning differences, possibly as a mechanism to avoid redundancy [Völker *et al*., 2008]. Hence, two words are considered synonyms if they can be exchanged for one another in a sentence without altering its truth conditions.

- *Hyponymy/Hypernymy*. One sense is a *hyponym* of another sense if the first is more specific, denoting a subclass of the other. For example, *car* is a hyponym of *vehicle*; *dog* is a *hyponym* of animal. Conversely, vehicle is a hypernym of car, and animal is a hypernym of dog.

  Hyponymy is a transitive relation: if *A* is a hyponym of *B*, and *B* is a hyponym of *C*, then *A* is a hyponym of *C*. More formally, hyponymy/hypernymy can also be defined in terms of logical entailment. Under this definition, a sense *A* is a hyponym of a sense *B* if everything that is *A* is also *B*. Thereby, being an *A* entails being a *B*. In other words, for all *x*, $A(x) \Rightarrow B(x)$.

- *Word Similarity*. As already seen, synonymy is a binary relation between words. For many computational purposes, it is preferable to use a looser metric of *word similarity* or *semantic distance*. Thus, two words are more similar if they share more features of meaning; and they are less similar, or have greater semantic distance, if they have fewer common meaning elements.

  When computing word similarity, a very important assumption is the fact that context may be exploited as a basis on which to assess the similarity of words. This assumption rests on the *distributional hypothesis* that claims that words are similar to the extent that they share similar context [Harris, 1968]. Actually, empirical investigations corroborate the validity of the above hypothesis (see [Miller and Charles, 1991] and [Grefenstette, 1994]).

  On this basis, an important question in this respect is how to represent the context of a certain word. In fact, the answer to this question has become the main contribution of many works, which often represent context as a vector in a *high dimensional space*. Such dimensions correspond to words found in the context of the word. The so-called *vector space model* used in Information Retrieval [Baeza-Yates and Ribeiro-Neto, 1999] constitutes the core of the vector-based context representations in previous work on word similarity.

2.2.4. Semantic Resources: WordNet and WordNet Domains

WordNet. WordNet[7] [Fellbaum, 1998] is a large semantic network interlinking words and groups of words by means of lexical and conceptual relations represented by labelled arcs. It constitutes the most used resource for English sense relations, including synonyms, antonyms, hypernyms/hyponyms, and meronyms.

[7] WordNet, a lexical database for English. http://wordnet.princeton.edu

The WordNet structure is organized in *synsets*, clusters of words that are synonymous in a particular linguistic context.

The WordNet 3.0 release contains more than 150,000 open-class words (nouns, verbs, adjective, and adverbs), comprising 117,097 noun synsets, 11,488 verb synsets, 22,141 adjective synsets, and 4,601 adverb synsets. The average noun has 1.23 senses, and the average verb has 2.16 senses.

Synset examples are {*car, automobile*}, {*hit, strike*}, and {*big, large*}. The meaning of the synsets is further clarified with short, general definitions (or *glosses*). Each sense, represented by a synset, is associated with a set of pointers to other synsets, which correspond to the various kinds of lexical relationships.

Synonymy is the major lexical relation among individual word forms; another is antonymy, as between the pairs {*wet*} and {*dry*}; and {*rise*} and {*fall*}. Concepts expressed by nouns are densely interconnected by the hyponymy relation (or hypernymy, or *is-a* relation), which links specific concepts to more general ones. For instance, the synset {*mailbox, letterbox*} is a hyponym of {*box*}, which in turn is a hyponym of {*container*}. {*Mailbox, letter box*} is a hypernym of {*pillar box*}, which denotes a specific type of mailbox.

Hyponymy builds trees with increasingly specific "leaf" concepts growing from an abstract "root". As can be noticed from Fig. 2.10, all noun synsets ultimately descend from {*entity*}. Fig. 2.11 shows the WordNet 3.0 entry of the word "person", which has three senses, the most frequent one being "a human being", the first one in the entry. Each sense is accompanied by its synsets and its *sense key*. A sense key is the sense index for accessing synsets in WordNet database.

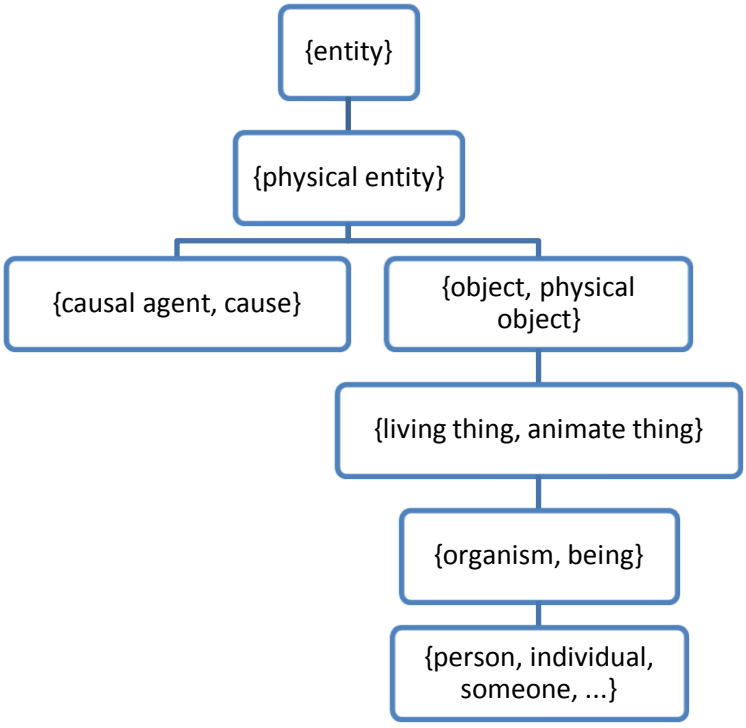


Figure 2.10. Example of a WordNet noun tree.

Although it is still one of the most valuable semantic resources for linguistic applications, WordNet presents some shortcomings: its unbalanced coverage of domains; it cannot be considered as an ontology because not only it lacks formal and explicit

semantics, but also proper distinctions between different types of hyponymy relations [Gangemi *et al*., 2001].

(n) person (*person%1:03:00::)*, individual (individual%1:03:00::), someone (someone%1:03:00::), somebody (somebody%1:03:00::), mortal (mortal%1:03:00::), soul (soul%1:03:00::) (a human being)

(n) person (*person%1:08:00::)* (a human body (usually including the clothing))

(n) person (*person%1:10:00::)* (a grammatical category used in the classification of pronouns, possessive determiners, and verb forms according to whether they indicate the speaker, the addressee, or a third party)

Figure 2.11. WordNet entry for "person".

WordNet Domains. Bringing complementary information to what already exists in WordNet, WordNet Domains (WND) [Bentivogli *et al*., 2004] provides semantic domains, a natural way to establish semantic relations among word senses, which can be very useful in Computational Linguistics.

Semantic domains consist of areas of human knowledge, including Politics, Economy, and Sport. This notion of semantic domain is related to the similar notions of semantic field, broad topic, subject domain, for example. In addition, semantic domains can be used to describe texts according to general subjects characterized by domain specific lexica. For instance, the English synset {*court, tribunal, judicature*} was annotated with the domain "Law", in WND.

WND contains 200 domain labels organized in a hierarchical structure, called WordNet Domains Hierarchy (WDH) (see Fig. 2.12).

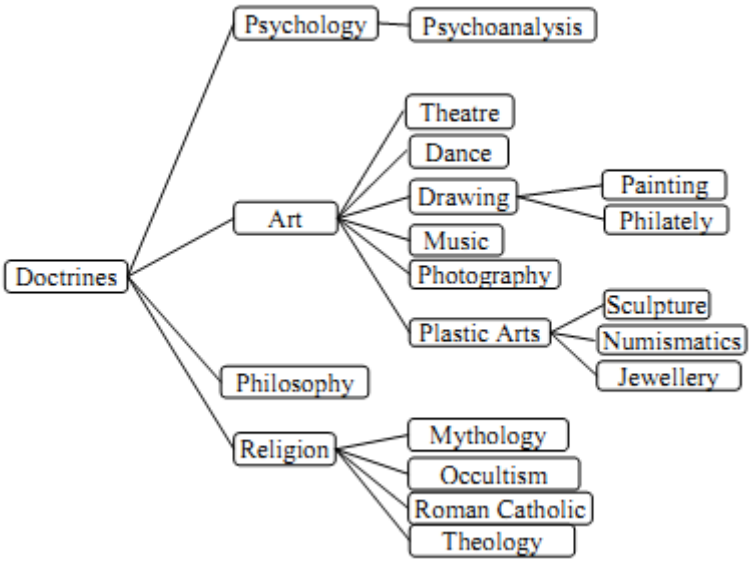


Figure 2.12. Fragments of the WordNet Domains hierarchy. Reproduced from [Bentivogli *et al*., 2004].

Each synset of WordNet 2.0 was labelled with one or more domain labels, using a methodology which combined manual and automatic assignments.

In WND, all issues concerning the "semantics", the "completeness", and the "granularity" of domain distinctions, have been addressed with reference to the Dewey Decimal Classification (DDC)[8].

The DDC is the most widely used classification system in the world, with libraries in more than 135 countries using it for organizing and providing access to their collections.

The advantage of using WordNet Domains is that it may include synsets of different syntactic categories and from different WordNet subhierarchies. This has the side effect of reducing word polysemy in WordNet, which also proved to bring a suitable level of abstraction and granularity, mainly in text categorization and information extraction tasks.

2.2.5. Semantic Roles of Event Participants

An aspect of lexical semantics is related to the semantics of events which, in their turn, are closely related to the notion of predicate-argument structure for representing an event and its participants. For instance, in the sentence: "Mary broke the window.", the verb "break" indicates the event, "Mary" is the agent of the action, and "the window" the object that was broken.

The event participants mentioned above usually share *thematic roles*, or *semantic constraints* [Jurafsky and Martin, 2009]. Such semantic constraints applied on the arguments of event predicates can be of two types: *semantic roles* and *selectional restrictions*.

In what follows, these two semantic constraints are discussed, starting with the particular model of *thematic roles*.

Thematic Roles. Thematic roles refer to the underlying semantic relationship between a predicate and its arguments [Kipper *et al.*, 2006]. They were introduced by Fillmore (1968) in order to create a closed set of participant types for a predicate's arguments. Therefore, these roles are used to describe lexical and semantic patterns in the behaviour of verbs.

Examples of thematic roles are given in the following sentences:

Mary broke the window.

John opened the door.

The thematic roles of the subjects of the verbs "break" and "open" are *Breaker* and *Opener*, respectively. These two roles are very specific to each possible kind of event, i.e., *Breaking* events have Breakers, *Opening* events have Openers, *Eating* events have Eaters, etc. On the other hand, Breakers and Openers have something in common: they are both volitional actors, often animate actors, and they have direct causal effect for their events.

Thereby, in thematic role terms, Breakers and Openers are both *agents*. Thus, an AGENT is the thematic role representing an abstract idea of volitional causation. Similarly, the direct objects of both these verbs are inanimate objects that are affected by the action. The thematic role for these event participants is THEME. Tab. 2.2 lists some of the most common thematic roles and their definitions.

The main reason for using thematic roles in text mining systems lies on the fact that thematic roles, or semantic roles in general, provide a shallow semantic language that allows for simple inferences that would not be possible relying just on the pure surface string of words, or even parse trees of sentences. Moreover, thematic roles can help us to generalize over different surface realizations of predicate arguments.

[8] http://www.oclc.org/dewey/resources/summaries.en.html

Table 2.2. Some thematic roles with their definitions [Jurafsky and Martin, 2009].

Thematic Role	**Definition**
AGENT	The volitional causer of an event
EXPERIMENTER	The experimenter of an event
THEME	The participant most directly affected by an event
INSTRUMENT	Na instrument used in an event
SOURCE	The origin of the object of a transfer event
GOAL	The destination of an object of a transfer event

On the other side, thematic roles have been criticized because there is no consensus about which set of roles is necessary to exhaustively characterize argument types of verbs. This is due because there are no clear criteria for determining which role should be applied to a particular argument of a predicate, and because definitions for these roles are often vague.

Selectional Restrictions. As seen earlier, the semantic roles express the semantics of an argument in relation to its predicate. On the other hand, a selectional restriction is a kind of semantic type constraint that a verb imposes on the type of concepts that are allowed for filling its argument roles.

In the sentence “Mary eats Italian food now”, note that “eat” is a transitive verb and the phrase "Italian food" is its direct object, being the THEME of the verb "eat". Actually, for this particular sentence, a stronger restriction can be formulated because the verb "eat" imposes a "THEME of EATING" role, i.e., this verb would accept as its THEME argument something that is edible. This type of restriction is called *selectional restriction* or *selectional preference* of the given verb, which constraints the semantic type of its arguments.

2.2.6. Semantic Role Labeling

Semantic Role Labeling (SRL) aims at automatically assigning semantic roles for each predicate in a sentence [Gildea and Jurafsky, 2002]. SRL is also called *thematic role labeling*, or *shallow semantic parsing*. More specifically, that means determining which constituents in a sentence are semantic arguments for a given predicate, and then determining the appropriate role for each of those arguments. SRL has the potential to improve performance in any language understanding task.

The state-of-the-art approaches to SRL are based on supervised machine learning, and they often perform SRL after syntactic parsing, either dependency parsing or constituent parsing, because they use syntactic parses as input.

Traditionally, SRL is performed in two steps: *argument identification and argument classification* [Gildea and Jurafsky, 2002] [Johansson and Nugues, 2008]. Argument identification is the task of finding arguments for each predicate in the sentence, whereas the argument classification task tries to assign a semantic role to each argument according to its predicate.

Based on supervised machine learning techniques, the current approaches to SRL require access to adequate amounts of training and testing data. Such training/testing data for SRL have been provided, over the last few years, by the *PropBank* and *VerbNet* resources. These semantic resources besides providing training and test data for SRL, equally specify what counts as a predicate and its arguments.

In the following, these resources are both presented.

PropBank. The Proposition Bank [Palmer *et al*., 2005], also referred to as *PropBank*, focuses on the argument structure of verbs, and provides a complete corpus annotated with semantic roles, including roles traditionally viewed as arguments and as adjuncts. PropBank provides a domain-independent resource, which can lead to more robust and broad-coverage natural language processing systems.

Due to the difficulty of achieving an agreement of a universal set of thematic roles as mentioned earlier, the semantic roles in PropBank were defined with respect to each individual verb sense. PropBank relies on the syntactic structure of the Penn Treebank[9] corpus.

In PropBank, verb arguments are annotated with a layer of semantic roles played with respect to their predicates. In addition, each predicate encompasses one or more senses defining their own predicate argument structures. Actually, every sense of each verb is specified by a set of roles, with a pre-defined list of labels which can be distinguished into two categories:

- *verb roles*, i.e., ARG[0-4] are called *numbered arguments*, and they represent the most common semantic elements that frequently co-occur with their predicates. In general, ARG0 denotes the AGENT role, while ARG1 denotes either the THEME or PATIENT role. The semantics of the other roles are specific to each verb sense. This implies that the Arg2 of one verb is likely to have nothing in common with the Arg2 of another distinct verb.
- *general event modifiers* such as a temporal adjunct are annotated with ARGM-* labels (e.g., ARGM-TMP).

The particular verb sense {*open.01*} with its argument structure, according to the PropBank, is displayed in Fig. 2.13. Thus, for this particular verb sense, ARG0 and ARG1 represent the *Opener* and *Thing opened* roles, respectively.

Role Label	Description
ARG0	opener
ARG1	thing opened
ARG2	instrument
ARG3	benefactive

Figure 2.13. Argument structure for the particular verb predicate sense open.01 in PropBank.

An illustrative example of the output of the SRL task based on a classification model learned with PropBank is shown in Fig. 2.14. This output clearly shows the arguments of the verb sense "open.01", where Arg0 (*Agent*) was assigned to "He", Arg1 (*thing opened*) assigned to "the door", and the complement "with his wet hand" assigned the label AM-MNR (*manner*).

Appendix C provides a detailed list of the semantic role labels used in PropBank.

VerbNet. VerbNet [Kipper *et al*., 2006] is a domain-independent verb lexicon consisting of approximately 5800 English verbs classified over 270 such verb classes. The verb classes in VerbNet are hierarchically organized, according the Levin's classification of verb classes and their *syntactic alternations* [Levin, 1993].

Each VerbNet class contains a set of syntactic descriptions, or *syntactic frames*, depicting the possible surface realizations of the argument structure for constructions such as transitive, intransitive, prepositional phrases, and a large set of *verb alternations*. This

[9] Penn Tree Bank Project. http://www.cis.upenn.edu/~treebank/home.html

verb classification ensures that each class is coherent enough so that all its members have a common semantics and share a common set of thematic roles, selectional restrictions of the arguments, and basic syntactic frames. The coherence of verb classes in VerbNet relies on the Levin's hypothesis which states that the syntactic behaviour of a verb is largely determined by its meaning [Levin, 1993].

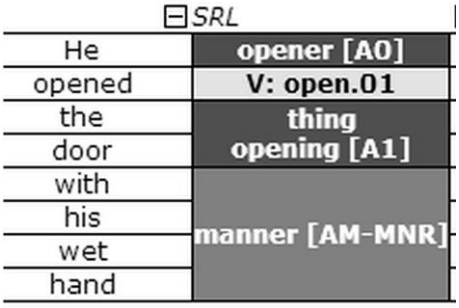


Figure 2.14. Semantic Role Labeling output of the sentence "He opened the door with his wet hand", according to the PropBank semantic role set.

A syntactic frame in a VerbNet class has a corresponding semantic representation that details the semantic relations between event participants (arguments) across the course of an event. The argument list consists of thematic roles and possible selectional restrictions on the arguments expressed by binary predicates. Each verb argument is assigned one (usually unique) thematic role within the class. Tab. 2.3 lists some thematic roles found in VerbNet.

Table 2.3. Some specific thematic roles in VerbNet with their definitions.

Thematic Role	**Definition**
ACTOR	Used by some communication verb classes.
AGENT	A human of an animate subject.
PATIENT	It denotes participants that are undergoing a process or that have been affected in some way.
LOCATION	Underspecified destination, source, or place.
DESTINATION	End point of the motion, or direction towards which the motion is directed.
PRODUCT	End result of a transformation.

VerbNet has recently been extended by a set of new classes and currently is the most comprehensive and versatile classification of English verbs in Levin style. Fig. 2.15 shows a simplified example of the VerbNet entry for the *Hit-18.1* class.

Class Hit-18.1			
Roles and Restrictions: Agent[+int_control] Patient[+concrete] Instrument[+concrete]			
Members: bang, bash, hit, kick, ...			
Frames:			
Name	*Example*	*Syntax*	*Semantics*
Basic Transitive	*Paula hit the ball*	Agent or Patient	cause(Agent, E), manner(during(E), directedmotion, Agent) !contact(during(E), Agent, Patient), manner(end(E), forceful, Agent), contact(end(E), Agent, Patient)

Figure 2.15. Simplified version of the VerbNet entry for *Hit-18.*1 class

In this thesis, it is argued that the advantages of using VerbNet for information extraction purposes resides in the fact that it can improve both: precision (via the correct choice of the semantic roles linked to a given verb); and recall (via the inclusion of synonyms that share the same restrictions semantic role information).

Appendix D provides a detailed list of the thematic role labels used by VerbNet.

2.3. Inductive Logic Programming

Inductive Logic Programming (ILP) [Muggleton and Raedt, 1994], also referred to as *Relational Learning* [Dzeroski, 2010], is a subfield of Machine Learning with solid foundations in Logic Programming (LP).

ILP is theoretically settled at the intersection of Inductive Learning and Logic Programming. From inductive machine learning, ILP inherits the development of techniques to induce hypotheses from observations. From Logic Programming, it inherits its representation formalism and semantics.

ILP relies on *first-order clauses* as a uniform representation language for *examples*, *background knowledge* (BK), and *hypotheses* [Lavrac and Dzeroski, 1994].

In ILP, the main goal is to induce, from a set of training examples, a *logic program* describing a *relational predicate*, using concepts defined in the BK. The returned logic program (or *theory*) can be used to classify new unseen examples into positive or negative.

One of the reasons for the ILP success is the *readability* of the induced models. In addition, ILP has the capability of learning from structural or relational data, and can take profit of domain knowledge given as BK. Another interesting advantage is that it can represent, using FOL, more complex concepts than traditional attribute-value (zero-order) languages.

ILP has been successfully applied in several application areas, mainly in Bioinformatics [King, 2004], Pharmacology (Drug Design) [King *et al.*, 1992], and Protein Structure Prediction [Turcotte *et al.*, 1998], that is, domains where relation information is of paramount importance.

In the rest of this chapter, after a short review of some of the Logic Programming concepts in Section 2.3.1, the relational rule induction process is presented in Section 2.3.2. Then the basic concepts concerning the ILP learning process is given in Section 2.3.3. This learning process is illustrated by an example of a classification task in Section 2.3.4. Section 2.3.5 discusses the advantages and limitations of ILP. Some current ILP systems are compared in 2.3.6, followed by a discussion rule induction strategies employed by ILP systems (2.3.7).

2.3.1. Logic Programming

For a better understanding of the fundamental concepts in ILP exploited in this thesis, the basic logic programming terminologies is introduced.

Logic Programming. Logic Programming (LP) [Fürnkranz *et al.*, 2012] is a programming paradigm based on a subset of first-order logic named *Horn clauses*. Horn clauses are logic clauses with at most one positive literal, and Horn clauses with exactly one positive literal are also called *definite clauses*.

More formally, a definite clause is a clause of the form $H \leftarrow B_1,..., B_n$ which contains precisely one literal (H) in its *consequent* and 0 or more literals (B_1, ..., B_n) as the *antecedent*. H is called the *head* and $B_1,...,B_n$ is called the *body* of the clause.

A definite clause can be conveniently seen as an implication instead of a disjunction of literals. For example, the clause

$$\neg p(X) \vee \neg q(X) \vee h(X)$$ is equivalent to the implication

$$(p(X) \wedge q(X)) \rightarrow h(X).$$

Moreover, in the form of an implication definite clauses have a straightforward procedural interpretation: in order to prove *h(X)*, show that *p(X)* is true, and *q(X)* is true.

Tab. 2.4 summarises the most important LP terms with their meanings.

Table 2.4. Some logic programming elements and their definitions

Element	Definition
constant	A symbol for denoting individuals, represented in lower case.
variable	A symbol for an unspecified individual, represented in upper case.
predicate	Symbols for denoting relations.
term	A constant, variable or function in predicates.
atom	A formula in the form $p(t_1, ..., t_n)$, where *p* is a predicate and $t_1 ... t_n$ are terms.
ground atom	An atom which contains no variable.
clause	A formula $(L_1 \vee ... \vee L_n)$, where each L_i, is an atom (positive literal) or the negation of an atom (negative literal).
fact	A definite clause where n = 0.
Horn clause	A clause with at most on positive literal.

Typically, the theories found by ILP systems are expressed as a set of definite clauses, or a definite logical program.

Prolog. Prolog consists of a programming language for general purpose that implements the paradigm of Logic Programming.

Prolog is also a declarative language in the sense that the programmer only needs to express what is known about the problem domain, i.e., facts and relationships among them, and the inner resolution mechanism takes care of the control part.

Prolog is restricted to accept Horn clauses because, as shown by [Kowalski, 1974], it allows for an efficient proof procedure, namely *Selective Linear Definite clause resolution* (SLD-resolution) [Kowalski and Khuehner, 1971] which is sound and refutation complete for Horn clauses.

Programs in LP are expressed as a set of clauses and facts (clauses without body literals). SLD-resolution is then used to mechanically prove queries and, bind logic variables to values that satisfy the query.

2.3.2. Relational Rule Induction in ILP

The relational rule induction task is the most common in ILP [Lavrac and Dzeroski, 1994]. This task consists of learning *logical definitions of relations*, where tuples that belong or do not belonging to the target relation are given as examples.

Fürnkranz *et al*. (2012) give another definition of the above task:

Given:

- a *target relation* or *concept,*
- *a background knowledge BK*, consisting of a finite set of extensional (ground) or intentional (with variables) Horn clauses.
- *a hypothesis description language* imposing a bias on the form of rules,
- a *coverage function Covered(r, e)* which defines whether a given rule *r* covers example *e*,
- a finite set of examples *E*, divided into positive *P* and negative *N* examples, both expressed by non-empty sets of *ground facts* (definite clauses without variables),

Find:

a *correct hypothesis H* (or a *theory*) composed of first-order clauses such that

- $\forall e \in P : \mathrm{BK} \wedge H \models e$ (*H* is *complete* if it covers all examples that belongs to the concept),
- $\forall e \in N : \mathrm{BK} \wedge H \not\models e$ (*H* is *consistent* if it does not cover any example not belonging to the concept).

Fig. 2.16 illustrates (in-)complete and (in-)consistent hypotheses (rule set *R*).

In practice, it is not possible to find a correct hypothesis that strictly attends both criteria above, i.e., *H* is complete and consistent, and then both criteria may be relaxed.

The ILP setting seen so far aims at learning target concept descriptions in the form of *classification* or *prediction* rules, the so-called *Predictive ILP*. This contrast with *Descriptive ILP* which aims at finding properties describing the data, like association rules [Dehasp and Toironen, 2000], and subgroup discovery [Wrobel, 1997].

In what follows, the ILP learning process outlined in this section is provided.

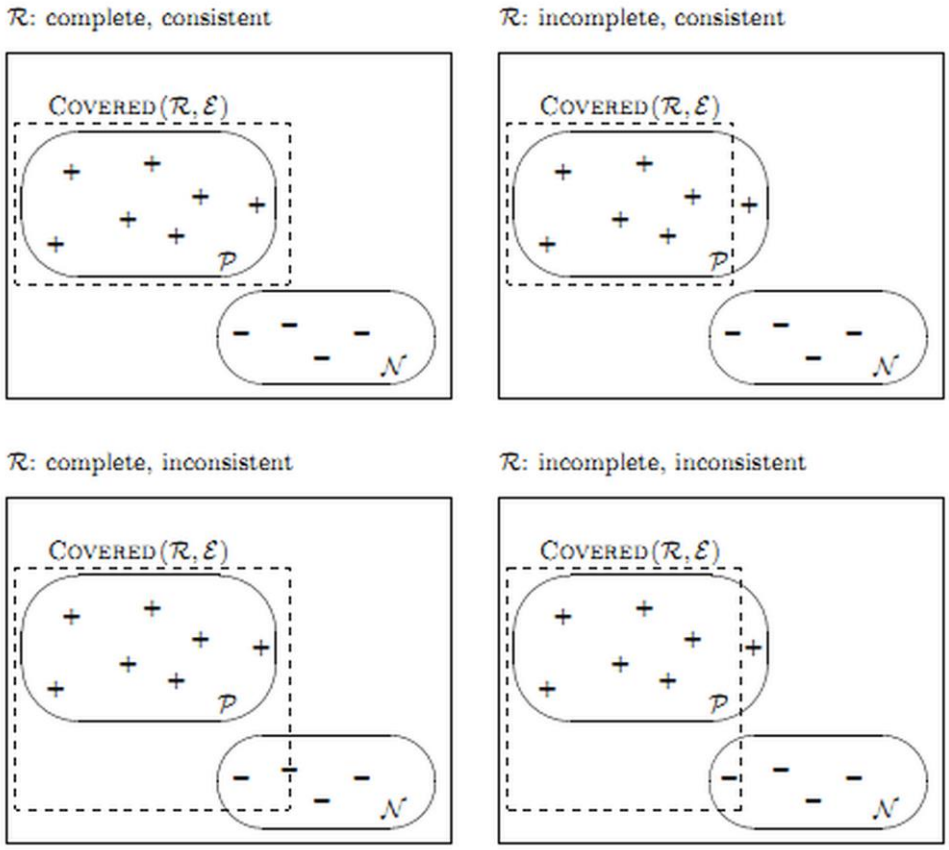


Figure 2.16. The possible cases of completeness and consistency of a hypothesis (rule set R). Extracted from [Fürnkranz, 2012].

2.3.3. Learning Process in ILP

ILP can be implemented by mapping the given problem into a search through a partially ordered hypothesis space [Dzerovski, 2010]. In this search space, each state is a concept description or a hypothesis, and the goal is to find one or more states satisfying some quality criterion. Thus, the search procedure traverses the hypothesis space, *generating* and *testing* the candidate hypothesis.

On ILP, this search procedure is implemented by a *covering algorithm*. The pseudo-code of the covering algorithm is shown in Fig. 2.17.

Covering (E)

Input: set of examples E
Output: a set of consistent rules

1. Learned_Rules = ↓
2. E^+ = Positives(E)
3. **while** $E^+ \neq \downarrow$
4. R = learn_rule(E^+)
5. Learned_Rules = Learned_Rules ∩ R
6. $E^+ = E^+$ - {examples covered by R}
7. **end while**
8. **return** Learned_Rules

Figure 2.17. A generic covering algorithm. The *learn_rule*( ) procedure returns the best rule that covers a subset of the positive examples (E^+).

Iteratively, the covering algorithm in Fig. 2.17 constructs a set of clauses. Starting with a empty set of rules (*Line 1*), the algorithm then generates and evaluates this clause on the positive examples (*Line 4*), adds this clause to the hypothesis (*Line 5)* if it satisfies some criteria, and finally removes the positive examples covered by the clause (*Line 6*). These steps are repeated until all positive examples have been covered (*loop while Line 3*).

In particular, the *learn_rule(e)* procedure in *Line 4* constructs individual clauses by (heuristically) searching the space of possible clauses, structured by a *specialization* or *generalization* operator.

Often, search starts with a very general rule (clause with no conditions in the body), then proceeds to add literals (conditions) to this clause until it only covers positive examples, i.e., the clause is consistent.

This section proceeds with a description of the following subtasks performed by an ILP implementation regarding ILP as a search problem:

i. structuring the hypothesis space;
ii. searching the hypothesis space;
iii. bounding the search;
iv. evaluating the hypotheses.

Structuring the Hypothesis Space. In ILP, enumerating the whole space of possible rules is often infeasible. Consequently, it is desirable not only to structure the search space in order to traverse it systematically, but also to enable pruning of some parts of it.

Therefore, structuring the search space consists of sorting the hypotheses according to a notion of clause ordering, which allows for determining, between two clauses, which one is more general/specific than the other.

Most ordering strategies used by ILP systems are based on *θ-subsumption* [Plotkin, 1971a], which introduces a syntactic notion of generality: given two clauses *C* and *D*, *C*θ

subsumes D if there exists a substitution θ [10], such that $C\theta \subseteq D$. In other words, C is a *generalisation* of D, and D is a *specialization* of C under θ-subsumption.

Another interesting property of θ-subsumption is that it imposes a *partial order* between the clauses, i.e., this notion of generality between clauses is reflexive, anti-symmetric, and transitive. As a result, this notion of generality introduces a *lattice* on the set of clauses [Plotkin, 1971b].

ILP systems use the so-called *refinement operators*, a term coined by Shapiro [Shapiro, 1983], to explore the lattice of hypotheses structured by the θ-subsumption. This operator is defined by a function that generates a set of *specializations* or *generalizations* of a clause.

The specialization refinement operators allow the navigation through the search space from the most general clauses to the most specific ones. This type of refinement operator performs two syntactic operations on a clause: apply a θ-substitution to the clause, or add a literal or set of literals to a clause.

Conversely, generalization refinement operators allows for traversing through the lattice from the most specific clauses towards the more general ones. It also performs two basic syntactic operations on a clause: apply an inverse substitution to the clause, or remove a literal from the body of the clause.

The lattice formed by the hypothesis space of clauses, structured by the θ-subsumption generality ordering is designated as a *refinement graph*. A refinement graph can be viewed as a directed, acyclic graph in which nodes are clauses and arcs denote the basic refinement operations.

Fig. 2.18 shows part of a refinement graph for the *daughter* relation, in which the task is to induce a definition of the *daughter* relation in terms of both *female* and *parent* relations. At the top of this refinement graph (lattice), one can see the clause C = *daughter (X, Y)*. The other clauses connected to C were generated by a specialization refinement operator in this case. In all refinements operations, it is assumed that the hypothesis language is restricted to definite clauses.

The search for a clause in the refinement graph starts at the top of the lattice. Next, all direct refinements are then considered, then their refinements in turn, and so on. This procedure is repeated until a clause that covers only positive examples is finally found.

In the example depicted in Fig. 2.18, the clause

$$daughter\ (X, Y) \leftarrow female(X), parent(Y, X)$$

is the final clause that entails all positive examples.

Searching the Hypothesis Space. Once created the refinement graph for a given target relation, the next step is to adopt a search strategy in order to efficiently traverse the lattice of hypothesis (clauses). There are two main ways of traversing the hypothesis space in ILP: *top-down* or *bottom-up* manner.

In top-down search, one starts with the more general hypothesis. The hypothesis is repeatedly specialized by the application of *downward* refinement operators. In this kind of search, the negative examples are also employed to remove inconsistencies in data.

In bottom-up search, the search starts from a most specific rule (or a bottom rule for a given example), and then generalizes the hypothesis until it cannot further be generalized without covering negative examples by means of a *upward* refinement operator.

Current ILP implementations typically search the refinement graph level-wise, using heuristics based on the number of positive/negative examples covered by a hypothesis.

---

[10] A substitution $\theta = \{V_1/t_1, V_2/t_2, ..., V_n/t_n\}$ consists in assigning terms t_i to variables V_i.

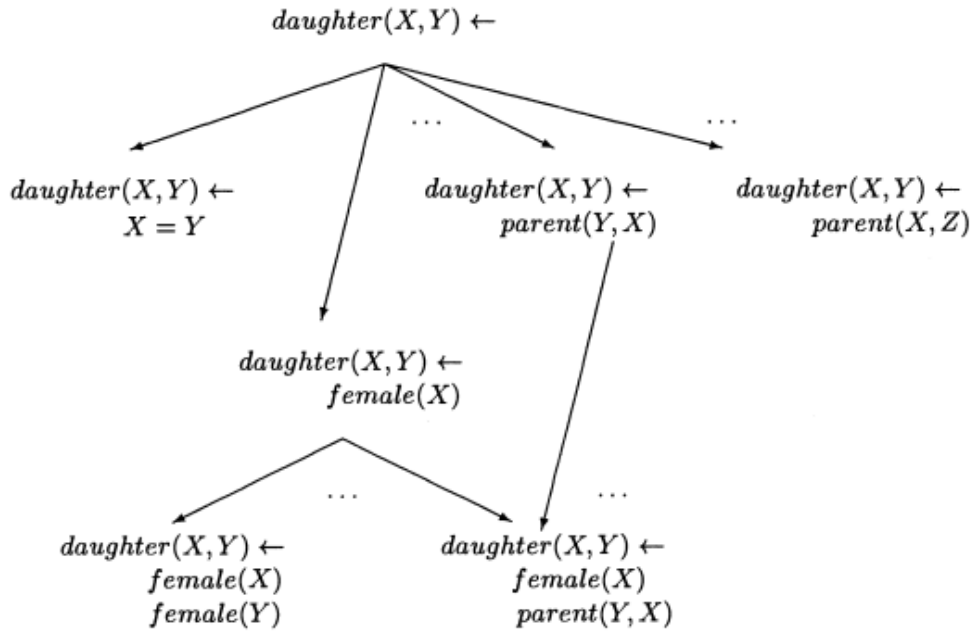


Figure 2.18. Part of the refinement graph for the daughter relation. Extracted from [Dzeroski and Lavrac, 2001].

Due to the large branching factor in the refinement graph, i.e., the number of refinements of a clause, ILP implementations typically rely on *greedy search methods* [Mitchel, 1982] that only consider a limited number of alternatives at each level of the refinement graph. Some search methods include *hill-climbing* that considers only one best alternative at each level; and *beam search* [Mitchel, 1982] which considers *N* best alternatives, where *N* is the beam width.

Another general search method used by ILP systems is the best-first search. Best-first search is similar to the well-known breadth-first search strategy with the difference that the node selected for expansion is based on the output of an evaluation function that estimates the "distance" to the solution.

Bounding the Search. The branching factor of a refinement graph is usually very large. This is especially true for deeper clauses in the lattice that contain many variables. Therefore, it is necessary to find ways to reduce the hypothesis space.

Accordingly, state-of-the-art ILP systems mitigate the combinatorial explosion of the hypothesis space by imposing all sorts of restrictions, mostly syntactic, on candidate hypotheses, aiming at reducing the search space. Such restrictions are called *bias*es.

Thus, the compelling reason to use bias elements in ILP is that they restrict the hypothesis space, addressing the vital aspect of efficiency in ILP systems.

Biases in ILP can be of three types [Lavrac and Dzerovski, 1994]:

- *language bias*, which specifies constraints on the structure or semantics on the clauses in the search space. For instance, one could restrict a given hypothesis to *N* literals.
- *search bias* that defines the way an ILP system searches the space of ordered clauses;
- *validation bias* which defines a stopping criterion during the search.

In what follows, we focus on two types of language bias that further restrict the hypothesis space in ILP, namely: *mode declarations* and *bottom clause*:

- *Mode Declarations*. A very popular approach to make refinement graphs smaller consists in imposing the types of the predicate arguments, as well as input/output *mode declarations* [Muggleton, 1995].

  Mode declarations characterize the format of a valid hypothesis (rule). They also inform both the type, and the input/output modes of the predicate arguments in a rule. There are two types of mode declarations employed by many ILP systems: *head* and *body*.

  Mode head declarations (*modeh*) define the *target predicate*, the head of a valid rule that the ILP system has to induce, whereas mode body declarations (*modeb*) determine the literals (or ground atoms), which may appear in the body part of the rule. Mode body declarations usually refer to predicates defined in the BK, but they can also refer to the target predicate in the case of recursive theories. In the following, the syntax of mode declarations syntax is detailed.

  A mode declaration has either the form *modeh(recall, atom)* or *modeb(recall, atom)*, where *recall* is an positive integer or "*" , and *atom* is a ground predicate. Atoms or ground predicates have the form $atom(type_1, type_2, ..., type_n)$, whereas "recall" specifies the maximum number of allowed instantiations of the ground predicate, where "*" means that the ground predicate can be instantiated any number of times. In other words, "recall" indicates the *non-determinancy* of a predicate in Prolog. A Prolog predicate may be *determinate* or *non-determinate*. A predicate is determinate if it succeeds at most once given a particular instantiation of its input arguments, i.e., it looks like a function. A non-determinate predicate may have more than one solution when their input arguments are instantiated.

  In mode declarations, each argument of a ground predicate has a *type* and an associated *symbol* that appears before the type indicator: *+type(input)*, *-type(output)* or *#type(constant)*.

  The meaning of each symbols (+, - , #) is as follows:

  - *Input (+)***.** An input variable of type *T* in body literal B_i appears as an output variable of type *T* in a body literal that appears before B_i, or appears as an input variable of type *T* in the head of the clause.
  - *Output (-)***.** An output variable of type *T* in the head of the clause must appear as an output variable of type *T* in any literal of the body of the clause.
  - *Constant (#)***.** An argument denoted by *#T* must be ground with terms in *T*.

  For instance, consider the predicate *lives_in(X, Y)* that requires the first argument being a PERSON entity, and the second, a CITY entity. Then, the following mode declaration *lives_in(+person, -city)* specifies that the person has to be given as input when calling this predicate.

- *Bottom Clause.* Type and mode declarations can be combined with the construction of a bottom clause that bounds the search of the refinement lattice from below.

  The bottom clause is the most specific clause covering a given example (or examples). As a result, only clauses on the path between the top and the bottom clause will be considered, significantly improving efficiency.

  The two most used techniques for constructing the bottom clause in ILP are the *relative least general generalization* (rlgg) of two (or more) examples [Muggleton and Feng, 1990] and the *most specific inverse resolvent* of an example [Muggleton, 1991], both taking into account a given background knowledge.

Discussion about the use of biases in ILP. A final discussion about the use of biases in ILP concerns the results presented in [Tausend, 1994]. Tausend reports a study of the impact of

several bias constituents in the size of the search space. His conclusion is that not all combinations of bias elements are useful.

In reality, if the bias is not restrictive enough, then the search space becomes computationally intractable. On the other hand, if the bias constrains the hypothesis space too much, than the correct solution cannot be found. For example, if the search is restricted to theories with 4 literals, but all correct theories contain clauses with 5 or more literals, then no solution will be found. From the exposed, it is clear that there exists a trade-off between efficiency in the search and quality of the theory to be found.

The conclusion is that it is crucial to choose a balanced bias, i.e., not too weak nor too strong, since an inappropriate bias may prevent the ILP system from finding the best hypotheses.

Hypothesis Evaluation. In addition to the constraints imposed by the declarative bias seen earlier, the user may provide general preferences, for example, towards shorter clauses, shorter final theories, or towards clauses with higher predictive accuracy on the training data. In fact, ILP systems need to choose among partial solutions, i.e., they use *evaluation functions* that try to estimate how close they are to the best solution. Such evaluation functions calculate clause´s coverage over positive and negative examples.

Typically, the computation of evaluation measures requires knowing the correct and the predicted classifications for each class. This is better visualized by constructing a *confusion matrix* that lists the correct against the predicted classifications for each class. Tab. 2.5 shows the classical confusion matrix for a two-class (binary) classification problem. In this matrix, each column represents the examples in a predicted class, while each row represents the examples in an actual class.

Table 2.5. Confusion Matrix for a two-class (binary) classification problem.

	Class Positive	**Class Negative**
Prediction Positive	True Positives (TP)	False Positives (FP)
Prediction Negative	False Negatives (FN)	True Negatives (TN)

The number of correct predictions for each class will be located along the main diagonal of the matrix (*True Positives* and *True Negatives*). Incorrect predictions can be of two types: *False Negative* when a positive example was incorrectly predicted as Negative; and *False Positive* is the opposite.

Tab. 2.6 lists some of the most common rule evaluation functions used by current ILP systems in function of *TP*, *TN*, *FP*, *FN*, and *L* (the number of literals in a given rule):

Table 2.6. Rule evaluation measures commonly used in ILP

Measure	**Formula**
Coverage	TP – FP
Compression	TP – FP – L
Compression Ratio	(TP – FP) / L
Laplace	(TP + 1) / (TP + FP + 2)
Accuracy	(TP + TN) / (TP + TN + FP + FN)
Precision (P)	TP / (TP + FP)
Recall (R)	TP / (TP + FN)
F-Measure (F1)	2 * P * R / (P + R)

where *TP*, *FP* are respectively the number of positive and negative training examples covered by a candidate hypothesis; and *L* is the number of literals in the current candidate hypothesis.

The two most often used measures are *coverage* and *compression*.

Besides being the simplest kind of evaluation measure in ILP, the *coverage* measure has a definite advantage: it is very robust to datasets containing an unbalanced distribution between positive and negative examples. The reason concerns the fact that this measure does not consider true negatives (TN) in its formula, contrarily to what occurs in the accuracy measure (see Tab. 2.6).

The difference between the two measures above is that the former, besides computing the difference between the number of positive (TP) and negative examples (FP) covered by the rule, also takes into consideration the size of the rule. Thereby, on the case of two rules with the same evaluation score, the compression measure will prefer shorter rules.

The *accuracy* evaluation measure reflects the overall correctness of the rule, i.e., the fraction of examples correctly classified (*TP + TN*) in relation to the total number of examples *(TP + TN + FP + FN)*. From this measure, it can be also derived the error rate as (1 – accuracy). This measure can give biased results in the case of learning from a highly unbalanced number of negative examples compared to the number of positive examples.

The *Laplace* measure assumes a uniform prior probability over the positive and negative classes [Fürnkranz *et al.*, 2012].

It worth noting that *precision*, *recall* and *F-measure* evaluation functions used in ILP have the same meaning as employed in Information Retrieval. Therefore, the precision measure will attain its optimal value when no negative examples are covered by the rule.

2.3.4. A Relational Learning Example

The following example illustrates the task of learning logical definitions of relations, or target predicates in ILP. Given training examples as input, an ILP system induces a predicate definition corresponding to a relational view that defines the target relations in terms of others relations given in the background knowledge.

Consider the classical Michalski´s East-West train learning problem [Michie *et al.*, 1994]. Here, the learning task is to induce a concise Prolog program for classifying trains as eastbound or westbound (Fig. 2.19).

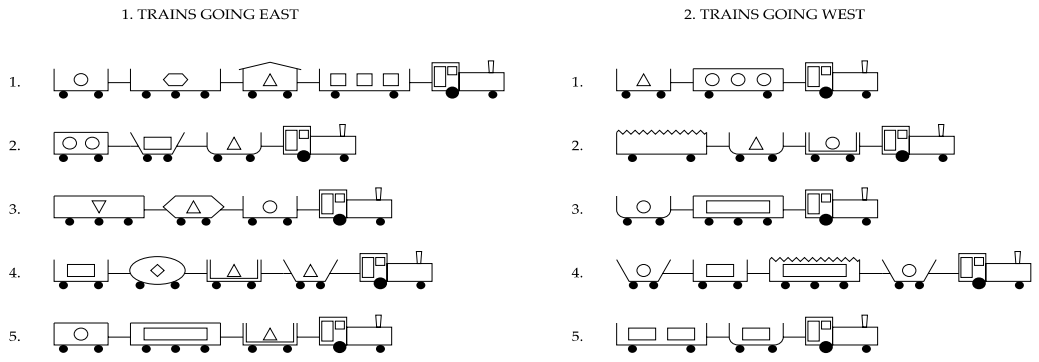


Figure 2.19. The ten train East-West challenge.

The set of training examples are composed of 10 trains (5 *eastbound*, 5 *westbound*). Each train consists of 2 to 4 cars; cars have attributes like *shape* (rectangular, oval, u-shaped, ...), *length* (long, short), number of *wheels* (2 or 3), type of *roof*(open, closed, peaked, jagged), *load shape* (circle, triangle, rectangle, ...), and *number of loads* (1-3).

Here is a list of some Prolog facts representing this problem:

- *eastbound(T)*: trains *t1* to *t5* (left side of the Fig. 2.19)
- *westbound(T)*: trains *t6* to *t10* (right side of the Fig. 2.19)
- *car(C_{ij})*: the i^{th} car of the j^{th} train
- *c_number(T, n):* the trains has *n* cars
- *has_Car(T, C)*: the train T has car C_{ij}
- *c_property(C, prop):* car length, car roof type, etc. *Ex.: c_length(C, short), c_roof(C, open)*
- *l_property(L, prop):* load property, such as *l_shape* (circle, oval, etc.)

The set of facts describing the properties of the first train are listed in Fig. 2.20. Each train can be represented by several facts which provide an extensional definition of the properties of a particular train.

```
eastbound(t1).                    car(c11). car(c12). car(c13). car(c14).
cnumber(t1,4).                  load(l11). load(l12). load(l13). load(l14).

has_Car(t1,c11).    has_Car(t1,c12).    has_Car(t1,c13).    has_Car(t1,c14).
c_shape(c11,rect).  c_shape(c12,rect).  c_shape(c13,rect).  c_shape(c14,rect).
c_length(c11,long). c_length(c12,short). c_length(c13,long). c_length(c14,short).
c_roof(c11,open).   c_roof(c12,closed). c_roof(c13,open).   c_roof(c14,open).
c_wheels(c11,2).    c_wheels(c12,2).    c_wheels(c13,3).    c_wheels(c14,2).
hasLoad(c11,l11).   hasLoad(c12,l12).   hasLoad(c13,l13).   hasLoad(c14,l14).
lshape(l11,rect).   lshape(l12,tria).   lshape(l13,hex).    lshape(l11,circ).
lnumber(l11,3).     lnumber(l12,1).     lnumber(l13,1).     lnumber(l14,1).
```

Figure 2.20. BK predicate definition of the first eastbound train in Mikalski´s problem.

Finally, an ILP system could induce the following target predicate that classifies between eastbound and westbound trains:

eastbound(T) :- has_Car(T, C), c_length(C, short), c_roof(C, closed).

This predicate states, "*a train is eastbound if it contains a short closed car*".

2.3.5. Advantages and Limitations of ILP

ILP presents several advantages as a machine learning method:

- *High expressivity in representation*: a distinguishing feature of ILP resides in its use of first-order language to represent more complex concepts than most other machine learning approaches based on attribute-value (propositional or zero-order) representation. In other words, ILP systems can be applied on multi-relational data to find patterns that involve multiple relations (tables), while most other statistical machine learning approaches can only deal with a single table at a time. In addition, ILP can learn recursive concept definitions [Raedt, 2008].
- *Effective use of background knowledge*: The prior knowledge about a problem may be coded and integrated into an ILP system which incorporates this kind of knowledge into the solution. For instance, any constraint to the problem can be expressed in the form of auxiliary predicate definitions provided by the user as additional BK.
- *Rules readability*: Usually, ILP systems generate rules in declarative form, which means that hypotheses are understandable and interpretable by humans. Furthermore,

if the background knowledge of a given problem is structured, a first-order language is certainly easier to read.

On the other hand, ILP also presents two major limitations:

- *High computational cost:* The downside of the greater expressiveness in ILP is that such flexibility comes at a high computational cost, and ILP systems are known for their difficulty in scaling-up [Page and Srinivasan, 2003].

  Indeed, the evaluation of a single hypothesis in ILP involves testing if the hypothesis, along with the background knowledge, entails the examples used in training. As a result, the time required to evaluate this hypothesis time mainly depends on the size of training data, i.e., the computational effort required to evaluate the hypothesis given a BK. The evaluation of a single hypothesis for a relatively large set of training examples may take a long time. The main reason is that most of ILP systems executes in main memory, limiting its application on large databases.

- *Difficulty in dealing with pure numerical data*. Another limitation is that the classical ILP framework has difficulties to deal with numerical data and uncertainty [Bratko, 2000]. Aiming to mitigate the limitations mentioned above, research work has proposed better heuristics to traverse the search space [Fürnkranz *et al*., 2012], the use of parallelism [Fonseca, 2006], hybrid approaches based on ILP and probabilistic methods [Raedt, 2008].

2.3.6. ILP Systems

In the following, some current ILP systems that implement the ideas of the inductive learning framework seen so far are briefly described. A more detailed list of ILP systems, in addition to the presented here, can be found in [Kavurucu *et al.*, 2011].

Progol [Muggleton, 1995]. It consists of an iterative top-down ILP system that performs batch learning, that is, all of the examples and the BK must be defined before starting the algorithm. In order to reduce the hypothesis space, it makes use of the BK and the examples *E* (positive and negative) and requires a set of *mode declarations* as input.

The mode declarations indicate the BK predicates appearing in the head of clauses, and in the body of clauses. They also restrict the valid argument types for the BK predicates.
In Progol, a bottom clause is constructed from a positive example and is derived using inverse entailment. Progol begins the search with an empty body hypothesis and traverses the refinement graph whose elements are literals contained in the bottom clause.

Golem. Golem [Muggleton and Feng, 1990] is a bottom-up ILP system, which constraints the search space by employing Plotkin's notion of *relative least general generalization* (rlgg) [Plotkin, 1971] as the hypothesis formation technique. The rlgg constitutes the less general hypothesis concerning the BK with respect to the positive examples.

For generating a single clause in the final theory, Golem first randomly picks several pairs of positive examples. Then, in the next loop, Golem computes the rlgg of the current best hypothesis and randomly selects examples, and so on. The system then continues until a loop does not increase the coverage of the current best hypothesis, choosing the one with greatest coverage. The covered positives are removed from the input and the algorithm will be applied to the remaining positive examples. More details about Golem can be found in [Muggleton and Feng, 1990].

FOIL [Quinlan, 1990]. FOIL uses the BK in an extensional way, i.e., relational tuples or facts are used and the target language is function-free Horn clauses. The induction of a single clause in FOIL begins with an empty body clause. The body of the clause is

iteratively specialized by a greedy algorithm that adds literals to the main clause. These literals are chosen by placing variables in appropriate positions of the arguments that must be specified earlier. To choose a literal among many possible candidates, FOIL uses the *information gain heuristic* which is based on how much the literal contributes to the distinction between positive and negative examples.

Dl Learner. DL-Learner [Lehmann, 2009] consists of an ILP system for learning class expressions in OWL/DL from instances. Therefore, this system extends the ILP to the Description Logic and the Semantic Web. In the most common scenario, the BK is expressed in OWL, as well as, the positive and negative examples. Each example In DL-Learner is an individual (class instance) of the working ontology. DL-Learner aims to find the OWL class expression such that all or many positive examples are entailed by this expression, and little or no negative example is covered at the same time. Thus, after learning class expressions of an ontology, it can classify new instances as well. For practical reasons, DL-Leaner is biased towards shorter class descriptions.

Aleph (A Learning Engine for Proposing Hypothesis). The top-down ILP system Aleph[11] uses a specialization approach (top-down refinement operators) which starts with the most general hypothesis, i.e., considering all examples as positives. Then, it refines this hypotheses by repeatedly adding a literal from the bottom-clause that best improves the hypothesis score. As a result, this new rule becomes more specific than the previous one, and will cover only a subset of the examples previously covered.

Aleph implements the *Progol* algorithm [Muggleton, 1995]. Progol main advantage is the use of a bottom clause to guide the search. The bottom clause, the most specific clause that entails a selected example, is built using *inverse entailment* [Muggleton, 1995]. Progol algorithm performs the following steps:

1. A positive example, or a seed example, is selected to be generalised, using the order of the examples in the training set of examples.
2. The bottom clause ($\perp$) concentrates all constraints related to the BK, language bias, and the seed example. This step is also called saturation step. The idea motivating the use of bottom clause is that, by construction, all clauses in a refinement graph search are guaranteed to cover at least the example associated with the bottom clause.
3. The search for clause begins with a general-to-specific top-down hypothesis space search, bounded by the most general possible hypothesis and the bottom clause. For that, Aleph refines a clause by repeatedly adding literals from the bottom-clause. This new rule will be more specific, covering only a subset of the examples previously covered. This step is the "reduction step".
4. The best clause found so far is added to the final theory, and the examples covered by it are removed from the training set. This process continues until there are no more examples in the training set, otherwise it returns to step 1.

ProGolem. ProGolem [Muggleton and Santos, 2009] is one of the ILP systems implemented in GILPS (*General Inductive Logic Programming System*) [Santos, 2010] which combines approaches from Progol [Muggleton, 1995] and Golem [Muggleton and Feng, 1995]. It uses a bottom clause relative to an example to guide the search like in Progol. It also performs a variant of Golem's bottom-up search, based on *Asymmetric Relative Minimal Generalization* (ARMG) [Muggleton and Santos, 2009]. The ProGolem algorithm uses ARMG to traverse the hypothesis space following a specific-to-general bottom-up subsumption order relative to the bottom clause.

[11] The Aleph Manual. http://comlab.ox.ac.uk/activities/machinelearning/Aleph/aleph.html

It first randomly selects a positive example e_i, like in Progol, and constructs its *bottom clause:* the most specific hypothesis that explains the example e_i. This step is also called *saturation step*. Next, in the *reduction step*, it successively drops a minimal set of literals from the body to allow coverage of one additional positive example. This hypothesis refinement step continues until the hypothesis score stops improving. Then, the hypothesis is added to the theory, and all the positive examples covered by it are removed from the training set. The cycle of saturation and reduction continues on the remaining examples until all positive examples are covered or no new rules can be found. At this point, ProGolem outputs the final theory, the set of the best rules found so far.

2.3.7. Discussion on Top-down vs. Bottom-up ILP Systems

Earlier in this section, two representative ILP systems, namely Aleph and ProGolem are described. Then, the section continues with a discussion the advantages and limitations of these ILP systems, aiming at motivating the choice of an ILP system as the core learning component in OntoILPER framework architecture.

The widely used ILP system is Aleph. Its popularity seems to be due to the fact that it was conceived as a workbench for implementing and testing, in a single Prolog file, several concepts and procedures from various ILP systems proposed in many papers. The main Aleph advantage is the plethora of other ILP systems that it can emulate. However, despite all interesting options Aleph gives to the user, Santos (2010) and Nassif (2012) identified the following limitations:

- Aleph performs a top-down search, with literal concatenation being the main refinement (specialization) operator. That means that a clause is refined in small incremental steps, which can cause many of the evaluated clauses to be very similar to each other.
- This “one-step lookahead” search strategy assumes that the literals are independents and, even several lookahead and backtracking steps cannot capture complex predicate dependencies.
- It adopts a local theory construction method. This method depends upon the ordering of the positive example. Thus, the best rules are not guaranteed to be found. By removing examples from the training set, it may appear the situation in which the best rules are not generated because the best rules would be generated by examples that were removed in previous sub-optimal learned rules.

In contrast, ProGolem implements a bottom-up search with the ARMG being the main generalization operator. The ARMG consists of a powerful generalization operator in large leaps [Santos, 2010]. In addition, ProGolem implements a *global theory construction* method, which ensures that the theory is only constructed after all rules have been generated. As a result, ProGolem is not dependent of the order of examples like Aleph.

Furthermore, ProGolem implements specialized coverage engines that can efficiently compute coverage of long, *non-determinate* clauses. A non-determinate clause has more than one possible solution.

The above ProGolem features suits well for RE because ProGolem drastically reduces the runtime per evaluated clause. For the reasons exposed above, ProGolem was adopted as the inductive logic programming system in OntoILPER framework.

Chapter 3

Information Extraction

A large amount of information on many different subjects and formats is published on the Web continuously. This information volume was made possible due to the advances in computer technology that simplifies the creation, storage, and distribution of information on the Web. Much of this information, stored in an unstructured form, is spread over thousands of individual computers (*hosts*) on the Web what results in a huge and complex online database. However, this lack of structure greatly limits the use of this information. Besides that, current technologies for searching web pages based on keywords are not mature enough to provide the user with the specific information she needs [Freitas, 2002].

In this context, Information Extraction (IE) as a means for finding structured information from unstructured or semi-structured text can be of great help. The main goal of IE consists in discovering and structuring information found in semi-structured or unstructured documents [Jiang, 2012]. IE is an important task in Text Mining and has been extensively studied in various research communities including Natural Language Processing (NLP), Web Mining, Information Retrieval (IR), among others. More concretely, IE can be regarded the task of automatically identifying and recovering certain entities, relations or events in text sources.

This chapter is organized as follows: Section 3.1 gives an overview of the IE tasks, general architecture, and applications of IE systems. Section 3.2 and 3.3 introduce the main approaches to Information Extraction, proposing a broad classification of the IE systems based on several published surveys. Supervised machine learning techniques applied to IE are the subject of the Section 3.4. Section 3.5 is dedicated to Ontology-based Information Extraction, a specialized branch in IE. The motivation, the potential, and the classification of state-of-the-art OBIE systems are also described in that section. Finally, Section 3.6 concludes the chapter and presents the focus of this thesis.

3.1. Introduction

3.1.1. Types of Extracted Elements

In IE, documents can be categorized as structured, semi-structured, and unstructured ones. Structured documents follow a predefined and strict, but usually unknown, format where itemized information presents uniform syntactic clues. Semi-structured documents, like web pages, consist of free text merged with HTML tags which can define tables, and other structures within the page. Unstructured pages usually consist of free text.

Comparing IE systems with those in IR, the first ones give a step forward because they are capable of finding out specific information that was previously defined by the user, instead of returning a link to a document, leaving the user with the task of finding the desired information.

For instance, consider the follow sentence in English:

"Microsoft was founded by Bill Gates and Paul Allen in 1975."

Then, an IE system is able to extract the following information:

isFounderOf(Bill Gates , Microsoft),
isFounderOf(Paul Allen , Microsoft),

FoundedIn(Microsoft, 1975),

Such structured information is now ready to be presented to an end user, or it can be used by another computer application, including search engines, in order to provide better services to end users.

From the textual fragment shown in Fig. 3.1, an IE can extract four basic elements:

- *Entities* are the basic building blocks that can be found in text documents. Examples include people, companies, locations, genes, drugs, etc.
- *Attributes* are features of the extracted entities, including person's title, the type of an organization, etc.
- *Facts* are the relations that exist between entities. An employment relationship between a person and a company is a typical example of a factual relationship.
- *Event* is an activity or occurrence of interest in which several entities participate. Examples of an event include a terrorist act and a merging between two companies.

Fig. 3.1 illustrates several entities and relationships found by an IE system from a document in free natural language text.

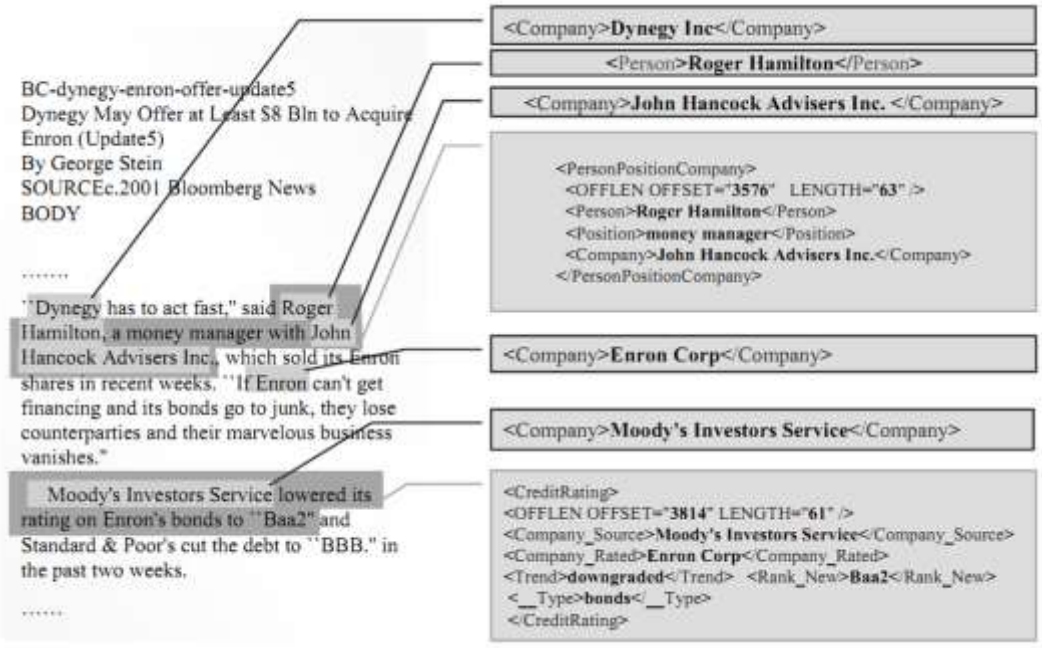


Figure 3.1. Example of a document annotated by an IE system. Note that entities and relations are first identified for composing more complex facts and events. Extracted from [Ben-Dov and Feldman, 2010]

3.1.2. Historical Perspective of IE and Types of IE Tasks

Much NLP research was funded by U.S. government agencies in the 1970s and 1980s, but these agencies became frustrated at the difficulty of evaluating competing approaches because researchers chose their own issues, processing methods, evaluation methods, and data. This frustration motivated an effective solution concerning the establishment of a methodology of competitive evaluation, where sponsored researchers and others would agree to develop systems to process the same data, and formalize their analysis results in a standard notation.

Accordingly, a series of seven Message Understanding Conferences (MUC) was held, with the last in 1998 [Hirschman, 1998] which were responsible for the major contribution to the development of IE by proposing standards to the IE community.

The MUC conferences [Grishman and Sundheim, 1996], sponsored by DARPA[1], provided corpora and standardized assessment criteria and invited developers of IE systems for competing with their proposed solutions. This led to the establishment of the evaluation metrics for assessing IE systems and conceiving techniques that allowed researchers to extract effectively information from textual sources, mainly from unstructured ones. One of these techniques, the manual development of extraction rules that rely on patterns in the form of regular expressions attracted the attention of many researchers after the FAUSTUS system [Appelt *et al.*, 1993] that used this technique and was second on the 4th edition of MUC. The names of the various tasks identified in MUC evaluations, and the methods used for evaluation in MUC, have become widely adopted and adapted outside MUC ever since.

Early MUCs defined information extraction as filling a predefined template that contains a set of predefined slots (see Fig. 3.2). *Template filling* can be regarded as a complex task and systems developed to fill one template cannot directly work for a different template. In MUC-6, a number of template-independent subtasks of information extraction were also defined [Grishman and Sundheim, 1996]. These include named entity recognition, coreference resolution, and relation extraction.

The IE area proved also to be very useful for business organizations. For this reason, the domain related to companies was used in the subsequent MUC conferences as well as in the Automated Content Extraction (ACE)[2] campaigns, the MUC´s successor.

At present, due to both the exponential growth of Bioinformatics literature and the infeasibility of processing all this information manually, IE systems have been heavily exploited as tools for populating bioinformatics databases, and biomedical ontologies, among others. Furthermore, various competitions such as the *BioNLP* [BioNLP, 2009] and *BioCreative* [BioCreative, 2006] have been organized for EI systems dealing with bioinformatics corpora.

Returning to the extracting elements mentioned earlier, the MUC conference established the following tasks summarized in Tab. 3.1.

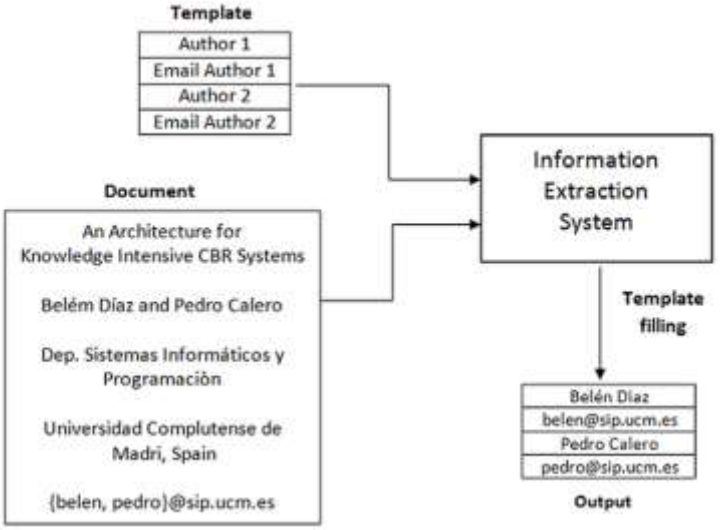


Figure 3.2. Multi-slot template extraction example.

1 Defense Advanced Research Projects Agency (*DARPA*). *http://www.darpa.mil*
2 Automatic Content Extraction. http://www.itl.nist.gov/iad/mig/tests/ace/

Table 3.1. MUC Information Extraction Tasks [Grishman and Sundheim, 1996]

Task Name	*Description*
Named Entity Recognition (NER)	extracts people´s names, organizations, locations, numeric and temporal expressions
Coreference Resolution (CR)	links references to the same entity
Template Element(TE)	extracts identifying and descriptive named entity attributes
Template Relation(TR)	extracts specific relationships between NE´s (simple facts).
Scenario Template (ST)	extracts events by filling one or more slots with instances of TE´s or TR´s.

3.1.3. Named Entity Recognition and Relation Extraction

Since the NER and TR extraction tasks are the focus of this thesis, they are explained in more details in this section.

Named Entity Recognition

Named Entity Recognition (NER) was formally introduced in 1995 by the sixth MUC as a subtask in IE. The aim of Named Entity Recognition (NER) is to identify named entities from text and to classify them into a set of predefined types such as person, organization and location, among others.

NER consists of the most fundamental task in IE, since the extraction of more complex structures such as relations and events depends upon accurate NER as a previous step. These types are the most useful for many application domains. Other expressions such dates, time, monetary values and percentages, which were introduced in MUC-6, are viewed as named entities, although strictly speaking these expressions are not named entities.

Other application domains have also been attracted to the NER, like the biomedical domain. In this domain, named entities to be extracted refer to biological or medical terms in unstructured text. Such entities include gene and protein names, medical problems and treatment, drug names, to name a few.

In the biomedical domain, NER has been notably challenging for two reasons [Yeh *et al.*, 2005]: the dynamic nature of scientific discovery that constantly increases in this domain; and the abundant use of synonyms, acronyms/abbreviations make it difficult to identify the concepts with these terms.

Relation Extraction

Relation Extraction (RE) consists of *detecting* and *characterizing* semantic relations between entities in text. By detecting, one refers to the task of only determining if a relation between two entities holds, whereas by characterizing, the classification problem of assigning a *relation type label* to a particular relation instance is addressed.

The research in RE has been promoted by the MUC which held several editions of this event from 1987 to 1997 under the supervision of DARPA. Later, the NIST Automatic Content Extraction (ACE) program continued to organizing campaigns (ACE from 2002 to 2008) on IE in general and RE in particular. The ACE workshop is considered the best world forum for the comparison and evaluation of new technologies in the field of IE, including NER, RE, event extraction, and temporal IE.

Much of the work on RE in those ACE programs focuses on *binary relations*, i.e. relations between two entities, or arguments. A set of major relation types and their subtypes are defined by ACE. Examples of such major relation types include *physical* (e.g. an entity is physically near another entity), and *employment/affiliation* (e.g. a person is employed by an organization).

Fig. 3.3 shows the overview of the RE task with its input and output accompanied with two examples.

The left side of Fig. 3.3 shows a more abstract representation of the RE task in terms of its input and output. This input consists of documents in natural language containing instances of relations to be extracted, whereas the output consists of the triples representing the instances of relations holding between two entity instances. The right side of the same figure depicts an example comprising two input sentences and the final extracted instances, where:

i. "*American saxophonist David Murray recruited Amidu Berry*";

ii. "*Cdc3+ encodes profilin, an actin-monomer-binding protein*".

Sentence (i) contains two relation mentions: "Citizen" between "David Murray" and "American"; and "Business" between "David Murray" and "Amidu Berry". Sentence (ii) is from a biomedical corpus, and one can also identify two protein-protein relation instances.

3.1.4. Evaluation Metrics for Information Extraction

The assessment of IE systems is performed by the evaluation metrics proposed at MUC [Hirschman, 1998]. The study conducted by the first four MUC editions formed the basis for the definition of existing evaluation metrics in IE.

Initially these metrics were developed based on *Precision* and *Recall* metrics originally proposed in Information Retrieval (IR) [Baeza-Yates and Ribeiro-Neto, 1999].

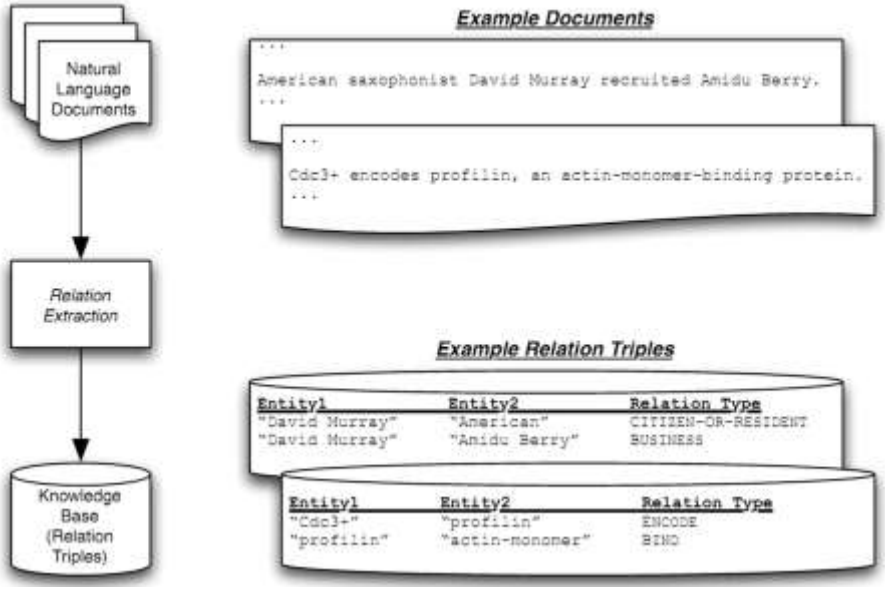


Figure 3.3. Overview of the RE task with extracted instances.
Extracted from [Hachey *et al.*, 2011].

Precision and Recall metrics are generally defined in terms of the elements indicated by *contingency table*, or *confusion matrix* (Fig. 3.4) for binary classifiers, like the following

	Golden Standard	
Predicted	True	False
True	*tp*	*fp*
False	*fn*	*tn*

Fig. 3.4 Confusion matrix of a binary classifier

where *tp* consists of true positive (instances correctly classified as belonging to the target class,), *fp* denotes false positives (instances incorrectly classified as belonging to the target class), *fn* consists of false negatives (instances incorrectly classified as not belonging to the target class), and finally, *tn* denoting true negatives, i.e., instances correctly classified as not belonging to the target class.

Thus, both *Precision P* and *Recall R* can be defined in function of the values from the confusion matrix above as follows:

$$P = \frac{tp}{tp + fp} \qquad R = \frac{tp}{tp + fn}$$

where *P* is the proportion of correct instances among all of the instances that the classifier was able to assign to the target class, while *R* is the proportion of correct instances among all of the instances of the target class present in the input corpus.

Another metric that combines *P* and *R* is the harmonic mean *F* of *P* and *R* defined as follows:

$$F = 2.\frac{P.R}{P + R}$$

It is important to emphasize that it is hard to optimise both precision and recall at the same time because they present a classical trade-off between them, i.e., if on the one hand, one tries to obtain a high precision classifier, then it is possible that relevant information in the input data will be missed or even ignored. On the other hand, if one works towards a high recall score, then it is likely that spurious instances of the input data will be extracted as well.

The bottom line is that one has to decide on which aspect (completeness or correctness) is more important for a particular IE task.

3.1.5. General Architecture of an IE System

The general architecture of an IE system is illustrated in Fig. 3.5. Typically, an IE system includes three processing steps [Hobbs *et al*., 1997]:

Text Preprocessing. From the very beginning, the main issue in IE appeared to be the design of efficient extraction rules able to separate relevant from the non-relevant information. However, the intrinsic richness, ambiguity and complexity of natural language, in which a given word may have different meanings (polysemy) can make the task very difficult.

To alleviate this problem, state-of-the-art IE systems rely on a pre-analysis of the input texts. For instance, based on the presence of a given specific lexical item, the word distance or order, a whole set of very specific rules can be designed for each new IE application. The working hypothesis here is that if the text is pre-analysed, information extraction rules can be expressed in a more abstract and powerful way, since the rules are applied on a normalised representation of the text produced by a previous natural

language analysis. In this case, very specific rules can be designed for each new IE application.
More concretely, Text Preprocessing starts with the text segmentation into sentences and words, followed by more advanced NLP linguistic analysis. This natural language processing is done by pipelining previous analysis with the current ones (Fig. 3.6). For instance, the linguistic analysis should segment the text into lexical units first, and then identify the named entities as well as the verbs appearing in the sentence, and finally establishing candidate relations among them.

Rule Selection. Information extraction rules are associated with triggers, or keywords. Such extraction rules specify the conditions that pre-processed text must match and how the relevant textual fragments can be interpreted to fill the extraction templates.
It is worth noting that the extracted information is determined by a set of patterns or specific extraction rules of a given domain. The definition of such rules can be performed manually by a specialist in the relevant field, or with different degrees of automation using machine learning algorithms, e.g., *supervised, semi-supervised or unsupervised.* Details about these machine learning techniques among others are presented in the next section.

Rule Application: Once a rule has been triggered, all contextual conditions of the rules are checked and the form is filled according to the conclusions of the matching rules.

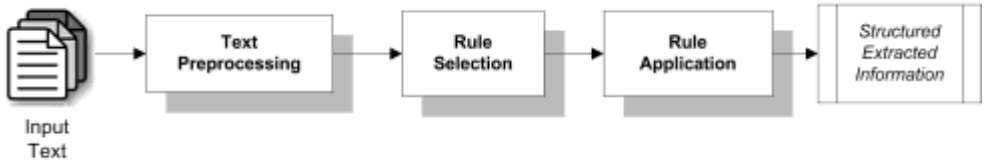


Figure 3.5. General architecture of an IE system.

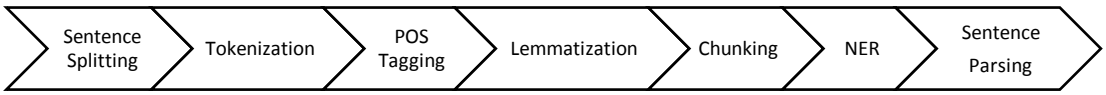


Figure 3.6. Common pipeline of natural language analysis performed by IE systems.

3.1.6. Information Extraction Applications

IE has applications in a wide range of domains. Depending on the need of a particular application, a specific kind of (structural) information must be defined. It follows three of the various successful current applications of IE:

- *Biomedicine*. Biomedical researchers often need to look for particular genes, proteins or other biomedical entities in a large amount of scientific publications. Due to the ambiguous names and synonyms for biomedical entities, the simple search based on keyword matching might not succeed. In this context, IE then can help by mining biomedical literature to automatically identify mentions of biomedical entities from text and to link them to their corresponding entries in existing biomedical knowledge bases.
- *Search Engines*. Undoubtedly, web search engines have become an integral part of people´s daily need for retrieving information on the Web. However, search based on bag-of-word representation of documents [Baeza-Yates and Ribeiro-Neto,

1999], performed by such search engines cannot, in most cases, provide satisfactory results. A more interesting possibility concerns the more advanced search problems such as entity search, structure search, and even question answering that can provide users with much better results. In this scenario, IE can again help as a preprocessing step aiming at enriching document representation or even populating an underlying database.

- *Business Intelligence*. Financial professionals often need to seek specific pieces of information from news articles to help their day-to-day decision making.

3.2. Main Approaches to Information Extraction

Two main approaches for building information extraction systems have been proposed: the *knowledge engineering* or *classical* IE, and the *automatic training* approach, also referred to as adaptive IE. In the following, these two IE approaches are presented.

Most of the information extraction systems that have participated in the MUCs are rule-based systems [Turmo *et al.*, 2006] [Tang *et al.*, 2007]. In such systems, extraction patterns or rules developed manually by humans were used to match text for pieces of relevant information. This way of creating extraction rules is also referred to as knowledge engineering approach because a knowledge engineer creates the rules by hand.

In this approach, the knowledge engineer has to be very familiar with both the rule-making process and the domain in order to be able to effectively create good extraction rules. It is evident that this approach takes plenty of time in the process of rule development because it consists of an iterative process. Usually, this process is carried out as follows: In the first interaction, the knowledge engineer generates a set of rules, then it applies these rules on a set of documents (the tuning set) and, if necessary, she changes the rules again to obtain a better coverage of the domain. In the subsequent iterations, the knowledge engineer tries to obtain a satisfying set of rules with the correct level of generalisation. It is crucial to this approach that the generated set of extraction rules does not overfit or underfit the specific task after each iteration step. Finally, the knowledge engineer stops this iterative process when a set of extraction rules with a satisfying level of accuracy is achieved. Clearly, in such an approach, the user skills play a crucial role in the successful identification and extraction of relevant information.

Although this approach can achieve acceptable performance on the specific target domain, the manual development of rules is labour intensive, and the resultant rules are usually highly domain dependent [Jiang, 2012].

Realizing the limitations of these manually developed systems, researchers turned to other means of developing extraction rules more adaptive in the sense that it should require less effort in the development process.

The adaptive IE approach instead exploits machine learning techniques to induce extraction rules starting from a set of information patterns that are marked for extraction by a user. In this approach, the focus is on automating the rule generation process partially or fully in order to reduce the development time and the dependency upon a knowledge engineer. Moreover, with the advent of tools for NLP subtasks, such as named entity recognizers, information extraction systems were decomposed into several components that performed specific information extraction subtasks by employing standard supervised machine learning algorithms.

In adaptive IE, one learns a language model or a set of rules from a collection of annotated documents used to train the learning algorithm and then apply the model or rules to new texts. In this manner, the models or the induced set of rules are usually effective when applied to documents similar to the ones used in the previous training phase.

However, the performance extraction results may be poor when applied to documents with a different genre or style.

Besides that, the price for automation typically comes with the requirement of annotating a large set of documents from a particular domain of interest, in which the desired information has to be annotated by a domain expert. With such annotations, the system can then derive extraction rules on its own. Generally, someone who is familiar with the domain and the task will be sufficient to make these annotations.

The defenders of this approach claims that it is required less effort and expertise to annotate documents than to create rules from scratch.

The following table (Tab. 3.2) summarizes the main benefits and drawbacks of both approaches seen so far. It also shows the classification of these two main approaches according to several aspects: the degree of supervision, the kind of learned rules, the learning paradigm, or the learning strategy, among others.

Next section is devoted to present the state-of-the-art IE systems according to the criteria listed above.

Table 3.2. Comparison of hand-coded rules and adaptive IE. Extracted from [Sanchez, 2007].

Classical IE	Adaptive IE
+ very precise (hand-coded rules)	+ reasonable precision (rule induction)
+ handles domain-independent phenomena (to some extent)	+ higher recall
	+ no need for developing grammars
- need to develop grammars	- provide training data (expensive)
- expensive development & test cycle	- typically "overfitted" to the domain
- develop lexicons, gazetteers, *etc*	- rules can be hard to interpret

3.3. Classification of Approaches to Information Extraction

This section is devoted to present the state-of-art methods and systems implementations related to this thesis.

Fig. 3.7 is an attempt to broadly classify IE systems (NER and RE) found in literature. This classification is based on several published information extraction surveys [Turmo *et al.*, 2006] [Bach and Badaskar, 2007], [Tang *et al.*, 2007], [Melli, 2007], [Sarawagi, 2008], [Jiang, 2012]. This figure shows the classification of approaches to IE into two main lines, namely *Knowledge Engineering* (KE) and *Machine Learning* (ML) approaches, as mentioned earlier.

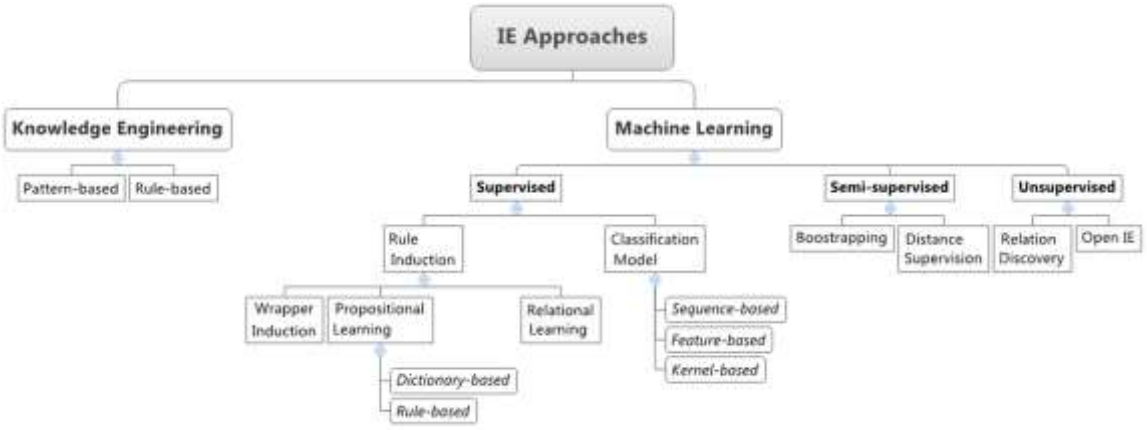


Figure 3.7. Proposed classification of IE approaches.

3.3.1. Knowledge Engineering Approach

The KE main approach is subdivided into two ones: *Pattern-based* and *Rule-based*.

Pattern-based Approach. According to [Sarawagi, 2008] and [Bui, 2012] IE systems based on manually defined patterns require domain expert. This is clearly a time-consuming process.

IE systems based on this approach differ from each other depending on the level of linguistic analysis employed to define patterns. For instance, earlier systems are based on word forms or regular expressions or regex.

In [Zhou *et al*., 2008] patterns are expressed by regular expressions such as Pro_1 * *relation* * Pro_2, where Pro_1 and Pro_2 refer to protein names, whereas relation refers to the verb identifying the relationship between the two proteins. This system aims to extract a limited set of relations given a list of predefined relation words such as "inhibit", "bind", "activate", etc. Tikk *et al*. (2010) evidenced that such kind of systems is too simple to achieve satisfactory results.

Later, other systems attempted to improve the previous work using syntactic analysis of a sentence, such as POS tags, and phrasal structure (e.g. noun phrases, verb phrases, preposition phrases). Again, such surface patterns do not generalize well on complex sentences [Hao *et al*., 2005]. In addition, a closer examination of these results revealed that more patterns were needed to take into account the large amount of grammatical variations in text.

To sum up, manually defined patterns achieve high precision but have relatively poor recall. One can conclude that this approach is not feasible in practical applications due to their limited generalization power. Moreover, they are not well adapted to be applied to a new domain.

Rule-based Approach. Manual rule-based systems rely on a set of rules to extract relations [Jang *et al*., 2006], [Ono *et al*., 2001]. They extend the pattern-based approach seen earlier by adding more constraints to the rules, such as checking negation of relations and determining the direction of relations [Koike *et al*., 2005] [Kim *et al*., 2007]. Thus, instead of using regular expressions to represent constraints, rule-based systems rely on a set of more flexible rules based on rather abstract levels of syntactic structures, grammatical relations, and semantic relations.

Rule-based approaches may generalize well when applied to new domains. Additionally, the number of rules is relatively smaller than those of pattern-based systems. On the other hand, these approaches also suffer from low recall since the defined rules can only cover obvious cases.

Aiming at improving the recall of these systems, the trade-off between precision and recall has to be taken into account once that by relaxing the constraints or by learning rules automatically from training data, the system might improve recall.

3.3.2. Machine Learning Approaches to IE

The second main approach (ML) is traditionally categorized into 3 main classes, according to the annotation effort made, by humans, to annotate the training input texts [Grishman and Yangarber, 2000]:

Unsupervised Methods. There has been recently an increasing interest in unsupervised IE from large corpora, because they do not require any annotated data during learning. In this approach, the goal is to discover salient relations of a given domain.

The two representative approaches in unsupervised IE are *Relation Discovery*, and *Open Information Extraction*.

– *Relation Discovery*. In RE, the types of relations to be extracted are usually known in advance. However, there are also cases where the system does not have any specific relation types to be extracted. In this case, it is imperative to discover salient relations types from a given corpus. This approach is based on clustering techniques. Its key idea consists of clustering entities or entity pairs based on their lexico-syntactic contextual features.

  Shinyama and Sekine were the first researchers to study this problem, which they called Unrestricted Relation Discovery [Shinyama and Sekine, 2006]. In their work, the IE system first collects a large number of news articles from different news sources on the Web. Then, it uses clustering techniques based on lexical similarity in order to find articles about the same event. This preliminary task aims at enriching the feature representation of an entity using its multiple occurrences in different articles. Next, the system performs syntactic parsing and extract named entities from these articles. Each named entity could then be represented by a set of syntactic patterns as its features, e.g., such a pattern may denote that the entity is the subject of a given verb. Finally, another iteration using a clustering algorithm is performed to cluster pairs of entities co-occurring in the same article.

– *Open IE*. Open Information Extraction aims at extracting any type of relation from a large corpus. Typically, the Web is used because of this huge diversity of subjects. In Relation Discovery, the extraction task is based on a single or multiple domain corpus because the goal is to discover the most salient relations from such a domain specific-corpus. However, in some cases, the goal is to find all potentially useful facts from a large and diverse corpus, like the Web. This corresponds to the focus of a new subfield in IE called Open Information Extraction, first introduced by [Banko *et al*., 2006].

  Open IE does not assume any specific target relation type in advance. The main idea here is to make a single pass over the corpus and try to extract as many relations as possible. Since no relation type is assumed in advance, the extraction results are usually phrases describing the extracted relation, i.e., usually such phrases are expressed in terms of *(arg1, relation, arg2)* tuples.

  Banko and Etzioni introduced an unlexicalized CRF-based method for open IE [Banko and Etzioni, 2008] based on the observation that, even though different relation types have very different semantic meanings, there exists a small set of syntactic patterns that cover the majority of the semantic relation mentions. The key idea here is to obtain more generalized patterns covering many cases as possible.

  Later work on open IE introduced more heuristics to improve the quality of the extracted relations [Fader *et al.*, 2011].

  More recently, the ExtrHech open IE system [Zhilla and Gelbukh, 2014] relies on lexical-syntactic patterns handcrafted from POS tagged texts. Using a similar method, [Xavier *et al.*, 2013] propose a system that performs unsupervised extraction of triples by applying a few lexical-syntactic patterns to POS-tagged texts. In order to validate their strategy, the authors developed a prototype and compared its performance with two other open IE systems. The proposed approach achieved promising results, overcoming those from the state-of-the-art systems.

Semi-Supervised Methods. This approach advocates that learning can be carried out from only a small set of annotated data. Then the IE system has to learn patterns on its own. It is also called *weakly supervised* approach. This approach can be divided into *boostraping* and *distant supervision*.

- *Boostrapping*. The idea underlying the bootstrapping approach for relation extraction is to start with a set of seed entity pairs that are related to a given target relation.

The most representative RE system based on this approach is the *Snowball* system [Agichtein and Gravano, 2000], which improved over the DIPRE [Brin, 1998]. Snowball starts with a set of seed entity pairs appearing in target relation. For instance, if the target relation is *HeadquarteredIn*, then the following seed pairs <*Microsoft, Redmond*>, <*Google, Mountain View*>, and <*Facebook, Palo Alto*> can be chosen as seed pair examples. Then, a large corpus is utilized to find co-occurrence of these entity pairs within close proximity. The underlying assumption is that if two entities related to the target relation co-occur closely, the context in which they co-occur is likely to be a pattern for the target relation [Agichtein and Gravano, 2000]. For example, the selected sentence fragment such as "Google's headquarters in Mountain View" and "Redmond-based Microsoft" can derive an extracted pattern like "ORG's headquarters in LOC" and "LOC-based ORG". With these patterns, the Snowball system searches the corpus and tries to find more <ORG, LOC> entity pairs that have the *HeadquarteredIn* relation.

The system adds these entity pairs to the set of seed relation instances and repeats the process. More patterns and entity pairs are added to the results until a certain condition is satisfied.

An important step in bootstrapping methods is how to evaluate the quality of extraction patterns in such a way to minimize many noisy patterns during the extraction process.

Usually two factors are considered, *coverage* and *precision*. Coverage is related to the percentage of true relation instances that can be discovered by the pattern. Precision is related to the percentage of correct relation instances among all the relation instances discovered by the pattern.

The major drawback with this approach is that it is prone to drift, i.e., a large number of noisy patterns are generated. As a result, the overall precision of the IE systems relying on this approach is reduced due to noisy patterns [Schuhma *et al*., 2010].

- *Distant Supervision*. In the bootstrapping approach, only a small set of seed entity pairs or examples is used. With the growth of the social Web, several knowledge bases, with a vast human knowledge contributed by a large crowd of users, are freely available.

Two well-known examples of such knowledge bases are Wikipedia[3] and Freebase[4] that stores structured human knowledge under the form of relations holding two entities [Bollacker *et al*., 2008].

With such available knowledge bases, researchers have been studying a way to take profit from by generating training data a large set of entity pairs known to have a target relation.

Mintz *et al*. (2009) proposed a distant supervision method for relation extraction based on this idea. The authors assume that if there are two entities participating in a given relationship, then any sentence that contains these two entities is likely to express that relation. It is evident that this assumption does not hold in many cases. Accordingly, Mintz *et al*. employed features extracted from different sentences containing the same entity pair aiming at creating a richer feature vector that

[3] http://www.wikipedia.org
[4] https://www.freebase.com

encompasses lexical, syntactic, and named entity tag features that is supposed to be more reliable.

Supervised Methods. In this approach, the learning process relies on fully annotated data: a human annotator has to mark all of the relevant information in a corpus used for training the machine learning algorithm.

Ought to the fact that the IE method proposed in this thesis is based on the supervised approach, Section 4.4 is devoted to present it in more detail.

In what follows, a comparative discussion between KE and ML-based approaches is provided.

3.3.3. Knowledge Engineering vs. ML-based approaches

In several works on IE, the comparison between KE-based approaches, hereafter *symbolic* approaches, and machine learning-based ones (statistical models) is commonly made.

As already mentioned, early work on IE is has exclusively focused on symbolic methods. Then, the fundamental machine methods appeared in the field and they are continuing evolving. Nevertheless, there is no consensus in the IE community which one is better than the other. Thus, each approach is equally suitable to be adopted, depending upon the circumstances.

However, some important aspects have to be considered before opting for one of the two approaches.

Starting the discussion with the symbolic approach, several authors [Turmo *et al.*, 2006], [Sarawagi, 2008] [Yildiz, 2007] have claimed that the manually created set of rules have the following advantages:

- The rule set is likely to cover the domain very well. Actually, this constitutes the main advantage of this approach that makes it the most important choice in applications where the highest precision is crucial.
- Symbolic rules present an explicit nature that allows human interpretation and verification. Further, explicit patterns can be mined and matched with methods that use the explicit nature for optimization [Blohm, 2010].
- Using extraction rules, for example, the distinguished tokens as argument slots in a given extraction template are straightforward identifiable, while statistical-based methods, in most cases, require a previous identification of the arguments by means of named-entity taggers or other markup or lookup operations [Bunescu and Mooney, 2006].

On the other hand, the most cited disadvantages of the symbolic approach are:

- A knowledge engineer might not always be at hand.
- It is a time-consuming task when knowledge engineer has to build a whole new set of rules in the case of a new domain or task has to be supported. Thus, this method does not scale well.

Turning the attention to ML approaches, the main advantages they present are:

- Plenty of time can be saved because the rule generation process can be automatized. Therefore, there is no dependency to a knowledge engineer. However, the system will require enough training data and often a domain expert to make the annotations.

- As statistical models numerically compare a large number of descriptive features of candidate examples to a learned model. This implies that they are more robust to noisy or variability in the data due to the larger set of features they use compared to the symbolic approaches. In this case, a deviation in one feature can be compensated by others. Thereby, each feature contributes to this score to a certain extent.

Finally, the main drawbacks from the statistical ML approaches for IE are:

- Typically in ML in general, the generated statistical models are tailored exactly for the given annotated corpus used as input in training. Consequently, the model will work fine for the documents it was trained on, but will have less overall accuracy when processing new documents in its model application phase. This is a typical example of the overfitting problem in ML.
- Still considering the input annotated corpus, it quite possible that the annotated corpus does not contain all the relevant information of the domain of interest. Therefore, when adopting this approach, the user has to select very carefully the training corpus in order to increase the likelihood that the induced model covers a large portion of the domain. Again, in that case, either human intervention is necessary or a good document retrieval system.

The bottom line is that both approaches have their strength and their weaknesses. Thus, one has to mainly analyse the setting in which the IE system is going to be used. Appelt and Israel (1999) suggest to prefer the automatic training approach when resources and rule writes are not available, training data is cheap and plentiful, extraction specification are stable, and the highest possible precision is critical.

3.4. Supervised Machine Learning Approaches to IE

This section focuses on the supervised machine learning approaches to the two most fundamental tasks in IE, namely NER and RE. According to Fig. 3.7, this approach is divided into two major methods: (i) *classification models*, and (ii) *rule induction*.

3.4.1. Classification Models

Statistical Classification Models consist of the most widely used ML methods for IE. This approach is also called *supervised machine learning*. The central idea in applying supervised ML to IE is to cast the information extraction problem as that of classification.

Consider a two class classification problem, or *binary classification* problem. Let $\{ (x_1, y_1), \ldots , (x_n, y_n) \}$ be a training data set, in which x_i denotes an instance (a *feature vector*) and $y_i \in \{ -1, +1\}$ denotes a classification label. Then, a classification model consists of two distinct stages: *learning* and *prediction*. During learning, the algorithm attempts to find a model from the labelled data that can separate the training data into two classes, say *A* and *B*. During prediction, the learned model identifies whether an unlabeled instance should be classified as *A* and *B*. In some cases, the prediction results may be numeric values, e.g. ranging from 0 to 1. In others, the output can be a set of class labels, e.g., { *A, B, C,* ...}.

Usually, the criteria for predicting the two classes are based on a given numeric value, the prediction value that has to be larger than a given threshold.

The ML approach based on the generation of statistical classification models is the most important and widely used by the IE community.

In the following, the focus is on the statistical model based on supervised classification employed in NER and RE. In this approach, one can distinguish three major methods: (i*) sequence-based, (ii) feature-based*, and *(iii) kernel-based.*

Sequence-based

Particularly for NER, the problem of identifying named-entities can be cast as a task of sequential labelling, in which complete sequences of words or tokens are considered instead of only individual words as it is done in classification-based approaches. In this supervised method, a document is viewed as a sequence of tokens. In addition, a sequence of labels is assigned to each token in order to indicate the property of the token [Tang *et al*., 2007].

More formally, given an observation sequence $\boldsymbol{x} = (x_1, x_2, ..., x_n)$, the sequence labelling IE task consists in finding a label sequence y that maximizes the conditional probability $p(y/x)$, i.e., $\boldsymbol{y}^* = argmax_y \ p(y/x)$.

One of the most used sequential labeling algorithms is the Hidden Markov Model (HMM) [Rabiner, 1989]. A HMM consists of a finite state automaton with stochastic state transitions and symbol emissions.

In [Seymore *et al*., 1999], an IE system based on HMM for extracting relevant fields from headers of computer science research papers is described. Peng (2004) used the sequence-based models for extracting metadata in research papers, such as *title*, a*uthor*, *email*, and *abstract*,

HMM has been successfully used as the main learning component in statistical frameworks for biomedical NER [Collier *et al.*, 2000] [Kinoshita and Cohen, 2005] [Shen *et al*., 2003]. However, Conditional Random Field (CRF) are often demonstrated to be a superior statistical sequence-based method for biomedical NER [Okanohara *et al*., 2006] [Settes, 2004].

Sequence labelling methods present two main disadvantages: the first one is related to the need for a large amount of training data, as the more training data, the better are the results. The second concerns the underlying model in such methods that is a flat mode, i.e., no structural information is taken into account. As a result, these methods are best applicable for the tasks in which the tagged sequences are not nested and when there is no explicit relation between the sequences. NER, chunking, and POS tagging belong to this category of task.

Feature-based

Feature-based approaches for RE rely on classification models constructed by first transforming relation examples into the corresponding numerical vectors that represent several types of features and then applying a machine learning technique, such as SVM [Joachims, 1999], to detect and classify the relations examples into a predefined types of relationships. They have achieved state-of-the-art performance results by employing a large number of diverse linguistic features derived from lexical knowledge, entity-related information, syntactic and dependency parsing trees, and semantic information [Kambhatla, 2004] [Zhou *et al*., 2005]. The utilization of hundreds or thousands of features is computationally burdensome and does not scale well with increasing amount of data. Furthermore, it is difficult for them to effectively capture structured parse tree information [Zhou *et al*., 2005], which is critical for further performance improvement in RE.

Roth and Yih (2007) proposed an entity and relation extraction system based on global inference. In their approach, predictors that identify entities and relations among them are first learned from local information in the sentence. The constraints induced from the

dependencies among entity types and relations constitute a relational structure over the outcomes of the predictors and are used to make global inference.

Giuliano *et al*. (2007) describe a system based on shallow linguistic processing (such as tokenization, POS tagging and lemmatization) that utilizes kernel methods to perform NER and RE tasks. This system also uses a combination of kernel functions that models two distinct information sources: (i) the global context where entities appear and (ii) the local contexts around the interacting entities. The whole sentence containing the entities (i.e., global context) discovers the presence of a relation between two entities. Text windows of limited size centered on the entities (i.e., local contexts) provide clues to identify the roles played by the entities within a relation.

Kernel-based.

Kernel-based approaches for RE are based on *kernel functions*, or simple *kernels*, that define the inner product of two observed instances represented in some underlying vector space. Kernel functions are often regarded as a measure of similarity between two input vectors that represent examples in a transformed space using the original attribute set. The major advantage of using kernels is that observed instances do not need to be explicitly mapped to the underlying vector space in order to their inner products defined by the kernel to be computed [Jiang, 2012].

Two types of kernels were previously explored in RE [Jiang, 2012]:

- Tree-based kernels rely on common substructures containing two entities in order to implicitly explore structured features by directly computing the similarity between two trees [Zelenko *et al*., 2003] [Culotta and Sorensen, 2004]. Another state-of-the-art tree-based kernel system is the one proposed by Zhang *et al*. (2006). The authors explored various structured feature spaces and the tree kernel over parse trees to model the syntactical structured information for RE. Tree kernels can achieve comparable or even better performance than feature-based ones, largely due to their distinctive merit in effectively capturing the structural information of relation instances. However, there exist two main problems in applying tree kernels in RE. The first one is that the subtrees enumerated in a tree kernel computation are context-free; therefore they do not consider the context information outside the target subtree containing two argument entities [Zhou *et al*., 2007]. The second problem concerns the choice of a proper tree span in RE, i.e., the tree span relating the subtree enclosed by the shortest path linking two involved entities in a parse tree [Zhang *et al*., 2006].
- Composite kernels result from the combination of different kernels. Such kernels are mainly used when it is difficult to include all kinds of features into a single kernel, i.e., they can integrate the advantages of feature-based and tree kernel-based methods. Zhao and Grishman (2005) defined several feature-based composite kernels in order to integrate diverse features. In [Zhang *et al*. 2006], the authors proposed a composite kernel that combine the convolution parse tree kernel with the entity feature kernel. More recently, Choi *et al*. (2009) introduced a composite kernel in which several lexical and contextual features were integrated by expanding an existing composite kernel. In their work, they extended the syntactic features with a range of lexical features for achieving a higher performance. Previous investigation revealed that composite kernels achieve better performance than a single syntactical tree kernel. This means that entity type information, i.e., flat information can be combined with structural (syntactic) features into a one single kernel function. The disadvantage of composite kernels methods is that the comparison is performed only on the basis of sentence component information of each node.

3.4.2. Rule Induction

Rule-based systems rely on a set of rules automatically generated from training data. They are traditionally categorized into *wrapper induction, propositional learning, and relational learning*. In what follows, these kinds of rule-based system are described in details.

Wrapper Induction. This method primarily aims at structured and semi-structured documents such as web pages, also called *Web Information Extraction*.

According to Jiang (2012), there is a major difference between Web IE and IE studied in natural language processing in the sense that Web pages often contain structured or semi-structured text such as table and lists, whose extraction relies more on HTML tags than linguistic features. Such Web information extraction systems are also called wrappers. Wrapper induction is the technique for automatically learning wrappers from training data.

In the following, some of the most influential works on wrapper induction are described

WIEN

WIEN is the first wrapper induction system and it was proposed by Kushmerick (1997). In short, multi-slot itemized page fragments are well covered by a set of individual wrappers (4 tabular and 2 nested) in WIEN.

Fig. 3.8 shows an example a wrapper generated by WIEN, which extracts "Country" and "Area Code" from two HTML pages, D1 and D2.

D1: <B>Congo</B> <I>242</I>

D2: <B>Egypt</B> <I>20</I>

Rule: *'<B>'(*)'</B>'*'<I>'(*)'</I>'

Output: Country_Code {Country@1}{AreaCode@2}

Figure 3.8. Example of a rule in WIEN [Kushmerick, 1997]

The rule in Figure 3.8 has the following meaning: ignore all characters until finding the first occurrence of '<B>' and extract the country name as the string that ends at the first '</B>'. Then ignore all characters until '<I>' is found and extract the string that ends at '</I>'. In this example, one can see that this WIEN rule can successfully extracted the desired information from both documents. This same rule is repeatedly applied to other documents.

SRV

SRV [Freitag, 1998] is based on FOIL [Quinlan and Cameron-Jones 1993] and transforms the problem of learning extraction rules into a classification problem. The input to SRV consists of a training set of documents and a set of attributive features and relational features related to tokens that controls the generation process.

SRV uses a top-down covering algorithm to learn IE rules from positive and negative examples. The desired text fragments to be extracted within training documents are manually annotated as positive examples. Negative examples are automatically generated as the complement of the positive ones.

Fig. 3.9 shows a rule in SRV from semi structured documents related to seminar announcements. This rule extracts exact values for the entity speaker.

```
speaker:-                                          // Fragment F is a speaker if
  some(?A, [], word, *unknown*)                    // F contains a token (A), and
  every(capitalizedp, true)                        // every token in F is capitalized, and
  length(=, 2))                                    // F contains exactly 2 tokens, and
  some(?B, [], word, *unknown*)                    // F contains another token (B), and
  some(?B, [prev_token], word, ":")                // B is preceded by a colon, and
  some(?A, [next_token], doubletonp, false)        // A is not followed by a 2-char token, and
  every(quadruple_char_p,false)                    // every token in F does not consists of
                                                   // exactly 4 alpha. characters, and
  some(?B, [prev_token prev_token], word, "who")   // two tokens before B is the word "who"
```

Figure 3.9. Example of a rule for the speaker entity in SRV. Extracted from [Turmo, 2006]

Boosted Wrapper Induction

Another successful implementation of the wrapper induction approach is the BWI system [Freitag, 2000]. This wrapper generation system aims at making wrapping induction techniques suitable for free texts as well.

BWI represents a document as a sequence of tokens, and the IE task consists in identifying the boundaries of the piece of information to be extracted. Let indices *i* and *j* denote the boundaries, then the pair <*i*, *j*> can be used to represent a given instance in a document.

In BWI, a learned wrapper *W* = <*F, A, H(k)*> consists of two sets of patterns that are used to detect the start and the end boundaries of an instance, respectively. The set *F* = *{F1, F2, ..., FN}* identifies the start boundaries, and the set *A = {A1, A2, ..., AN}* identifies the end boundaries. *H* is a length function that estimates the maximum-likelihood probability that the field has length *k*.

BWI estimates these probabilities by constructing a frequency histogram recording the number of fields of length *k* occurring in the training set.

Moreover, it uses the AdaBoost algorithm to generate and combine the predictions from many extraction patterns. In BWI, AdaBoost algorithm runs in iterations. In each iteration, it outputs a weak learner from the training data and a weight for the learner representing the percentage of the correctly classified instances by applying the weak learner to the training data.

Finally, to perform extraction using a wrapper W, every boundary i in a document is first given a "start" score and an "end" score. Then the wrapper *W* classifies a text fragment <*i*, *j*> as follows:

$$W(i, j) = \begin{cases} 1 & \text{if } F(i)A(j)H(\text{j}-\text{i}) > \theta \\ 0, & \text{otherwise} \end{cases}$$

where θ is a numeric threshold.

Propositional Learning. The methods belonging to this rule induction category can be grouped into two types:

Dictionary-based

IE systems under this category first construct a pattern (template) dictionary from training examples, and then use it to extract information from unseen texts. Examples of such systems includes AutoSlog [Rillof, 1993], and CRYSTAL [Soderland *et al.*, 1995].

AutoSlog [Rillof, 1993] was the first system to construct extraction dictionaries from training examples. AutoSlog builds a dictionary of extraction patterns called *concept nodes*. Each concept node has a conceptual trigger that activates it, a linguistic pattern, and enabling conditions. Then, when the conceptual trigger word is found in a text, the enabling conditions are checked on the text. If all conditions are met, then the text fragment defined by the concept node is extrated. Usually, the trigger word is a verb, but it can be a noun too. AutoSlog uses a predefined set of 13 linguistic patterns, where the information to be extracted can be one of the following syntactic categories: subject, direct object, or noun phrase.

The CRYSTAL system improves AutoSlog by enriching the features used to describe a concept node as done in AutoSlog. Rules in CRYSTAL consist of a set of features related to the slot-fillers and the trigger. Such features can be terms, heads, semantic classes, syntactic relations, and verbal modes, etc. CRYSTAL uses a bottom-up covering algorithm in order to relax such features in the initial specific rules. This relaxation is carried out by both dropping out irrelevant features and generalizing semantic constraints. The rules learned by CRYSTAL are more expressive than those learned by AutoSlog.

Rule-based.

$(LP)^2$ [Ciravegna, 2001] is an rule-based IE system which uses the bottom-up induction as its learning strategy. Its learning phase is performed from examples in a user-defined corpus.

Training in $(LP)^2$ is performed in two steps: initially a set of tagging rules is learned; then additional rules are induced to correct mistakes (correction rules) and imprecision in extraction. In the first step above, the tagging rules identify the start and end boundaries of the text fragment to be extracted.

These rules are composed of conditional-action rules, where the condition part consists of constraints on a window of *k* tokens (before and after) the current token; the action part inserts a single tag indicating the beginning or ending of a string to be extracted.

The correction rules aim at reducing the imprecision of the tagging rules. They basically shift misplaced tags rather than adding new tags.

The bottom-up learning process adopted by $(LP)^2$ is based on a sequential covering algorithm and a beam-search for selecting the best generalizations that can be applied at each step. This learning algorithm generalizes the constraints on feature such as POS tagging, shallow NLP, and user defined-classes

The first column of Tab. 3.3 represents a sequence of words. The second to the fifth columns denotes Part-Of-Speech, Word type, Lookup in a dictionary, and Name Entity Recognition results of the word sequence, respectively. All previous columns correspond to the conditional part of the rules. Finally, the action part of the rule is represented by the last column.

Therefore, from the example illustrated in Tab. 3.3, one can note that the action "<Speaker>" indicates that if the text match the pattern, the word “Patrick” will be identified as the start boundary of a speaker.

Rapier

Rapier [Califf, 1998; Califf, 2003] is another IE system that adopts the bottom-up learning strategy. In this systems, the rules are iteratively merged, instead of generalized from training examples. In addition, Rapier takes into account the linear sequence of tokens to perform generalizations on tokens, POS tags, and the senses derived from WordNet hierarchy.

Table 3.3. Example of a tagging rule in $(LP)^2$. Extracted from Tang *et al*. (2007).

Pattern					Action
Word	**POS**	**Kind**	**Lookup**	**Name Entity**	
;	:	Punctuation			
		Word	Person's first name	Person	<Speaker>
		Word			
		Punctuation			
assistant	NN	Word	Jobtitle		
professor	NN	Word			

Relational Learning. NLP has been exploring successful implementations of inductive logic formalisms in its tasks. One of such formalism is *Relational Learning*. In Relational Learning, concept examples are represented using first-order logic [Dzeroski, 2001].

In this supervised learning paradigm, one of the most widely-used learning techniques is *Inductive Logic Programming* (see Section 2.3.).

As a matter of fact, ILP has been used in several natural language subtasks, including: POS tagging [Cussens and Pulman, 2000], chunking [Stasinos, 2003], word segmentation (Kazarov and Manandhar, 2001), semantic and logic representation of sentences [Zelle, 1995] [Mooney, 1999], verb *qualia* [Bouillon, 2002], verb subcategorization frames [Faure and Nedellec, 1999], text classification [Junker *et al*., 2000], and extraction of entity attributes [Aitken, 2002].

More recently, ILP was used as a learning component for word sense disambiguation [Specia *et al*., 2009], wrapper induction of entity extraction rules [Badica *et al*., 2005], and subgroup discovery [Vavpetic and Lavrac, 2013].

ILP has been quite useful in all the aforementioned problems because:

- the induced rules are expressed in a symbolic formalism that are interpretable by a linguistic expert;
- the higher expressivity (in first-order logic) of the final induced rules;
- the integration of external prior knowledge is naturally achieved.

Due to the importance of inductive learning approaches to NLP, it was proposed the research field at the intersection of logic, NLP and machine learning called *Learning Language in Logic* (LLL) [Dzeroski, 2001].

The first LLL workshop [Cussens and Dzeroski, 2000] took place in 1999, in Bled, Slovenia. The 4th edition of the LLL workshop (LLL 05 challenge[5]) was organized by Claire Nédéllec. This challenge proposed several datasets and a dictionary for the *Genic Interaction Extraction Challenge* task[6]. The goal of this task is to learn rules to extract protein/gene interactions from biology abstract from the Medline database. This dataset is still used nowadays for evaluating protein-protein interactions by relation extraction systems.

[5] http://www.cs.york.ac.uk/aig/lll

[6] http://genome.jouy.inra.fr/texte/LLLchallenge

Given this historical review of the inductive logic-based approaches to NLP applications, the attention is on the more recent ILP-based systems for entity or relation extraction from text.

In what follows, the most representative work more closely related to the one proposed in this thesis is surveyed, presenting a discussion of its advantages and drawbacks. Except where explicitly cited, all systems surveyed in this section deal with natural language texts in English. These systems were divided into two categories, according to the IE task they perform: NER or RE.

ILP-based IE Systems applied to Named Entity Recognition

Berardi *et al*. (2006) relies on ILP in an approach to induce recursive theories able to extract biomedical entities in sentences containing both explicit and implicit relationships among them. In their approach, recursive rules are discovered by the induction of mutually dependent predicate definitions using the ATRE [Malerba, 2003] ILP system. Their built IE models consist of classification rules, including mutually recursive definitions of the classes, in which each class plays the role of a template extraction. Thus, the IE models or classification rules encode the conditions to fill a pre-defined template extraction. In their evaluation system, article abstracts on mitochondrial mutations are preprocessed and annotated by GATE[7] that provides tokenization, POS tagging, NER and gazetteers. Actually, two dictionaries are used: an independent domain dictionary already available in GATE; and a second one, more specific, and domain-dependent that contains a list of biological entities, including diseases, enzymes, and genes.

Their experimental results were obtained by means of k-fold cross validation (k = 6) over a set of 71 biomedical articles. The results show good accuracy (82.7%), against a less attractive recall of around 37% (F1-measure 0.51). An explanation to this low recall score is probably due to the preprocessing of noisy data that makes the learned theories to be very specific, therefore, causing overfitting.

Both works [Ramakrishnan *et al*., 2007] and [Dedek, 2012] used the Aleph ILP system for inducing extraction rules as Horn clauses for entity template extraction. They used the same dataset in their evaluation, the *Corporate Acquisition Events* corpus taken from the Reuters dataset [Lewis, 1992]. They both take profit of dependency parsing in text preprocessing, but using different NLP tool: the MINIPAR [Lin, 1998] and WordNet were the choice in Ramakrishnam and colleagues's work, whereas Dedek *et al*. used the TectoMT, a Czech and English parser [Popel and Zabokrtsky, 2010]. In addition, in Dedek´s work, the extracted information is serialized as a RDF ontology. The averaged performance results reported in [Dedek, 2012] on the Corporate Acquisition Events dataset were quite low (30.5/20.0/23.5) in terms of Precision/Recall/F1-measure, respectively; against the same averaged results (47.6/37.7/41.9) on the same dataset reported in [Ramakrishnam *et al*., 2007].

Patel *et al*. (2010) reported experiments using ILP to generate extraction rules for various named entities in Marathi language. They also used the GATE framework in preprocessing. They opted for ILP because it provides an appropriate mechanism for the incorporation of linguistic knowledge. In a first experiment, Patel and colleagues measured the time spent on the creation of rules for a corpus consisting of 3,884 sentences and 54,340 words, which took the period of 1 month to be manually developed. In the second experiment, extraction rules for the same corpus were induced using two ILP systems, WARMR and TILDE, available as a part of ACE[8] data mining system. For the rule

[7] General Architecture for Text Engineering (GATE). https://gate.ac.uk

[8] Machine Learning Group - ACE Data Mining System. http://dtai.cs.kuleuven.be/ACE.

induction task using 80% of the labeled corpus, TILDE took 1 hour, while WARMR took 140 hours. This means that the use of ILP reduced the rule development time by a factor of 240 for TILDE and 1.7 for WARMR. The authors justified these results claiming that ILP provided not only the ability to incorporate domain knowledge by experts, but also more reliable rules comparable to the rules developed by human experts.

ILP-based IE Systems applied to Relation Extraction

Kim *et al*. (2007) is one of the first relation extraction systems based on the Aleph ILP system. Kim opted for a text preprocessing involving NER, POS, chunking analysis, and grammatical function assignment (subject, object, time, location, etc.) provided by the Memory-based Shallow Parser (MBSP)[9] which has been adapted to the biological domain. In the evaluation of their RE system, Kim and colleagues used a set of sentences taken from the PRINTS database, a protein family database. The evaluation task involved extracting relation between proteins and any other biological entities, provided that they were relevant to three topics: disease, function, and structure. Their reported results were fairly precise, with about 75% in precision, but with a very low recall of less than 30% for two of the three datasets evaluated. In addition, once relations were extracted from the test set, they were manually evaluated by the PRINTS annotators.

Inspired by the work of Cullota and Sorensen (2004), Horvath and colleagues (2009) consider both dependency trees as relational structures composed of a single binary predicate that represent the edges of such trees. In their work, text preprocessing is performed by GATE and the Stanford parser. They also used WordNet as a semantic resource for obtaining hypernym relations. In addition, the authors assume a partial order on the set of unary predicates which are defined by a hierarchy between words, e.g., the unary predicate *Person (X)* is more general than the *Physicist (X)* predicate, the last one derived from the WordNet.

Applying the notion of generalization operator, i.e., Least General Generalization (LGG) of Plotkin [Plotkin, 1970], they generate a set of rules expressed as non-recursive Horn clauses satisfying some criteria of consistency, e.g., all rules must cover a minimum number of positive examples, while considering number of negative examples at the same time. Then, they used these rules to generate a binary vector of attributes for each example, and used the resultant vectors for training a SVM classifier to separate positive examples from the negatives. The performance of their RE method was empirically compared to other methods for RE using a corpus provided by ACE 2003[10], which contains documents related to 519 newspaper articles. The results achieved by the authors showed that their method for RE are comparable to other state-of-the-art RE methods [Cullota and Sorensen, 2004], [Bunescu and Mooney, 2005] reaching a value of 52.2% of F1-measure.

Seneviratne and Ranasinghe (2012) work's is very similar with Hovarth *et al*. (2010) with respect to the text preprocessing tools used, as they both used GATE and the Stanford parser for POS tagging and dependency parsing. The only difference is that the former also exploits gazetteers. Another marked difference between these two works concerns the dataset used in the evaluation: while Horvarth *et al*. evaluated their proposal using a standard dataset containing 5 distinct relations, Seneviratne and colleagues used thirteen web pages for evaluating only one relation (*Located_in)* between a bird and a location. Their final reported results are just the induced extraction rules, that is, no performance score such as precision/recall/F1-measure are provided.

[9] Memory-based Shallow Parser for Python. http://www.clips.ua.ac.be/ctrs/memory-based-shallow-parser-for-python

[10] Linguistic Data Consortium. https://catalog.ldc.upenn.edu

Smole *et al*. (2012) proposed an ILP-based RE systems suitable to learn rules for identifying and extracting information from definitions of geographic entities in texts in Slovene language. This ILP-based RE system constitutes as a component in a spatial data recommendation service. They focus on the extraction of five most frequent relations/properties, namely "isA", "isLocated", "hasPurpose", "isResultOf", and "hasParts", present in the 1308 definitions of spatial entities. The following sentence is an example of a definition found in this corpus. "*Sewerage is a system of ditches and canals for supplying and draining off wat*er". For that, they choose the classical Progol ILP system [Muggleton, 1995] which induces Horn clauses. Their NLP component is based on the Amebis Slovene POS tagger. The authors implemented a tool for chunking detection in Slovene that takes as input the text already tagged by Amebis. As a final step of text preprocessing, they use the annotated text version to manually assign relations/properties to all chunks of the selected 1308 definitions. A major drawback of their method is that the manual development of the chunk rules and manually mapping to semantic classes is time-consuming, burdensome, and not scalable.

Kordjamshidi *et al*. (2012) proposed a spatial RE system for the so called spatial role labeling (SpRL) problem recently introduced [Kordjamshidi et. al, 2011]. The core problem in SpRL is: i) the identification of the words that play a role in describing special concepts; and ii) the classification of the role that these words play in the spatial configuration. They utilize kLog [Frasconi *et al*., 2012], a framework for kernel-based relational learning that uses graph kernels. kLog is not only a relational learning system, i.e., but it is also able to exploit background knowledge in the form of logic programs. The authors rely on the Charniak Parser, for POS tagging and dependency parsing, and LHT tool, an automatic semantic role labeling tool. No further semantic resource is used nor ontologies in text preprocessing. In order to extract spatial relations, the kLog is first used for extracting relational features that kLog utilize to perform automatically a propositionalization step [Kramer *et al*., 2001]. Then, both a SVM and a HMM classifiers are constructed from the propositionalized features.

The Alvis project comprises [Nédéllec *et al*., 2008] a relation extraction system that aims at extracting relations between biological entities. Alvis provides a semantic analysis based on the NLP Ogmios platform [Nazarenko *et al*., 2006] which performs several NLP subtasks, including: NER of biological entities, POS tagging, terminological analysis assisted by terminology dictionaries, syntactic parsing, and semantic mapping to biological domain ontologies.

In Alvis, once the semantic units of text are identified, they are typed with fine-grained concepts that are associated by domain-specific relations from the ontology. The Alvis machine learning component consists of the LP-Propal method based on supervised ILP algorithm Propal [Alphonse and Rouveirol, 2000]. LP-Propal takes as input the corpus after the full processing by the linguistic pipeline and induces extraction rules suitable to tag semantic relations found in the ontology. Alvis heavily relies on terminology dictionary at first place for identifying biomedical instances in text, which is not adequate for very active domains producing documents containing new named entities instances. As a result, such terminology dictionaries are quite hard to keep fully updated.

Summary of the ILP-based IE systems

Tab. 3.4 summarizes the ILP-based IE systems presented earlier according to the following comparison criteria:

- IE task performed, either NER or RE;
- type of the induced models or the final induced set of rules;

- NLP subtasks performed in text preprocessing;
- NLP tool(s) used;
- linguistic or semantic resources used;
- ILP learning component;
- evaluation dataset(s); and use of ontologies.

Table 3.4. Summary of the ILP-based IE Systems

Main Reference	IE task	Learned theory	Preprocessing task(s)	NLP tool used	Semantic resources used	ILP System	Evaluation dataset	Use of ontology in IE
Berardi et al. (2006)	RE	Horn clauses	POS, NER, Gazeteers	GATE	None	ATRE	70 abstracts	No
Ramakrishnan et al. (2007)	Entity Template	Horn clauses[b]	NER, Dep. parsing	MINIPAR	WordNet	Aleph	600 news articles[c]	No
Patel et al. (2010)	NER	Decision List	POS	GATE	None	Warmir, TILDE	25.000 sentences	No
Dedek (2012)	Entity Template	Horn clauses	POS, Dep. parsing	GATE, TectoMT	None	Aleph	600 news articles[c]	No[h]
Kim et al. (2007)	RE	Horn clauses	POS, NER, Chunking, Grammatical functions	MBSP	None	Aleph	Function (1268)[g]	No
Nédéllec et al. (2008)	NER , RE	Horn clauses	POS, NER,Gazeteers, Dep. parsing, Semantic ,apping	Ogmios framework	None	LP-Propal	LLL	Yes
Hovarth et al. (2010)	RE[a]	Horn clauses	POS,Dep. Parsing	GATE and Standford Parser	WordNet	LGG	ACE 2003	No
Seneviratne and Ranasinghe (2011)	RE	Horn clauses	POS, Gazeteers, Dep. parsing	GATE and Standford Parser	None	Newgen	13 Wikipedia pages	No
Smole et al. (2012)	RE	Horn clauses	POS, chunking[d] , Manual SRL	Amebis	None	Progol	1308 definition sentences	No
Kordjamshidi et al. (2012)	RE[e]	Horn clauses	POS, Dep. Parsing, SRL	Charniak Parser and LHT tool	None	kLog in Prolog	1213 sentences[f]	No

[a] Actually it generates binary features as input to a SVM classifier.
[b] Aleph was used for feature construction to a SVM classifier.
[c] Corporate Acquisition Events corpus taken from the Reuters dataset [Lewis, 1992].
[d] Both POS tagger and a chunker for the Slovene language.
[e] Only one spatial relation involving three arguments: *sr(indicator, trajector, landmark).*
[f] These sentences consists of textual description of 613 images in the TC-12 image dataset.
[g] Two other datasets were used: *Disease* with 777 sentences, and *Structure* with 1159 sentences.
[h] At the end of the IE process, the annotated documents are serialized in a RDF ontology.

Discussion

A closer look at the Tab. 3.4 that summarizes all surveyed ILP-based system for IE presented in this section reveals that the most of the systems performs RE. In these systems, it is assumed that NER is already solved, and they take as input a corpus

containing pre-annotated named entities. On the other hand, 4 of the 10 surveyed systems either perform NER or the similar task of Entity Template Extraction (ETE). This second information extraction task is considered easier than the first one because of the nature of the semi-structured documents used in ETE which present some degree of regularity of the extraction slots.

Surprisingly, Alvis is the only system that performs both NER and RE [Nedellec *et al*., 2008]. In fact, the focus of the Alvis system consists in interactions (relations) between two biological entities. However, due to the difficulty in NER in the particular biological domain, they decided to integrate in Alvis several named entity classifiers based on the C4.5 decision tree implementation of the WEKA[11] data mining tool. The extracted instances of biological entities are integrated in a dictionary of terms.

Tab. 3.4 also reveals that all systems induced expressive extraction rules in Horn clauses for both NER and RE tasks, except in [Patel *et al*., 2010], in which the final induced rules constitutes a decision list [Fürnkranz *et al*., 2012].

Half of the systems shown in Tab. 3.4 rely on GATE for text preprocessing. This is a reasonable choice as a NLP preprocessing tool, given that GATE offers a comprehensive set of NLP tools in just one package. MINIPAR and the Charniak parsers were the alternative option for other systems.

The Ogmios framework is another comprehensive NLP platform that offers several NLP tools but it is tailored to the biomedical domain, while GATE is domain-independent. In addition, it is noticeable the trend of the ILP-based IE systems to rely in deeper NLP tasks such as full dependency parsing, and semantic role labeling. Again, this is not surprising because previous work on RE and SRL has proved to be very beneficial to IE [Jiang and Zhai, 2007] [Harabagiu *et al*., 2005].

Concerning external resources used in the IE process, the great majority of the systems make no use of such prior knowledge. Among the surveyed systems, only Hovarth *et al*. (2010) and Ramakrishnan *et al*. (2007) take profit from the WordNet.

For the learning component, the systems do not have any preference for a particular ILP system. However, the Aleph ILP system was the preference for 3 of the 10 surveyed systems.

From the experimental methodological point of view, none of the studies carried out substantial experiments using several corpora either of the same domain, or from different domains. For instance, in Seneviratne *et al*. (2012), only one relation type was evaluated on a set of 13 web pages. The only works that used a standard competition datasets in their experimental evaluations were [Hovarth *et al*., 2010] and [Nédéllec *et al*., 2008].

Thus, one can draw to the conclusion that the use of ILP for IE has not been yet fully exploited by substantial assessment using several datasets. In this sense, Nédéllec *et al*. (2008) was the only system to take profit of ontologies in the IE process. As already seen, this system relies on both a term dictionary and a biological ontology for mapping domain-specific terms to the domain ontology. Another distinguishing feature of this system is that these mapping are used primarily for semantic annotation.

In [Dedek, 2012], the exploitation of ontologies only consists in serializing the input documents as RDF tuples after text preprocessing. Thus, no active use of ontologies is performed, such as exploiting the ontology for guiding the IE process.

3.4.3. Open Problems in Classical Information Extraction

In the last two decades, there were considerable advances in NLP. Information Extraction, regarded as a type of limited natural language understanding that produces a structured

[11] WEKA: Data Mining Software in Java. http://www.cs.waikato.ac.nz/ml/weka

view of facts, has been taking advantages of such advances, allowing much more sophisticated text analysis to be performed. Despite of that Information Extraction is still difficult, also due to the inherent aspects of natural language such as *ambiguity* and *variability*.

Ambiguity arises when a single expression can have multiple interpretations, while variability occurs when a single interpretation of a fact is denoted by multiple expressions. These two aspects are, in one sense, opposites, but they constitute both of equally very complicating factors for information extraction in general [Pyysalo, 2008].

From the structural point of view, language analysis, and consequently IE, struggles with the ambiguity at all levels of language analysis ranging from the meaning of individual words to interpretation of complex syntactic structures.

Language variability is another essential property of natural languages, since authors seek for variability when writing their texts because readers can get bored with repetitive text. As a result, the creative potential of languages considerably complicates the automatic extraction of facts stated in texts. Psysalo (2008) gives a surprising variability example from the biomedical domain where a single relationship between two proteins can be expressed in more than twenty ways taken from just a small sample of sentences in the BioInfer corpus [Pyysalo *et al*., 2007]. As a result, due to the great potential for producing variability in natural language, it is simply not possible to enumerate a complete set of full word sequences that can be express facts. Therefore, such flexibility of human language guarantees that novel variants occurs, which escape from any attempt to be defined by a collection of known forms for them.

Another problem to be addressed is related to the scalability problem, i.e., an IE system should be scalable in the sense that it could be able to process increasing amount of data.

A shortcoming of the overwhelming majority of the IE systems reviewed in this chapter resides in the fact that they are not able to extract *implicit information* from texts, since they take only into account in their analysis one sentence at a time and, thereby, relevant implicit information cannot be extracted.

Portability is another main issue to the current IE systems reviewed. It concerns the capability of how ease an IE system can be applied to a new domain. Thus, the portability to a new domain is still considered hard to be achieved for IE systems belonging to the two approaches seen in Section 3.2. On the one hand, for KE approaches, it is quite difficult to convert a previous set of efficient and reliable extraction rules to a new domain, or even to a new task, while preserving the overall accuracy level of the original rules. On the other hand, for the statistical model-based approaches, an expert has to annotate the new document set in case of the definition of a new relevant information to be extracted. To sum up, this portability problem, also known as adaptability problem is a serious bottleneck for state-of-the-art IE systems [Turmo *et al.*, 2006].

Besides such problems, Cunnighan (2006) and Yildz (2010) point out the following major challenges that classical IE systems should address in the near feature:

- facilitating *maintainability* for reacting on evolving domain knowledge;
- focusing on increasing *portability* to other domains;
- enhancing usability of *extraction templates* by allowing ad-hoc template generation;
- separating *background knowledge* specifications from templates and applying background ontologies for knowledge representation purpose.

In this respect, the emergence of Ontology-based Information Extraction as a subfield of IE seems to be an important alternative to be considered in order to address the aforementioned challenges. The next section is devoted to review the literature about

Ontology-based Information Extraction, and situating the main contributions of the present thesis.

3.5. Ontology-based Information Extraction

Storing information in plain text format does not allow document accessibility, since the semantic aspects of its content are not explicitly expressed. The lack of some kind of structure hinders the exploration and interpretation of semantic information by computational agents [Fensel *et al*., 2002]. According to Feldman and Sanger (2007), the Web is currently at a *syntactic level*, i.e., its contents can be read by machines, just considering keywords and combinations of these, and not on a semantic level, in which computer systems can interpret unambiguously the information available. This characteristic constitutes an important limitation of the current Web. Thus, the task of automatically finding relevant information to specific needs, especially those that require some level of semantic interpretation, becomes costly, and time consuming.

To address those limitations, the Semantic Web [Berners-Lee *et al*., 2001] was proposed as a global initiative. The main goal of the Semantic Web is to make explicit the meaning of data content on the Web. Thus, it is possible that both people and computational agents can process web data, i.e., the data semantic aspects.

One of the fundamental layers in the development of the Semantic Web is composed by ontologies [Koivunen and Miller, 2001], which are responsible for providing the necessary expressiveness to the representation of relevant knowledge about a domain [Freitas, 2003]. Thus, the first step to make the Semantic Web goals achievable is to define the appropriate semantic structures for representing most domains of knowledge, which implies the development of domain or task-specific ontologies. Once the ontology for a specific domain is available, the next step is to semantically annotate related web resources. This task is also known as Semantic Annotation [Petasis *et al*., 2011]. Thus, computers must have access to ontologies that enable knowledge to be represented and shared.

On the other hand, although domain or task-based ontologies are recognized as essential resources for the Semantic Web, the development of such ontologies relies on domain experts or knowledge engineers that typically adopt a manual construction process. Such manual construction process is time-consuming and error-prone [Cimiano, 2006]. An automated or semi-automated mechanism to convert the information contained in existing web pages into ontologies is highly desirable in such a scenario. *Ontology-based Information Extraction* (OBIE) [Wimalasuriya and Dou, 2010], a subfield of Information Extraction, is a promising candidate for such a mechanism. An OBIE system can process unstructured or semi-structured natural language text through a mechanism guided by ontologies to extract certain types of information, and present the output using ontologies.

In this section, the motivation for OBIE and its potential for Ontology Population are presented. Next, state-of-the-art OBIE systems are studied and discussed, highlighting their advantages and drawbacks.

3.5.1. Motivating the Use of Ontologies in IE

As already mentioned, Information Extraction can be defined as the automatic process of identifying and retrieving relevant information from texts in natural language, presenting such information in a structured format.

Many of the tasks performed by a traditional IE system (especially the ones that relate to templates) have a strong dependency on knowledge about the domain of interest, i.e., they are highly coupled with the target domain of interest. This is considered as a

portability issue which constitutes one of the major challenges for the IE field Wimalasuriya and Dou, 2010] [Petasis *et al*., 2011].

Recently, OBIE, as a subfield in IE, tries to alleviate the portability problem through the use of ontologies, by providing means to make the IE system less dependent of the knowledge domain in consideration. In fact, making domain knowledge explicit through an ontology, not only enhances portability, but also provides new opportunities for IE systems, ranging from using ontologies for storing the extracted information by using reasoning for implementing various IE tasks.

Besides the above advantages that it brings to IE, ontologies themselves bring other advantages as well [Freitas, 2003]:

- *Common vocabulary*. They provide a common vocabulary for domain knowledge representation which can be interpreted by different agents, whether humans or computers.
- *Correct interpretation*. They define a formal representation of a domain, thus avoiding ambiguous interpretations.
- *Reuse*. Researchers can reuse other ontologies, or even make adaptations and extensions of existing ones. Since the construction of knowledge bases constitutes a complex task, this aspect of ontologies promotes a significant time reduction in development compared to create ontologies from scratch.

The emergence of OBIE as a subfield in IE has been witnessed by the increasing number of publications. Other key indicator for the trend towards OBIE is the increasing number of conferences tracks and workshops at ML conferences devoted to IE, such as the very recent conference on *Empirical Methods in Natural Language Processing* (EMNLP/2014) and, the first two editions of the Semantic Web and Information Extraction Workshop (SWAIE) in 2012 and 2013. The goal of these workshops is to bring researchers from the fields of Information Extraction and the Semantic Web together to foster inter-domain collaboration.

3.5.2. Potential of OBIE

Several authors [Wu and Weld, 2008] [Cimiano, 2006] agree with the fact that OBIE presents a lot of potential. Besides the automatic processing of the information contained in natural language texts discussed earlier, the potential for fully exploit OBIE is two-fold:

- **Creating semantic contents for the SW.** The Semantic Web aims at providing semantic content to the current World Wide Web, in a way that it can be processed by software agents [Berners-Lee, 2001]. On the other side, it is quite hard to imagine that such content has to be manually annotated, given the prohibitive size of the Web. As a result, a massive metadata generation is required in order to make the Semantic Web come true [Popov *et al*., 2004]. In this context, OBIE has the great potential as a means for automatic generation of semantic contents by converting the information contained in existing web pages into ontologies. This has been pointed out by several works including Wu and Weld (2008), and Cimiano (2004). This process is also known as the *Semantic Annotation* of web pages.

- **Improving overall ontology quality**: Interestingly, OBIE can be used in the assessment of the quality of an ontology [Kietz *et al*., 2000] [Maynard *et al*., 2008]. If one assumes that a given domain ontology can be successfully used by an OBIE system to extract the semantic contents from a set of documents related to that domain, then it can be deduced that the ontology itself is actually a good

representation of that domain. Consequently, the errors in the ontology can be identified by analysing the types of semantic content the OBIE system, which integrates the domain ontology, has failed to extract.

3.5.3. OBIE for Ontology Population

The focus of this section is on the application of OBIE systems for performing Ontology Population. Ontology Population (OP) consists of the process of inserting new instances of classes, properties and/or relations in an existing ontology [Petasis *et al*., 2011]. Therefore, an OP system does not alter the structure of the ontology, i.e., no change in the hierarchy of classes and/or a relationship is carried out. The updating task is restricted to the set of instances of concepts, relationships and properties of an input ontology. Instantiating ontologies with new factual knowledge is a relevant step towards the provision of valuable ontology-based knowledge services. Indeed, it is well known that the manual construction process of ontologies is very time-consuming and error-prone [Cimiano, 2006]. Thus, the automated or semi-automated mechanism that OBIE systems provide to convert the information contained in unstructured or semi-structured sources into ontologies is highly desired.

Indeed, OP systems are closely related to OBIE systems and, as pointed out by Petasis *et al*. (2011), because the latter provides mechanisms to associate pieces of the information with concepts and relationships of an ontology. One can draw to the conclusion that every OBIE system can be considered as an OP system, as it can be extended to assimilate extracted instances into the ontology.

Besides the crucial role for building and maintaining knowledge bases (Maynard *et al*., 2008), OP allows relating the knowledge described in natural language with ontologies, assisting the process of semantic content generation [Wimalasuriya and Dou, 2010]. Moreover, a populated ontology can be used in several applications such as content management, information retrieval, text mining, and automated reasoning, among others.

Cimiano (2006) highlights the following three major tasks in populating ontologies:

- learning instances of concepts in the domain. This task is similar to NER. Here, the formal hierarchical concept definition given by an ontology constitutes the major difference with the flat structure of named entities extraction templates.
- learning instances of formalized relations between two instances of concepts, which are somewhat similar to Relation Extraction.
- annotating entity references with instantiations of concepts and relations in a domain ontology. This task is also called *Semantic Annotation* [Popov *et al*., 2003].

The above tasks are applicable to the intended goals of this thesis, which investigates ontology population techniques in the three scenarios.

Fig. 3.10 illustrates a general ontology population process. This figure shows that, besides a corpus in natural language, an OP system requires a domain ontology that will be populated by an instance extraction engine. Thus, the class instances, i.e., the domain concept realizations, are identified and retrieved from the input corpus usually after the prepocessing step. In this process, the ontology structure, including its class hierarchy and non-taxonomic relations, are not modified. Thus, the set of concepts and relation instances changes. The information extraction engine is responsible for locating instances of concepts and relations in a textual corpus. This component can be implemented by various techniques such as linguistic rules, machine learning (classification, clustering, etc.) or simple regular expressions based on linguistic patterns, to name a few. Finally, the list of extracted classes/relation instances is subsequently used to populate the ontology.

To summarize, OP plays an important role in building and maintaining knowledge bases [Maynard *et al*., 2008]. Furthermore, it allows for relating knowledge described in natural language with ontologies, assisting the annotation process of semantic content [Wimalasuriya and Dou, 2010]. Last but not least, the final populated ontology can be used by various knowledge intensive applications, such as content management, information retrieval, automated reasoning, among others.

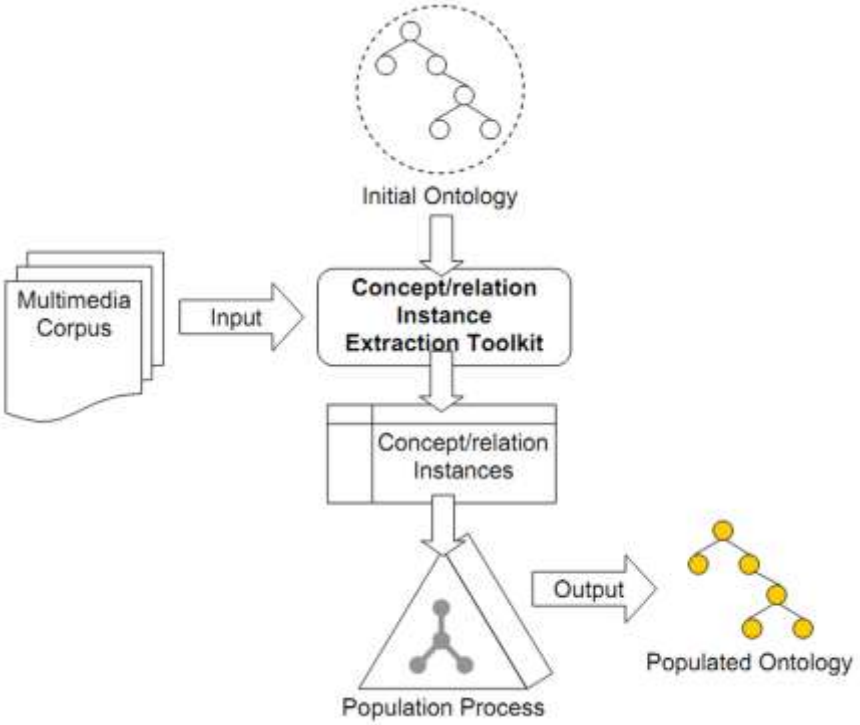


Figure 3.10. The ontology population process. Extracted from [Petasis *et al*., 2011].

3.5.4. Ontology-based Information Extraction Systems

Ontology-based Information Extraction (OBIE) consists of a subfield of IE in which ontologies are used to guide the information extraction process and the output is usually used to enrich an ontology [Petasis *et al*., 2011].

Wimalasuriya and Dou (2010) define an OBIE System as "a system that processes unstructured or semi-structured natural language text through a mechanism guided by ontologies to extract certain types of information and presents the output using ontologies".

Ontologies contain concepts arranged in class/sub-class hierarchies (e.g. a Country is a type of Geographical Location), relations among concepts (e.g., a Country has a President), or properties. OBIE normally takes place by specifying an ontology for the domain targeted by an IE system and using an IE technique to discover individuals for classes and instances for properties.

OBIE is considered a multidisciplinary research once it involves concepts from Semantic Web, Information Extraction, Natural Language Processing, and Machine Learning.

OBIE can equally be exploited for *Ontology Learning* [Cimiano, 2006] [Maedche *et al*., 2002], in which an OBIE system is used for changing the ontology TBox schema, i.e., adding new classes and properties.

General Architecture of OBIE Systems

Fig. 3.11 represents the union of different architectural components found in current OBIE systems, according to [Wimalasuriya and Dou, 2010]. However, there are many OBIE systems that do not comprise all of the components depicted in Fig. 3.11. For instance, the *Ontology Generator* component is missing in many OBIE systems.

Compared to the general architecture of classical IE systems, the new introduced components in the OBIE architecture correspond essentially to the ontology and the Ontology Generator. In some OBIE systems, a *semantic lexicon* is also employed.

The OBIE process starts with the textual input that first goes though a text preprocessing component that converts the text to a format suitable to be the *Information Extraction Module* (information extractor). It is in the IE module that the actual information extraction is performed. So far, this process is the same of what occurs with a traditional IE system. As already seen, this component can be implemented using several techniques.

What really distinguishes an OBIE system from an IE one, is that the former will be guided by an ontology and, in some cases, by a semantic lexicon, like WordNet that can also support the IE component. Additionally, the creation of both the ontological domain knowledge and the semantic lexicon may be supported by external sources.

The separation between an ontology and a semantic lexicon splits the knowledge representing a domain of concern from the more general linguistic knowledge about a language in lexicons.

Another interesting aspect is that the ontology in this architecture may be generated internally by an Ontology Generator component that may also take profit of the semantic lexicon.

As expected, the output of the OBIE system consists of extracted information that can be represented using an ontology definition language such as OWL, or stored in a knowledge base.

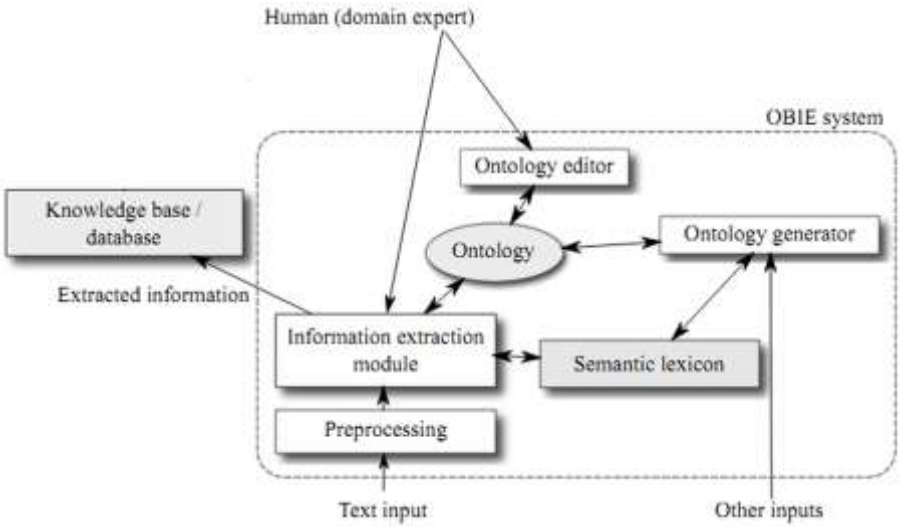


Figure 3.11. General architecture of an OBIE system. Adapted from [Wimalasuriya and Dou, 2010].

3.5.5. Classification of State-of-the-art OBIE Systems

As pointed out by Wimalasuriya and Dou (2010), OBIE systems can be employed in two main tasks: *Ontology Learning* and *Ontology Population*. In the first case, as already seen, the OBIE system aims at extending an ontology through the addition of new concepts,

relations, and rules. In the second, a given domain ontology is already present before the IE takes place, and the goal of the OBIE system is to populate ontological elements (classes and properties) with instances found in the input set of documents. There are also OBIE systems that perform both tasks at the same time, but they are not the rule. Examples of such systems include [Maedche *et al*., 2002] [Castano *et al*., 2008].

According to the thesis purposes, the classification presented here is focused is on the most representative and influential OBIE systems that only perform OP. The dimensions along with the OBIE systems are classified are introduced. This classification takes into account several surveys [Karkaletis *et al*., 2011] [Petasis *et al*., 2011] [Wimalasuriya and Dou, 2010] and other papers on OBIE. Finally, this section discusses the surveyed OBIE systems, finishing with the description of some of the contributions that the present thesis brings to the OBIE community.

For the sake of simplicity, instead of presenting the surveyed OBIE systems in typical chronological order, it was preferred to group them according to the dimensions of the main IE method employed by the OBIE systems.

Extraction Ontologies. The first category of OBIE systems summarized in Tab. 3.5 comprises the systems with the distinguishing feature of using the domain as a repository for pattern matching expressions, usually in the form of regular expressions (regex) in its IE extraction process.

Embley's OBIE system [Embley, 2004] was the first to use this technique. This system combines linguistic rules that based on regular expressions with the elements of the domain ontology, including classes and properties. This modified version of the ontology is also called *extraction ontology*.

An extraction ontology consists in an augmented conceptual-model instance that serves as a wrapper for a narrow domain of interest. Usually, the conceptual-model instance includes objects, relationships, constraints, and data-frame descriptions of strings for lexical objects [Embley, 2004].

In the same track, Yildiz and Miksch (2004) proposed the ontoX system with the main difference that in ontoX regex involving some elements of the domain ontology are induced. Actually, this pattern expressions induction process is performed by two main modules in ontoX.

First, the Ontology Management Module (OMM) takes the input ontology and tries to exploit the knowledge in it to determine what exactly has to be extracted from the input data. In ontoX implementation, this module supports ontologies formulated in OWL. Second, the Rule Generation Module uses the output of the OMM and performs several steps to formulate rules, i.e., regular expressions, able to locate candidate values that are relevant according to the input ontology. The Extraction Module takes these rules and determines candidate values in the input texts, applying several heuristics to choose the most accurate values among them. This module finally returns the extracted values and suggestions to the user regarding possible changes in the ontology.

The next two systems of this category, *iDocument* [Adrian *et al*., 2009] and FLOPPIES [Nederstigt *et al*., 2014], besides the common use of pattern matching expressions in regex, attempt to improve the matching process between terms in the input set of documents with classes and relations in the domain ontology by means of tailored similarity functions. The key idea in these two systems is to avoid the problem of having and exact matching if the system only uses regex.

Concerning pre-processing tasks, only Adrian's iDocument takes profit of gazetteers, POS tagging, and chunking parsing. The other systems based on extraction ontologies use no preprocessing at all (Embley’s and FLOPPIES), or remove stop words and recognizes named entities instances by employing similarity functions, such as in ontoX.

Criteria for Classifying OBIE systems

Tab. 3.5 presents the most representative and influential OBIE systems proposed so far according to some features that distinguish them:

Table 3.5. Summary of state-of-the-art OBIE systems.

System and/or Main reference	IE method	Extracted elements	Preprocessing Tasks / NLP tool(s)	Additional resources used	Types of Sources
Embley (2004)	Pattern matching (regex)	Class instances, data property values	None	Extraction Ontology (wrappers)	Car´s ads web pages
OntoX [Yildiz and Miksch, 2007]	Induction of Regular expressions	Class instances, datatype property values	Stop words, named entity similarity function	Digital camera extraction ontology	Web pages on digital cameras
iDocument [Adrian *et al*., 2009]	Regex, similarity measure	Class instances, datatype property values	POS, Gazetteers, Chunking	Extraction ontology (Wrappers)	Corpus from a domain
FLOPPIES [Nederstigt *et al*., 2014]	Regex, Similarity measure	Class instances, datatype property values	none	OntoProduct ontology	Web pages with tabular product data
PANKOV and C-PANKOW [Cimiano *et al*., 2004] [Cimiano *et al*., 2005]	Web-based search (Hearst patterns)	Class instances	POS	Domain ontology	Web pages
OntoSyphon [McDowell and Cafarella, 2006]	Web-based search (Hearst patterns)	Class instances	none	Domain ontology	Web pages
SOBA [Buitelaar and Siegel, 2006]	Wrapper (regex), grammar based-rules	Class instances, data property values, object property values	POS, NER , Morphological analysis, Parsing / SProUT	SWInto ontology	Web pages on soccer events
Saggion *et al*., (2007)	Linguistic rules	Class instances, datatype property values	NER, Gazetteers / GATE	MUSING ontology	Corpus from e-business domain
KIM [Popov *et al*, 2004]	Linguistic rules	Class instances, data property values, object property values	POS, Gazetteers, NER, coreference resolution / GATE	KIM ontology	Corpus from a domain
Artequakt [Kim *et al*., 2002]	Linguistic rules	Class instances, object properties	NER, Parsing, Gazetteers, Coreference resolution (simple pronoun replacement)	WordNet, Artequart ontology	Web pages about artists
Hermes [IJntema *et al*., 2012]	Linguistic, rule compiler	Class instances, object properties	Gazetteers, POS, NER / GATE	Financial ontology	Financial and political news

BioOntoVerb [Ruiz-Martínez, 2012]	Linguistic rules	Class instances, object property values	Gazetteers, POS, NER, Syntactic parsing (GATE, GENIA NER)	Gazeteers, WordNet, VerbNet, FrameNet, BioTop ontology	Biomedical corpus domain
SMES [Maedche *et al*., 2012]	Parsing tree based-rules, machine learning (inference)	Classes, class instances, data and object properties values	POS, Morphological, NER, chunk and dependency parsing	Lexical database	Corpus from tourism domain
BOEMIE [Castano *et al*., 2008]	Pattern-based , machine learning (inference)	Classes, class instances, data and object properties values	POS, NER, gazetteers, chunking, coreference resolution (GATE)	Several ontologies related to the athletes domain	Web pages and multimedia sources on sports events

The above-mentioned OBIE systems use manually defined extraction patterns based on regex which is not so expressive and demands one or more domain experts to read all documents of the corpus and figure out suitable extraction rules. In addition, this tedious and time-consuming process does not scale well and put serious limitations concerning the portability of the IE system based on extraction ontologies.

Web-based Search. PANKOV [Cimiano *et al*, 2004] and OntoSyphon [McDowell and Cafarella, 2006] are OBIE systems that use queries on web-based search engine for extracting class instances. The key idea is to not be limited to a local corpus, but instead, take profit of the web as a big corpus.

The OBIE system called "Pattern-based Annotation through Knowledge on the Web (PANKOW)" semantically annotates a given web page only using web-based searches [Cimiano *et al*, 2004]. Using Hearst patterns [Hearst, 1992], PANKOW applies web searches for every combination of identified proper nouns in the document with all the concepts of the ontology for a set of linguistic patterns. Examples of Hearst patterns include *"<CONCEPT>s such as <INSTANCE>*", and *"<CONCEPT>s including <INSTANCE>*". The concept labels for the proper nouns are determined based on the aggregate number of hits recorded for each concept and returned by the web search engine.

The core of PANKOW is a pattern generation mechanism that creates pattern strings out of a certain pattern schema conveying a specific semantic relation, an instance to be annotated, and all the concepts from a given ontology. The ontological instance in question is semantically annotated according to the maximal evidence score measured by the concept having the largest number of hits.

C-PANKOW [Cimiano *et al*, 2005], PANKOW's successor, tackles the ambiguity problem by taking into account the context the entity to be annotated appears in. It overcomes this problem by actually downloading the pages, analysing them linguistically, and matching the patterns instead of merely generating them and counting their Google hits.

The OntoSyphon system uses a similar approach but aims to learn all possible information about some ontological concepts instead of extracting information from a local corpus [Cimiano *et al*, 2005].

Even though the Hearst patterns can be quite useful, the IE systems based on them have the limitation that the set of patterns is fixed and, therefore, cannot retrieve others constructions present in natural language texts. In addition, as Cimiano *et al*. (2015) has

pointed out, such patterns are quite rare, with a very low recall if one applies them on a local corpus, for instance.

The SOBA system [Buitelaar and Siegel, 2006] can be regarded as a hybrid OBIE system because it combines both techniques of extraction ontology with a manually tailored set of grammar-based rules able to identify and extract class instances from semi-structured documents, like web pages about soccer events.

Besides extracting class instances, SOBA also extracts object and data property values, i.e, classes attributes and instances of relations from text by first parsing the input text with the SProut parser [Drozdzynski *et al*., 2005] which provides POS tagging, morphological analyses, and full sentence parsing.

The ontology-based information extraction and integration system SOBA consists of a web crawler, linguistic annotation components, and a component responsible for transforming linguistic annotations into a knowledge base according to the SWIntO ontology [Buitelaar and Siegel, 2006]. This extraction ontology is the core knowledge resource used by SOBA because it integrates various domain and task ontologies for representing knowledge about football, navigation, discourse, and multimedia.

The OBIE process in SOBA is performed in two stages: First, linguistically annotated documents (by SProut) are further processed by the semantic transformation component, which generates a knowledge base of football-related entities (players, teams, etc.) and events (matches, goals, etc.) by mapping annotated entities and events to instances of ontology classes and their properties. Next, domain-specific and manually created rules extract football-specific entities, such as actors in soccer (trainer, player, refer, etc.), teams and tournaments. On top of these entity types, there exist other rules for extracting football- specific events, such as player activities (shots, headers, etc.), match events (goal, card, etc.), and match results.

Linguistic Rule-based. Linguistic rule-based OBIE systems are based on the idea of manually creating rules; however, here the extraction rules are not embedded within the domain ontology. Moreover, the linguistic rules are more expressive than the pattern expressions as the former normally relies on syntactic features derived from state-of-art NLP tools. Ought to the preprocessing step with such NLP tools, the enriched annotated text can be searched using more expressive extraction rules that consider linguistic features.

Example of such a OBIE systems is the one introduced in [Saggion *et al*., 2007].This system uses linguistic rules for extracting relevant semantic information to be used in business intelligence process in the areas of financial risk management, among others. It is based on the MUSING ontology that provides a knowledge base about companies, and ranked list of countries/regions for companies interested in investing into new country/regions. This system, like other systems of this group, uses GATE as NLP tools for providing gazetteers and NER preprocessing.

Other OBIE systems including Artequakt [Kim *et al*., 2002], KIM [Popov *et al*, 2004], and the one proposed by JInterma *et al.* (2012) has further components in their architecture, compared to Saggion's.

Artequakt [Kim *et al*., 2002] aims at extracting knowledge from the web about artists, populating a knowledge base, finally using it to generate personalized biographies. Once instances have been identified, the system uses a domain specific ontology and a generic one in order to extract binary relations between two instances. This system also incorporates GATE as the provider of gazetteers lists and the Pie parser for sentence parsing. In addition, Artequakt uses a simple resolution function that takes into account three attributes (gender, number, and structural information) when determining the best-

guessed referent. Artequakt reduces the problem of linguistic variation between relations defined in the ontology and the extracted facts, by using three lexical chains (synonyms, hypernyms, and hyponyms) as defined in WordNet. It uses heuristics to remove redundant instances from the ontology

KIM [Popov *et al*, 2004] is another system that heavily relies on the GATE framework for performing semantic annotation, indexing, and retrieval of natural language texts. The annotation process in KIM is guided by the KIMO ontology, a pre-populated knowledge base with entities of general importance that allow enough clue for the IE process consisting of above 80,000 entities. The essence of the KIM IE approach is the recognition of named entities with respect to its formal upper-level KIMO ontology. The grammar-based rules in KIM are based on the KIMO classes, rather than on a flat set of NE types.

The OBIE system proposed by IJntema *et al*. (2012) differs from the above systems based on linguistic rules because it provides its own Hermes Information Extraction Language (HIEL) that employs semantic concepts from an ontology. The HIEL system integrates both a rule compiler and rule matcher engines implemented in Java. It extracts instances of classes and non-taxonomical relations from the financial/political domain.

BioOntoVerb [Ruiz-Martínez, 2012] consists of a framework for ontology population based on the integration of ontological and linguistic resources on the biological domain. The framework comprises the BioOntoVerb ontological model [Ruiz-Martínez, 2010]. This ontology is derived after a previous mapping process between the BioTop ontology [Beisswanger *et al*., 2008] and the semantic roles resources VerbNet, FrameNet, and WordNet. BioTop ontology consists of an upper-level ontology for the life science domain intended to link and integrate various specific domain ontologies. This top level ontology defines the basic semantic relationships in the biomedical domain.

In the proposed OBIE classification here, the BioOntoVerb ontology can be viewed as an extraction ontology that contains trigger terms (nouns and verbs) and their variations, like synonyms that are retrieved from the WordNet. Then, with both the BioOntoVerb and a domain ontology, i.e., the ontology to be populated the OBIE process proceeds as follows:

First, the relations of the domain ontology have to be mapped onto the relations defined in BioOntoVerb. This process is manually performed by a domain expert. As a result, all the defined properties as well as the associated roles of the BioOntoVerb become part of the domain ontology. Second, the input text is pre-processed by GATE which performs POS tagging, NER, and syntactic parsing. Due to the particular aspects in biological entities, the GENIA tagger [Tsuruoka *et al*., 2005] is also used. The identified NE mentions by the GENIA tagger are considered as candidate instances in the ontology.

The ontology population step uses a combination of JAPE rules[12] and lists of gazetteers to perform a pattern matching process between the candidate instances and their contextual verbs with the ones present in the domain ontology already mapped to the BioOntoVerb. The matched candidate instances are inserted into the domain ontology.

In a final step, the Hermit reasoner[13] is executed in order to check for the consistency of the ontology, and compute inferred types.

Machine Learning for Inference. The last two OBIE systems surveyed in this thesis, namely SMES [Maedche *et al*., 2002] and BOEMIE [Castano *et al*., 2008] share the same underlying idea that it is not enough to detect named entities, associate them with properties, and relate them with other named entities, according to the concepts, and

[12] JAPE rules are a rich and flexible regular expression-based rule mechanism offered by the GATE framework.

[13] http://hermit-reasoner.com/

properties defined in the ontology. Their working hypothesis is that these extracted facts must be combined in order to be semantically interpreted (by inference) according to the domain ontology. In both systems, human subjects validate the inference results.

In both systems, the above processes are implemented as two distinct phases:

(i) *the information extraction phase* that extracts named entities and relations between them. In the SMES system, this is achieved by a preprocessing task involving POS, morphological analyses, NER, and both chunk and dependency parsing. In this phase, SMES also accesses a lexical database containing about 700,000 word forms as well as manually constructed extraction rules.

In BOEMIE, the preprocessing phase is based on the GATE framework that annotates text with POS tags, NER, gazetteers, chunking and coreference resolution. In BOEMIE, the IE process is based on JAPE rules that are defined manually.

(ii) The inference phase that, in both systems, can generate more complex concepts, i.e., concepts of higher level and relations among them are generated based on the previous extracted concepts, using a inference engine. Therefore, as new classes and properties are generated by both systems, this characterizes them as ontology learning systems as well. It is worth noting that, in contrast to lower-level concepts of the first level, the higher-level concepts usually cannot be mapped to textual fragments.

In fact, this last inference phase actually consists of a post-processing procedure that makes the OBIE system strongly dependent of the specific characteristics of the domain of interest. This certainly increases the manual effort for porting the system to new domains.

3.5.6. Discussion

The comparative analysis of the state-of-the-art OBIE systems seen so far leads to several conclusions:

1. Almost all work has been performed on text corpora; work on other modalities is practically non-existent. Among the surveyed related systems, only BOEMIE is able to process other types of input, including images and videos.
2. Almost all OBIE systems utilize gazetteers. As already discussed, such repositories are easy to integrate into the IE process, which certainly explains their widespread use as preprocessing components. However, these repositories require a lot of effort to keep them up-to-date with new variations of named entities emerging every day.
3. Since no conferences or standard text corpora for OBIE existed until very recently, most studies surveyed in this section have compiled their own corpora for experimental evaluation. In this light, it is clear that having standard text corpora and well defined tasks for the OBIE community, as it was done in MUC, would have a similar positive impact on the development of OBIE systems. This certainly would allow OBIE researchers to objectively evaluate different OBIE systems and identify their strengths and weaknesses.
4. Related work on ontology population heavily relies on linguistic preprocessing (especially syntactic parsing for RE), and the exploitation of WordNet as semantic lexicon resource. Other deeper natural language processing subtasks like semantic role labelling have not been exploited by the surveyed systems in an automatic way. For instance, only BioOntoVerb uses VerbNet and FrameNet, but with a manual mapping of the semantic structures found in VerbNet/FrameNet into ontological elements of the domain ontology. It is worth noting that NLP tools are becoming more mature and accurate, serving as an infrastructure for new IE techniques that can also exploit knowledge bases.

5. Surprisingly, none of the OBIE systems in Tab. 3.5 actually performs automatic construction or induction of extraction rules in symbolic form. SMES and BOEMIE systems are the only systems that provide an automatic way of pattern induction, but such patterns are learned in a post-processing stage in the IE process, and they are used for inference purposes, instead of IE. Therefore, the IE task itself is not really automatic for these two systems. The fact that all OBIE systems in Tab. 3.5 preferred to manually create linguistic-based extraction rules can be explained by the fact that it is easier and more straightforward to integrate such extraction rules than integrate statistical extraction models into the IE process. Moreover, the manually created extraction rules have an inherent symbolic aspect that facilitates their use and customization by a domain expert.
6. Most of the OBIE systems apply a single ontology for guiding their IE process. However, as pointed out by Wimalasuriya and Dou, 2010, there is no rule that forbids a system from using multiple ontologies. The same authors also argue that the use of multiple related but distinct ontologies can bring several benefits to IE. Therefore, more analyses and experiments are needed to explore this line of work.
7. Very few OBIE systems are able to extract implicit information extraction. With the exception of SMES and BOEMIE that can derive implicit information by performing a post-processing step after the normal information extraction step.

In this thesis, some of the aforementioned problems identified in the state-of-the-art in IE will be addressed.

In addition, this thesis subscribes to the idea that by making domain knowledge explicitly via an ontology, the direct benefits to IE are two-fold:

- ontologies can offer new opportunities for IE systems, ranging from using them for storing the extracted information to using reasoning for improving various IE tasks.
- IE system's portability may be enhanced by allowing the adaptation of the system’s behaviour via changes in the ontologies.

On this basis, this thesis attempts to directly address the issues discussed in the conclusion points 3-7 seen above:

- (Problem 3) The experimental evaluation conducted in thesis is based on several standard datasets for NER and RE. Thereby, comparative assessments are feasible.
- (Problem 4) One of the key aspects addressed in this thesis concerns the construction of a more effective OBIE system by exploiting relevant prior knowledge. In the proposed method, such knowledge is expressed by annotations derived from more sophisticated (deep) text preprocessing, and mapping to semantic resources, and a top-level ontology. These last resources certainly enable a richer text annotation producing several useful features that can be exploited by the ILP-based component for information extraction in the framework proposed in this thesis.

  More concretely, this thesis proposes a NLP component that offers:
  - a very rich set of natural language processing subtasks, such as dependency parsing, pronoun normalization, coreference resolution, word sense disambiguation, and semantic role labelling;
  - mapping to semantic resources, including WordNet synonyms/hypernyms and WordNet domains, and;
  - mapping to the top-level SUMO ontology.

- (Problem 5) The OBIE method introduced in this thesis relies on Inductive Logic Programming, a supervised ML technique which allows the induction of symbolic extraction rules expressed in Horn clauses. The decision for adopting the symbolic rule induction technique is motivated by the fact that, being expressed in declarative way as Prolog rules (or Horn clauses), a knowledge engineer can easily intervene in the IE process by, for example, validating the extraction rules, optimizing the rules, or use them as components to form other rules or axioms in a domain ontology.
- (Problem 6) In the proposed solution, two ontologies are integrated: the first one representing the domain elements (classes and relations), while the second provides a means for integrating and formalizing BK in terms of flexible definitions of lexical, syntactic, semantic, and relational features used by the rule learning component. Moreover, the second ontology is domain-independent, presenting the additional advantage of being invariant to domain changes.
- (Problem 7) The symbolic machine learning component proposed in this thesis is able to infer implicit information from text.

3.6. Conclusion

The goal of this chapter was to present the state-of-the-art approaches to IE. IE, as a text mining problem, has been thoroughly investigated in areas including NLP, Web Mining, and IR. IE has the primary goal of discovering and structuring information found in semi-structured or unstructured documents. This chapter reviewed some representative work on IE/OBIE, particularly on NER (or class instances extraction) and RE (or non-taxonomical relations).

Two main approaches to automatically detecting and classifying instances of entities and relations in textual data are surveyed. A classification of the state-of-the-art IE systems according to several relevant published surveys was proposed. The advantages and limitations of these approaches were also highlighted.

It was seen that the IE systems adopting the knowledge engineering approach try to exploit the regularities in natural language in order to seek for common patterns or rules that can match such regularities. IE systems following this approach achieve good performance, particularly when processing semi-structured text, such as web pages. However, their drawback resides on their simple learning mechanisms that cannot provide enough generalization capabilities, resulting in low recall performance. This limitation is particularly noticeable on a more difficult IE scenario, such as extraction from natural language texts.

On the other side, the statistical-based models cast the IE task as a classification problem, in which annotated examples are passed as input to a machine learning algorithm that generates, by induction, a statistical-based model suited for classifying new unseen examples. IE systems adopting this approach usually incorporate diverse types of features, including lexical information, syntactic, and semantic features. Such systems are more robust to domain changes. However, their inherent limitation is that their induced models are commonly complex and difficult for a given user to understand.

A detailed discussion on ILP-based IE systems closely related to the one proposed in this thesis was presented. This discussion was about the benefits and the drawbacks of such systems which use FOL as the language for both BK and example representations.

This chapter has also presented the emerging research OBIE field as a promising line of investigation that aims at enhancing classical IE by combining machine learning and

ontologies. On the one hand, machine learning techniques have been very attractive to IE community since a specific information extraction model is required for each IE process and, in many cases, providing annotated examples is easier than manually creating models.

Classical IE and OBIE share challenges and open problems, including ambiguity, variability, and portability. Another challenge for the IE field concerns the fact that implicit information is hard to extract. However, according to several researchers, OBIE systems have the great potential to mitigate these problems.

Finally, a detailed comparative analysis of the state-of-the-art OBIE systems was provided. The analysis revealed some gaps in the field that will receive special attention by this thesis. Thus, some of the aforementioned problems will be addressed in the next chapter.

Chapter 4

An Ontology- and Inductive Logic Programming-based Method for Entity and Relation Extraction

This chapter introduces OntoILPER, an *Ontology- and Inductive Logic Programming-based Method to Extract Instances of Entities and Relations from Texts*. This is the most fundamental contribution of this thesis; besides that, the following contributions directly related to the proposed OBIE method are also discussed:

- the *shallow* and *deep natural language processing techniques* used in it,
- the *graph-based model for sentence representation* that encompasses several kinds of features;
- a *simplification method* for that representation aiming at alleviating overfitting of the generated models, and
- the *annotation ontology* that turns the solution into a fully-fledged OBIE framework.

OntoILPER consists of a specific inductive logic programming-based approach to knowledge acquisition, which uses ontologies for extracting instances of classes and relations from texts, and feeds back the extracted information, performing the ontology population task. It relies on rich text preprocessing, linguistic-based knowledge sources, and ontologies that formalize the background knowledge for achieving state-of-the-art performance.

The working hypothesis that guides OntoILPER is that an automatic acquisition of a substantial body of linguistic knowledge from textual data, and its formalization using ILP, can enable the generation of effective information extraction models.

The extraction models in OntoILPER can perform the following classification tasks:

1. *class membership prediction* that aims to assign class individuals to ontological classes, and;
2. *relation prediction* which attempts to characterize the relationship between two class individuals. In other words, it aims to identify *non-taxonomical relations* between two class individuals.

Compared to related work, the OntoILPER distinguishing advantages resides in the following:

- the higher expressiveness of its extraction rules and the rich set of features that are explored by an ILP-based component for inducing symbolic extraction rules. More concretely, the ILP learning technique was adopted as the core component for building classification models.
- the inductive learning component in OntoILPER allows prior knowledge about the domain to be integrated into the construction of the classification model in such a way that, the classification of examples are performed by reasoning involving their syntactic, semantic, and structural features defined by a rich relational representation model. This representation model and a simplification method aimed at alleviating overfitting of the examples are also proposed.
- The information extraction process in OntoILPER is *ontology-based* in the sense that ontologies not only provide the initial knowledge to the extraction process, but they also formalize the graph-based representation of sentences and examples.

Concerning the integration of ontologies in OntoILPER, it is argued that the more expressive representation model provided by formal ontologies may open up new opportunities for an effective and adaptive knowledge extraction process.

In what follows, the IE tasks addressed by this thesis are formally introduced, followed by the underlying assumptions OntoILPER is based on. Then, an overview of the OntoILPER functional architecture is presented.

4.1. Task Definition and Work Assumptions

This section considers the task of identifying and extracting instances of predefined entities (or *entity types*) and relations (or *relation types*) from text. In a conceptual view, entities and relations can be visualized as shown by the directed graph in Fig. 4.1. In this graph, *nodes* denote entities, or *phrase constituents*, whereas the *edges* represent *binary relationships* between entities.

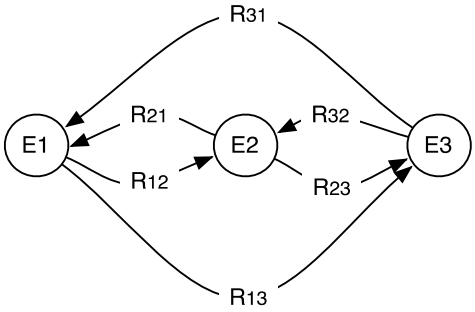


Figure 4.1. Conceptual view of relation extraction examples.

In the proposed approach, the identification of both entity and relation types is treated as the *target* learning problem. Therefore, the learning problem is cast as a *classification problem.*

Putting it more formally: given a sentence S formed by an ordered sequence of words w and entities e_i $\{e_1, e_2, ..., e_n\}$ in S, and a binary relation between a pair of entities contained in S, i.e., $R_{ij} = (e_i, e_j)$, where e_i and e_j are the first and second argument of relation R_{ij} respectively, the task of relation extraction is to correctly assign a label $t_i \in T_R$ to the set of all distinct relation instances $\{R_{ij}\}$ in S.

The set of predefined entity and relation types are restricted to T_E and T_R, respectively. The relation instances R_{ij} are usually *directed*, i.e., $R_{ij} \neq R_{ji}$, since the involving entities, e_i *and* e_j may play different roles in the same sentence S.

Other starting assumptions concern the input elements to OntoILPER, namely: the *domain ontology*, and the *corpus* (a set of text documents):

i. the fundamental assumption is that the domain ontology already exists before the entire OBIE process takes place. This ontology conveys concepts and relations relevant to the application domain;
ii. the entity types in a sentence can be already given by the input corpus, or can be recognized by the NLP component in the preprocessing phase. In other cases, an early classification of the entity types has to be performed. In this work, an entity instance is represented as one or more *consecutive lexical items* with a predefined boundary. As a result, one can assume that multi-word nouns, with their corresponding head word, denote an entity instance;

iii. only relations between entities within the same sentence are considered. This is the case of several standard datasets for evaluating RE systems, such as TREC and ACE evaluation campaigns. In these corpora, relations evolving entities present in different sentences are not annotated.
iv. reflexive relations are not considered, i.e., R_{ii}.
v. entity and relation classifier are learned independently of each other, i.e., they are trained and tested on different copies of the examples in a given corpus;
vi. finally, it is assumed that negative examples of entities or relations can be derived as the complement of the positive ones. The set of both positive and negative examples are called *candidate instances*.

4.2. OntoILPER Overview

The present thesis focuses on the proposal and evaluation of an IE system based on ontologies. More concretely, this thesis introduces the OBIE method, OntoILPER, and its implementation as a framework suitable for extracting instances of entities[1] and relations[2] from natural language texts.

In this work, an entity type denotes a set of objects sharing the same characteristics in a given domain, whereas a binary relation denotes the relationship of two instances of entity distinct entity types. The interest on binary relations, in special *non-taxonomic binary relations* [Baader *et al.*, 2008], can be motivated by the fact that the Semantic Web has studied many ways of formalizing knowledge representation based on classes of individuals and binary relations among them by means of ontologies. Indeed, OWL/DL ontologies can model complex domains by means of basic axioms defining several binary relations [Baader *et al*., 2008].

OntoILPER relies on Inductive Logic Programming, a supervised ML technique that allows the hypothesis space to be searched for good hypotheses. In other words, the generated hypotheses constitute the set of final extraction rules expressed in Horn clauses.

The decision for adopting a symbolic rule induction technique is motivated by several reasons, improving related work with respect to the following aspects:

- contrarily to statistical methods, the symbolic rule induction technique employs a declarative representation which means that hypotheses are understandable and interpretable by humans, and it offers a useful mechanism for defining background in declarative form.

- both BK and examples are expressed in the same symbolic level which allows to enrich the IE process by integrating additional semantic resources, such as thesauri or ontologies, without modifying the core of the IE process. For instance, any constraint to the problem can be expressed in the form of auxiliary predicate definitions provided by the user as additional BK. Moreover, the first-order formalism is able to take into account the structural information (relation features) of the examples.

- it overcomes the representational limitations of attribute-value (propositional) learning systems in related work that employ a table-based example representation. In this formalism, the learning examples correspond to rows in a table; and the

[1] For the purposes of this thesis, entity types and classes (or concepts) in ontologies can be considered as equivalents, as one can always represent an entity type as a class in an ontology.

[2] In ontological terms, relations correspond to non-taxonomical relations. In this context, non-taxonomical relations are also called *object properties*.

features, to columns in which a single value is assigned to each one of the attributes [DeRaedt, 2010].

To summarize, the working hypothesis that guides this thesis is that, by using the richer ILP formalism, the proposed OBIE method should be able to directly represent a vast amount of BK extracted from ontologies, semantic resources, and both shallow and deep analysis originated from natural language annotation tools.

Compared to related work, the present study also introduces the use of several types of background knowledge integrated into the IE process. More concretely, this thesis proposes and evaluates a *unified relational model* for representing both entities and relations found in textual data. This model consists of an expressive *graph-based model for sentence representation* that comprises four types of features suitable for describing the examples in the proposed solution, including *lexical*, *syntactic*, *semantic*, and *relational* features. Such features are exploited by a machine learning technique able to induce *symbolic extraction rules*.

In addition, a method to simplify the graph-based representation of sentences is proposed. This method comprises several simplification rules that reduce the complexity of the graphs representing sentences by eliminating both spurious nodes and relations. The key idea is to speed up the learning phase by applying several rules for graph simplification that constrain the hypothesis space before generating extraction rules.

Another distinct feature of *OntoILPER*, improving related work, is that it integrates two ontologies:

- the *domain ontology*, representing the domain elements to be extracted (instances of classes and relations); and
- the *annotation ontology*, integrating and formalizing background knowledge in terms of a flexible mechanism for defining lexical, syntactic, semantic, and structural features used by the learning component. In other words, this ontology mirrors the relational model of the annotated instances of entities and relations.
  Moreover, this ontology is *domain-independent* which means that there is no need to be updated when the domain of the application changes. As a result, such a rational use of ontologies in OntoILPER enables it to be easily portable to new domains.
  Concerning the above ontologies, the thesis guiding principle is that ontological knowledge bases, such as OWL/DL or RDF data sources, can be mapped to graph structures and vice-versa.

OntoILPER Functional Architecture

An overview of the pipelined processing flow performed by OntoILPER is shown in Fig. 4.2.

In OntoILPER, the ontology-based information extraction process is performed in two distinct operational modes: *training* and *application*.

1. In Training Mode, a theory (a set of extraction rules) is induced from a given annotated corpus. In this phase, OntoILPER generates extraction rules from an annotated corpus containing the examples.
2. In Application Mode, or Rule Application Mode, the learned theory is then applied to extract ontology instances from new-tagged documents. For both modes, a previous corpus pre-processing stage takes place in which several Natural Language Processing (NLP) tools are exploited, followed by an automatic representation of the examples according to an expressive hypothesis space also proposed by this thesis.

In what follows, the OntoILPER processing flow depicted in Fig. 4.2 is explained, according to the two operation modes mentioned above.

In training mode (the top part of Fig. 4.2), the *Text Preprocessing* stage annotates the input documents by performing several NLP subtasks (to be detailed in next chapter) that enriches the documents with morphosyntactical and semantic annotations. The linguistically annotated documents are passed as input to the *Sentence Representation and Simplification* stages. This last stage unifies the various annotated elements produced after the Text Preprocessing stage into a graph-based representation model of sentences and examples. This graph-based representation model enables, by applying syntactic-based simplification rules, the simplification (reduction in size) of the graph instances representing sentences without losing relevant information for the IE process.

Then, the next stage, the *BK Generation* stage, is in charge of transforming all graphs produced and simplified by the previous stage into logical ground predicates (factual Prolog clauses). In addition, this stage has access to both domain and annotation ontologies that provide valuable information in terms of TBox and ABox elements as background knowledge. This ontological BK information allows for the creation of a more flexible and adaptive IE system [Wimalasuriya and Dou, 2009].

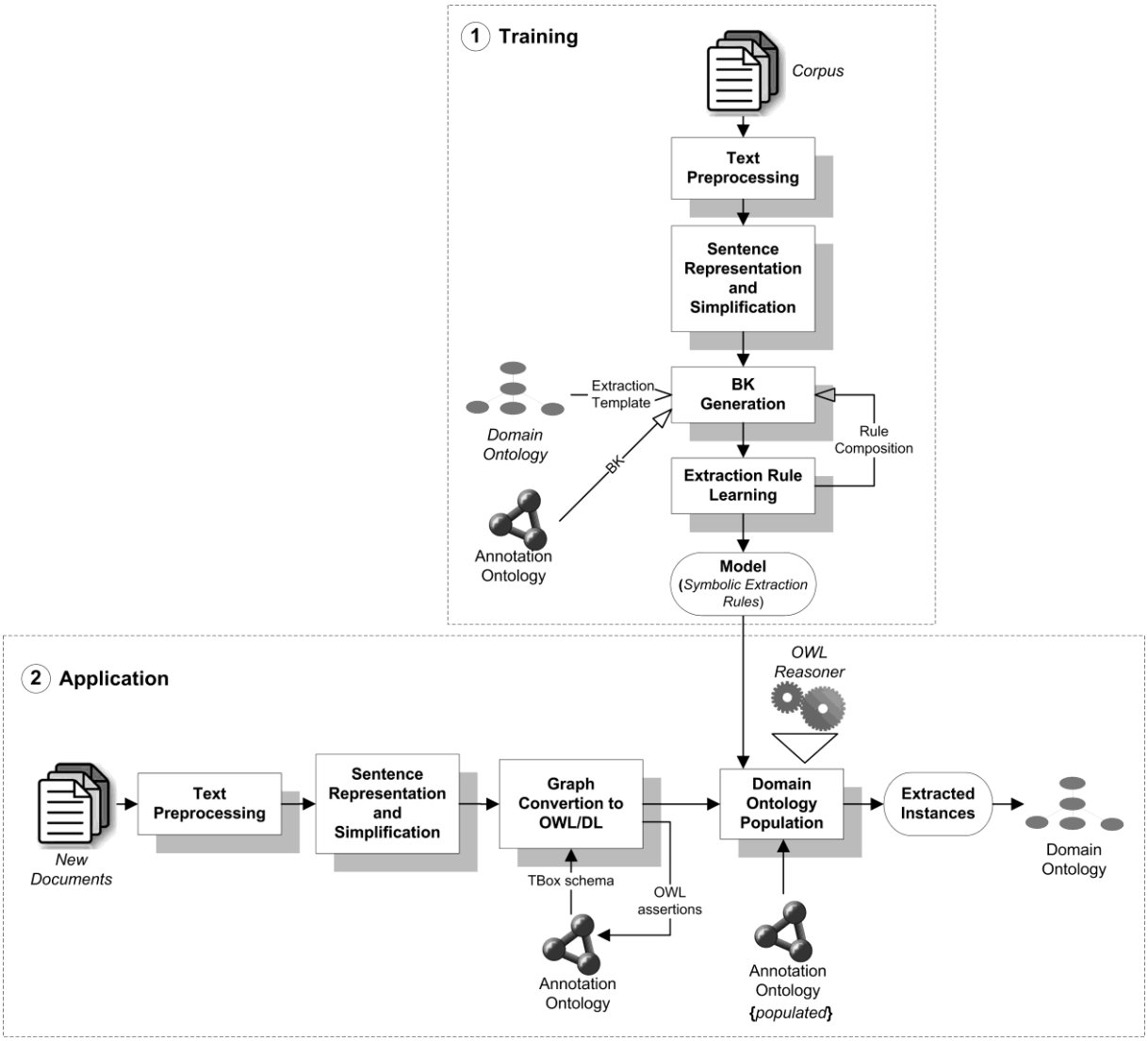


Figure 4.2. Main processing stages of the OntoILPER.

Next in the *Extraction Rule Learning stage*, a general Inductive Logic Programming system provided with the previous generated BK as well as customizable parameters induces symbolic rules expressed as a set of logical predicates (Prolog rules). This last stage in OntoILPER training mode produces a classification model composed of Progol rules.

Considering now the OntoILPER application mode (the bottom part of Fig. 4.2), the same two stages of the training mode (the Text Preprocessing and the Sentence Representation and Simplification) are executed on a new input corpus from which instances of classes and relations will be extracted.

The difference here is that, in the *Graph Conversion* stage, the simplified graphs are converted to ABox assertions (instances) and integrated into the *Annotation Ontology*. The Annotation Ontology consists of a lightweight and domain-independent ontology that has two major purposes in this methodology:

- it provides BK information by means of annotation-related ontological elements; and
- it serves as a repository of the candidate instances of classes and relations formalized by the Annotation Ontology.

Finally, using the set of extraction rules, or the Extraction Model (see Fig. 4.2) produced by OntoILPER in training mode, the *Domain Ontology Population* stage performs the ontology population task.

For performing the final classification and extraction of the entities and relation instances, the Domain Ontology Population stage has to integrate the following elements:

- the Annotation Ontology, regarded here as the knowledge base of candidate examples,
- the Extraction Model, converted to the same rule formalism of the Annotation Ontology, and
- the domain ontology, before its execution.

Actually, a previous rule conversion step has to be performed in order to convert the Extraction Model (in Prolog) to the same formalism employed by the Annotation Ontology (e.g., SWRL rules).

Provided with these elements, the Domain Ontology Population stage applies the Extraction Model on the Annotation Ontology by utilizing an OWL reasoner that is in charge of the final classification of the candidate instances.

As a result, the extracted instances can be used for populating the domain ontology with new instances of classes and relations.

The rest of this chapter is structured as follows: Section 4.3 shows how shallow and deep NLP subtasks were integrated to the OntoILPER preprocessing stage, which led to a graph-based model for sentence representation (Section 4.4) presenting the distinct advantage of being both very expressive and flexible. Another advantage of the proposed graph-based model for sentence representation is that it can be simplified by a method (proposed in Section 4.5) which has the potential of alleviating the classical overfitting problem of the classification models, reducing noise in the representation of the examples as well. Then, Section 4.6 present the way how ontologies are incorporated as BK into the OntoILPER rule learning process. More precisely, the roles played by both domain and annotation ontologies (bottom side of Fig. 4.2) will the described. Finally, this chapter presents the advantages and limitations of OntoILPER in Section 4.7 and some conclusions and final remarks in Section 4.8.

4.3. Text Preprocessing: Integrating Shallow and Deep NLP for Effective IE

For IE, there is no doubt that natural language processing technologies have been of great importance for analysing textual resources (usually consisting of free texts) and extracting their meaning. The result of such textual analysis, or *annotation process*, is usually performed automatically and can provide both syntactical and semantic descriptions in the form of text features. On the one hand, if the text is pre-analysed by NLP tools, information extraction rules can be expressed in a more abstract and powerful way. In that case, the rules can be applied on a *normalised representation* of the text obtained from the pre-analysis [Jurafsky and Martin, 2009] [Nedellec and Nazarenko, 2005]. On the other hand, there comes a difficulty in this annotation process due to the richness and complexity of natural language, in which a given word or phrase may have different meanings (polysemy), and the same information can be expressed by several formulations (paraphrases). This difficulty is even more evident if the information extraction rules rely on *surface clues* (i.e. the presence of a given specific lexical item, the word distance to some referential element in the sentence or word order). In this case, the whole set of very specific rules must be designed for each new IE application domain.

For instance, a first stage of a normalised version of the document consists in the typical operations of converting plural nouns to their singular forms, verb tenses to its infinitive form, and word derivations (suffixes and affixes) to their stemming or lemma. Another example: based on the subject and object syntactic dependencies in a sentence given by dependency parsing, information extraction rules can be more general and easier to interpret because syntactic dependencies, expressed by a dependency graph, are independent of word order [Jurafsky and Martin, 2009]. The same occurs with sentences in passive voice, in which syntactic dependencies can abstract the relative position of subject and object (with respect to the main shared verb) in both active e passive voices.

Due to its inherent complexity, natural language analysis is not carried out in a single large stage. Instead, computational linguistics deals with natural language at several layers of processing.

Chap. 2 showed a typical decomposition of such an analysis into the identification of words (the lexical level), the organization of word groups in phrases (the syntactic level), and the meaning that can be assigned to these words (the semantic level).

For the purposes of this thesis, the NLP for IE is considered from a perspective of its decomposition into those three major components, as depicted in Fig. 4.3. This figure displays a broad view of the NLP pipeline proposed in this thesis[3].

In Fig. 4.3, each NLP module enhances the text representation with a *layer of annotation*, which represents explicit linguistic and/or semantic information attached to text in machine-readable form.

First, after tokenization, lexical variants (inflexion and derivation) are unified by the morphological analysis. Thus, these variants are transformed to a canonical base form. This already presents an advantage because extraction rules can be expressed in a more abstract way. Another advantage is that, via lexicon look-up operations, one is able to relate these canonical forms to a corresponding entry in a linguistic (semantic) lexicon.

Second, the syntactical analysis identifies the structural relationships holding between groups of words at the sentence level. In fact, categorical information are attached to each lexical item in terms of its part-of-speech (e.g., noun, verb, etc.). This is a requirement to

[3] A detailed description of this NLP component will be presented in Chap. 5.

determining groups of words (chunking analysis) that grammatically belong to the same category. In this way, one can constitute larger syntactic units.

The final analysis links the terms or tokens to semantical lexicons (such as WordNet and VerbNet), and the SUMO upper-level ontology. Such lexicons offers different semantic relations, including synonyms and hyponyms from WordNet, while VerbNet opens up further opportunities for semantic interpretation because semantic relations among verbs and arguments are treated as predicates or propositions.

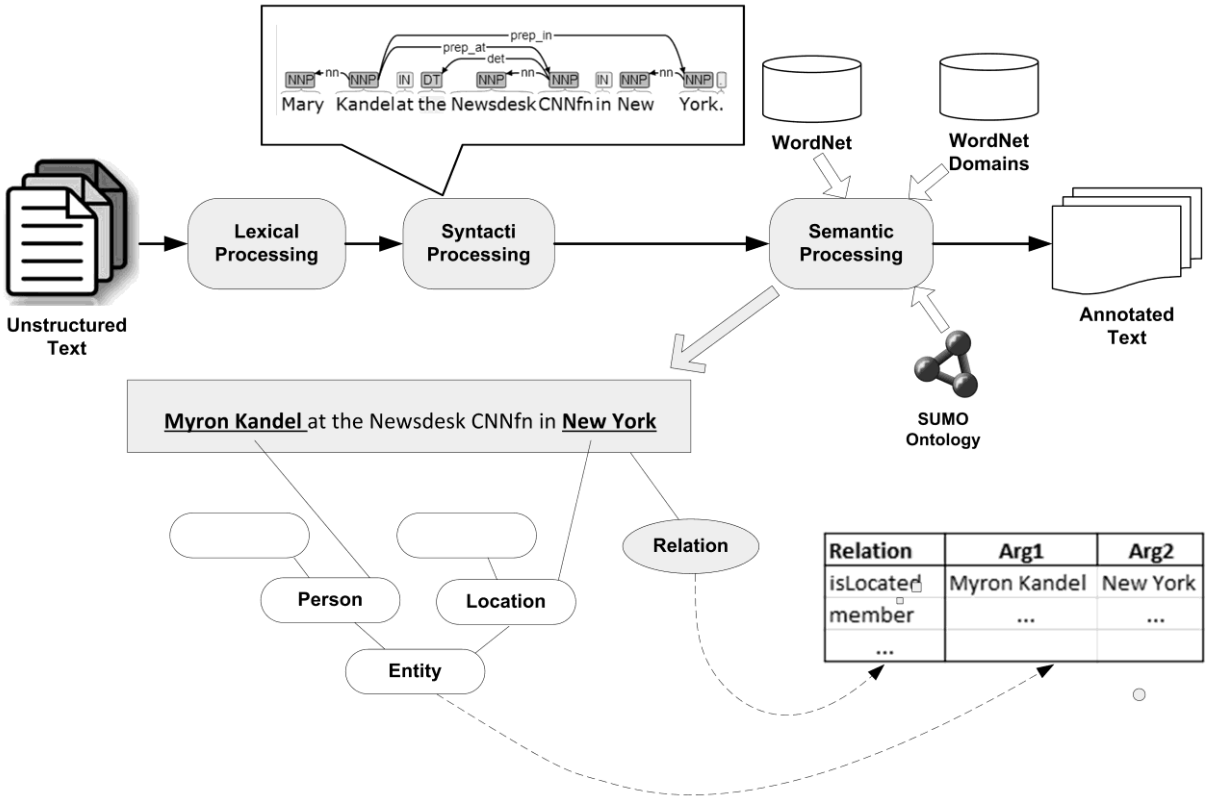


Figure 4.3. Overview of the NLP pipeline in OntoILPER.

To sum up, this thesis proposes a comprehensive NLP component that not only considers the lexical and syntactic features typically used by related work, but also enables the integration of several semantic resources and ontologies. More precisely, several layers of text annotation derived from text preprocessing, including shallow and deep NLP tools, semantic resources (WordNet, WordNet Domains, VerbNet), and semantic mapping to the top-level SUMO ontology comprise a rich set of features for describing sentences and examples in OntoILPER.

The ultimate goal of such a sophisticated text preprocessing is to alleviate high terminological variability, typical in many domains, and focus on the linguistic annotations that can be employed for neutralising the effects of such variation.

The remainder of this section will focus on the aspects related to a rich annotation of textual corpora aiming at effectively integrating several layers of linguistic annotations carried out in OntoILPER.

4.3.1. Rich Annotation of Textual Corpora

NLP tools annotate corpora according to their underlying linguistic theories, which determine somehow the kind of annotation or information to be encoded. In this case, the resulting encoding of annotation is thus designed to fit the specific theory used for describing the data. Consequently, the encoding of other theoretical linguistic descriptions cannot be supported.

Aiming at standardizing corpora annotation that take into account the possible integration of different theoretical linguistic descriptions, the standardization proposals should concern, according to [Schmidt, 2005]:

- The *logical data structure*: it defines the data models that are used for modeling linguistic phenomena and their properties. For example, the hierarchical structures like trees or graphs for syntax annotations in the TIGGER/SALSA [Erk and Pado, 2004].
- The *physical data structure*: due to its portability, XML has become the widely-recognized standard format.
- The *content in XML-based representation*: This concerns XML applications for specific linguistic annotation. An example of such a specific linguistic annotation proposal is the Text Encoding Initiative [Sperberg and Bernard, 1994] that defines highly detailed DTDs for encoding all kinds of bibliographic information.

The first models for corpora annotation only featured POS and synthetic annotations, e.g., *Penn Treebank* [Marcus *et al.*,1993]. Other more recent model proposals for corpora annotation switched to properties beyond the morpho-syntactic level. The basic idea here consisting in combining different types of annotation, also called *multi-level annotation*.

For representing multi-level annotations, models like the *Linguistic Annotation Framework* (*LAF*) [Ide *et al*., 2003] and *PAULA* [Dipper *et al*., 2007] have been introduced. These models define general, *multi-rooted graphs* whose nodes can be augmented by features, including timing information. Inspired by the LAF format, PAULA serves as an *interlingua* for the representation of several kinds of corpora annotations formats.

All mentioned models for corpora annotation use either (i) *stand-off* or (ii) *inline annotation*. In stand-off annotation, the source text and its annotation are stored in separate files. This type of annotation presents some important advantages [Dipper *et al.*, 2007]:

- it leaves the source text untouched;
- it allows for alternative annotations to be represented, e.g. variants of POS annotations resulted by different POS taggers;
- the annotations at different levels can be created and modified independently of each other by distributing annotations over different files. Individual elements are referenced to their annotations by the use of unique identifiers, or pointers.

In short, in stand-off annotation, not only is the source text separated from its annotations, but individual annotations are separated from each other as well.

The disadvantages of stand-off annotation concern the fact that there is no immediate connection between the text and its annotation. Thus, extra care has to be taken in order to synchronize all annotations when the source text is modified. As annotation is scattered out in several annotation files, human inspection of the annotations becomes quite cumbersome.

On the other hand, inline annotation allows that all information referring to the same token or span of them are annotated as attributes of one element, and they are inserted in the same file.

The inline representation is guided by the following principles [Dipper *et al.*, 2007]:

- all the annotations are encoded at the units that they refer to. Therefore, there is no use of *pointers*. For instance, the basic unit of granularity assumed in other types of text representations is usually a token. Hence, all tokens in a sentence are represented by a *token layer*, where each token is uniquely identified by an *id* which in turn is used by other compound layers (e.g., a sentence layer) that links its constituents to the token layer.
- no redundant information is permitted.
- use of the genuine XML data model where the XML-child relation is used for encoding relations in trees.

The advantages of inline annotation compared to stand-off annotation is that the former is more human readable and querying inline annotation using standard XML queries libraries tend to be more efficient. Indeed, it is easy to note that querying stand-off data usually involves following up the links between different layers of annotations, which involves considerably complex query expressions, even for a simple query of a single word or token [Dipper *et al*., 2007]

4.3.2. Representation of Annotated Corpora

Particularly to the OBIE scenario primarily concerned by this thesis, the integration of annotation data produced by different NLP tools is a big concern.

Similar to related work on standardization of annotation seen earlier, the Natural Language Processing stage performed by OntoILPER needs to properly represent different output formats and different morpho-syntactic and semantic annotations.

Due to the diverse nature of linguistic annotation performed by several NLP tools, as seen in the previous section, the resultant natural language analysis is represented by distinct output formats, which unnecessarily complicate its posterior use. To alleviate this problem, a *canonical view model* of linguistic annotations was designed for OntoILPER, and it satisfies the following constraints:

- it is flexible enough to integrate the encoding of different linguistic annotation formats and frameworks;
- it is efficient in terms of computation time required for querying it because, contrarily to the pure stand-off representation, no external source is provided. Therefore, one benefits from the pointers without having to cope with the overhead caused by accessing external sources.
- it is feasible in terms of querying linguistic XML data with current off-the-shelf XML technologies;
- it is suitable for human readability, which facilitates the verification/correction of the resultant annotation.

This canonical annotation model consists of a hybrid XML representation of annotated corpora that meet the aforementioned requirements. It is based on the combination of two standardizations for linguistic annotation, namely PAULA (Erk and Pado, 2004) and TIGGER /SALSA (Dipper *et al*., 2007). From the PAULA format, the inline representation was adopted as it enables human readability and efficiency on queries to the model. Additionally, a flexible encoding of trees and graphs both representing several syntactic annotations was defined, as done in the TIGGER/SALSA model.

For the purposes of this thesis, a DTD (Document Type Definition)[4] as a means to define document structures is preferable because one can concisely define a coarse structure of valid

[4] W3C Recommendation, November, 2008. Extensible Markup Language (XML) 1.0, 5th edition. http://www.w3.org/TR/REC-xml

documents, whereas a XML Schema, for example, is more verbose, and harder to read and less intuitive. Furthermore, using DTDs, any validating XML parser can check the annotated corpora in the proposed framework for adherence to the annotation schema.

The DTD structure of the proposed canonical view model of annotations is shown in Fig. 4.4, where several *element type declarations* define all admissible elements and their contents. The elements that may be repeated are marked with a '+' symbol. Additionally, the typical XML element embedding structure is exploited in order to represent the different annotation layers (including syntactic parsing, chunking analysis, etc.) of a given sentence. Nodes features are realized as typical element attributes in XML (e.g. IDs, POS tags, word lemma, etc).

```
<! ELEMENT doc (sentences)>

<! ELEMENT sentences (sentence)+ >
<! ELEMENT sentence (tokens, syntactic_parsing, chunks, dependencies, instances) >

<! ELEMENT tokens (token)+ >      <!-- terminal nodes layer -->

<!-- layers of non-terminal nodes -->

<! ELEMENT syntactic_parsing (tree, (phrase)* ) >   <!—non-terminal phrase nodes -->
<! ELEMENT tree (phrase)+ >
<! ELEMENT phrase ( (phrase)*, (token)? ) >

<! ELEMENT chunks (chunk)+ >  <!-- non-terminal chunking nodes -->
<! ELEMENT chunk (tokens) >

<! ELEMENT dependencies (input,(pair)+ ) >   <!-- non-terminal dependency nodes -->
<! ELEMENT pair (arg1,arg2) >

<!ELEMENT srl (roleset)*>     <!-- non-terminal semantic role nodes -->
<!ELEMENT roleset (arg)*>

<! ELEMENT instances (nes, rels) >   <!-- non-terminal example nodes -->
<! ELEMENT nes (ne)+ >
<! ELEMENT rels (rel)+ >
```

Figure 4.4. DTD structure of the OntoILPER XML model for corpora annotation.

According to this DTD, an annotated document contains one or more sentences. Each sentence itself contains several layers of annotation, namely:

- *Token Layer*. The token layer corresponds to terminal nodes referenced by all other layers via *unique identifiers* which play the role of *pointers*. Actually, each main element in the above list of layers has *globally unique IDs*. This enables an efficient mapping between non-terminal elements and terminal ones during the parsing of the annotated corpora.
- *Syntactic Layer*. In the syntactic layer, constituent parsing of sentences is naturally represented as a nested structure with both *inner nodes* (*phrases*) and *outer nodes* (*terminal tokens*). This means that leaves of syntactic trees are token nodes of the syntactic structure.
- *Chunk Layer*. Such layers are composed of one or more tokens. Here redundant information about tokens is encoded for making overall processing more efficient.
- *Dependency Layer*. The dependency layer represents sentence structure by means of *directed acyclic graphs* (DAGs). This enables the representation of structure-sharing, where two nodes link to the same third node. In that case, two types of edges are distinguished by an attribute of the element "pair(arg1, arg2)" indicating the typed

dependency. The *arg1* and *arg2* arguments of the relation pair "pair(arg1, arg2)" are linked to the terminal token nodes.

- *Semantic Role Layer*. In this layer, the verbs in a sentence are directly linked to their arguments. Such verb arguments, identified by the VerbNet, are augmented with their corresponding semantic roles they play in the sentence. The additional information of the semantic role is assigned to it by the VerbNet. The *N* arguments of a given verb predicate are represented in the XML file as children of that verb.
- *Instance Layer*. Given that entities and relations are usually already annotated in IE corpora, the *instance layer* in the proposed DTD model was introduced in order to replicate the corresponding annotation already available in input corpora. The element <*nes*> encompasses all named entities annotated in corpora by the attributes of its inner element <*ne*>. In particular, the attribute of the element <*ne*> specifies the class of the named entity. Similarly, instances of relations are annotated by the XML element <*rel*>, whose attributes may indicate *types*, *subtypes* and whether an instance is a positive example or not.

It should be mentioned that the annotation layers could, in principle, be completely decoupled because all references between annotation layers is via identifiers that are unique throughout the corpus. In addition, this model can easily assimilate new annotation layers, such as layers for *semantic annotation*, by introducing additional semantic elements, and leaving the representation of all previous layers unchanged.

Actually, the last layers in the above DTD model contemplate supplementary annotations corresponding to shallow semantic features [Zouaq *et al.*, 2010], such as *synonym/hypernymy* from the WordNet [Miller, 2005], and VerbNet [Kipper *et al*., 2006].

An instance of a XML file illustrating all the layers mentioned above is presented in Section 5.2.

4.4. Sentence Representation: An Expressive Hypothesis Space for Generating Symbolic Extraction Rules

Previous studies on RE have demonstrated that the choice of features derived from analysis of NLP tools can strongly influence the extraction results. The features that have been used for RE include word, entity type, mention level, chunks, syntactic parse trees, and dependency relations [Zhou *et al.,* 2005] [Jiang, 2007] [Zhang and Zhou, 2008].

In RE community, there have been essentially two lines of research. The first one is mainly based on the direct selection of features (*feature engineering*) obtained from NLP tools [Kambhatla, 2004] [Zhao and Grisman, 2005] [Zhou *et al.,* 2005], [Jiang and Zhai, 2007], whereas the second line relies on *kernel functions*, which proposes a customized feature space [Jiang, 2012]. The kernel functions try to capture the similarity between two structured representations (sequences or trees) of relations instances [Cullota and Sorensen, 2004] [Zhang *et al.,* 2006] [Zhang and Zhou, 2008].

Despite the good results obtained, the aforementioned methods have some shortcomings according to a recent study [Choi, 2013]:

- feature-based SVM- and Maximum Entropy-based methods requires a lot of effort for feature extraction and selection;
- pure tree kernel-based methods make very limited use of structural information;
- kernel functions based on dependency trees have slow similarity calculation speed;
- shortest-path dependency tree kernels provide too simple structures for kernel functions.

Based on a careful investigation of previous work on features used for RE, and taking into consideration the limitations of kernel-based methods mentioned in Section 3.4, this thesis proposes a feature space that not only overcomes these limitations, but also provides a well-structured hypothesis space for the problem. The proposed hypothesis space combines both structural relations and properties of nodes in a graph-based model that integrates morpho-syntactic and semantic features.

Comparing OntoILPER with the aforementioned methods based on kernel functions, the following substantial improvements are proposed:

i. the feature selection engineering phase is based on a detailed study of the most useful and effective features for RE. Furthermore, the choice was motivated by the fact that each individual feature should have a clear meaning, that is, its meaning would be easily understood by an expert in the domain of interest. Indeed, some complex combinations of features, statistical ratios, for instance, were not considered in the feature selection phase, because their poor results demonstrated by previous work in RE [Jiang and Zhai, 2007] [Zhou *et al.*, 2005] [Zhou *et al.*, 2008].

ii. As stated before, pure Tree kernel-based methods use very little structural information. The same occurs with the Shortest-Path dependency tree kernel methods. On the contrary, one combines three levels of structural information (sequential, syntactic chunks, and dependency parse trees) in the proposed hypothesis space which allows a well-structured hypothesis space that can be systematically explored by the ILP-based learning component in the proposed architecture.

iii. This thesis also takes into consideration performance issues by choosing a compact set of informative and relevant features, as opposed with the hundreds or even thousands of sparse features commonly used by kernel-based methods. This condensed set of features demonstrated to be very effective due to the significant learning time reduction obtained in all experiments reported in Chapter 6. Another clear advantage of using a condensed set of features in learning is that one avoids dealing with redundant features.

The next sections 4.4.1 and 4.4.2 introduce another contribution of this thesis concerning the proposal of an Enhanced Entity-Relationship (EE-R) model in conjunction with the derived graph-based model for sentence representation adopted in OntoILPER.

4.4.1. A Relational Model for Representing Sentences and Examples

OntoILPER relies on an *Enhanced Entity-Relationship* (E-ER) model [Elmasri and Navathe, 2010] for representing not only the domain elements, including documents, sentences, chunks, and words, but also their taxonomical, non-taxonomical relationships, syntactic dependencies, and their underlying structural information.

The proposed E-ER model for sentence representation, depicted in Fig. 4.5, is grounded on dataset theory [Elmasri and Navathe, 2010], and offers an abstract representation of rich sentence annotation in OntoILPER.

In this model, entities are represented by rectangles and relationships by diamonds. For both entities and relationships, ovals denote their attributes. Key attributes that uniquely identify an instance of an entity in this model are underlined. Taxonomical relationships, such as a noun "is-a" word, are indicated by circles with the letter *"d"*, i.e., the distinguishing attribute.

This relational model delimitates the sentence representation in OntoILPER at a conceptual level, which also offers a flexible way of visualizing the concerned domain elements. The

rationale for using such a relational model is rooted in the fact that when learning about the properties of objects in relational domains, such as relation extraction, feature construction should be guided by the structure of the examples.

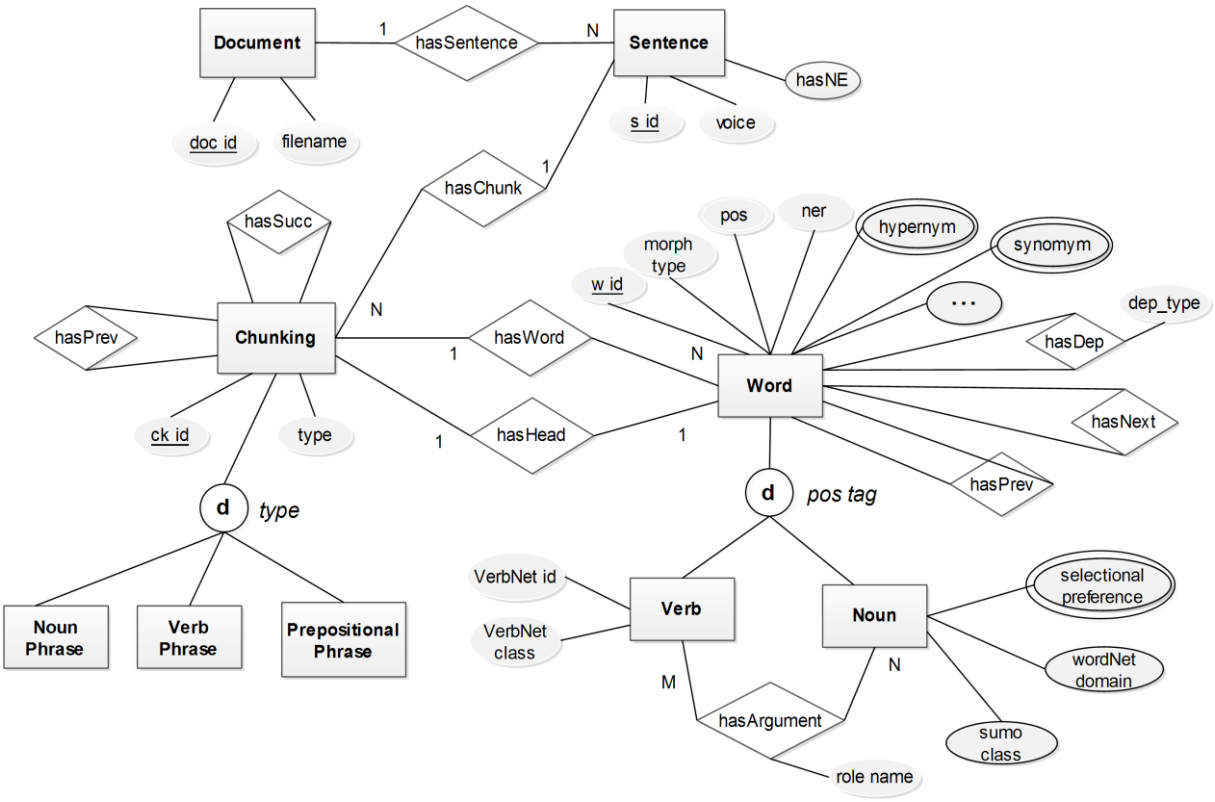


Figure 4.5. Enhanced Entity-Relationship model for sentence in representation in OntoILPER .

Accordingly, a binary relationship can be specified between conceptual entities (instances of classes) and each major phrasal constituent (either nominal or verbal) in a sentence, regarded as a candidate instance for extraction. Actually, OntoILPER focuses on the relational modelling, in addition to declarative feature construction and feature generation from graph-based sentence representations, as it will be shown in the next section.

The proposed E-ER model presents the following advantages: it enables the transformation of relational into graph-based representations of examples, allowing the integration of structural features into the learning process. Thus, the ILP-based learning component is capable of dealing with an extended high-dimensional feature space, which is much richer than all propositional representation employed by other machine learning approaches [FürnKranz, 2012].

Furthermore, the benefits of such a graph-based representation are to fully exploit the relational representation of linguistic-related annotations is two-fold: it provides a natural way to express language structures, and it allows for exploring the contextual features.

From the perspective of this EE-R data model, entity attributes denote predicates defining *properties*, whereas *relationships* between entities correspond to *structural* predicates.

In the following, it is illustrated way in which structural and properties predicates can be derived from the EE-R model shown in Fig. 4.5:

- *hasSent(D, S)* a document *D* contains the sentence *S*; Analogously, the predicate *hasChunk(S, C)* states that a sentence is composed of several chunks;

- *hasDependency(T, U, Dep)* denotes that the token *T* has a grammatical typed dependency *Dep* with the token *U*.

The previous predicates are example of structural predicates. In what follows, some examples of property predicates are given:

- *chunk_type(CK, chunk_type)*: a chunk (group of tokens) may be nominal, verbal or prepositional, i.e., *type = {noun, verb, prep}*
- *token_pos(T, t_pos)*: a given token *T* has the POS tag *t_pos.*

It is easy to notice that several other predicates like the ones presented above can be derived in an analogous way using the proposed EE-R model.

4.4.2. A Graph-based Model for Sentence Representation

OntoILPER employs a rich *relational (graph-based) model* of sentences based on both structural and properties features that describe mentions of entities and relations. Such features are considered here as *logical predicates*, which can be exploited by a full automatic learning system, which guarantees the discovery of symbolic extraction rules from examples. This method is based on the principle that the establishment of a relationship between two entities in the same sentence can be obtained, for instance, by a path between them in this graph, which encodes both morpho-syntactical attributes of individual words, and semantic relations between phrases constituents [De Marneffe and Manning, 2008].

The proposed representation that supports OntoILPER consists of a *graph-based model* of sentences. In this model, a relationship can be specified between *conceptual entities*: each major phrasal constituent (nominal and verbal chunk) in a sentence is considered as a candidate instance for extraction. In other words, all phrases that contain tokens or chunking constituents are either potentially referencing real-world concepts or semantic relations defined by a domain ontology of interest. Thus, this relational representation of the syntactic structure of a sentence *S* provided by the proposed graph-based model can be defined as the mapping $G: S \rightarrow$ *tuples of relations*, where *G* consists of a directed graph.

The proposed solution relies on graphs because they play an important role in many disciplines including biomedical domain, where graphs is found to be an invaluable tool to model the complex biological processes [Björne, 2009].

More formally, a graph *G* is a finite set of vertices *V(G)* connected by set of edges *E(G)*, defined as *G = {V(G), E(G)}*. If the edge connecting two vertices is directed, the graph is a directed graph; or an undirected graph, otherwise. The graph-based representation proposed here are all treated as directed graphs.

This model integrates a dependency grammar analysis that consists of generating the typed dependencies parses of sentences, which finally produces a *dependency graph* [De Marneffe and Manning, 2008]. This directed graph is the result of an all-path parsing algorithm based on a dependency grammar [Kruijff, 2002] in which the syntactic structure is expressed in terms of dependency relations between pairs of words, a *head* and a *modifier*. All derived dependencies of a sentence, define a dependency graph whose root is a word that does not depend on any word.

It was adopted the typed dependencies proposed in [De Marneffe and Manning, 2008], the *Stanford dependencies*. It should be noted that typed dependencies and phrases structures are different ways of representing the inner structure of sentences, in which a phrase structure (constituent) parsing represents the nesting of multi-word constituents, whereas a dependency parsing represents dependencies between individual words. In addition, a typed dependency graph labels dependencies with grammatical relations, such as *subject* or *direct object*. An example of a dependency graph is displayed in Fig. 4.6.

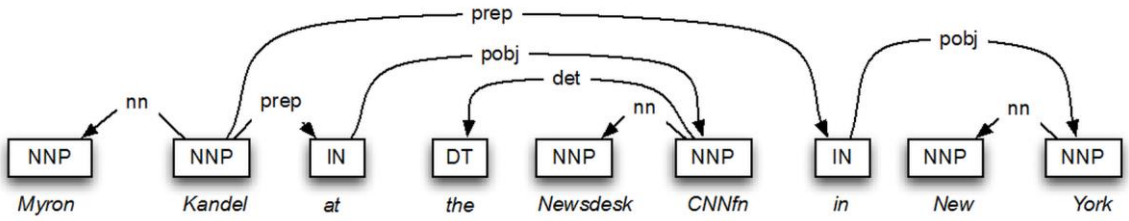


Figure 4.6. Dependency graph of the sentence "Mary Kandel at the Newsdesk CNNfn in New York"

Different variants of the Stanford typed dependency representation are available in the dependency parsing system provided with the Stanford parser[5]. The *collapsed tree* representation was chosen, in which dependencies involving prepositions, conjuncts, as well as information about the referent of relative clauses, are collapsed to get direct dependencies between content words. This representation was preferred because it has the advantage of reducing the number of typed dependencies in a given dependency graph by cutting down some highly frequent dependencies [De Marneffe and Manning, 20008]. Therefore, this collapsed tree representation simplifies the relation extraction process. The difference between these representations of dependency parsing analyses is illustrated in Fig. 4.7. The reader should notice that the second collapsed tree dependency graph (b) of the sentence required less nodes and edges elements than the uncollapsed dependency version (a) of the same sentence. Indeed, both "in" prepositions nodes in the first basic dependency graph do not participate in the second collapsed dependency graph. In fact, this information is captured by the novel "prep_in" dependency directly relating the two involving nodes ('Kandel' and "York'), removing the "in" token of the dependency graph (Fig. 4.7).

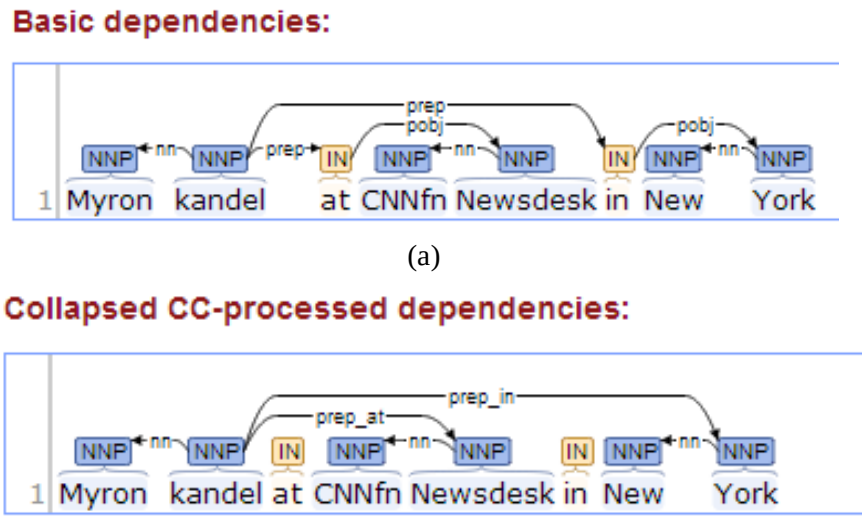


Figure 4.7. Comparison between two dependency graph types of representation

Additionally, the proposed graph-based model exploits *chunking analysis*, which is useful to define entity boundaries, and the head constituents of nominal, verbal and prepositional phrases. For example, consider the sentence "*Myron kandel at CNNfn Newsdesk in New York"*. Fig. 4.8 shows the head tokens of this sentence obtained after a chunking analysis. Usually, verb phrases are possible candidates for relations, and nominal ones, can represent an entity or a class instance.

[5] Stanford NLP Group. The Stanford Parser: A statistical Parser. http://nlp.stanford.edu/software/lex-parser.shtml

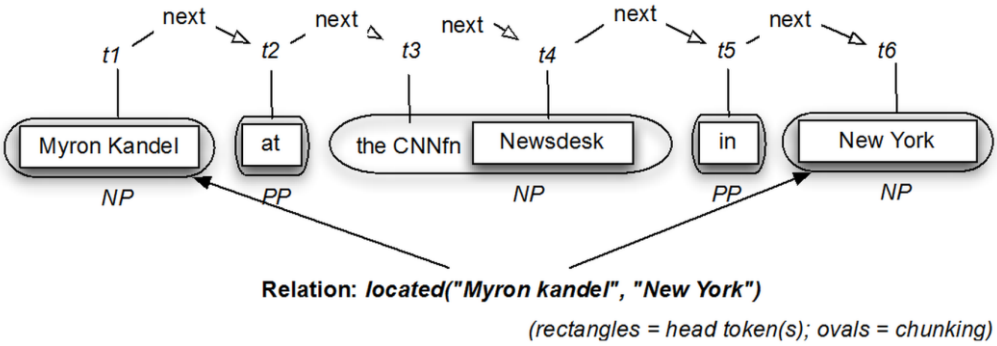


Fig. 4.8. Chunking analysis and head tokens of the sentence.

In this graph-based model, edges are considered as *relational features* that can be exploited in the automatic induction of symbolic extraction rules from sentences. Indeed, an ILP-based formulation for this IE problem was designed and shows how to cast this problem in it. In addition, the proposed approach is based on the premise that, when learning about properties of objects in relational domains, feature construction can be guided by the structure of individual objects [Raedt, 2010], which can be used for asserting relationships between two (or more) class instances in the same sentence.

Fig. 4.9 shows the final graph-based representation of a sentence obtained by integrating:

- a dependency analysis with *collapsed dependencies* (e.g. *prep_on*) according to the Stanford dependency parser;
- a *chunking analysis* (head tokens in bold);
- the *sequencing of tokens* in a sentence (*NextToken* edges);
- *morpho-syntactic, named entity, and semantic features* as nodes attributes (arrows in gray color).

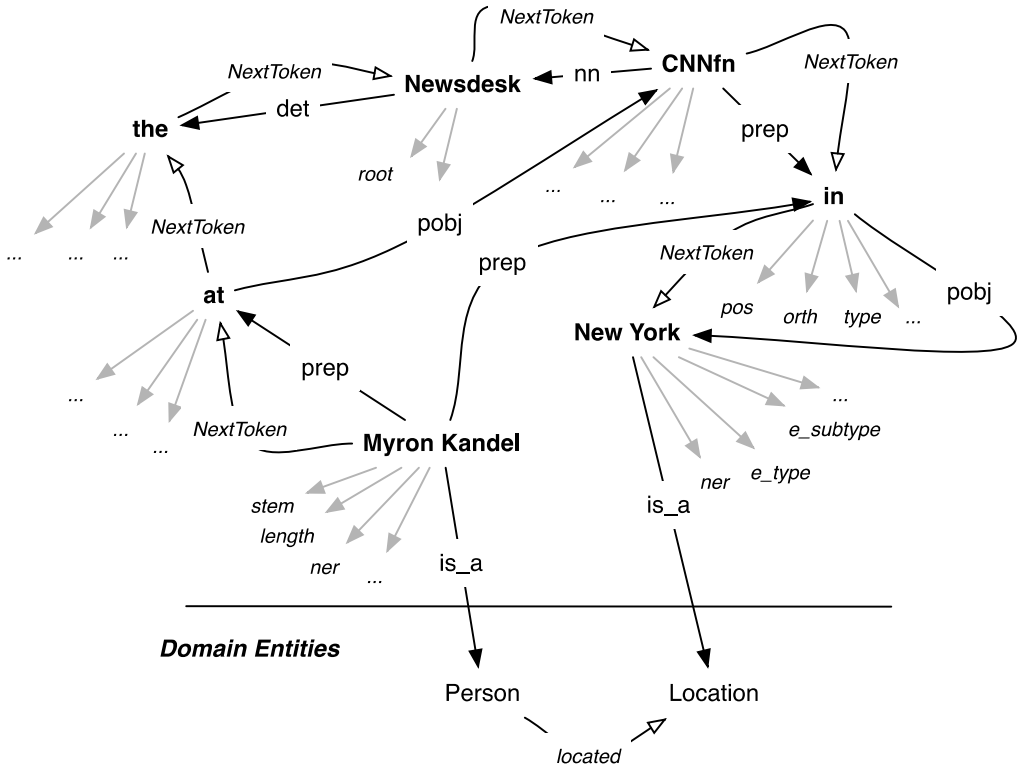


Figure. 4.9 The graph-based representation of the sentence: "*Myron Kandel at the Newsdesk CNNfn in New York*".

This graph-based representation can be expressed by a set of *binary relations* or *predicates*. For the same sentence (Fig. 4.9), here is a list of both unary and binary predicates extracted from the sentence:

- *root (Myron-Kandel)*
- *det (Newsdesk, the)*
- *nn (Newsdesk , CNNfn)*
- *prep_at (Myron-Kandel , Newsdesk)*
- *prep_in (Myron-Kandel , New-York)*
- *NextToken (the, Newsdesk)*
- *is_a(Myron-Kandel, Person)*
- *is_a(New-York, Location)*

Moreover, examples of entities, or instances of classes are simply represented by an is-a relationship, i.e., is-a(token, class), where "token" denotes a head token of a nominal chunk, while "class" is the semantic class assigned to the "token". Other possibility consists in a multi-word noun that denotes an entity. In this case, the multi-word noun is derived by concatenating its individual elements by an "-", e.g., "New-York", being regarded as a semantic unit.

Finally, directed relationships where the order between the arguments matters, are represented in its normal order of arguments, e.g., *located(Myron, New-York)*, indicating that "Myron" is located in "New York", considering this order between the arguments. Undirected relationships, like brother_of (Paul, John) are represented with an additional commutative property.

The graph-based model represents a collection of binary relations, and their arguments can be enriched with additional constraints on the types of the arguments. These additional binary relations are used by the ILP-based induction learning component responsible to link terms in a sentence with classes and relations from a domain ontology. For example, if the predicate to be learned is *read (X, Y)*, or in ontological terms, the object property *read(X, Y)*, then the first argument *X* should be an instance of the *Person* class, whereas the second one *Y* should be an instance of the *Publication* class in the domain ontology. Therefore, instances of classes and relations can be viewed, respectively, as nodes and edges in the model. Each node can have many attributes, e.g., the ontology class label which it belongs to.

In the present work, the task of identifying the labels of candidate classes and relations instances is defined as the *target predicate* in the learning problem formulation. More concretely, such target predicates are learned as a combination of several sentence elements given by the graph-based model described above.

Most previous work on entity and relation extraction has only considered attribute-value features, or propositional features derived from input text data [Finn, 2006] [Giuliano, 2007] [Roth and Yih, 2007] [Kambhatla, 2004] [Zhou *et al.*, 2005]. Instead, the proposed solution relies on a first-order logic representation of examples which provides a richer representation formalism, allowing the classification of objects whose structure is relevant to the classification task [Fürnkranz *et al*., 2012].

4.4.3. Discussion and Related Work

The proposed graph-based model can be viewed as a labelled graph representation of sentences and candidate examples, such as named entities and relations between two entities.

Since this graph is defined by syntactic structures defined by NLP annotations, it is domain-independent, i.e., there is no need to redesign it when domain changes. Moreover, this

model has the potential to fit most tasks in NLP, as it can be easily extended by specifying additional relational features.

The overwhelming majority of proposed approaches for RE, including the one proposed in this thesis, have been focusing on the extraction of simple binary interactions between named entity pairs. However, some recently published corpora provide complex, and typed event annotations that aim to accurately capture *n-ary* relations. Such n-ary relations, called events, involve more than two participants. An example of such type of relation appears in the sentence "proteins *A*, *B*, and *C* form a complex", where A, B, and C are the possible arguments of an ternary relation.

Nevertheless, differently from most related work that uses a tabular representation of examples, the graph-based model proposed in this thesis can be easily extendible to deal with n-ary relationships.

In this light, the proposed graph-based model corresponds to a semantic representation of an event structure, with nodes representing named entities and events, and edges corresponding to event arguments [Björne *et al*., 2010].

4.5. Sentence Simplification: A Graph-based Method

As stated earlier in this chapter, RE aims at finding predefined relationships between target entities in a text. It has lately drawn the attention of the community of information extraction as offering potential solutions to a number of problems in that area. The typical target entities in RE consists of real world named objects, or named entities, such as people, organizations, geographical locations, among others. In the domain of molecular biology, for instance, the focus has largely been on the investigation of protein-protein interactions (PPI) [Airola *et al*., 2008] [Buyko *et al*., 2011] [Fundel *et al*., 2007]. For that purpose, biomedical NLP communities have made available several annotated corpora on PPI.

Most of the state-of-the-art RE approaches proposed so far on news [Jiang, 2007][Zhou and Zhang, 2007] [Choi *et al*., 2013] and biomedical domain [Airola *et al*., 2008] [Buyko *et al*., 2011] has relied on the exploitation of the full constituent and dependency parsing trees [De Marnerffe and Manning, 2008] without any form of simplification or filtering. Thus, the syntactic structures, as they directly result from natural language parsing tools, may not always be adequate for relation extraction [Buyko *et al*., 2011].

The path connecting a pair of entities in a parsed sentence has been extensively used for constructing feature vectors or kernel functions to identify relations [Airola *et al*., 2008] [Jiang, 2007] [Zhou and Zhang, 2007]. However, some problems were reported: the tree-like structures derived from parsed sentences usually contain unnecessary sub-paths that, although quite useful, may also have misleading information [Jonnalagadda and Gonzalez, 2009].

This thesis hypothesizes that filtering and simplification operations pruning non-essential nodes and edges of a graph-based model for sentence representation can improve overall extraction results in RE.

Earlier in this chapter, the proposed OBIE method OntoILPER that induces symbolic extraction rules suitable to identify semantic relations between entities was introduced. OntoILPER is grounded on a graph-based model of sentences as a hypothesis space for generating candidate extraction rules.

In this context, a method to simplify graph-based representations of sentences is proposed. This method replaces dependency graphs of sentences by simpler ones, keeping the target entities in it. The key idea is to speed up the learning phase of the proposed RE framework, by applying several rules for graph simplification that constrain the hypothesis space for generating extraction rules.

Moreover, this thesis investigates the effects of the simplified graph-based representations on relation extraction performance. In particular, the effect of such simplification operations as a means to alleviate overfitting in the extraction rules is investigated. As a "proof-of-concept", the relation extraction task on protein–protein interactions was chosen to validate the proposed method to simplify the graph-based representations of sentences.

Other contributions related to this trimming process consist of:

- the proposal of several rules for syntactic and non-syntactic transformations of a dependency-based graph model for sentence representation;
- an intrinsic evaluation of the proposed rules showing promising results on trimming graph-based representations of sentences (Section 6.5.1);
- a further assessment of the effectiveness of the simplification rules on the performance of relation extraction tasks (Section 6.5.2)
- the development of a tool for visualization of graph-based representations of sentences.

In the remainder of this section, the problem of sentence representation for IE is first described and motivated. Then, the proposed method for graph-based sentence simplification is presented. This section finally discusses related work on sentence simplification.

4.5.1. Transforming Graph-based Representations of Sentences

Syntactic parsing, as performed by state-of-the-art NLP tools, commonly constitute one of the first stages (preprocessing) in information extraction systems. However, such tools often neglect the characteristic complexity of long sentences found in biomedical literature. As a result, the syntactic structures as they come directly from such parsers may not always be suitable for relation extraction, mainly for two reasons: they contain many irrelevant lexical node; and, the distracting structural noisy information, as it might occur in the original dependency graphs, may cause overfitting during the classifier learning phase. It is well known that overfitting prevents classifiers of finding more general extraction patterns [Buyko *et al*., 2011].

To alleviate this problem, a set of simplification rules to be applied on the sentences represented by the graph-based model introduced in Section 3 is proposed. Such simplification rules attempt to reduce the complexity of the dependency graph-based component employed in OntoILPER, by eliminating both spurious nodes and relations.

Therefore, the ultimate goal here is reducing and rearranging the final representation of sentences in OntoILPER in order to avoid overfitting of the relational feature space caused by overly specific syntactic and lexical information. One may argue that all proposed trimming/ transformation operations are well motivated by linguistic aspects, and they should guarantee minimal information loss. On the other hand, one assumes that the meaning of the target sentences itself is less important than keeping the truth-value of the relations, i.e., whether or not there exists a semantic relation between two entities.

In what follows, the proposed transformation and simplification rules are described in more detail.

4.5.2. Transformation and Simplification Rules

The graph-based model used in OntoILPER is used here as the hypothesis space for generating symbolic extraction rules. The proposed simplification rules introduced in this section, only concern the dependency graph and the chunk sequencing, both constituting the structural predicates in the hypothesis space here.

The term "simplification" must be understood as something that is closer to the *canonical form* of sentences. Therefore, given an input sentence, the aim is to produce a shorter sentence which contains as much information as possible from the original one.

Since the primary objective here is to perform relation extraction on the canonical sentences, the adopted approach to sentence simplification is rather *entity-oriented*, in the sense that it seeks to preserve the minimal relevant contextual information around entities in a dependency graph. This favors the discovery of more general patterns (extraction rules), as they proved to be very useful in improving the overall performance in all biomedical datasets tested here. Furthermore, not only the resulting simplified graphs further constrain the original hypothesis space, but also impacts in less learning time required.

In the proposed simplification approach, a rule transforms a dependency graph (or dependency tree) of a sentence in the sense that some nodes and its outgoing edges, if there are any, can be completely removed from the graph. Another trimming operation on graphs targets the incoming and/or outgoing edges of a given token, which can be reassigned to another different node, or nodes in the same graph.

All rules were defined after a careful study of typical grammatical types of sentences or phrases, and how they are represented in a dependency graph output by the Stanford parser. The rules have the form: $R_i : \{C_i\} \rightarrow \{A_i\}$, where C_i denotes the *conditional part* which is mainly defined by constraints on nodes POS tags, type of outgoing/incoming edges, parents of nodes, etc; and A_i is just a series of simple actions that are applied on the matched nodes. For instance, most of the rules play the role of *filters* that, given a dependency relation between two nodes e.g., *rel(A, B)*, it removes the *B* node and the corresponding relation *rel*, leaving the *A* node unaltered.

One should emphasize that any node not matched by the condition part of the rules remains as they were in the original dependency graph, i.e., no transformation is performed on them. For convenience, Tab. 4.1 provides the meaning of the typed dependencies, according to [14], mentioned in the rule descriptions to follow.

Table 4.1. Typed dependency acronyms with related words in italics.

Dependency	Meaning/Example.
det	a noun determiner: "*the* boys..."
aux	modal or auxiliary verb: "it *should* appear now."
auxpass	a non-main verb of the clause which contains the passive information: "He has done the job: Kennedy has *been* killed".
{a|adv|part|t}mod	a noun modifier, such as adverbs, adjective, quantified modifier: "I do not eat *red* meat".
predet	it modifies the meaning of a noun determiner: "*All* the boys are here".
mark	a subordinate conjunction such as "that", "which": "This is the man *that* I mentioned".
ccomp	clausal complement: *I am certain that* he did it.
rcmod	relative clause modifier: "I saw the *man* you *love*".

Two kinds of rules are addressed: (i) *clause-level* rules, and (ii) *entity-level* rules. In the following, each rule with some illustrated examples[6] is described.

[6] The corpora are assumed in the English language.

Clause-Level Rules

This kind of rule deals with *compound* or more complex sentences, where it is assumed that one of the clauses (main clause) have both entities participating in a relation. Usually the main clause contains the main verb of the sentence, or the root of the dependency graph. The subordinate clauses may contain non-crucial information, and thereby, may be discarded. In fact, before removing a clause, including initial and final adverbial phrases, one checks for the existence of important entities in it. In the case of finding such entities, the option made here was not to exclude them from the graph.

The intuition behind the proposed rules is to try to identify independent clauses from complex sentences. Thus, in order to properly apply the clause-level rules, one needs to first identify whether the sentence is simple, i.e., with just one main verb, or compound. A compound sentence may have or not a relative clause. Thus, to be classified as simple, there should be only one verb sentence and no clausal complement dependency from the set {**comp, csubj, csubjpass, rcmod*}. Otherwise, the sentence is complex, or compound. The test whether a compound sentence has the *rcmod* dependency characterizes a relative or subordinate clause.

R1. Removal of Non-Informative Clauses

There are three possible positions where subordinate clauses can be embedded in a compound sentence: start, middle, and end. This rule checks for the existence of non-informative clauses in any of these positions. This rule for sentence simplification selects the more important of the two compound clauses in a compound sentence.

A first look at Fig. 4.10 reveals that the most informative clause is the second one: “the costs will rise”. Thus, in order to remove the first sentence, the following rule (pseudo code) removes the first non-informative clause. In the following pseudo-code examples, *tkA* and *tkB* denotes two distinct tokens in the same sentence.

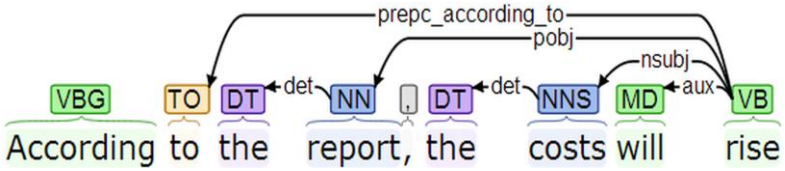


Figure 4.10. Example of an adverbial clause in a sentence.

The pseudo code is given below:

```
for each (token tkA of the sentence) do
   if (tkA contains a dep in {prepc_*| dep} with a token tkB ) then
      if ((tkA has POS in{VB*}) and (tkB has POS in
            {DT|JJ|IN|TO}))
      then
            remove {prep_*(tkA, tkB) | dep(tkA, tkB)}  from the graph
      end if
   end if
end for
```

R2. Removal of Attribution Clauses

This rule is in charge of swapping the role of the main verb, i.e., the root element of the dependency graph in a compound sentence between clauses, as illustrated in Fig. 4.11. In this example, the verb of the attribution clause is the root of the dependency graph, but the more

relevant clause is the other one. Thus, one should swap the roles between the two verbs, accordingly.

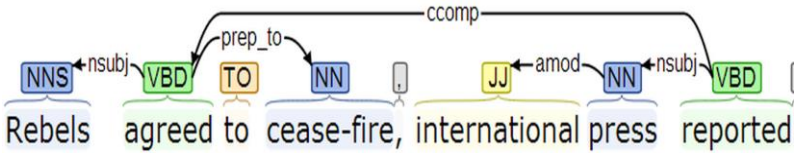


Figure 4.11**.** Dependency graph of a compound sentence with the attribution clause "international press reported".

The rule that performs the removes attribution clauses is given below.

```
if (tkA is root) and
  ( tkA POS is in {JJ | VB*} and (tkA dep is not in {dobj|pobj})   and
    ( tkA has the dep {ccomp(tkA, tkB)})  and  (tkB POS is in{VB*} )
  )  then
       tkB is the new root of the sentence S
       remove {ccomp(tkA, tkB)} from the graph
 end if
```

R3. Subject-Verb-Object Rearrangement

Particularly in biomedical texts, complex coordinated syntactic structures that link together two or more conjuncts of the same type abound. Such structures pose several problems to dependency parsers. Fig. 4.12 displays an example where two coordinated verbs (with the *CC* coordinating conjuncts) share the same object. However, just the first verb has a direct link to its object CC via its *dobj* dependency relation. This may also difficult the generation of extraction rules in the proposed framework for relation extraction in the sense that only the first verb has the (*subject-verb-object*) path as a typical pattern used in extraction. Accordingly, this rule looks for such configuration, by checking if chunk sequences (verbal and nominal) appear before and after the conjunction that link similar types of words. The task here consists in propagating the *obj* dependency relation of the first verb by creating a new *obj* dependency for the second one, pointing to the same target object (see Fig. 4.13). Indeed, this rule is generalized by taking into account up to 3 verbs with 3 respective objects in the same sentence. One must be aware that, although this rule does not actually simplify the graph in terms of removing nodes as it was done in other rules presented so far, it was often used in the biomedical corpus tested here.

Tab. 4.2 provides additional sentences matching the clause-level rules described above.

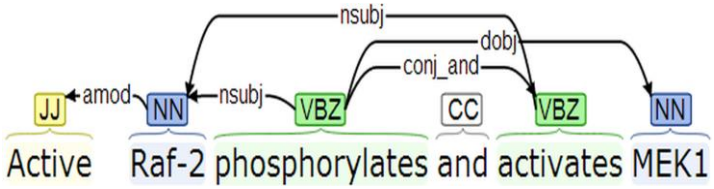


Figure 4.12. Sentence with a coordinating conjunct between verbs.

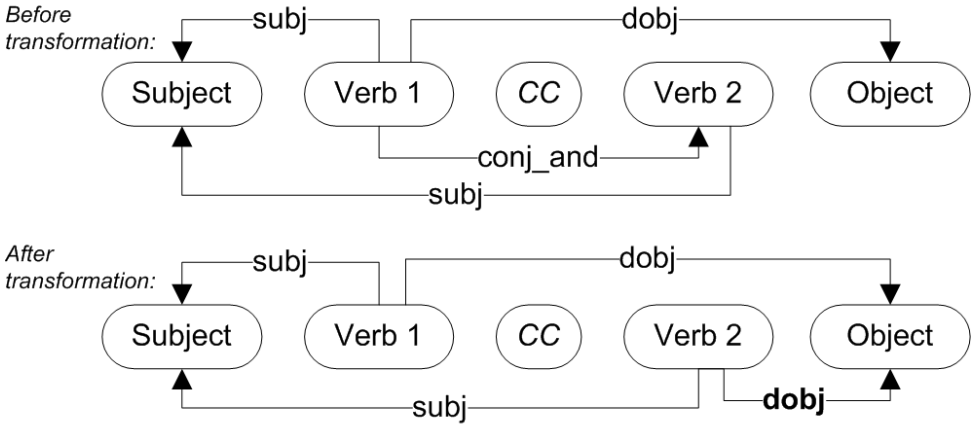


Figure 4.13. *verb-dobj* missing dependency before transformation and its inclusion after transformation (in bold).

Table 4.2**.** Examples of simplified sentences by clause-level rules. Removed tokens are underlined.

Rule	Original/Simplified Sentences
R1	As a matter of fact, John came into the room while you were talking about him.
R2	On the assumption that mothers stay home with children.
R2	Rebels agreed to talk with government officials, international observers said Tuesday.
R2	It was suggested that Yak1 phosphorylates Crf1 to promote its nuclear entry.

Entity-Level Rules

This kind of rule makes a small change in the graph by acting on one or more nodes of the graphs.

R4. Replacement of Protein/Genes Names

Single terms or nouns in a sentence are stemmed, i.e., its *lemma* or *root* is determined. Thus, plurals and verb tenses are reduced to singular and infinitive form, respectively. Although this rule does not reduce the sentence in number of tokens, it helps generalization over specific protein/genes names.

R5. Treatment of Multi-word Entities

This rule deals with the *nn* dependencies or *noun compound modifiers* of another noun node. For instance, the dependency graph in Fig. 4.14 shows two multi-words terms, or noun chunks: “Nevada Corporation” and “United States” . There are two *nn* dependencies (labeled edges) with the same governor term “Corporation”, which is also identified as the head noun

of the noun chunk. Thus, this rule checks for head tokens with *nn* edges outgoing from them, and concatenate these words in a multi-word term by a hyphen, but keeping a reference to its head noun.

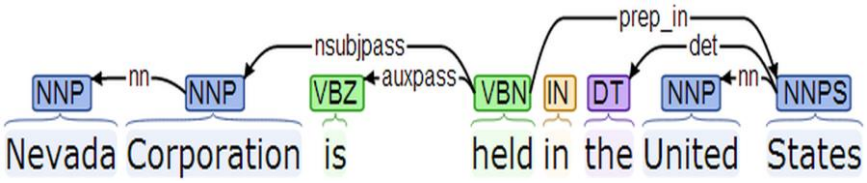


Figure 4.14. Parsed sentence with two *nn* dependencies.

R6. Removal of Distracting Dependencies

Aiming at pruning syntactic determiners, auxiliaries, modals, and all the tokens other than the head noun, this rule removes from a given sentence, any of the dependency relations belonging to the set: {*det, aux, auxpass, amod, predet, advmod, partmod, tmod, mark*}. However, before removing such dependencies, one needs to verify if the node candidate to deletion is a leaf in the graph. Examples of sentences containing such types of dependencies are shown in Tab. 4.3.

Table 4.3. Examples of sentences simplified by entity-level rules. Removed or modified tokens are underlined.

Rule	**Original/Simplified Sentences**
R2+R6	I think he´s been in Washington too long.
R6	However, cytokines, in particular IL-2 and IL4, ...
R5	Nevada-Corporation is held in the United-States.
R5+R6	Mutations in CBP have recently been identified in RTS-patients.
R6	You are the girl that I am looking for.

A Minimal Example

Fig. 4.15 presents the reduced dependency graph of the sentence introduced in Section 5.3. It should be highlighted that this graph minimally retains the core information for the relation *located_in("he", "Washington").*

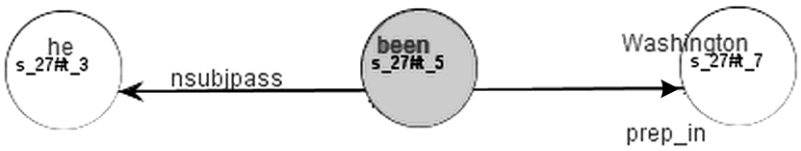


Figure 4.15. Reduced dependency graph of the sentence: "I think he´s been in Washington too long" in the visual graph reduction and representation tool developed.

Rule Application Order. The application order of rules in the approach proposed must be considered with special attention due to the "loss of information" that is achieved after each individual rule has taken place. Thus, in order to avoid misleading results, the rules are applied in the following order: *R4*, *R5*, *R1*, *R2*, *R3*, and *R6*.

Application Scenarios. It worth noting that, although the developed prototype has been developed with the primary purpose of simplifying graph-based representations of sentences

in this thesis, it is argued that it can be employed to other text mining problems, like automatic text summarization, for instance.

Another usage scenario concerns the exploratory corpus analysis and interactive text mining. For example, by design graph-based patterns in the representation model, a user can answer questions like "what are the entities in this particular sentence?", “what are the verbs, subject and objects of this sentence?”, etc.

A final suggestion concerns the task of automatically extract keyword lists for the creation of further patterns.

4.5.3. Related Work and Discussion

Automatic simplification of sentences was first proposed for improving the performance of parsing tools [Chandrasekar and Srinivas, 1997]. Later on, researchers have found other applications for sentence simplification. One of them consists in creating sentences that are shorter, grammatically correct, and information preserving to help people with reading problems [Carroll *et al.*, 1998]. Another application of the technique is related to the automatic text summarization systems [Zajic *et al.*, 2007]. The focus in that work was to preserve only the important content in the final summaries.

Reference [Jonnalagadda and Gonzalez, 2009] presents an application for sentence simplification closer related to the one proposed in this thesis, since the authors in [9] also describes a sentence simplification method, called “BioSimplify”, which was evaluated on relation extraction. The main goal of their method is to improve the performance of biomedical extraction systems by reducing the complexity of sentences that could be hiding protein-protein interactions (relations). Their method can also remove noun phrases that are important for a given relation of interest.

The method presented in this thesis differs from that attempt to remove all information outside the target verb and arguments as presented in reference [Vickrey and Koller, 2008]. It also differs from the method proposed in [Jonnalagadda and Gonzalez, 2009] in which the authors attempted to keep all information in a sentence aiming at improving a parser. The approach presented here generates canonical graphs representing sentences.

4.6. Incorporating Linguistic and Ontological BK into OntoILPER

In this section, the problem of integrating formally represented knowledge into the learning process is addressed. It aims at improving the OntoILPER results in terms of portability and accuracy. For that, it is investigated how semantic structures, like ontologies, can enable more accurate IE.

Under such outlook, Yildiz and Miksch (2007) argue that the combined use of ontologies and IE presents several advantages, which includes:

- ontologies enable the development of domain-independent IE systems, increasing system portability.
- ontologies allow IE systems to have a suitable model for the domain of concern and the extracted facts are represented in the form of ontologies.
- it is possible to obtain a clear separation of domain knowledge and the operational knowledge of the system.

Another reason to integrate ontologies into OntoILPER is that ontologies not only can capture knowledge about a domain of interest, but they can also be used in applications that need to process information content. Moreover, ontologies provide mechanisms to reason about it, instead of just presenting information.

Many OBIE systems proposed so far [Castano *et al*., 2008] [Buitelaar *et al*., 2008], [Maedche, 2002] make use of a single ontology for a particular domain. However, according to [Wimalasuriya and Dou, 2010], there are any rules that prevent the use of multiple ontologies for IE. The authors show that, using multiple ontologies, one can improve the IE process because multiple ontologies provide different perspectives on a domain. Thus, by integrating multiples ontologies, an OBIE system has the potential of providing accurate answers to queries related to different users' perspectives.

Contrary to the current trend of using a single ontology, OntoILPER represents additional background knowledge by means of a second *domain-independent* and *linguistic-oriented annotation* ontology, hereafter *annotation ontology*, that formally describes the rich annotation set of features derived from several NLP tools. The annotation ontology manifests itself as a first-class component that provides a higher level of expressivity by means of a richer linguistic knowledge model. This feature distinguishes the OntoILPER from many other state-of-the-art OBIE methods.

The working hypothesis is that the ontological elements defined by both ontologies (domain and annotation ontology), working in synergy, may contribute to OntoILPER, enabling an enhanced flexibility and adaptiveness to other domains.

More concretely, during learning, OntoILPER takes advantage of TBox definitions and ABox assertions of the above mentioned ontologies as background knowledge for its ILP-based machine learning component that builds the classification models (see Fig. 4.2).

In the following, the roles that these two ontologies play in OntoILPER are detailed, showing illustrative examples that highlight the advantage that these ontologies can bring to IE.

4.6.1. The Role of the Domain Ontology in OntoILPER

This section presents the role played by the domain ontology in the two operational modes in OntoILPER: *training* and *application*.

Training Mode. After the Text Preprocessing and the Sentence Representation and Simplification stages (see Fig. 4.2), the graph-based sentence representation are passed as input, along with the elements of both ontologies (domain and annotation ontology), to the BK Generation stage. They formally describe the domain and background knowledge exploited by OntoILPER.

With respect to the domain ontology, in particular, this ontology guides the BK generation process by defining the level of abstraction (classes and super classes) of the BK predicates from which the rules will be induced. Therefore, TBox elements of the domain ontology (class and property labels, data/object properties, taxonomical relationships, and domain/range of non-taxonomical relations) are taken into account during the BK Generation stage mentioned above.

The above integration of domain ontologies into OntoILPER is in accordance with the first three levels of ontological knowledge used by few the state-of-the-art OBIE systems, as discussed in [Karkeletis, 2011]:

- At the first level, the ontological resources explored by OntoILPER encompass the domain entities (e.g., person, location) and their synonyms or co-referents. These resources are mainly used in OntoILPER for entity classification;
- At the second level, the main semantic resources used in OntoILPER consist in the domain entities organized in conceptual hierarchies, which can be exploited by the IE process for generalizing/specializing extraction rules;

- At the third level, OntoILPER exploit concepts' properties and/or relations between concepts of the ontology. Moreover, extraction rules are acquired from corpora that have been previously annotated according to the domain ontology.

In addition, the domain ontology can be viewed as a rich and structured extraction template for the IE process.

Application Mode. The overall IE process of OntoILPER aims at mapping pieces of textual information to the domain ontology. OntoILPER selects and interprets relevant pieces of the input text in terms of their corresponding classes in the domain ontology. Fig. 4.16 illustrates this process.

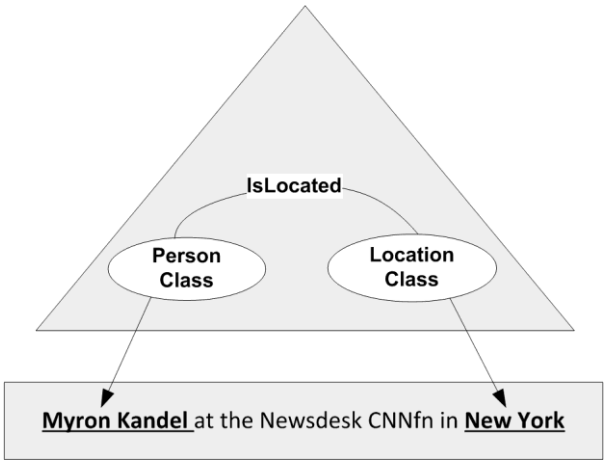


Figure 4.16. Example of semantic mapping or semantic annotation.

This mapping can be formalised into the text annotation by the ontology (Fig. 4.16). Here, text fragments are labelled with ontological concepts and relations, according to the IE task of interest, either NER or RE. Thus, the ontological concepts (*Person* and *Location*) and the relation (*isLocated*) are linked to their corresponding semantic units, i.e., "*Myron*" and "*New York*" for the concepts), and "*has been*" as a relation. In other words, the ontological TBox elements denoting the *Person* and *Location* classes are instantiated by the object property *isLocated ("Myron", "New York")* in the domain ontology.

Thus, in application mode, the domain ontology can be seen as a repository for the extracted instances, at first place[7]. Furthermore, the user has the potential benefit of having the extracted instances populated into the domain ontology that, in conjunction with a reasoner, provides explanations about the classified instances.

4.6.2. The Role of the Annotation Ontology In OntoILPER

At the semantic level, it is claimed that any text representation should be supported by ontologies as a formal means for representing domain knowledge. Actually, as stated in [Spasic *et al.*, 2005], the correct interpretation of the semantic content in natural language should require linguistic knowledge and some degree of general knowledge.

In OntoILPER, ontologies are regarded as conceptual models that provide the necessary framework for semantic representation of textual information. Moreover, OntoILPER subscribes to the central idea that the main link between text and ontology is *terminology*, which maps terms to domain specific concepts or classes [Spasic *et al.*, 2005].

Accordingly, OntoILPER semantically annotates texts and link them to explicit semantic layers supported by ontologies, which offers a higher expressive power, because they are able

[7] Besides checking redundancy, OntoILPER performs no further inference task at this point.

to support automatic semantic interpretation [Spasic *et al*., 2005]. Figure 4.17 illustrates this key point.

On the one hand, OntoILPER relies on ontologies as a mechanism to provide a platform for semantic interpretation of textual information, i.e., the task of linking domain-specific terms or words to their correct classes in a given ontology. Putting it differently, OntoILPER is based on an explicit semantic layer, composed by a domain and a task ontologies, which allows IE to be interpretable from the domain concepts perspective. Furthermore, the extracted information from textual sources can be used for updating the content of the domain ontology, i.e., populating it.

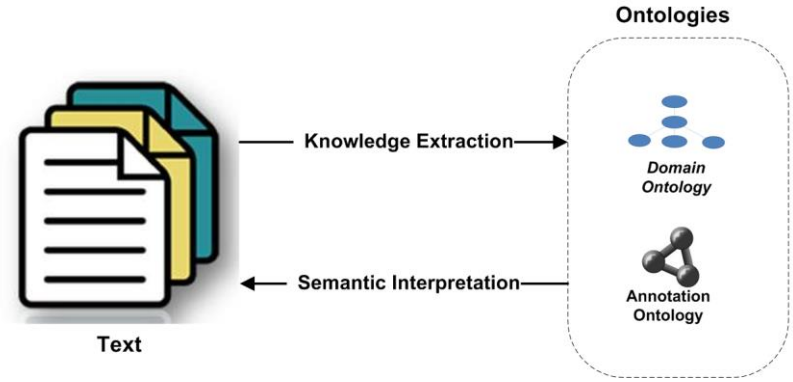


Figure 4.17. Complementary tasks in semantic annotation of texts.

As a result, the overall IE is able to extract more relevant and precise information, since the ontologies provide a mechanism to achieve a better contextualization for the interpretation of the results. This contrasts, for instance, with the methods based on linguistic patterns that take into account a local and limited context of words [Nedellec and Nazarenko, 2006].

Accordingly, this thesis proposes a domain-independent and expressive ontology (in OWL/DL), the *annotation ontology*. It formalizes the annotations produced by various NLP analyses through the definition of several ontological elements (classes, object and data properties) that are able to represent, with the enhanced expressiveness, the BK employed by the symbolic rule learning component in OntoILPER.

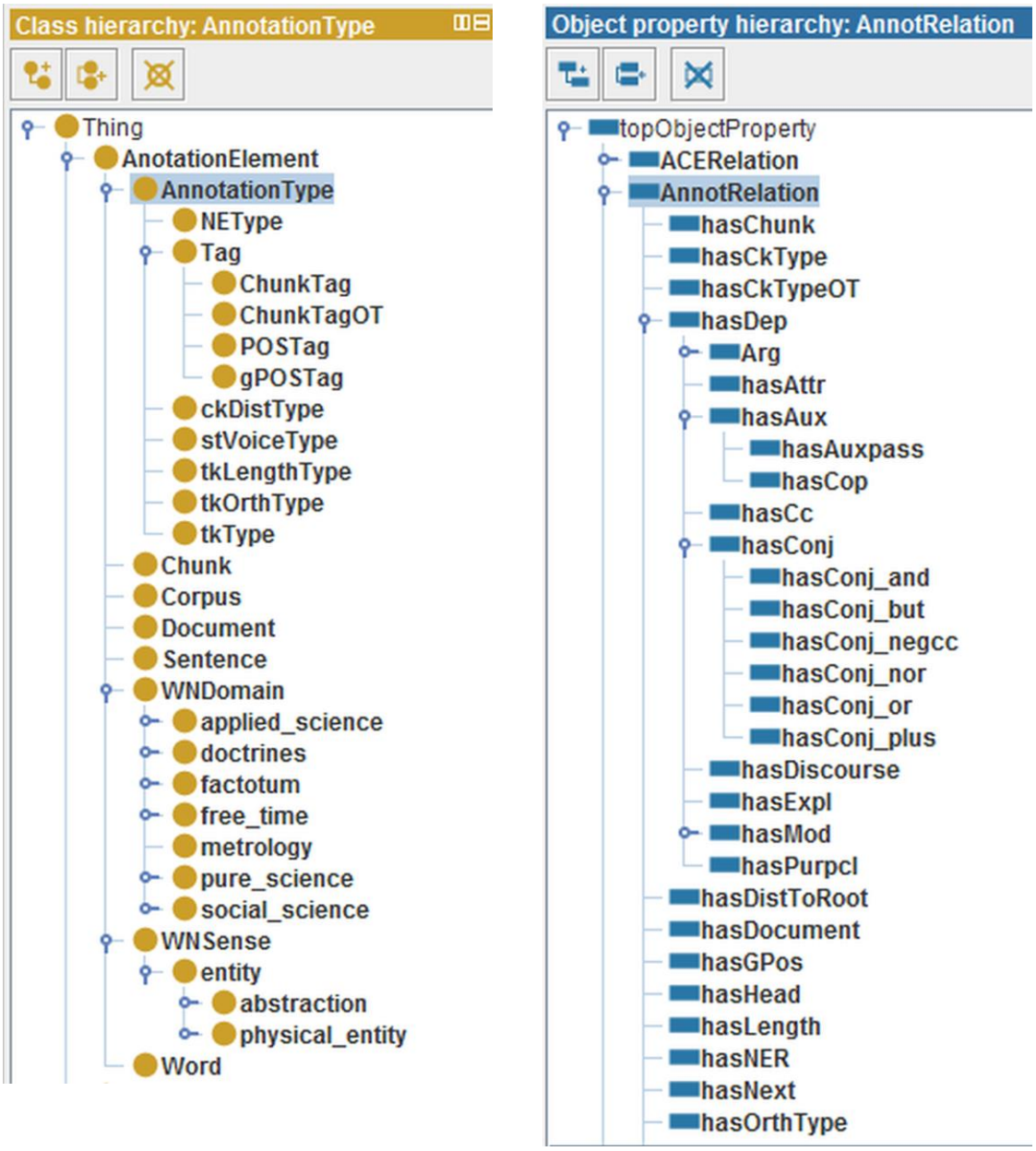
Annotation Ontology Elements. Fig. 4.18 shows the class and object property hierarchies respectively, while Fig. 4.19 displays the datatype properties of the Annotation Ontology.

In the following, further explanations about the role played by the Annotation Ontology are provided. Such roles are defined according to the two operational modes in OntoILPER: *training* and *application*.

Training Mode. The Annotation Ontology mirrors the graph-based model of sentence representation (Section 4.4), consisting of an "antologized" version of the graph-based model.

After the execution of the Text Preprocessing and the Sentence Representation and Simplification stages (see Fig. 4.2), the annotation ontology provides the following elements to the BK Generation stage which are exploited as further BK by the next stage, the Extraction Rule Learning.

- the labels of classes, object and data properties that constitute the ontologized version of the graph-based sentence representation;
- all the linguistic entity types present in the annotated corpus, such as typed dependencies and lexicon hierarchies, chunk types, POS tags, NER types, etc.



Part (a) **Part (b)**

Figure 4.18. Class and Data property hierarchy of the Annotation Ontology.

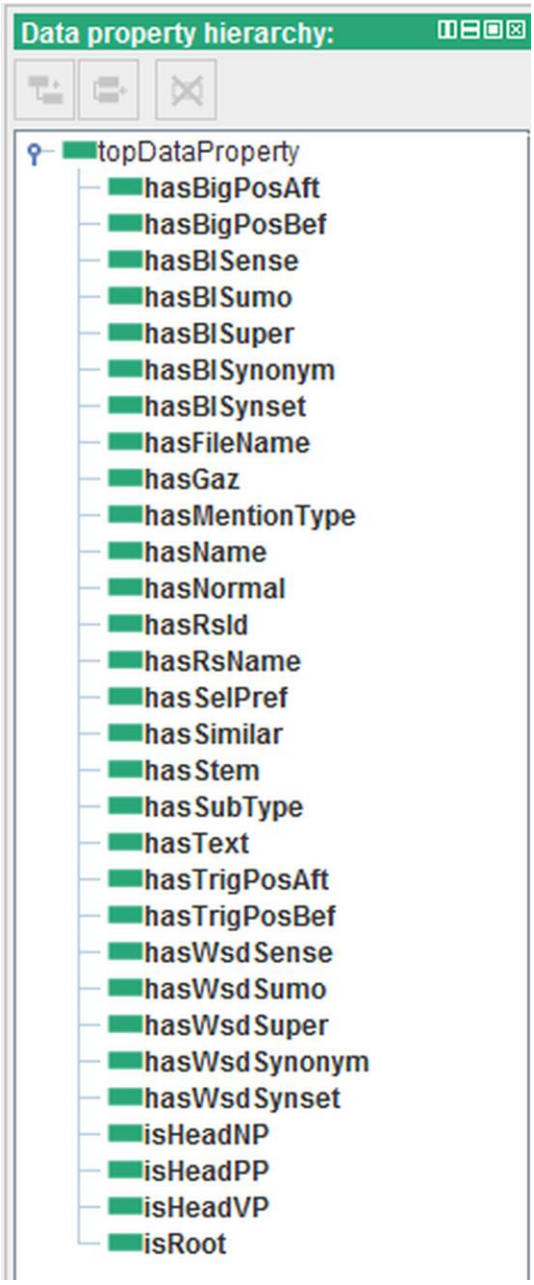


Figure 4.19. Data property schema of the annotation ontology.

- flexibility in defining new linguistic elements as additional relational features in the graph-based sentence model. For instance, the Annotation Ontology could be extended with a new *object property* relating two token instances for representing the coreference[8] between them in a document.
- the possibility for the ontology engineer or expert to choose the level of abstraction of classes and relations as well as the level of linguistic analysis employed (morphological, syntactic parsing, semantic role labelling, selectional preferences, etc). In other words, the user has the option of selecting a subset of the features to be used in the Extraction Rule Learning stage.

8 Entity coreference are found by the Coreference Resolution NLP substask that tries to find all expressions that refer to the same entity in a text.

With all these elements at hand, the BK Generation stage is able to map them to Prolog predicates that form the factual knowledge base used for learning by the Extraction Rule Learning stage.

Application Mode.

The annotation ontology, as a task and lightweight ontology, reflects the annotated sentences represented as graphs after the Text Preprocessing phase.

Then, the Graph Conversion stage converts the graphs to assertions that are inserted into the annotation ontology in the OntoILPER application mode.

There exists an one-to-one mapping of every element represented by the simplified graphs to their corresponding ontological elements in the annotation ontology. In short, nodes denoting documents, sentences, chunks, and tokens in the graph, are represented as instances of the *Document*, *Sentence*, *Chunk*, and *Token* classes in the annotation ontology, respectively. The same correspondence exists for the graphs' edges denoting syntactic relationships or dependencies, structural relationships, etc. The nodes attributes are represented as data properties in the annotation ontology.

To sum up, one can say that the annotation ontology is an "ontologized" version of the simplified graph-based representation of sentences.

By transforming graphs representing sentences to ontological instances, one can enjoy the following advantages:

- *semantic type disambiguation:* terms are instantiated taking into account its disambiguated sense in the annotation ontology. In fact, the WSD task is performed in the Text Preprocessing stage, but the annotation only is in charge of appropriately representing it as an instance of the correct class.
- *treatment of synonyms*: mapping of words on concepts also solves the synonymy problem.
- *term normalization*: including hypernyms (super) concepts allows for relating very similar terms into a unique normalized representation.
- *term generalization:* The abstraction degree of the induced rules in OntoILPER strictly depends on the representation of the training examples. Several semantic features and mapping to semantic recourses like WordNet, WordNet Domains and the SUMO top ontology enable that more abstract features are incorporated into the annotation ontology as instances of classes in the sense/domain hierarchies present in the Annotation Ontology. Hence, the integration of super-concepts provides also a very good basis for feature generalization and, consequently, rule generalization as well.
  This generalization aspect in OntoILPER is illustrated by the following figure (Fig. 4.20):

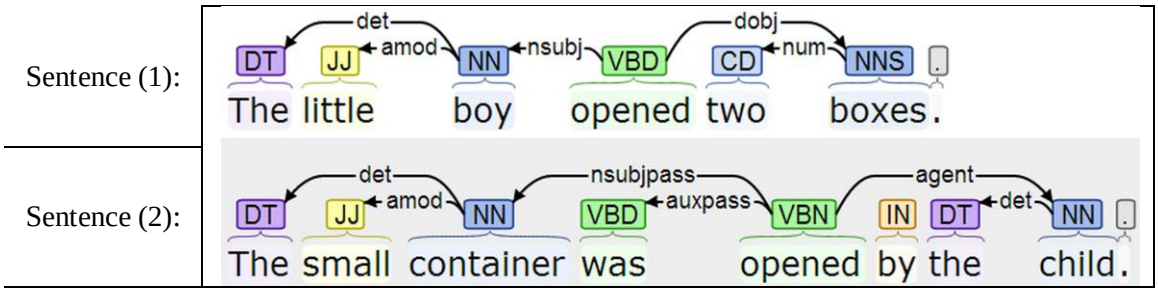


Figure 4.20. Examples of syntactic dependencies (annotations) and their generalizations.

The two sentences displayed in Fig. 4.20 contain specialized/generalized terms according to the WordNet hypernym hierarchy, i.e., the subject in Sentence 1 is “the little boy”, “the

child" is the subject of the second sentence. In addition, notice that the second sentence is in passive voice, while the first one, in active voice.

Both sentences practically convey the same meaning, but they are annotated with different elements. Only considering the main dependency relations between the verb and its subject and object, the two sentences above can be annotated as follows:

– Sentence (1) : *nsubj* (opened, boy), *dobj* (opened, boxes)

– Sentence (2): *nsubjpass* (opened, container) *agent* (opened, child)

Thus, in Sentence (1), the subject-verb dependency relation was annotated as *nsubj* (noun subject), whereas the same dependency type was annotated as *agent* in the second sentence. In turn, the verb-object dependency was annotated as *dobj* (direct object) in Sentence (1) and *nsubjpass* (noun subject of the passive) in Sentence (2).

Now, consider the task of inducing the extraction rule *open(X,Y)* denoting that X opens Y from these two sentences.

Two possible rules, in first-order logic formalism, that cover the first and second sentences with respect to the target relation *open(Person, Container), i.e., a Person opens a Container* would be:

Rule for the Sentence (1):

```
open(X, Y):- Boy(X), Box(Y), Verb(V), nsubj(V,X), dobj(V,Y).
```

Rule for the Sentence (2):

```
open(X,Y):-Child(X),Container(Y),Verb(V),agent(V,X),nsubjpass(V,Y).
```

However, once the following further BK predicates (Fig. 4.21) are provided to OntoILPER:

```
Container(Z) :- Box(Z).
Person(Z):- Boy(X).
Person(Z):- Child(X).

subj(X, Y):- nsubj(X,Y).
subj(X, Y):- agent(X,Y).

obj(X, Y):- dobj(X, Y).
obj(X, Y):- nsubjpass(X,Y).
```

Figure 4.21. Additional BK predicates for rule generalization used in OntoILPER.

Then, OntoILPER will induce just one extraction rule that covers both relation instances seen earlier:

```
open(X,Y):- Person(X),Container(Y),Verb(V),subj(V,X), obj(V,Y).
```

The further BK predicates that enable the rule generalization discussed above are actually provided by the annotation ontology that comprises several domain-independent term taxonomies extracted from WordNet, WordNet domains, and the mapping to the top level SUMO, which also contributes with a more broad taxonomic classification of nouns in the annotation ontology.

Moreover, a taxonomical classification of the syntactic relationships, i.e., the dependency relations as seen in the above example, is also provided by the annotation ontology. The taxonomic classification of these syntactic dependencies can be found in Appendix B. The

WordNet Domains taxonomy, i.e., that associates a term with a given domain field of research, can be found at http://www.wndomains.fbk.eu.

Another key aspect to emphasize here is related with the idea of the lexical knowledge represented in the annotation ontology as a mediator between text and the domain ontology. That is, several types of lexical resources can be exploited in IE, from the named entity dictionaries and gazetteers, to the domain terminologies or ontological thesauri.

Finally, it is worth highlighting that this generalization aspect of the rules on the examples discussed above would not be possible only using the original graph-based model for sentence presentation, without an additional algorithm for applying the rules on them and, accordingly, take into account the specialization/generalization relationships of the targeted instances in the examples.

To summarize, the goal of using ontologies in OntoILPER consists in allowing users to:

- use domain ontologies for the definition of extraction templates;
- represent the result of annotated corpora by a second (task) ontology; and
- obtain the extraction results by reasoning on SWRL rules applied over the Annotation Ontology. The result of this rule application process consists in instances of classes and relations, which are used for populating the domain ontology.

4.6.3. Discussion

This section discusses the aspects involving the two main issues in IE: *portability* and *extensibility*, relating them to OntoILPER.

Portability. Another important aspect taken into account in the annotation ontology design concerns the traditional advantages of the declarative solutions. It is well known that the IE task requires frequent changes of their solutions. In this scenario, the declarative solution proposed by OntoILPER for OBIE can provide a closer integration of the ontologies with a more natural and direct translation of the BK knowledge of an application domain of interest. Thus, with declarative knowledge, the necessary changes can be more easily taken into account.

Extensibility. The great expressivity of the declarative knowledge representation also becomes a major advantage that benefits OBIE, concerning the aspects of the extensibility of the IE systems. In most of the possibilities of inferences, concepts implied by the extracted facts can be expressed in a declaratory way and, consequently, these concepts can be organized into ontologies. In addition, the use of ontologies in the development of IE systems also increases their flexibility. For instance, the entities of a certain domain can be structured with the suitable granularity representing the subtle differences in hierarchy between the entities.

Another interesting aspect studied in this thesis concerns the hypothesis that an IE system can be portable to other domains if the recognition of named entities is extended with fined-grained ontological categories.

Indeed, in OntoILPER, once the semantic units (named entities and terms of a certain domain) have been identified, they are related to the concepts of the domain and annotation ontologies by semantic tagging, and the concepts play the role of semantic types, restricting the types of the entities participating in a binary relation, for example.

Compared to the traditional named entity broader types, finer grained ontological categories receives a special attention in OntoILPER, since several semantic mappings to external knowledge resources are provided. Such semantic mappings aim at annotating the terms in the input corpus associating them with finer grained taxonomical classification.

Actually, this aspect in OntoILPER methodology is demonstrated by the integration of the WordNet, WordNET domain, and SUMO hierarchies in the learning process.

As the knowledge acquisition methodology applied in OntoILPER favours the wiliness of integrating new BK sources, OntoILPER has a great potential to explore this semantic annotation aspect further. In the case where concepts are organised into generality hierarchies, semantic tagging selects the generality level relevant to a given domain of application. This semantic tagging task should both highlight the contrasts among critical objects of the domain, and attenuate or remove unessential difference among them.

4.7. Discussion on the OntoILPER Advantages and Limitations

This section is devoted to the discussion about the OntoILPER advantages and limitations. First, the following OntoILPER advantages are presented:

– **Rule composition during learning.** An interesting feature of the ILP learning component in the OntoILPER framework is its capability to employ rules learned in a previous learning session (*iteration i*) as additional BK predicates at a posterior learning session (*iteration i + 1*). Some authors call this capability as the *pipeline method* [Roth and Yih, 2007] [Giuliano *et al*., 2007].

This method mimics the typical strategy for solving more complex NLP problems in which a task is divided into several stages and solving them sequentially [Roth and Yih, 2007]. For instance, a NER is usually trained using a different corpus in advance, and given to a relation classifier as a tool to classify individual entities, hence providing entity features. Therefore, the pipeline method first trains an entity classifier, and then uses the prediction of entities as well as other local features to learn the relation model.

Observe that, although the true labels of entities are known when training the relation classifier, as it occurs in ACE Relation Detection and Characterization task, this may not be the case in real world information extraction scenarios. Thus, the pipeline method introduced above, as it is implemented in OntoILPER, may be of great help in a more realistic information extraction task.

Fig. 4.22 shows the flow of information exchanged between the BK Generation component and the Rule Learning component in OntoILPER.

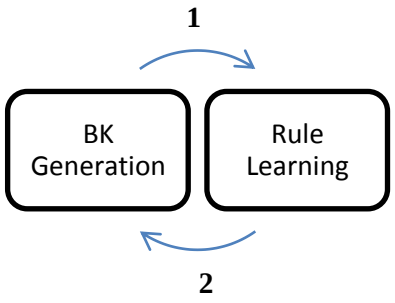


Figure 4.22. Composition of learned rules in OntoILPER framework.

For a better understanding of the following discussion, the reader should recall that the graph-based model of sentences can represent a set of binary relations and their arguments, along with constraints on these arguments, and the relation types. Additionally, one has to assume that the learning mechanism is able to identify, based on local contextual features, the involving entities that correspond to noun phrases in a sentence.

Consider the first learning scenario (edge 1 in Fig.4.22), indicated by the information flow according to the *Rule Composition* arrow in Fig. 4.22. This means that, given a proper BK as input (*BK Generation*), the problem resides in recognizing either entity or relation instances by means of the learned extraction rules (*Rule Learning*). This corresponds to the most learning scenarios of RE campaigns, such as Automatic Content Extraction (ACE) evaluation promoted by NIST[9], entity instances are already labelled in their evaluation corpora.

However, one should point out that this "facilitated" learning scenario may not reflect a real world scenario for IE, as one can always expect that the involving entity arguments in a relation are already annotated or explicitly indicated in the corpus.

Conversely, the information flow from the Rule Learning stage to the BK Generation stage, as indicated by the *edge 2* in Fig. 4.22, represents a more realistic IE scenario in which the relation classifier does not know the labels of its entity arguments. Therefore, the Rule Learning task should identify the class labels of the argument entities first, which means to generate extraction rules for classifying, separately, the two involving arguments of a relation. Next, the new class labels of the relation argument are then used as complementary BK (BK Generation) and passed to the Rule Learning stage, which attempts to discover a relation between them. Thereby, based on the previous identified argument entities, relation can be now recognized.

The following example well illustrates this point. Consider the relation $R_{12}(e_1, e_2)$ that depends on the class label of e_1 and e_2, where e_1 and e_2 are two argument entities. Note that the class labels of entities and relations in a sentence must satisfy some constraints. For example, if e_1 is the first argument of R_{12} denotes a *Location* class instance, then R_{12} cannot be *born_in* because the first argument of relation *born_in* has to be a *Person*.

In short, if e_1 and e_2 entity classifiers are trained independently, then the resultant extraction rule models of these two entities may be very useful as addition relational features by the R_{12} relation classifier.

An example of a composite rule for the target relation *live_in* induced by OntoILPER is shown in Fig. 4.23.

Hypothesis or target relation: *live_in*

#Literals = 7, PosScore = 8, NegScore =0 Prec = 100%

live_in(A,B):- t_pos(A,nn), isa_Person(A), t_hasDep(amod,B,C), t_next(C,B), isa_Location(B), t_isHeadNP(B).

Figure 4.23. Example of a composite rule in OntoILPER framework.

The above rule is evaluated in terms of (number of literals), (positive examples covered), (negative examples covered), and the (rule precision score). It means that “A” lives in “B”, if “A” is an entity instance classified as “Person”, and the head token of the nominal chunk “B” is classified as an instance of *Location* class. The other literals in the rule give additional contextual restrictions on the relation arguments.

Under the light of the above discussion, one can note that OntoILPER can be regarded as a pipeline method as well. In fact, this thesis provides several experimental assessments in the two learning scenarios discussed above (Chapter 6), comparing the obtained results with related work on three distinct datasets.

[9] National Institute of Standards and Technology

- **High expressiveness of the learned rules**. One important question one might ask is to know if the learned rules are expressive enough for concept learning problems. To begin with, consider the following rules describing a positive example e_i, as it is induced by a OntoILPER system:

$$e_i \leftarrow f_1 \wedge f_2$$
$$e_i \leftarrow f_4 \wedge f_4 \quad (1)$$

The first rule above means that e_i is a positive example if it is matched by features f_1 and f_2, whereas the second one means that the same positive example is determined if it is matched by features f_3 and f_4. The expressiveness of the two rules can be increased if *disjunctions* among rules are allowed.

In principle, the conditional part of a rule can be any logical combination of one or more features. This means that one can obtain a rule body with a single feature, or it may be composed by a complex disjunctive and conjunctive mixture of several features. However, in practice, given that most of the ILP systems are implemented in Prolog, the rule body is restricted to the conjunctive combination of features only. In this case, disjunction can only be expressed with multiple rules for the same head. Thus, the aforementioned theory could also be written as a single disjunctive rule (Formula 2).

$$e_i \leftarrow (f_1 \wedge f_2) \vee (f_3 \wedge f_4) \quad (2)$$

Note that use of disjunctions is a not a restriction of expressiveness of the theory. Indeed, also note that all boolean concepts over a finite domain can be represented with disjunctions because one could specify a concept using a disjunction of all examples that belong to the concept. In fact, it is well known that any logical formula can be transformed into a disjunction of conjunctions, i.e., the so-called *disjunctive normal form*.

In the context of concept learning, this implies that *any* possible concept over a feature set can be represented as a set of conjunctive rules [Furnkranz, 2012]. Moreover, due to the fact the conjunctive operator is *commutative*, the problem of constructing a rule body reduces to the problem of selecting the most relevant subset of features, i.e., it does matter their order in the rule. Particularly in ILP, the side effects of the commutative property of the conjunctive operator are twofold:

  - it imposes a syntactic restriction of the hypotheses space, significantly reducing the complexity of the searching algorithms through the hypotheses space just for an appropriate combination of features.
  - the decomposition into a set of conjunctive rules is much more understandable for a human than a longer boolean formula with an arbitrary complex structure.

- **Implicit information extraction.** Human readers can infer implicit information from explicitly stated facts using commonsense knowledge. On the contrary, automated IE systems, trained to extract explicitly-stated information, are limited in their ability to extract implicitly stated facts, for many reasons. First, being limited to the scope of a sentence at a time, state-of-the-art extraction systems based on features or kernel functions presented in Section 3.4.1 are not suitable to discover implicit relations [Raghavan *et al*., 2012]. Second, implicit relations exist in different sentences, paragraphs, or even across documents, and they require further relational knowledge to be inferred [Raghavan *et al*., 2012]. Third, these systems do not have access to commonsense knowledge, and hence are incapable of performing deeper inference.

For the following discussion, consider the two sentences shown in Fig. 4.24 (part a).

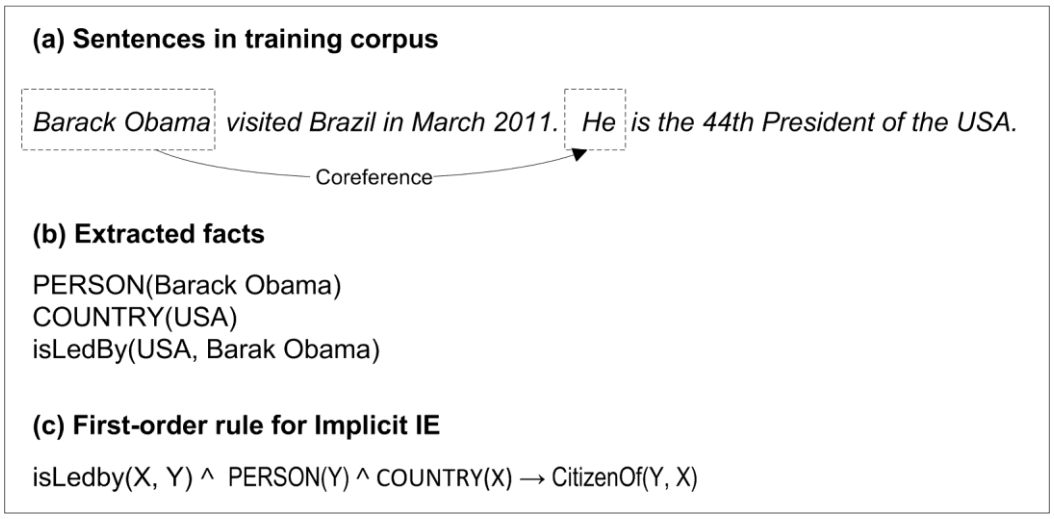


Figure 4.24. Example showing various stages of implicit IE in OntoILPER.

Considering one sentence at a time, features-based and kernel-based RE methods cannot detect the *CitizenOf* relation between "Barack Obama" and "USA" from the explicit extracted facts of the two sentences above. Actually, only using the information given by the contextual clues scattered on those two sentences, and considering one sentence at a time, it is not possible for such extraction methods to infer that Barack Obama is a citizen of USA.

The reason is that the models in statistical machine learning for IE, including feature-based or kernel-based approaches, cannot make use of background knowledge information about the problem during learning [Fürnkranz, 2012]. In order words, all description features are expressed in a fixed-size vector, and it is well know that such an attribute-value representation formalism has propositional expressivity, i.e., zero-order logic expressivity. Therefore, extracting implicit relations is challenging for the current statistical-based relation extraction models.

Contrarily, OntoILPER enables the discovery of implicit relations because it relies on first-order logic for representing examples and its hypothesis language. Thus, during the learning phase, the required additional relational knowledge can be easily integrated to the initial (default) background knowledge.

Returning to the example shown in Fig. 4.24, thanks to both the first-order formalism that allows OntoILPER to represent and use relational knowledge, and its capability to handle coreference mentions in text (see Fig. 4.24 part a), the predicate *isLedBy(USA, Barack Obama)* can be easily extracted (Fig. 4.24 part b). Then, the first-order rule (Fig. 4.24 part c) is such that its body typically contains relations that are explicitly stated, while the head employs a less-frequent mentioned relation that is easily inferred. Finally, during *training*, such a rule can be integrated with the default (normal) BK, and they are used to derive additional facts from extracted information using the normal (prolog) inference mechanism adopted by OntoILPER.

To summarize, it is clear that implicit knowledge implied by the text can be expressed by additional background knowledge predicates that effectively provide access to the logical implications scattered in textual data. Consequently, by employing the first-order logic formalism, OntoILPER is able to discover potential implicit information from text, thanks to its representation formalism that can cope with relational knowledge.

Since OntoILPER heavily relies on NLP tools and ILP systems, it presents the following shortcomings:

- **Error Propagation in Text Preprocessing.** The introduction of parsing errors in OntoILPER preprocessing pipeline might hamper its extraction performance, mainly due to the propagation of errors that inevitably occurs in this typical pipelined architecture. In fact, related work on RE only based on deeper NLP subtasks inevitably also suffered from errors introduced by NLP tools.
- **High computational cost**: The downside of the greater expressiveness in ILP is that such flexibility comes at a high computational cost, which difficult ILP systems in scaling-up [Page and Srinivasan, 2003]. In other words, not only the great expressivity of the hypothesis language, but also the flexibility of integrating BK in ILP contributes to a combinatorial complexity of the learning task. The main reason is that the evaluation of a single hypothesis in ILP involves testing if the hypothesis, along with the background knowledge, entails the examples used in training. Consequently, the time required to evaluate this hypothesis time mainly depends on the size of training data. Thu, in learning problems where attribute-value representations are adequate, propositional learning is recommended for efficient reasons.
- **Difficulty in dealing with pure numerical data**. As the majority of the current state-of-the-art ILP systems are implemented in Prolog, the classical ILP framework has difficulties to deal with numerical data and uncertainty [Bratko, 2000].

4.8. Conclusion and Final Remarks

In order to contribute to the IE field, this thesis has investigated the main open challenges in this research field and has identified some problems that were directly addressed by OntoILPER in this chapter.

Accordingly, this thesis proposed OntoILPER, an OBIE method that automatically extracts instances of classes and relations from textual sources. OntoILPER relies on several linguistic-related knowledge sources, including shallow and deep NLP subtasks; and two ontologies (domain and annotation) that integrates and formalizes the background knowledge automatically exploited by the inductive learning component in OntoILPER.

The OntoILPER distinguishing features, compared to related work, consist in the high expressiveness of its final induced extraction rules, and a rich set of features that are utilized by the ILP-based component in OntoILPER functional architecture.

Besides OntoILPER, which constitutes the major contribution of the thesis, other relevant contributions were also presented in this chapter, including:

- a rich unified model for sentence representation that encompasses several types of features, including morphological, syntactic, semantic, and relational ones. The proposed graph-based model is flexible enough to integrate mappings between terms and semantic resources, such as semantic repositories and ontologies, as well.
- a method to transform and simplify graph-based representations of sentences. This simplification method aims at improving the overall extraction performance results in terms of precision and recall by reducing the size of the graphs.
- the design of a domain-independent and expressive annotation ontology in OWL/DL. This ontology formalizes the resultant analysis carried out in the preprocessing stage in the proposed solution, in which various types of features were mapped to formal structures to an OWL/DL ontology.

This chapter concluded with a discussion of the further advantages of OntoILPER and its limitations.

A final remark before concluding this chapter concerns the exploitation of the extracted instances in a post-processing step. With this respect, some OBIE systems like BOEMIE [Castano *et al*., 2008] and SOBA [Buitelaar *et al*., 2008] combine the extracted instances, considering them as facts, to semantically interpret them according to a very specific domain [Karkaletsis *et al*., 2011]. Although this seems an interesting option for constructing new concepts, or even extracting implicit information, this also makes those OBIE systems strongly dependent upon a specific domain of interest.

The BOEMIE system, for instance, performs an inference step in which high level composite concepts from sport events and relations among them are generated, based on previously extracted facts, whereas in the SOBA system, discourse analysis [Hilbert *et al*., 2006] is used for identifying relations in the domain of soccer match events.

Differently from these two cited OBIE systems, the main goal of OntoILPER consists in performing a *domain-independent* extraction of instances, thus not being restricted to a single domain of interest. Indeed, the user can simply change the input domain ontology in order to obtain instances of the elements defined by the new input ontology.

Moreover, the development of some inference mechanism for further analysis of the resultant extracted instances would make OntoILPER strongly dependent upon a specific domain. Therefore, an additional processing stage in OntoILPER for combining the extracted facts or making further inference tasks strongly related to a very specific domain is out of the scope of this thesis.

Next chapter proceeds with the detailed descriptions of the methodological design decisions presented in this chapter. Such design decisions were implemented as a comprehensive OBIE framework which can also perform OP.

Chapter 5

OntoILPER Framework

This chapter presents a detailed description of the *OntoILPER Framework*, the developed prototype that implements the OntoILPER method introduced in the previous chapter.

As already mentioned in Section 4.2, due to its many positive aspects, the ILP technique was adopted as the core component for machine learning in the proposed solution. The main reason is that this machine learning technique offers to user the possibility of automatically inducing extraction rules in symbolic form. Therefore, the generated rules present the advantage of being fully interpreted by a knowledge engineer who can refine them after the rule induction, aiming at improving the whole extraction process. Moreover, symbolic rules can be automatically converted into other rule formalisms, such as SWRL, one of the proposed rule languages for the Semantic Web.

This chapter proceeds with a detailed description of the OntoILPER Framework.

5.1. The Architecture of the OntoILPER Framework

As already presented in the previous chapter, the OBIE process is based on the supervised machine learning approach, and thus it is carried out in two distinct phases or modes in OntoILPER:

- **Training Mode or Rule Learning Mode**. In this mode, a theory (a set of extraction rules) is induced from a text corpus containing the learning examples. Then, OntoILPER generates extraction rules from this annotated learning corpus.
- **Application Mode or Rule Application Mode**. This mode is in charge of applying the learned theory derived by the above Rule Learning phase on a set of equally rich annotated documents provided as input. Then, the extracted instances of classes and relations are added into the domain ontology.

It is worth emphasizing that, in both of the phases above, a previous corpus pre-processing stage takes place in which several NLP tools are utilized, followed by an automatic representation of the examples according to the hypothesis space proposed and presented in Section 4.4.

The proposed OBIE method introduced in this thesis (Chap. 4) was implemented as a modular pipelined architecture that integrates several components, as shown in Fig. 5.1. As a convention for this chapter, the dashed line boxes denote the modules of the architecture, while the rectangular boxes represent the key components performing a given process. Both oval-shaped and small figures denote either an input or an output element. This architecture is implemented by the OntoILPER Framework.

In a broader view, the OntoILPER framework performs the following tasks distinguished by the two operation modes: *training* and *application*.

Training mode. During training, in the *Text Preprocessing* module, the annotated documents in XML format are given as input to the *Sentence Representation* and *Simplification* module. This last module is in charge of transforming the annotated sentences as directed graphs which can be further simplified or reduced (Section 5.3). Then, the reduced graphs representing the sentences containing positive and negative examples of relation pairs, e.g.,

rel(arg$_1$, arg$_2$) are passed to the *Background Knowledge Generation* module that exploits a given domain ontology.

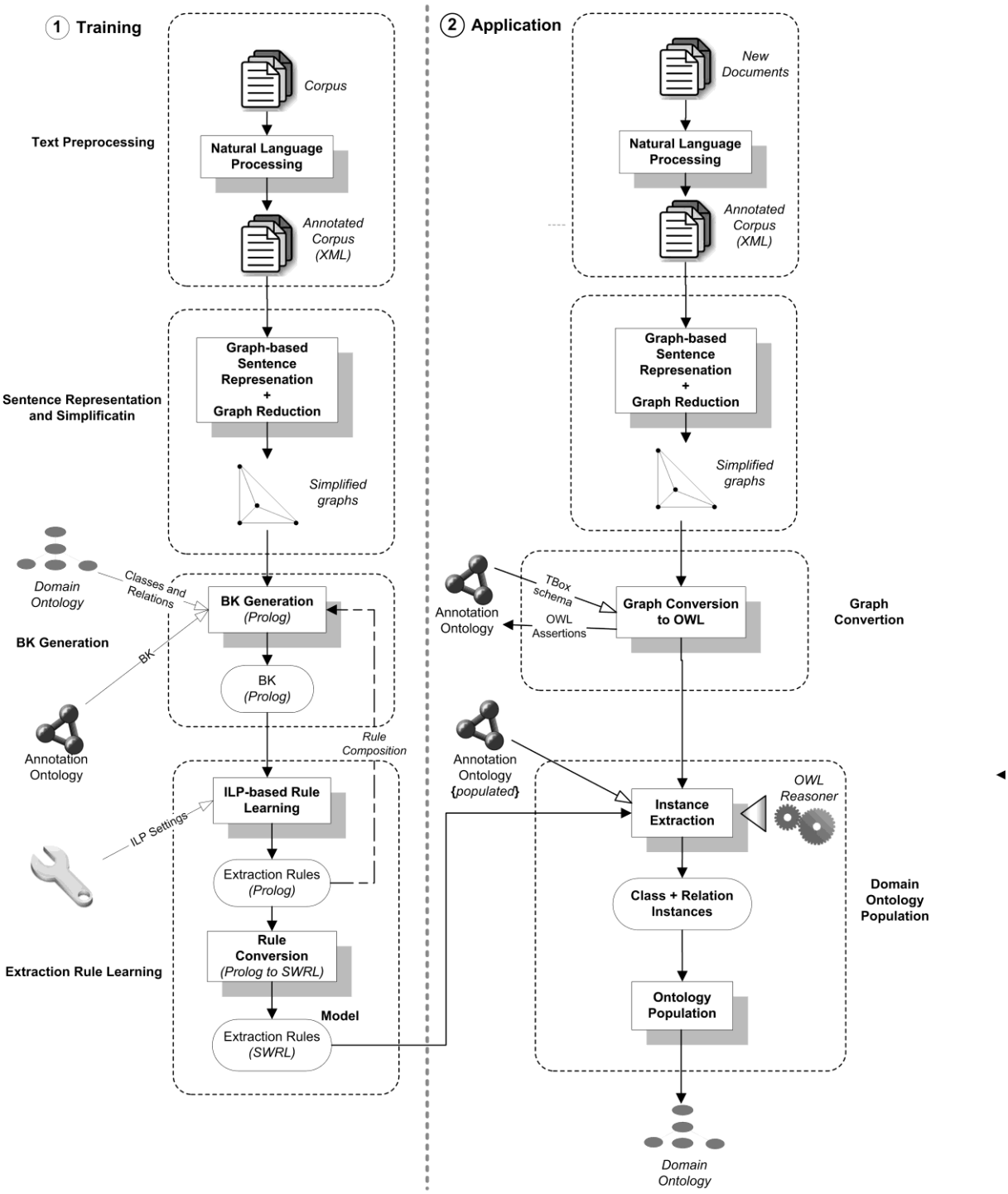


Figure 5.1. Overview of the OntoILPER framework architecture.

The domain ontology provides ontological elements (classes and relations) as background knowledge that guides the information extraction process. In addition, the BK Generation step takes profit of a second domain-independent ontology (*Annotation Ontology)* that offers a formal specification of all types information and annotations of the input corpus (Section 5.6).

Next, in the *Extraction Rule Learning* module, a general ILP system, provided with appropriate BK in the form of logic programs as well as customizable parameters, automatically induces symbolic extraction rules also expressed as a set of logical programs, or a theory, in ILP terms. This theory denotes the e*xtraction rules models* that are derived after a previous conversion between the rule formalisms of Prolog and OWL/DL.

Application mode. During the application phase, the same first two modules of the training phase also perform the annotation and the representation/reduction of sentences of the new input set of documents. Then, the resultant document annotations, for each sentence in the new input corpus, are converted into ABox assertions (instances), in the *Graph Conversion* module, and are finally integrated into the Annotation Ontology.

As shown in Chap. 4, the Annotation Ontology has two purposes in the methodology: it provides BK information by means of its ontological elements, and it constitutes a repository for the candidate instances (classes and relations).

Finally, the *Domain Ontology Population* module applies the extraction model (in SWRL) on the Annotation Ontology that encompasses all candidate instances to be extracted. Indeed, the Domain Ontology Population module delegates this classification task to a standard SW reasoner that is in charge of making the inference of the SWRL rules on the populated Annotation Ontology. The classified instances of classes and relations resultant from this process, are used for populating the domain ontology.

The remainder of this section describes each module and key component of OntoILPER functional architecture in more details. First, the related-training modules depicted in the left side of the Fig. 5.1 are introduced. Next, we proceed with description of the OntoILPER framework application module. This section concludes with a discussion on the possible application scenarios envisaged by this research work.

5.2. Text Preprocessing

This section describes the *Natural Language Processing* step that is in charge of the annotation process of input corpora by means of fully automatic state-of-the-art NLP tools. Then, the output of this annotation process must be carefully designed for achieving both a robust and efficient representation of the several aspects concerning shallow and deep NLP tasks.

Natural Language Processing in OntoILPER

For carrying out the general-purpose NLP subtasks of interest in the context of this thesis, several state-of-the-art NLP tools, API´s, and semantic resources were integrated as a single software component in the OntoILPER Framework.

These subtasks are performed in *pipeline mode*, i.e., starting with simpler analysis such as tokenization and sentence splitting, whose output results become the input of the more complex subtasks (POS tagging and parsing, for instance). Fig. 5.2 shows the complete pipeline according to the order that was implemented in OntoILPER.

In total, twenty different NLP subtasks were seamlessly integrated which constitutes the Natural Language Processing Component (NLPC) in OntoILPER (Fig. 5.2). The reader should refer to the Section 2.2 for a brief review on NLP subtasks treated here.

The NLPC relies on the Stanford CoreNLP[1] tools for carrying out eight NLP subtasks of the proposed pipeline, namely: *tokenization, sentence splitting, POS tagging, lemmatization, NER, constituent syntactic parsing, dependency parsing,* and *coreference resolution*.

The chunking analysis is performed by Apache OpenNLP[2], while *morphological analysis, gazetteers look-up,* and *pronoun normalization* were implemented as ad hoc programs within the NLPC.

The NLPC includes the *base line sense*[3] of the head words (for nouns and verbs) retrieved from the WordNet using the Java WordNet library (JWNL)[4]. It also retrieves the list of all synonyms and hyponyms of a given head word using the JWNL.

OntoILPER NLP Component		
Seq	**NLP Subtask**	**Tool or Resource**
1	Tokenization	Stanford CoreNLP
2	Sentence Splitter	
3	POS	
4	Lemmatization	
5	Chunking	OpenNLP Chunker
6	NER	Stanford CoreNLP
7	Morphological Analysis	*ad hoc* programs
8	Gazetteer Look-up	
9	Pronoun Normalization	
10	Syntactic Parsing - Constituent	Stanford CoreNLP
11	Syntactic Parsing - Dependency	
12	Coreference Resolution	
13	WordNet baseline senses (head words)	Java WN Libray
14	WordNet synsets synonyms and hypernyms	
15	Similar words	Lin´s database
16	WSD	Sense Learner
17	SRL with Propbank/VerbNet	ClearNLP
18	Selectional Preferences	SuperSense Tagger
19	Semantic Mapping to Domains	WordNet domains
20	Semantic Mapping to SUMO	*Ad hoc* program

Figure 5.2. Complete pipeline of the NLP subtasks performed in OntoILPER.

Using the Lin´s similar words dataset [Lin *et al.*, 2003], the NLPC fetches the list of *N* (*N = 5*) most similar words to a given head word [Lin *et al.*, 2003], which is followed by the

[1] Stanford CoreNLP Tools. http://nlp.stanford.edu/software/corenlp.shtml.
[2] Apache OpenNLP. The Apache Software Foundation. http://opennlp.apache.org
[3] In WordNet, the base line sense of a given word is the first sense in a particular word entry, regardless it has one or more senses.
[4] Java WordNet Library (JWNL). http://sourceforge.net/projects/jwordnet

WSD task performed by the Sense Learner [Mihalcea and Faruque, 2004]. This tool disambiguates noun and verbs.

The semantic role labelling for all verbs in the corpus are retrieved by the ClearNLP[5]. Still for verbs, the *SuperSense Tagger* [Ciaramita and Altun, 2006] finds their *selectional preferences*. SuperSense Tagger[6] annotates text with 41 broad semantic categories (WordNet supersenses) for both nouns and verbs, performing both sense disambiguation and named-entity recognition. This tagger implements a discriminatively-trained Hidden Markov Model.

In addition, NLPC takes advantage of the existing mapping between all WordNet synsets and the SUMO ontology[7], for retrieving the corresponding classes into the SUMO ontology and associating them to a given head word in the input corpus.

Another NLP subtask integrated into the NLPC concerns the annotation of words with labels from the WordNet Domains, by means of ad hoc mapping programs.

At the end of the entire pipeline, the NLPC produces a rich annotation version of the input corpus, and this output is structured according to a set of markup declarations that defines the resultant XML file(s) according to the Document Type Definition (DTD) to be presented in the next section.

An excerpt of this XML file for the sentence: “I think he has been in Washington too long” is listed in Appendix E.

5.3. Sentence Representation and Simplification

In order to facilitate the inspection of the examples in the experiments, a tool for visualizing the resultant graph-based representation of sentence (introduced in Section 4.5) was developed. This tool is also able to transform the graphs by trimming unnecessary nodes, edges, or even sub-paths that may not containing relevant information.

The approach to simplifying graph-based representations of sentences was implemented as a separate module in the OntoILPER functional architecture. The decision to decouple the simplification process from the preprocessing step was primarily motivated by the fact that the former allows for performing the latter separately, taking advantage of previous preprocessed datasets from other domains of interest.

The prototype implemented for validating the proposed approach offers to the user the following options:

- **Trimming operations on graph representing sentences**. These transformation and simplification rules are applied over graph-based representations of sentences. Such graphs are generated by the Stanford Dependency parser which provides other NLP subtasks as well.
- **Graph-based model visualization**. It offers the convenient visualization of each sentence in the input corpora, according to the graph-based model for sentence representation in ILPER. In addition, the user interface shows the original sentence as well as the simplified version of it. For graph visualization and manipulation, this thesis relied on the software library JUNG[8] (Java Universal Network/Graph Framework). It provides a common and extendible language for both data modeling and visualization that can be represented as directed and undirected graphs, among others.

[5] ClearNLP Project. http://clearnlp.wikispaces.com
[6] SuperSense Tagger. http://sourceforge.net/projects/supersensetag
[7] Suggested Upper Merged Ontology (SUMO). http://www.ontologyportal.org
[8] http://jung.sourceforge.net/

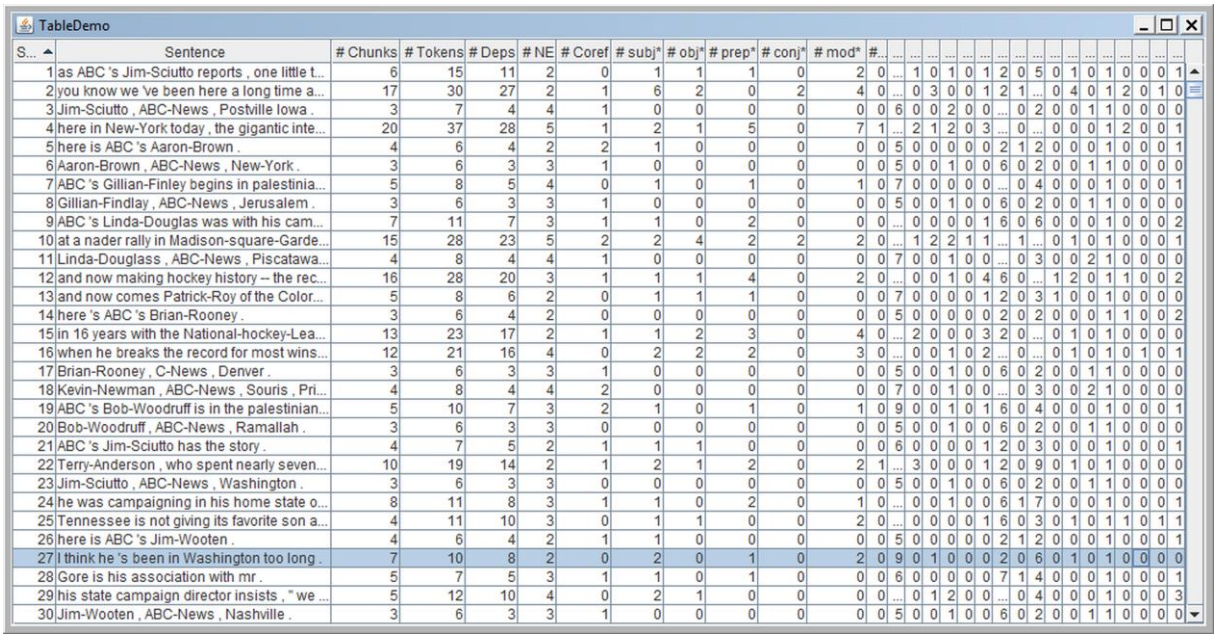
- **Basic reduction statistical information.** After choosing the desired set of rules, this module generates, in a .csv file, basic statistics about the simplification process in terms of reduction ratios of tokens, chunks, typed dependencies, entities, etc.

In what follows, the major components that constitute the tool implementing the graph-based model introduced in the sections 4.4. and 4.5 is briefly described.

Graph-based Representation of Sentences and Basic Reduction Statistics. According to the pipeline of the OntoILPER framework architecture in Fig. 5.1, after the Text Preprocessing step, the annotated documents in XML format are passed as input to the step of "Graph-based Sentence Representation". This last step is responsible to parse the XML file representing the annotated corpus, and convert the XML-based structure to a graph-based representation.

As already mentioned, the proposed solution relies on the JUNG tool which consists of a useful library containing several graph-related services and graph visualization facilities. The proposed solution takes profit of the tree-like structure of the XML file containing the annotated corpus for creating a directed graph (for each sentence) with nodes denoting tokens or words, and edges denoting the relational dependencies among tokens and chunks, among token themselves, etc.

Fig. 5.3 shows the main window interface of the developed tool. It shows the sentences in the input corpus, along with its ID, or order in the corpus, and some basic statistics of the corpus, before the simplification step takes place. The tool provides information at sentence-level about number of tokens, chunks, named-entities, coreference, subjects, and objects, among others. It also generates a .csv file containing all statistics about the corpus, such as number of tokens, chunks, sentences, named entities, subject and object per sentence, etc.



S...	Sentence	# Chunks	# Tokens	# Deps	# NE	# Coref	# subj*	# obj*	# prep*	# conj*	# mod*
1	as ABC 's Jim-Sciutto reports , one little t...	6	15	11	2	0	1	1	1	0	2
2	you know we 've been here a long time a...	17	30	27	2	1	6	2	0	2	4
3	Jim-Sciutto , ABC-News , Postville Iowa .	3	7	4	4	1	0	0	0	0	0
4	here in New-York today , the gigantic inte...	20	37	28	5	1	2	1	5	0	7
5	here is ABC 's Aaron-Brown .	4	6	4	2	2	1	0	0	0	0
6	Aaron-Brown , ABC-News , New-York .	3	6	3	3	1	0	0	0	0	0
7	ABC 's Gillian-Finley begins in palestinia...	5	8	5	4	0	1	0	1	0	1
8	Gillian-Findlay , ABC-News , Jerusalem .	3	6	3	3	1	0	0	0	0	0
9	ABC 's Linda-Douglas was with his cam...	7	11	7	3	1	1	0	2	0	0
10	at a nader rally in Madison-square-Garde...	15	28	23	5	2	2	4	2	2	2
11	Linda-Douglass , ABC-News , Piscatawa...	4	8	4	4	1	0	0	0	0	0
12	and now making hockey history -- the rec...	16	28	20	3	1	1	1	4	0	2
13	and now comes Patrick-Roy of the Color...	5	8	6	2	0	1	1	1	0	0
14	here 's ABC 's Brian-Rooney .	3	6	4	2	0	0	0	0	0	0
15	in 16 years with the National-hockey-Lea...	13	23	17	2	1	1	2	3	0	4
16	when he breaks the record for most wins...	12	21	16	4	0	2	2	2	0	3
17	Brian-Rooney , C-News , Denver .	3	6	3	3	1	0	0	0	0	0
18	Kevin-Newman , ABC-News , Souris , Pri...	4	8	4	4	2	0	0	0	0	0
19	ABC 's Bob-Woodruff is in the palestinian...	5	10	7	3	2	1	0	1	0	1
20	Bob-Woodruff , ABC-News , Ramallah .	3	6	3	3	0	0	0	0	0	0
21	ABC 's Jim-Sciutto has the story .	4	7	5	2	1	1	1	0	0	0
22	Terry-Anderson , who spent nearly seven...	10	19	14	2	1	2	1	2	0	2
23	Jim-Sciutto , ABC-News , Washington .	3	6	3	3	0	0	0	0	0	0
24	he was campaigning in his home state o...	8	11	8	3	1	1	0	2	0	1
25	Tennessee is not giving its favorite son a...	4	11	10	3	0	1	1	0	0	2
26	here is ABC 's Jim-Wooten .	4	6	4	2	1	1	0	0	0	0
27	I think he 's been in Washington too long .	7	10	8	2	0	2	0	1	0	2
28	Gore is his association with mr .	5	7	5	3	1	1	0	1	0	0
29	his state campaign director insists , " we ...	5	12	10	4	0	2	1	0	0	0
30	Jim-Wooten , ABC-News , Nashville .	3	6	3	3	1	0	0	0	0	0

Figure 5.3. Interface of the visualization and simplification tool.

Fig. 5.4 shows the graph-based representation of the sentence *id=27* of a corpus from the news domain. In this figure, the original version of the sentence (no simplification operation applied) is displayed. Nodes denote tokens and two types of edge are also displayed: typed dependency relations and *next* which indicates the next token in the sentence. The gray node

is the root (the main verb of the sentence). Node content "*s_id#t_id*" means: *s_id* denotes the sentence order in the corpus; and *t_id* is the token order in the sentence.

Graph Reduction. The visual tool introduced above also provides several transformation and simplification rules aiming at reducing the size of the resultant graphs created by the aforementioned option. This reduction proved to be very useful in several experiments conducted on several corpora in biomedical domains [Lima *et al.*, 2014]. Fig. 5.5 shows a simplified sentence.

Such rules were implemented in Java as another option offered to the user by the visualization tool. All transformation/simplification rules described in Section 4.5 were implemented using heuristics and graph traversal algorithms.

The simplification step is followed by the BK Generation step that utilized the simplified versions of the graph-based sentence representations. This last step is described in the next section.

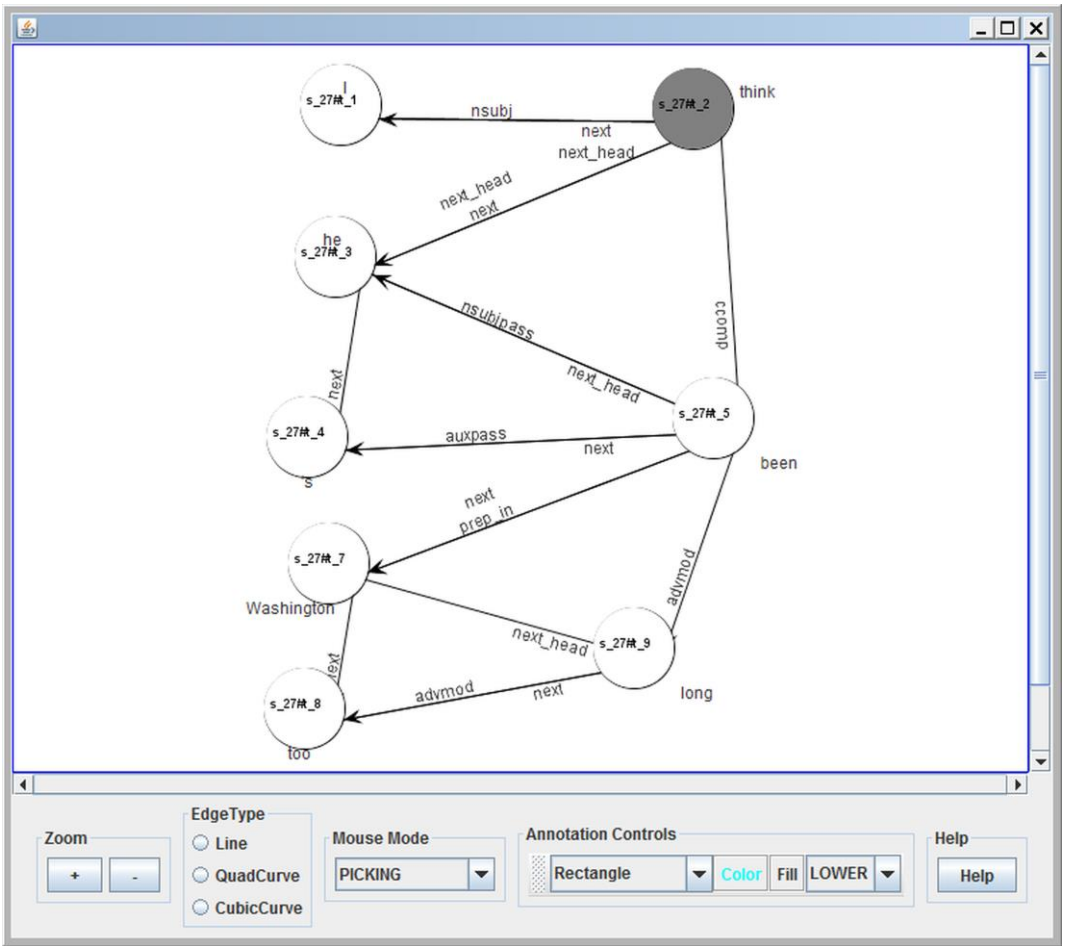


Figure 5.4 Visual dependency parsing representation of the sentence: "I think he´s been in Washington too long".

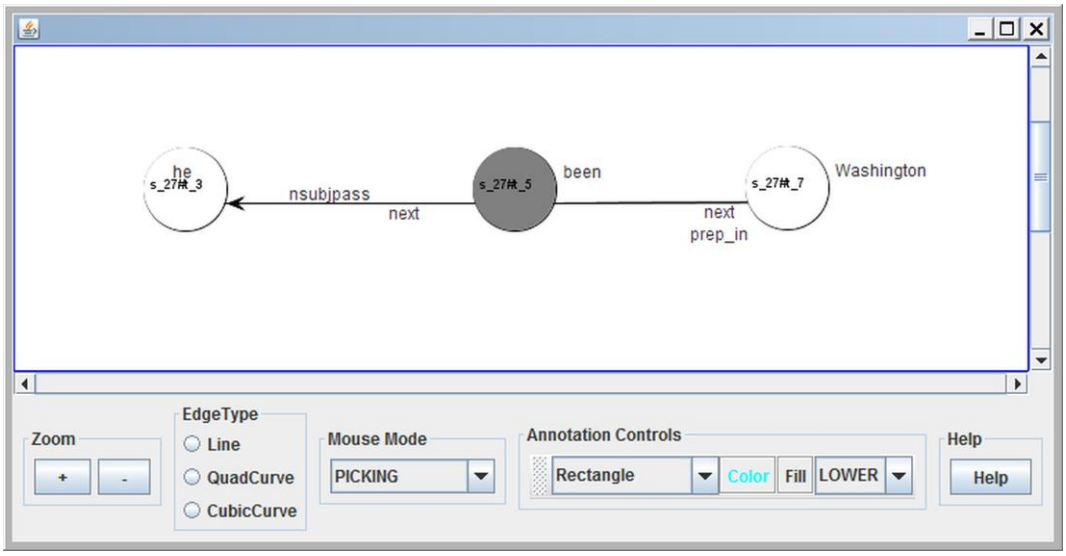


Figure 5.5. Simplified version of the sentence shown by the tool.

5.4. Background Knowledge Generation

In the previous steps of OntoILPER Framework, i.e., after the creation of the directed graphs representing the sentences and examples in OntoILPER, the resultant graphs may be reduced, aiming at removing potential distracting information from them.

Then, in the *BK Generating* step, it is carried out the critical task on identifying, extracting, and appropriately representing relevant background knowledge. The main goal of this step is to exploit the ILP feature that makes it different from traditional learning algorithms: an easy way of defining background knowledge in declarative form.

To illustrate this, consider the typical scenario in propositional machine learning in which the incorporation of expert knowledge is usually done by introducing new features, whose values are computed from other attributes values. In most of the related work on NER and RE, expert knowledge is defined by adding new columns as functions of the other data columns [Zhou *et al*., 2007] [Qian and Zhou, 2012] [Jiang, 2012]. This is particularly evident in kernel-based methods for RE in which the structural representation of sentence are converted to hundred of features in a vectorial representation by applying similarity functions on tree representations. As a result, part of the relational knowledge is lost in this transformation process [Special *et al*., 2006].

Another limitation of the vectorial representation model concerns the further restriction of having a unique representation format for all examples. That is, one feature is created for each element in the domain, and the same structure is used for characterizing all examples under consideration. Usually, this results in a very sparse data table because most of the attributes will contain null values, due to the difference among the examples. Indeed, Brown and Kros (2003) pointed out that this data sparseness problem is even more critical when deep knowledge is explored, which can cause serious problems for propositional machine learning algorithms.

By contrast, in the specific ILP-based approach to information extraction and ontology population proposed in this thesis, each example is represented independently of the others. Thus, the data sparseness problem for representing the examples is highly reduced. In fact, the aforementioned limitations are overcome by means of a first order formalism for representing both BK and examples. This enables several sources of information, either propositional or relational in nature, to be effectively represented without the drawbacks of the propositional approaches mentioned earlier [Fürnkranz *et al*., 2012].

It should be mentioned that the ability to take into account relational BK and the expressive power of the language of the discovered patterns are distinctive features of the proposed ILP-based OBIE method. Another advantage is that the (symbolic) extraction rules induced by the ILP-based component in OntoILPER can potentially inspire new insights in the knowledge acquisition task, by providing results easily understandable by human experts.

To summarize, these thesis testes the working hypothesis that, by using the richer ILP formalism, the proposed OBIE method should be able to directly represent a vast amount of BK extracted from ontologies, semantic resources, and both shallow and deep analysis originated from natural language annotation tools.

In what follows, the BK generation process conducted in OntoILPER framework is explained. This step takes into account the structural and linguistic features describing each element of the domain in interest and how to represent them as Horn clauses predicates for the core ILP-based learning component of the proposed OBIE architecture. Then, the way how additional BK can be integrated into the OBIE task carried out in OntoILPER is exemplified.

5.4.1. Linguistic and Structural Features

As previous research has shown, automatic morphosyntactical analysis of texts can provide very useful features in several IE related tasks [Zhou 2005] [Finn 2006] [Zhang *et al*., 2008] [Jiang, 2012]. In this same trend, semantic mapping to available semantic resources, like WordNet-related lexical resources [Bentivogli *et al*., 2004], Semantic Role Labeling [Harabagiu *et al*., 2005], and the SUMO ontology [Zouaq *et al*., 2010] can be also very useful for IE.

In this respect, after a careful investigation of the most successful features used in IE, the features listed in Table 5.1 were selected. Such features provide a suitable hypothesis (feature) space for the classification problem of ontological instances, describing each word, sentence, entity, and relationship found in textual data.

Furthermore, thanks to the relational model of sentences and examples proposed in this thesis, i.e., the extended E-R model (seen in Section 4.4.1), all these features are properly structured according to a relational unified view that seamlessly encompasses, under a single model, linguistic and ontological elements as background knowledge for information extraction and ontology population.

The major advantages of this unified relational model are two-fold:

- it is linguistically motivated in the sense that it is able to combine the three aspects of linguistic analysis: lexical, syntactic, and semantics; and
- It provides an increased flexibility towards facilitating the inclusion of other kinds of background knowledge, especially in the form of semantic resources and ontological elements, as done in OntoILPER.

Four main features categories are distinguished

- *Lexical features.* They concern the word itself, its lemma, length, and the indication if the token is a word, a symbol, or a number.
- *Syntactic features.* They consists of word´s POS tagging; head word of nominal, prepositional or verbal chunk; and its generalized form which does not consider plural for nouns neither verb tenses. Both bi-grams and tri-grams of consecutive POS tags of words as they appear in the sentence are also provided. Some experiments were conducted using 4-grams, but bi-grams and tri-grams achieved better results in the preliminary experiments reported by this thesis.

  Under this feature category, the chunking-related features are also included. Here, the chunks that segment sentences into noun, prepositional, and verb phrases provide

very useful information: its type (nominal, verbal or prepositional), its head word, and its relative position to the main verb of the sentence. This last feature describes the distance of verb arguments, either in the role of subject or object in the sentence.

- *Semantic features*. They include the entity mention feature that can be provided by the input corpus. For instance in reACE RDC datasets, each annotated entity has: its *type* (Person, Organization, Location, Facility, etc); its *subtype* (Business, Family, Part-Whole, etc); and its *mention type* denoting that the entity is a proper noun, pronouns, or quantified nouns (nouns quantified with a determiner, quantifier, or a possessive). Other semantic types of this category may be easily designed in the case of new similar features found in new corpora to be considered. In addition, all semantic features derived from WordNet (sense/hypersense, synomyns), supersenses, domain sense given by WN Domains, et semantic roles are also treated as semantic features in the proposed model.
- *Structural features*. This last group of features defines the structural framework for all others features in the graph-based model introduced in (Section 4.5). The sequencing of token preserves the order of the tokens as they appear in the input sentence. Similarly, one represents the existing part-whole relationship between tokens and the chunks containing them, i.e., the tokens belonging to a chunk. In addition, the chunking sequence in a sentence is represented by specific relationships connecting their respective head tokens.

  The last type of structural features consists of the grammatical dependency between two words in a sentence, according to the typed dependencies between them given by the Stanford dependency parser. This includes the name of each typed dependency between the heads of two tokens in a sentence. This corresponds to the label (name) of each typed dependency between both the heads of (multi-)words.

OntoILPER relies on Prolog as the representation language of examples, domain entities, relations, and all types of features mentioned above. In fact, all these features are converted to its corresponding Prolog predicate. As a result, the factual knowledge base (in Prolog) constitutes the language bias that restricts the hypothesis space and guides the search in the hypothesis space.

Tab. 5.1 illustrates this conversion subtask by generating all logical predicates that characterizes the instance of the *Person* class, "*Myron*".

For most of the predicates, the first order logic representation of both linguistic and structural features mentioned earlier is fairly straightforward: a *unary predicate* in logic programming correspond to an entity definition, *token(t1)*, in the hypothesis space (data model), whereas *binary predicates* correspond to features (attribute, value), and relations, e.g., $rel(arg_1, arg_2)$.

The only case of a ternary predicate in the proposed hypothesis space corresponds to the t*_hasDep/3 predicate*, in which the first argument denotes the label of the typed dependency, while the other two arguments correspond to the word´s ids participating in the dependency relation.

Differently from other machine learning techniques that use feature vectors for representing *context windows* (*n* tokens on the right/left of a given word *w* in a sentence), the *next/2* predicate, which relates one token to its immediate successor in a sentence, is provided. Fig. 5.6 illustrates some of the predicates (in Prolog syntax) generated by this step.

Table 5.1. Prolog predicates for the token "Myron" (t_1) when applicable.

Group	Prolog Predicates	Meaning
Corpus entities	doc(d_1)	*d_1* is a document identifier
	sent(s_1)	*s_1* is a sentence identifier
	chunk(ck_1)	*ck_1* is a chunk identifier
	token(t_1)	*t_1* is the token identifier
Lexical features	*t_stem(t_1, "Myron")*	token *t_1* stemming is "Myron"
	t_length(t_1, 5)	token *t_1* has length of 5 characters
	t_orth(t_1, upperInit)	token *t_1* begins with an initial uppercase letter
	t_morph_type(t_1, word)	token *t_1* is a word, a symbol, or a number
	t_norm_word(t_1, word_stem)	*word_stem* is the normalized form of the token t_1
	t_gaz(t_1, gaz_category)	*t_1* was found in *gaz_category* gazetteers list
	*t_ner(t_1, person)**	*t_1* was annotated by the NER as a Person entity
Syntactical features		
POS and POS n-grams	*t_pos (t_1, nnp)* *t_gpos(t_1,nn)* *t_bigPosBef (t_1, ….)* *t_bigPosAft (t_1, vbz-vbg)* *t_trigPosBef (t_1, ….)* *t_trigPosAft (t_1 vbz-vbg-dt)*	token *t_1* is a singular proper noun token *t_1* is a canonical noun (no plurals) POS tag bigram before token *t_1* POS tag bigram after token *t_1* POS tag trigram before token *t_1* POS tag trigram after token *t_1*
Chunking analysis	*ck_hasHead(ck_1, t_1)* *ck_hasType(ck_1, np)* *t_isHeadNP(t_1)* *ck_dist_to_root(ck_n, near)* *t_ck_tag_type(t_1, np)*	*ck_1* has *t_1* as its token head *ck_1* is a nominal chunk *t_1* is the head token of a nominal chunk *ck_n* is near the main verb of the sentence token *t_1* has the chunking type *np*
Semantic features		
Predefined corpus annotation types	*t_type(t_1, person)* *t_subtype(t_1, none)* *t_mtype(t_1, name)*	Token *t_1* has the PERSON type Token *t_1* has no subtype Token *t_1* is a named proper noun
Semantic Resource Mapping	*t_bl_synset(t_id, synset_value)*	Token *t_1* has the WN synset "synset_value"
	t_bl_domain(t_1, domain_value)	Token *t_1* is a term from the "domain_value" domain
	t_bl_sense(t_id, sense_value)	Token *t_1* has the WordNet baseline sense "sense_value"
Synonyms and Hypernyms	*t_bl_synonym(t_id, synonym)* *t_bl_hypernym(t_id, hypernym)*	Token *t_1* has the synonym "synonym" Token *t_1* has the hypernym "hypernym"
Similar words	*t_similiar(t_1, word_stem)*	Token *t_1* has the similar word "word_stem"
Semantic Roles	*t_rs_name(t_id, rs_name)*	Token *t_1* has roleset « name »
	t_rs_arg_name(t_1,t_3, rs_arg_name)	Token *t1* is linked to the *t_3* by the argument type "rs_arg_name"
	t_rs_arg_role(t_1, t_3, arg_role)	Token *t_1* has the semantic role *"arg_role"*
	t_rs_arg_vn(t1, t_3, rs_arg_vn)	Token *t_1* is linked to the *t_3* with a VerbNet role "vn"
Selectional Preferences	*t_sel_pref(t_id, sel_prefs)*	Token *t_1* has the selection pref. "sel_pref"
SuperSenses	*t_bl_supersense(t_id, supersense)*	Token *t_1* has the supersense "supersense"
Ontology Mapping	*t_bl_sumo(t_id, sumo_class)*	Token *t_1* is mapped to a SUMO class "sumo_class"

Structural features	*t_next (t_1, t_2)*	token *t_1* is followed by the token *t_2*
	t_next_head (t_1, t_3)	head token *t_1* is followed by head token *t_3*
	t_prev(t_3, t_2)	token *t_2* precedes *t_3*
	ck_hasToken(ck_1, t_1)	*t_1* is one the tokens in *ck_1*
	ck_hasSucc(ck_1, ck_2)	*ck_1* is followed by the chunk *ck_2*
	ck_prev(ck_2, ck_3)	*ck_2* precedes *ck_3*
	t_hasDep (nn, t_2, t_1)	*t_1* has a multi-word dependency with *t_2*
	t_root (t_n)	*t_n* is the root (main verb) of the dependency tree

* NER was classified here as a lexical feature, but in fact, it is generated from machine learning methods that employ several morpho-syntactic features in their classification.

% Sentence 1

```
st(s_1).
st_hasVoice(s_1, ative).
s_hasChunk(s_1, ck_1).
s_hasChunk(s_1, ck_2).
ck_hasSucc(ck_1, ck_2).
ck_hasSucc(ck_2, ck_3).
...
chunk(ck_1).
ck_hasType(ck_1, pp).
ck_hasTokens(ck_1, t_1).
ck_hasHead(ck_1, t_1).
ck_posRelPred(ck_1, -4).

chunk(ck_2).
ck_hasType(ck_2, np).
ck_hasTokens(ck_2, t_2).
ck_hasHead(ck_2, t_2).
ck_posRelPred(ck_2, -3).
...
```

% Token

```
token(t_2).
t_stem(t_2, 'abc').
t_pos(t_2, nnp).
t_morph_type(t_2, word).
t_ck_hasType(t_2, np).
t_ner(t_2, organization).
t_orth(t_2, uppercase).
t_isHeadNP(t_2).
t_length(t_2, 3).
t_gpos(t_2, nn).
t_bigPosAft(t_2, pos-nnp).
t_trigPosAft(t_2, pos-nnp-nns).
t_hasDep(poss, t_5, t_2).
t_next(t_1, t_2).
...
```

Figure 5.6. Example of some generated BK predicates in Prolog syntax.

5.4.2. User-defined BK

In ILP, the user can specify additional declarative knowledge in order to guide the induction process. The customized rules displayed in Fig. 5.7 were added to the default BK predicates shown in Tab. 5.2. Such rules correspond to two intentional predicates that discretize numerical features, namely *tok_length* and *ck_dist_to_root*.

In Fig. 5.7, the first rule categorizes the token length as *short*, *medium* or *long* size; in the second, the relative chunk position to the main verb of the sentence can be one of the elements of the set {*near, far, very_far*}. Such kind of user-defined predicates intend to enable better rule generalization in OntoILPER learning step.

Finally, two extra structural predicates complement the search in both token and chunk subspaces: *t_prev/2* and *ck_prev/2*, respectively. The former predicate relates one token with its antecessor in a sentence, while the latter relates a given chunk with its antecessor in a sentence. These predicates are declared as intentional predicates which depend on the extensional predicates *t_next/2* and *ck_hasSucc/2*. This illustrates how easy one can declare further background knowledge in ILP (see Fig. 5.7).

```
% Token length type definition
length_type(short). length_type(medium). length_type(long).

tok_length(T, short) :- token(T), t_length(T, X), X  =< 5.
tok_length(T, medium):- token(T), t_length(T, X), X > 5, X =< 15.
tok_length(T, long)  :- token(T), t_length(T, X), X > 15.


%  Chunking distance to the main verb
ck_dist_root(CK, near):-  ck_posRelPred(CK, X), X >= -3, X  =< 3.
ck_dist_root(CK, far) :-  ck_posRelPred(CK, X), ( ( X >= -8, X < -3) ;
                          (X > 3,  X =< 8) ).
ck_dist_root(CK, very_far):-  ck_posRelPred(CK, X),(( X < -8); (X > 8)).

% Previous token - intentional predicate
t_prev(Y, X) :- t_next(X,Y).

% Previous chunk - intentional predicate
ck_prev(Y,X) :- ck_hasSucc(X,Y).
```

Figure 5.7. Specific intentional predicates added to the default BK in OntoILPER.

5.5. Extraction Rule Learning

This section describes the components of the learning module in OntoILPER framework architecture. Here, the focus is on the adaptation of the ILP-based learning algorithms available in GILPS for inducing effective extraction rules. This learning component relies on the ILP predictive setting for inducing extraction rules, or *theories*. The ILP predictive setting consists in using ILP for constructing symbolic classifiers suitable for distinguishing between positive and negative examples. Therefore, it is crucial to fully exploit this supervised machine learning technique when applying it to the problem of entity and relation extraction.

Concerning the rule induction process in OntoILPER, the following restrictions to the properties of the final induced rules are imposed:

i. they have to reflect the BK in terms of both structural and properties features defined by proposed graph-based model introduced in Section 4.4.2
ii. they must be well-formed with respect to the *linkedness* of a variable in a clause, i.e., there is a chain of literals connecting the head input variables to the head output variables [Santos, 2010].
iii. their *qualitative aspects*, expressed by pertinent linguistic patterns easily understandable by a domain expert.

In the remaining of this section, the mode declarations employed by the learning component in OntoILPER are first presented. Such mode declarations are responsible for defining the hypothesis space, i.e., how one can bias the rule induction process for finding the best rules more efficiently. Next, the ILP settings necessary to maximize OntoILPER experimental results that take into account the criteria listed above with respect to the final induced rules are introduced. Furthermore, an illustrative example of the induction rule step in OntoILPER is presented. Finally, the way the final extraction rule model in Prolog is converted to the rule formalism of the Semantic Web is also explained.

5.5.1. Generating Rule Models

The information extraction task addressed in this thesis is cast as a search problem in hypothesis space according to the ILP framework. In this learning scenario, the resulting set of induced hypotheses is composed of several clauses, each one representing patterns that recognizes entity or relation instances. The ILP formalism ensures that a theory (or extraction

rules) can be induced from a set of positive and negative examples described by meaningful background knowledge about the examples.

With this in mind, the ILP-based learning component in the OntoILPER framework architecture is provided with all available knowledge about the training examples represented independently by program clauses. Such kind of example representation is determined by the graph-based model of sentences. At this point, extraction rules can be generated and applied to classify unseen examples.

The ILP-based rule learner in the OntoILPER framework is grounded on the ProGolem engine available in GILPS [Santos, 2010] running on the YAP (version 6) Prolog compiler[9].

During the learning step, the search in the hypothesis space for useful rules that ProGolem has to perform is an expensive task. In fact, an exhaustive search of all possible hypotheses coupled with the coverage test though the entire hypothesis space is unfeasible. Moreover, for finding the best hypothesis from a given example, it is necessary to test each found hypothesis with respect to the positive and negative examples. Usually, this is the most expensive task in the entire inductive process. Therefore, it is necessary to traverse intelligently this space, taking advantage of its particular structure, avoiding to pursue the exploration of partitions in this space that does not contain interesting solutions. Fortunately, the hypothesis space in ILP can be structured by a *partial order relation* (see Section 2.3.3) between two hypotheses which allows an efficient traversal of the hypothesis space. Further biases are also employed not only to reduce the hypothesis space, but also to ensure an efficient induction process.

Among the many types of biases available in ILP, the *Mode Directed Inverse Entailmen*t (MDIE) [Muggleton, 1995] is one of the most used by ILP systems for defining syntactical constraints on the desired hypotheses. MDIE consist of a set of user-defined mode declarations (in Prolog) in conjunction with other settings in order to constrain the search for a good hypothesis.

The ProGolem ILP system employs:

i. *mode declarations* for delimiting and biases the possibly huge hypothesis search space; and,
ii. *specific ILP settings* for modifying its default theory construction process.

The remainder of this section presents the mode declarations and the ILP learning parameter setting utilized in the OntoILPER Framework.

Mode Declarations. Mode declarations characterize the format of a valid hypothesis (rule). They also inform the type, and the input/output modes of the predicate arguments in a rule. There are two types of mode declarations in ProGolem: *head* and *body*.

Mode head declarations (*modeh*) define the *target predicate*, the head of a valid rule that the ILP system has to induce, whereas mode body declarations (*modeb*) determine the literals (or ground atoms), which may appear in body part of the rule. Mode body declarations usually refer to predicates defined in the BK, but they can also refer to the target predicate in the case of recursive theories. Section 2.3.3 explains in more detail the mode declarations in ProGolem.

Mode declarations also impose restrictions on the *types* of the variables used as arguments of a predicate. The notion of types is not very strict here: a type is any identifier assigned by the user to an element in the domain of interest. As different types are treated distinctly, they must be specified for each argument of the predicates appearing in mode declarations. Such types are simply declared by unary Prolog predicates of the form *type(value), i.e., sent(s_1), token(t_1), chunk(ck_1), pos_type(nn),* where *s_1, ck_1,* and *t_1* are identifiers of sentences,

[9] http://www.dcc.fc.up.pt/~vsc/Yap

chunks and tokens, respectively. The third unary predicate, *pos_type(X)* declares the set of allowed POS tag labels of a token.

Tab. 5.2 shows several mode declarations employed by ProGolem in the experimentations discussed in Chapter 6. The first mode declaration in this table specifies the head of the hypothesis to be learned, whereas the second denotes that the literal *t_next (t_id, #lenght_type)* can appear in the body of the theory clauses at most once, for each input token.

Table 5.2. ProGolem´s mode declarations used in OntoILPER framework.

Mode declaration	Description
modeh(1, located(+token, +token))	the target (head) predicate: the relation *located* between two input tokens
Structural features	
modeb(1, t_next(+token, -token))	token chaining
modeb(1, t_next_head(+token, -token))	token chaining between two consecutive head tokens
modeb(1, t_prev(+token, -token))	token chaining (previous token)
modeb(*, t_hasDep(#dep, +token, -token))	typed dependency between two tokens
modeb(1, ck_hasTokens(-chunk, +token))	given a token as input, output the chunk which the token belongs to
modeb(1, ck_hasSucc(+chunk, -chunk))	chunk chaining
modeb(1, ck_pred(+chunk, -chunk))	chunk chaining (predecessor)
Morphological features	
modeb(1, tok_length(+token, #length_type))	token length as one of the values {short, medium, long}
modeb(1, t_stem(+token, #string)	token stemming as an input string
modeb(1, t_orth(+token, #orth))	morphological property of a token as a constant
modeb(1, t_morph_type(+token, #token_type))	morphological category of the token as a constant
modeb(1, t_ner(+token, #ner))	named entity associated to the token as a constant
Syntactic features	
modeb(1, t_pos(+token, #pos))	POS category of a token as a constant
modeb(1, t_gpos(+token, #pos))	POS general category of a token as a constant
modeb(1, t_bigPosBef(+token,#bigposbef)) modeb(1, t_bigPosAft(+token,#bigposaft)) modeb(1, t_trigPosBef(+token,#trigposbef)) modeb(1, t_trigPosAft(+token,#trigposaft))	POS tag bi-grams and trigrams before and after a given token as constants
modeb(1, t_isHeadNP(+token))	head token of a noun chunk
modeb(1, t_isHeadVP(+token))	head token of a verb chunk
modeb(1, t_isHeadPP(+token))	head token of a preposition chunk
modeb(1, ck_hasType(+chunk, #ck_tag))	chunk type as a constant
modeb(1, ck_hasHead(+chunk, #token))	head token of the chunk
modeb(1, t_root(+token)).	Main verb of a sentence as an input token
modeb(1, ck_dist_root(+chunk, #dist_type)).	distance of the chunk to the main verb (root) {near, far, very_far
modeb(1, t_ck_ot(+token,#ck_tag))	chunking tag, one of the values {NP, VP, PP}
Semantic features	
modeb(*, t_similar(+token, #type_similar))	Similar word of the token
modeb(1, t_normalized(+token, #type_normal))	normalized version of the token
modeb(1, t_gaz(+token, #gaz_type))	gazetteer category of the token

modeb(*, t_sel_pref(+token, #type_sel_prefs))	selectional preference of the token
modeb(*, t_bl_domain(+token,#type_domain)) modeb(1, t_bl_sense(+token,#type_sense)) modeb(1, t_bl_sumo(+token,#type_sumo)) modeb(1, t_bl_supersense(+token,#type_super)) modeb(1, t_bl_synset(+token,#type_synset))	domain, WN sense, SUMO class, SuperSense, and synset attributes of a token (baseline sense)
modeb(*, t_wsd_domain(+token,#type_domain)) modeb(1, t_wsd_sense(+token,#type_sense)) modeb(1, t_wds_sumo(+token,#type_sumo)) modeb(1, t_wsd_supersense(+token,#type_super)) modeb(1, t_wsd_synset(+token,#type_synset))	domain, WN sense, SUMO class, SuperSense, and synset attributes of a token (disambiguated sense)

modeb(*, t_rs_id(+token, #type_rs_id))
modeb(*, t_rs_name(+token, #type_rs_name))
modeb(*,t_rs_arg_name(-token, +token, #type_rs_arg_name))
modeb(*, t_rs_arg_role(-token, +token, #type_rs_arg_role))
modeb(*, t_rs_arg_vn(-token, +token, #type_rs_arg_vn)).
	roleset ID, semantic role name, argument type, and VerbNet class associated a given pair of tokens

modeb(1, t_bl_hyper_level_N(+token, #type_hyp)) modeb(1,t_wsd_hyper_level_N(+token,#type_hyp)).	hypernym baseline or disambiguated sense of a given token at level *N*, *N* ={2, 3, 4, 5, 6, 7, 10}

During the learning stage, the mode declarations listed in Tab. 5.2 severely limits the number of potential solutions, reduces overfitting problems, and ensures that only well-formed hypotheses are generated. In the context of the extracting rules learning, a well-formed hypothesis is defined as a clause giving information about the entities, and the words appearing in their contexts.

An illustrative example: the hypothesis *located (A, C) :- t_ner(A, Org), isa_Location(C), t_next(A,B), t_next(B, C)* is a well-formed hypothesis because there exists a semantic restriction on the variables *A* and *C*, and *B* is restricted by the two *next/2* predicates. More precisely, during the learning step, it is desirable to obtain a theory containing only *head output connected clauses* (HOC) [Santos *et al*., 2009]. A definite clause *C* is said to be HOC if its head output variables are instantiated in the body, i.e., if there is a chain of literals connecting the head input variables to the head output variables. For example, consider these predicate signatures *d(+type), c(+type,-type), a(+type,-type), b(+type, -type)*. Then, *c(X ,Y)*←*a(X,Z), b(Z,Y)* is HOC since there is a chain of literals from variable *X* to variable *Y*. Contrarily, *d(X)*←*a(X,Y), b(Z,Y)* is not HOC, because *Z* is an input variable for *b/2*, but it is not connected to the head input variables.

By declaring typed arguments of the predicates and constraining the input/output variables, one guarantees that the clauses generated as hypothesis are at least executable by the Prolog engine. It is worth mentioning that these restrictions also considerably reduce the hypothesis space.

Fig. 5.8 shows an example of the hypothesis space for the relation *located(A, B)*, which orders well-formed candidate hypothesis. Thanks to the way the hypothesis space has been modeled in the proposed solution, in which the great majority of its BK predicates are determinate, almost all candidate hypotheses are also determinate. With such candidate hypotheses, the quasi-order property of the θ-subsumption refinement operator implies that the hypothesis space is structured as a *lattice* [Muggleton, 1995]. At the top of this lattice, the most general clause denotes that all pairs of the domain are arguments of the *located* relation,

whereas the most specific clause, the bottom clause $\perp$ (a clause without constant arguments) contains all literals used for generalizing a given example e_i.

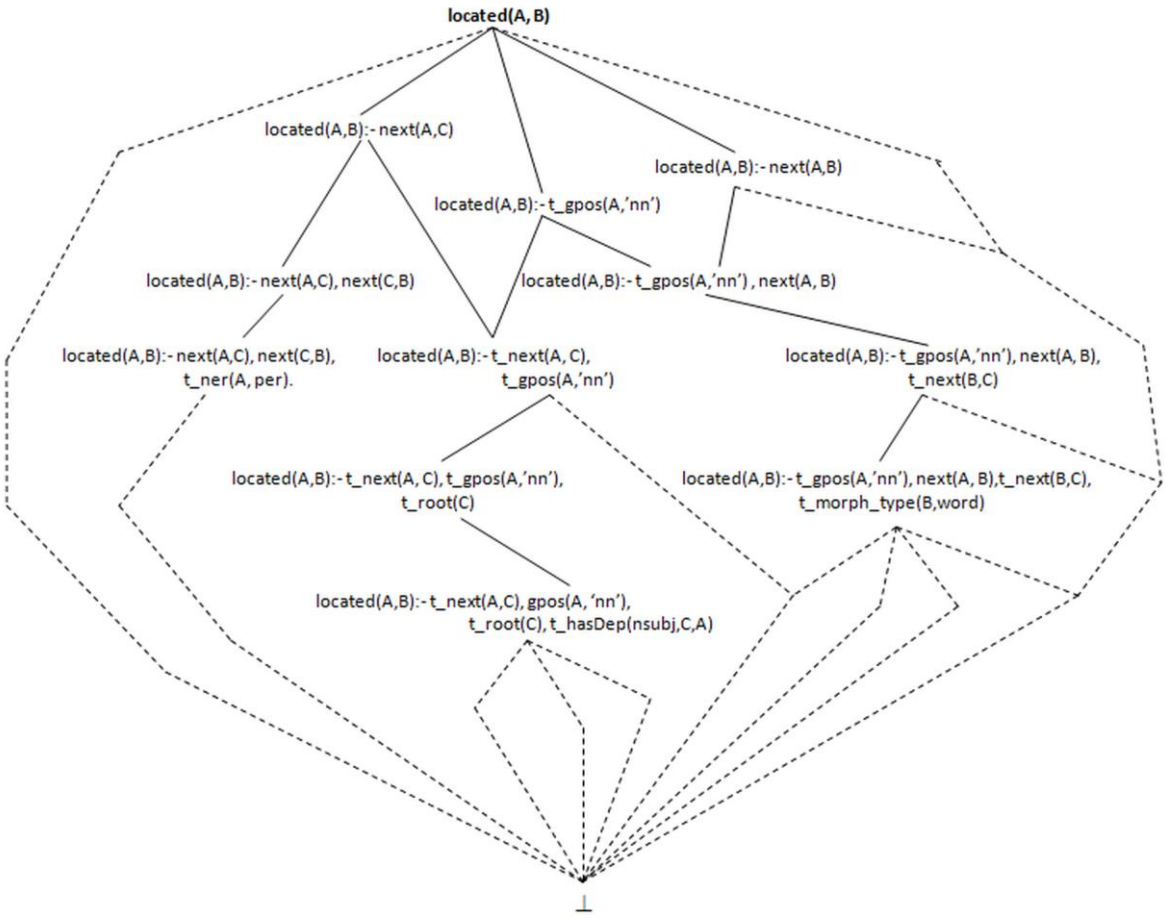


Figure 5.8. Hypothesis lattice structured according to the θ-subsumption for the target relation *located*

With the mode declarations shown in Tab. 5.2, the two first criteria on the induced extraction rules in OntoILPER are met.

The other requirement concerning the qualitative aspects of the rules induced by OntoILPER will be assessed and discussed in the experimental evaluation reported in Chapter 6.

ILP Parameter Settings. In its learning stage, the OntoILPER Framework allows users to customize the learning task by choosing the combination of "layers" of BK (structural, morphosyntactical, syntactic, and semantic) that is more appropriate to users' IE needs. In addition, users may directly intervene in the learning task, by means of the definition of the most important parameters of the ProGolem ILP system integrated into the OntoILPER framework.

ProGolem parameters are tuned for:

- determining the way rules are specialized/generalized, i.e., how the hypotheses space are traversed, top-down or bottom-up manner;
- imposing cardinality constraints on individuals rules or even on the entire learned theory;
- evaluating how good a candidate hypothesis is;

- optimising the rule generation phase,

- defining stop criteria during new rule generation;

- establishing the level of noise tolerance for dealing with noisy input data.

The most important parameters of the ProGolem were selected and summarized in Tab. 5.3. A complete list of ProGolem parameters can be found in [Santos, 2010].

Table 5.3. The most important parameters used in this thesis

Parameter	Description
Clause evaluation heuristic	It defines the clause evaluation engine to use when computing the coverage of a clause. One possible choice is the "left to right" clause evaluation that uses standard Prolog depth-first search heuristics for SLD resolution or another more sophisticated selection heuristic.
Clause length	It delimits or prunes the hypotheses having a maximum number of literals (including the head) of a valid clause.
Evaluation function	This parameter defines which evaluation function to use when scoring a clause.
Variable depth (i)	It allows the user to define the number of layers of new variables to consider during the construction of the bottom clause. In other words, this parameter controls the maxim depth *i* of a variable in a given clause.
Minimum precision	It is a real number between 0 and 1 and it specifies the minimum precision a hypothesis has to have on the training examples to be considered a valid hypothesis.
Minimum number of positive examples	It allows the user to define the minimum number of positive examples a clause has to cover
Noise tolerance	This parameter enables the hypothesis be more tolerant to noisy examples in the training data, since to obtain consistent hypothesis, i.e., covering no negative example, is practically impossible due to typical noisy examples found in training data.
Theory construction mode	It controls the way he theory is constructed after all hypothesis have been generated. ProGolem provides two theory construction modes: *incremental* and *global*. In the former, at each iteration for a given example, the best hypothesis found from an example is included in the final theory, and all the positive examples this hypothesis covers are removed from the training set. In the latter, the final theory is only constructed after all the hypotheses have been generated.

Among all the parameters shown in the Table 5.3, the parameter *depth (i)* is one of the most important in ILP because it directly determines the size of the hypothesis for each example, i.e., the size of the bottom-clause. This was evidenced in the Mikalski´s trains problem example presented in Section 2.3.4. The reader should recall that the target concept of the Mikalski's problem was *eastbound(X)← has_carriage(X, Y), closed(X), short(Y)*. For this problem, with *i = 1*, the target concept is not present in the hypothesis space. This is explained by the fact that none of the predicates *closed/1* and *short/1* are present as well. Indeed, both of these predicates are only available at a variable depth of 2 (*i = 2*) in the bottom clause.

The second most important parameter, the *noise* parameter, is related to a well-known problem in machine learning: real-world databases very often contain *noisy data*, i.e., erroneous or incomplete instances. Other reasons for the presence of noise in datasets include erroneous manual data entry, missing attributes, and inconsistency of human experts' judgments. Particularly in ILP, noisy examples or incompressible noise cannot be explained by non-trivial patterns or rules. Thereby, the noise parameter enables induced hypothesis in ILP to cope with noise in the training data.

As already mentioned, the ProGolem ILP system allows the customization of its induction hypothesis process by tuning some parameters. On the one hand, this ProGolem feature can be seen as an advantage, given that it provides a great flexibility to adapt the different

domains; on the other hand, the optimization task of such parameters can become a new problem itself.

Actually, a major issue in machine learning concerns the influence of different combinations of parameters. This is due to the fact that, in general, optimal classification results cannot be obtained without a properly tuning of its parameters [Hoste, 2002]. Therefore, the optimization of parameters becomes a crucial requirement, not only for ILP, but also for the vast majority of machine learning algorithms. However, manual tuning of parameters for finding high-quality settings of a machine learning algorithm can be very time-consuming, particularly when the learning algorithm involves a large number of parameters [Host, 2002].

Compared with other machine learning algorithms, the choice of parameters in ILP is more intuitive [Santos, 2010]. Consequently, the use of traditional parameter optimization techniques used by other propositional algorithms, such as SVM [Gaspar *et al.*, 2012] and Decision Tree [Host, 2002] is not really necessary. Despite of that, for the sake of accuracy in the experiments, the experimental chapter of this thesis reports the adopted evaluation methodology for performing a systematic search (variation) of a small subset of parameters listed in Table 5.3.

Induced Rules. It has been shown earlier the two possible customizations of the learning task by means of:

- the choice of an appropriate combination of BK predicates in the form of mode declarations which defines a hypothesis space;
- a specific parameter setting that determines how the final theory is learned in ProGolem. After taking such a decision, OntoILPER run the ProGolem ILP system on the top of YAP Prolog for inducing a final theory, or a set of extraction rules for classifying instances of classes and relations. Such extractions rules are actually symbolic classifiers, and they can be applied for classifying new unseen examples.

For illustration purposes, a complete induced theory in OntoILPER for the *part_whole* relation is presented in Fig. 5.9. This theory is composed of two rules using a subset of the complete set of mode declarations listed in Tab. 5.3. The clauses in this theory were evaluated using the scoring function *compression ratio*: *(positive examples - negative examples)/clause length*. The other parameters of interest were: *theory construction* = *global*, *i* = 3, *minimum precision* = 0.0, *minimum positive examples* = 5, and *noise* = 20%, leaving the remaining parameters with their default values.

Rule 1:
#Literals = 4, Positive Score = 90; Negative Score = 1; Precision = 98.9%
part_w(A,B):- t_gpos(A,nn), t_next(A,B), t_subtype(B,state-or-province).

Rule 2:
#Literals = 5, Positive Score = 31; Negative Score = 7; Precision = 77.4%
part_w(A,B):- t_next(A,B), t_pos(A,nnp), t_ne_type(B,gpl), t_subtype(A,pop-center).

Figure 5.9. Complete theory for the part_whole relation in reACE 2004 dataset.

The rules presented in the above theory are justified in terms of *number of literals*, *positive examples covered*, *negative examples covered*, and *rule precision P*:

The first rule classifies an instance of the *Part-Whole* relation. Its high precision (*P = 98.9*) is due to the high number of sentences containing *two adjacent tokens (*or *phrases)* where the first one (*A*) is a *noun*, and the second one (*B*) is tagged with respect to the domain ontology

as an instance of the *"State-or-Provence" subtype* class. This rule highlights that places (*A*) like cities are located, or are part of either a *State* or *Provence*.

The second rule is very similar to the first one, in the sense that the entity instances indicated by the tokens variables *A* and *B* are also adjacent. The token *A* is both a *proper noun* and an instance of the named entity *Geographical Political Location* (*GPL*), whereas the token *B* is mapped to the *Population-Center* subclass in the domain ontology.

5.5.2. Converting Prolog Rules to SWRL Rules

According to [Lisi and Esposito, 2009], the ML community in the last years is increasingly investigating how semantic structures, like ontologies, can improve the performance of classical ML tasks. The present thesis is inserted in this same realm because its main goal consists in investigating how existing Semantic Web knowledge bases and related technologies can be used in conjunction with inductive learning techniques.

In this context, this proposal concerns the important aspect of the type of objects treated by the inductive learning component in OntoILPER framework. More precisely, this thesis argues that the knowledge about the dependencies between instances become part of the overall machine learning model. Hence, the graph-based model presented in Section 4.4 implements this central idea in the form of constraints on this hypothesis space of possible solutions. Moreover, in Section 4.6.3, the idea to transform the Prolog-based BK to SW-type data was motivated, which constitutes the Annotation Ontology in OWL/DL.

In this respect, aiming at providing a feasible implementation of an OBIE system that allows both the linguistic and domain knowledge (represented as ontologies) to be employed by automated reasoning inference engines, OntoILPER performs the conversion of final induced Prolog rules to their counterparts *SWRL rules*. This conversion is persisted into the merged ontology enabling direct access to the inference services provided by OWL/DL reasoners like Pellet[10] and Hermit[11] in Protégé Ontology Editor.

Among many rule languages already proposed by the SW community, SWRL was adopted due to its distinguishing features:

(i) It is particularly well adapted to the extraction rules generated in the OntoILPER framework because OWL expressions in SWRL are based on classes and relations (properties). This SWRL feature was also explored in [Fiorentini *et al*., 2010] which used SWRL to provide rules to associate instances to new classes, and to create properties between instances, as it was done in OntoILPER.

(ii) Its syntax is very similar to Prolog, which means that the parsing effort for conversion is reduced. Actually, SWRL extends OWL/DL with standard first-order semantics under the form of Horn-style rules. In addition, SWRL rules present a human readable syntax, like the simple form of Horn-style rules.

(iii) SWRL rules are supported by some of the most prominent and powerful OWL/DL reasoning engines, specially the Pellet inference engine which has demonstrated state-of-the-art performance on instance classification [Bock *et al*., 2008] compared to other similar engines. OntoILPER relies on the Pellet reasoner to classify new instances resulting from the application of SWRL rules upon specific concepts.

In short, the rule conversion to SWRL brings the advantages that rules in SWRL align semantically with the ontological elements in the working ontologies in a very natural way.

[10] Pellet., OWL 2 Reasoner for Java. http://clarkparsia.com/pellet

[11] Hermit OWL Reasoner. http://hermit-reasoner.com

As a result, the rules induced by OntoILPER contain unary and binary predicates that are converted to their semantically equivalents in SWRL. For instance, the Prolog unary literal *t_token(A)* is translated as a class atom in SWRL: *Token(?a)*, whereas a binary literal in Prolog *t_pos(A,nn)* is converted to the equivalent binary atom *hasPOS(?a, nn)* in SWRL. This last example concerns the conversion of the Prolog ternary predicate *t_dep/3* shown in Tab. 5.4 to its semantically equivalent SWRL binary atom *hasDepXXX*, where the third argument of the former predicate is used to compose the name of the latter as an object property, i.e., *hasDepSubj(?a, ?b).*

Table 5.4. Example of Prolog predicate conversion to SRWL atoms.

Prolog Term	SWRL Term
t_token(A)	Token(?a)
t_pos(A, nn)	hasPOS(?a, nn)
t_next(A, B)	hasNext(?a, ?b)
t_dep(A, B, **subj**)	hasDep**Subj**(?a, ?b)

In what follows, the theory presented earlier in Section 5.5.1 expressed as rules in SWRL is shown. One can note that the rules in Prolog and SWRL differ little syntactically: in Prolog, the head of the rule comes before the consequent, whereas in SRWL, this order is reversed. Other minor differences such as the symbol '?' before the variable names are presented in Fig. 5.10.

Rule 1
hasPOS(?a, nn), hasNext(?a, ?b), hasSubType(?b, state-or-province) -> part_whole*(?a, ?b)*

Rule 2
hasNext(?a, ?b), hasPOS(?a, nnp), hasType(?b, gpl), hasSubType(?b, population-center)
-> *part_whole(?a, ?b)*

Figure 5.10. Theory presented in Fig. 5.9 converted to rules in SWRL.

During the conversion from Prolog rules to its corresponding in SWRL entails an important aspect with respect to the differences between these two rule formalisms. This aspect is discussed in more detail in Appendix F.

5.6. Graph Conversion

The remainder part of this chapter will address the modules of the OntoILPER framework architecture that concerns the application mode

It is worth noting that both the training and application mode operations of OntoILPER share the first two modules, i.e., Text Preprocessing and Sentence Representation and Simplification (see Fig. 5.1). These two modules were already introduced in Sections 5.2 e 5.3, respectively. Thus, the description of the *Graph Convertion* module is presented next.

The Graph Conversion module receives the graphs simplified by the previous module responsible for representing and simplifying the annotated sentences of the input corpus.

Remember that the Annotation Ontology encompasses not only the elements derived from the NLP annotation process, but also all linguistic-related entity types present in the annotated corpus, such as typed dependencies hierarchies, chunk types, POS tags, NER types, etc. Its primary goal consists in formalizing all BK knowledge derived from the previous step, i.e., the Sentence Representation and Simplification step.

At this point, in the OntoILPER Framework processing, one carried out a straightforward conversion between the elements of the graphs and their corresponding ontological elements in the Annotation Ontology. This conversion process is illustrated in Fig. 5.11.

First, each graph's node denoting the words or tokens in the sentence is converted as an instance of the *Token* class in the Annotation Ontology. All attribute values of the nodes are also retrieved for generating the assertional axioms that will describe each instance (document, sentence, chunking, tokens, etc.) in the Annotation Ontology. The conversion process is analogous for each document, sentence, and chunking represented as nodes in the graph.

Then, each graph edge connecting two nodes is converted to its corresponding object properties in the Annotation Ontology. Here, range and domain restrictions of such object properties are imposed during this process. This is an important step towards ontology consistency. Part (a) of Fig. 5.11 illustrates the retrieved graph elements just before the conversion step takes place.

Part (b) of Fig. 5.11 shows the ontological elements of the Annotation Ontology in Functional Notation[12] for ontologies in OWL/DL. The reader should note that, in reality, this last step of the conversion actually adds assertional axioms into the Annotation Ontology.

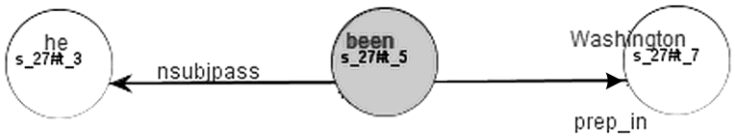
At this point in this in the conversion process, the Annotation Ontology contains all converted graphs, what constitutes a formal knowledge base. Furthermore, it represents and formalizes the set of richly annotated documents in the OntoILPER Framework.

5.7. Domain Ontology Population

In the OntoILPER Framework, the learned classification model, i. e., the extraction rules are kept in the same ontology that formally captures the content information of the input documents. As seen before, after several processing steps, the input documents are finally converted to instances in the Annotation Ontology. This means that the domain and the annotation ontologies, in conjunction with the learned rules are expressed in the same reasoner-readable language. This unified view of these ontologies (or knowledge bases) along with the extraction rules in SWRL, allows for inferences with the help of SW reasoners, like Pellet or Hermit.

[12] http://www.w3.org/TR/owl2-syntax

Input graph:



Parsing step

Nodes of the graph (Tokens and attributes)

token id= t_3 token stem = 'he' token pos = PRP token type = word token chunking type = NP isHeadNP = true token length = 2 token ne type = PERSON ...	token id= t_5 token stem = 'been' token pos = VBZ token type = word token chunking type = VP isHeadVP = true token length = 4 ...	token id= t_7 token stem = 'washington' token pos = NNP token type = word token chunking type = NP isHeadNP = true token length = 10 token ne type = LOCATION ...

+

Edges of the graph (typed dependencies)

(a)

nsubj ('been' , 'he')
prep_in ('been' , 'washington')

Ontology Population (OWL API)

Classes instances with their data property values

ClassAssertion(:Token :t_3) DataPropertyAssertion(:hasStem :t_3 "he") ObjectPropertyAssertion(:hasPOS :t_3 :prp) ObjectPropertyAssertion(:hasType :t_3 :word) ObjectPropertyAssertion(:hasCkType :t_3 :np) DataPropertyAssertion(:isHeadNP :t_3 true) DataPropertyAssertion(:hasLength :t_3 2) ObjectPropertyAssertion(:hasNE :t_3 :person)	ClassAssertion(:Token :t_5) DataPropertyAssertion(:hasStem :t_5 "been") ObjectPropertyAssertion(:hasPOS :t_5 :vbz) ObjectPropertyAssertion(:hasType :t_5 :word) ObjectPropertyAssertion(:hasCkType :t_5 :vp) DataPropertyAssertion(:isHeadVP :t_5 true) DataPropertyAssertion(:hasLength :t_5 4)	ClassAssertion(:Token :t_7) DataPropertyAssertion(:hasStem :t_7 "washington") ObjectPropertyAssertion(:hasPOS :t_7 :nnp) ObjectPropertyAssertion(:hasType :t_7 :word) ObjectPropertyAssertion(:hasCkType :t_7 :np) DataPropertyAssertion(:isHeadNP :t_7 true) DataPropertyAssertion(:hasLength :t_7 10) ObjectPropertyAssertion(:hasNE :t_7 :location)

(b)

+

Object Property values

ObjectPropertyAssertion(:hasNsubj :t_5 :t_3)
ObjectPropertyAssertion(:hasPrepIn :t_5 :t_7)

Figure 5.11. Example of converting a graph as instances of the Annotation Ontology.

```
Person("Myron Kandel")                    // "Myron Kandel" is an instance of the Person class
Location("New York")                      // "New York is an instance of the Location Class
is_located("Myron Kandel", "New York").   // "Myron Kandel is located in New York.
```

Figure 5.12. Extracted instances of classes and relation in OntoILPER.

The Instance Extraction step in the OntoILPER Framework is responsible for applying the SWRL rules over the knowledge base in order to classify both instances of classes and relations. Such instances will be finally integrated into the domain ontology. For instance, considering the sentence "*Myron Kandel at the Newsdesk CNNfn in New York*", the obtained extraction rules would be able to classify the following instances (Fig. 5.12):

In the OntoILPER framework, any OWL/DL reasoner with support to SWRL can be used to apply the extraction rules over the knowledge base determined by the annotation ontology, resulting in a set of extracted instances. This thesis is aware to the fact that the same extracted instances could have been obtained by a direct application of the induced Prolog rules on the Prolog factual database. However, it is a design decision to utilize SW reasoners instead of the default Prolog reasoning services because the reasoners are endorsed with some capabilities particularly interesting to the purposes of this thesis. First, they provide reasoning tasks such as class *satisfatibility*, and class *subsumption*. Second, they can check whether all the statements and definitions in the ontology are mutually consistent. Finally, and more importantly, they can provide explanations of their inference processes. In fact, the Protégé Ontology editor conveniently offers all the aforementioned inference services by means of a human-friendly program interface.

Moreover, it is argued that it is very advantageous to have the knowledge base and rules in a standard reasoner-readable format, due to the flexibility of not depending upon a specific extraction tool or process. Another criterion for this decision is that the integration of the knowledge base in OWL/DL and rules in SWRL constitutes an attempt to contribute to the SW field by promoting *shareable ontologies*. It is also expected that future advances on how to encode knowledge bases and rules would bring new perspectives to the application of the OntoILPER framework and, particularly, the annotation ontology.

Scenarios for the Application of the OntoILPER Framework.

In the following, three of the possible scenarios for applying the OntoILPER Framework are described:

i. *SW Applications.* In a SW application scenario, many alternative SW technologies for storing, manipulating and querying the instances extracted by OntoILPER can be effectively employed, including Sesame[13], Jena[14], SPARQL[15], to mention a few. In this scenario, users could specify their information demand by querying the working ontologies (Domain and Annotation ontologies) using SPARQL.

ii. *Ontology Population.* The extracted instances found by OntoILPER are added as new instances of the input domain ontology. This process is also known as Ontology Population. Thus, every OBIE system can be regarded as an OP system, as it can be extended to assimilate extracted instances into the ontology. It is worth mentioning that, due to the OWA, instance classification in practice will never return all the individuals that theoretically belong to a given class, but only those named individuals whose class membership can be inferred. Particularly in the OntoILPER framework, the reasoning

[13] OpenRDF Sesame. http://www.openrdf.org
[14] Apache Jena. http://jena.apache.org
[15] SPARQL Query Language for RDF. http://www.w3.org/TR/rdf-sparql-query

and the ontology population services are performed by using OWL API[16] and Pellet API [Sirin *et al*., 2007], respectively.

iii. *Ontology Engineering*. The final extraction rules induced by the OntoILPER Framework in conjunction with other TBox axioms from the domain ontology can be refined and combined by an ontology engineer for discovering other potential useful concepts.

As already seen, the main goal of the ILP-based learning component in OntoILPER is to automatically induce first order rules suitable for extracting instances of classes and relations. Such rules are properly converted to their equivalent in SWRL, which enables the enrichment of the annotation ontology.

A major benefit from this enrichment is that, for instance, the ontologist or domain expert can define complex concepts by manipulating the extraction rules. Besides that, she can use the OWL/DL constructs in the definition of new classes/subclasses in the domain ontology, as well.

It is finally illustrated how the ontological elements (extraction rules and instances) can be handled in the Protégé ontology editor.

Fig. 5.13 shows domain and the annotation ontology as they appear in the Protégé Ontology Editor. At the left side of this figure, one can see both ontologies (domain and annotation) sharing the same environment in Protégé. The central part of the figure list Token class instances, while the right side show its attributes, as object properties and data properties. This same figure shows the state of the ontologies just before running the Pellet reasoner, whereas Fig. 5.14 shows the ontology after activating the reasoner. One can notice the new relation instance appearing in the Property Object window (right side of the figure), in which one of the applied rule found the *Located*(t_2, t_3) relation, i.e., there is a Located relation between the tokens t_2 an t_3 in the annotation ontology.

Finally, Fig. 5.15 shows the explanation of the result. This inference explanation highlights the premises matched by the rule, and its related conclusions.

5.8. Conclusion

This chapter presented the main modules and components that constitute the OntoILPER Framework. This OBIE framework is one of the contributions of this thesis.

First, a broad view of the OntoILPER framework by indicating its two operation modes: learning and application is provided. After the description of each module and its implemented components, this chapter discussed three possible application scenarios for OntoILPER as a comprehensive OBIE framework for providing extraction services to Semantic Web applications. In one of these application scenarios, the OntoILPER framework can be used for ontology engineering purposes, in which an expert benefits from the extraction rules, discussed here as SWRL rules, as building blocks for creating complex concepts as the combination of basic ones.

Next chapter reports a comprehensive assessment of the OntoILPER framework in two distinct domains: news and biomedical. In fact, several experimental results are reported and discussed, aiming at testing the working hypothesis introduced in Chap. 1.

[16] The OWL API. http://owlapi.sourceforge.net

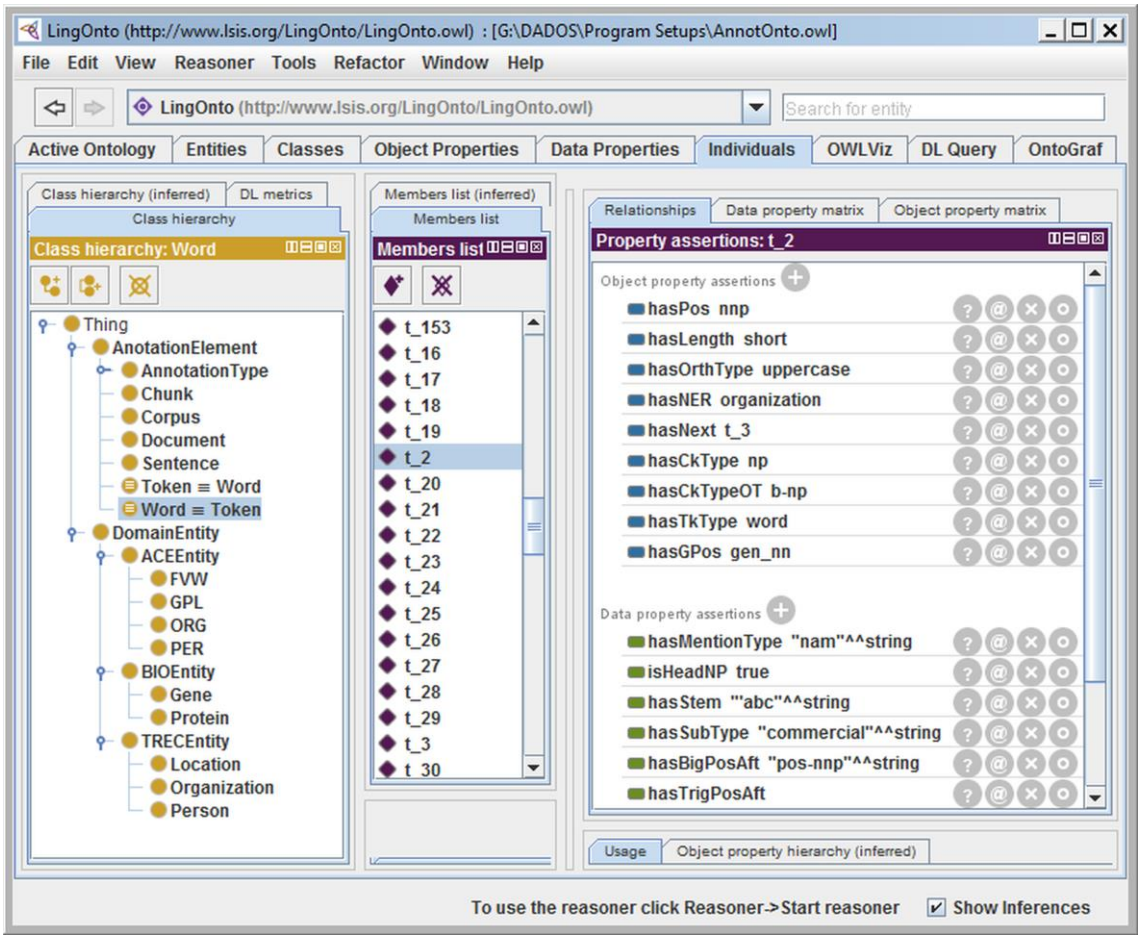


Fig. 5.13. Domain and Annotation ontologies merged by the Protégé ontology editor, before running the reasoner.

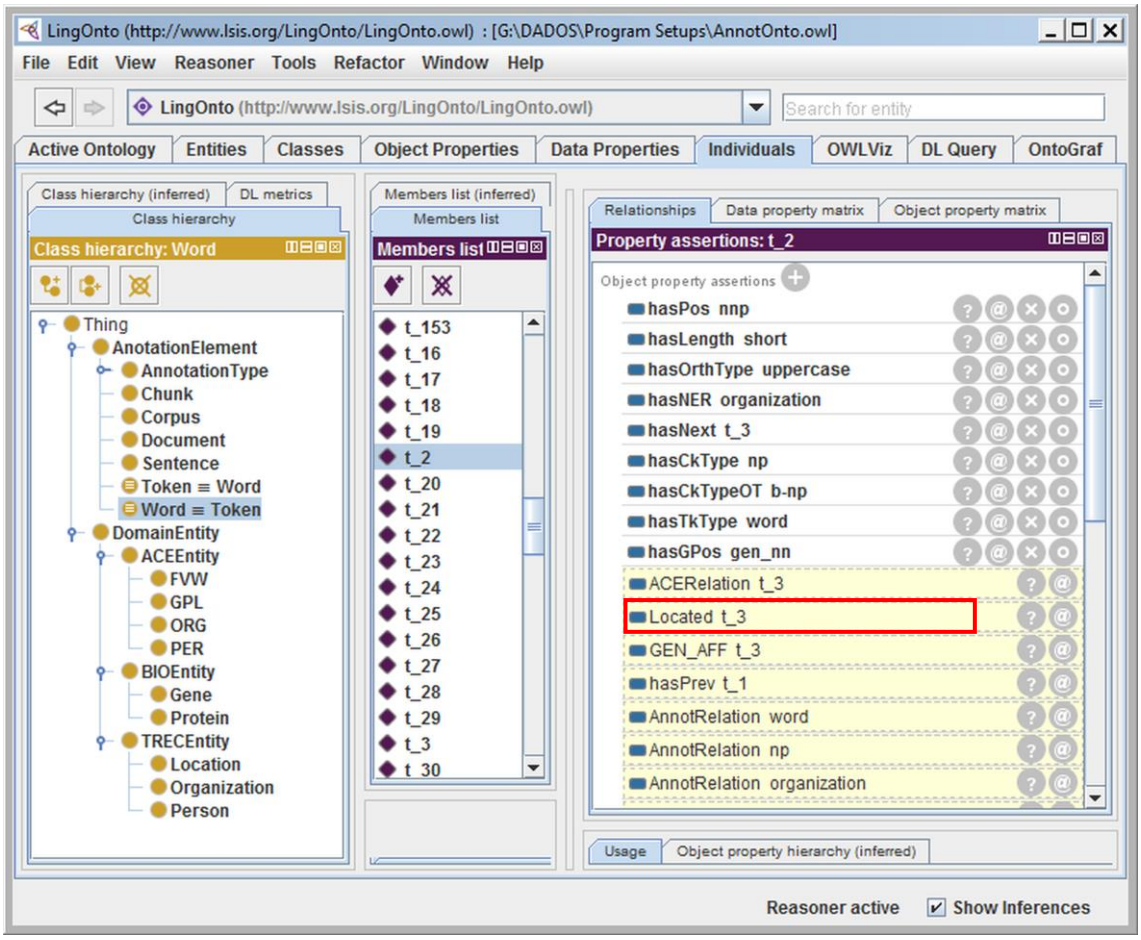


Fig. 5.14. Domain and Annotation ontologies merged by the Protégé ontology editor, after running the reasoner. The new classified relation instance *Located(t_2, t_3)* is highlighted.

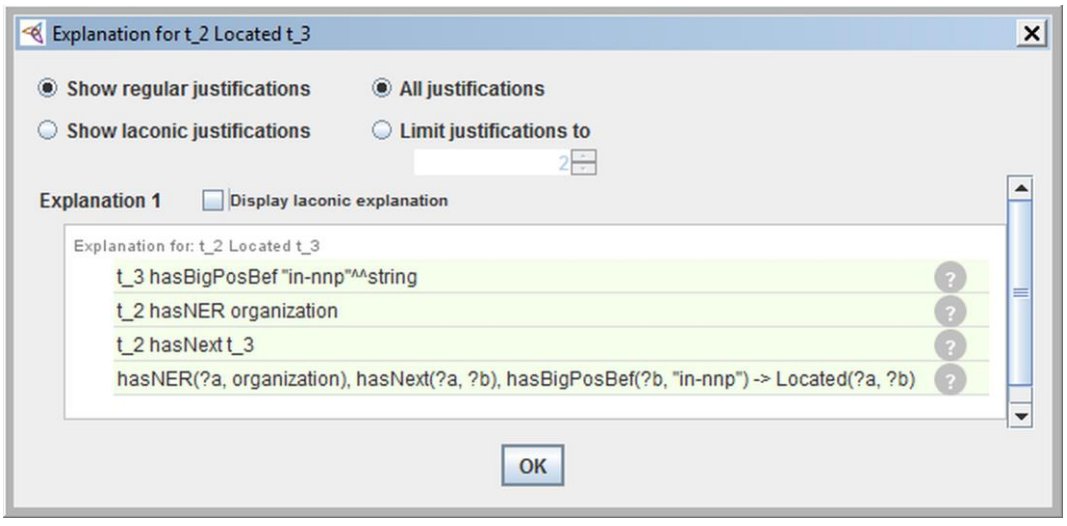


Fig. 5.15. Explanation of the *Located* rule results in Protégé.

Chapter 6

Experimental Evaluation

This chapter reports and discusses the experimental results on entity and relation extraction achieved by OntoILPER, addressing the specific problem formulations related to NER and RE in this thesis. The objective of the experiments is to investigate the effectiveness of OntoILPER according to the following *experimental questions* that drives the proposed assessment methodology:

1. What are the optimal parameters to be employed in OntoILPER experimentations?
2. What is the contribution of relational features on the final induced rules?
3. What is the influence of semantic- and ontological-related features on RE?
4. Which combination of linguistic-based background knowledge is best?
5. Do the features present a complementary contribution on the performance results?
6. In the OBIE scenario, in particular, what is the influence of ontological hierarchical information on the results?
7. How well do the rules generalize between different datasets in the same domain?
8. What are the most important qualitative aspects of the final induced rules?
9. What is the impact of using named-entities labels predicted in a previous learning phase of OntoILPER on a second learning iteration for relation predictions?
10. How does OntoILPER compare to state-of-the-art in the literature for NER and RE?
11. Can overfitting be alleviated by a previous simplification of the graphs representing the sentences and examples?
12. Can OntoILPER obtain comparable performance with respect to other RE approaches without using the named entities labels provided by the corpus?

In what follows, the adopted evaluation methodology is presented, which includes the corpora used, and the evaluation metrics adopted. The answers to above mentioned experimental questions are provided in the remainder of this chapter.

6.1. Evaluation Methodology and Experimental Settings

This section presents the evaluation methodology as well as the experimental settings adopted during the conduction of the assessment of OntoILPER in the NER and RE tasks.

6.1.1. ReACE, TREC, and PPI Corpora

This section describes the corpora used in the experimental evaluation of this thesis, namely:

- the *reACE 2004/2005* datasets [Hachey *et al*., 2011] for relation detection and characterization;
- the *TREC* dataset for NER and RE proposed by [Roth and Yih, 2004];

- Protein-Protein Interaction corpora from the biomedical domain [Pyysalo *et al*., 2007].

Datasets for Relation Extraction (reACE) [Hachey *et al*., 2011]. The Automatic Content Extraction (ACE) programme defines the task of Relation Detection and Categorization (RDC) which aims at detecting and classifying relations between entities according to a predefined ontology. These datasets consist of text from newswire and broadcast news taken in two consecutive years. There are 348 and 298 documents in ACE 2004[1] and ACE 2005[2], respectively. The data in both datasets are distinct, i.e., there is no overlap between them.

Contrarily to previous evaluations, these datasets introduced a type and subtype hierarchy to both entities and relations, providing a crucial step towards OBIE [Wang *et al*., 2005] which makes the task more challenging. In both datasets, an entity instance is called an *entity mention*, which can be a name, a proper noun, a quantified nominal, or a pronoun.

The ACE 2004 dataset defines a two level entity hierarchy consisting of 7 types and 44 subtypes; whereas, for relations, it is proposed a hierarchy with 7 types and 22 subtypes. the ACE 2005 dataset, in turn, shares the same entities types and most of the relations types and subtypes.

Aiming at evaluating some of the hypotheses raised up in this thesis, the revised versions of the ACE 2004/2005 datasets [Hachey *et al*., 2011] were chosen. These datasets, also known as *reACE* datasets, are the result of several transformation steps (refactoring, preprocessing, and reannotation) [Hachey *et al*., 2011] which normalized the two original ACE datasets so that they adhere to a common notion of relation that is more intuitive and simple, as defined by Hachey (dataset for RE): a *relation mention*, or relation instance in the context of this work, is a predicate over two arguments, where the arguments represent concepts in the real world. A predicate can describe the type of association or interaction between the things represented by the arguments. Other positive aspect of the transformations for deriving the reACE version is that: (i) it facilitates the evaluation and tuning of machine learning algorithms addressing the RE problem; (ii) it enables both comparative and cross learning evaluations between both versions of the ACE dataset (2004 and 2005).

The relations instances in reACE are required to be exactly between two entities in the same sentence. Here, all pairs of entity instances occurring in the same sentence are considered as *candidate relation instances*. Such a restriction offers some advantageous gains: (i) enforces consistency across datasets; (ii) allows a principled and tractable definition of the relation extraction task; (iii) removes relations between reflexive arguments, and embedded (overlapping) entities as well [Hachey *et al*., 2011]. As a result, the data in reACE is normalised such that relations instances are between named or pronominal entities wherever possible.

Tab. 6.1 parts *(a)* and *(b)* show, respectively, the type/subtype distributions and some examples of the relations in the final reACE2004/2005 datasets. The entity types in both reACE datasets were also refactored to 4 types, namely: PER (person), ORG (Organization), GPL (Geo-Political/Location), and FVW (Facility/Vehicle/Weapon).

Tab. 6.2 shows some instances of relations subtypes found in the reACE 2004 corpus.

The reACE datasets, corresponding to the ACE 2004/2005 datasets, fully comply with the work assumptions introduced in Section 4.1, and for that reason, they are used in the experiments instead of the original versions of the ACE 2004/2005 datasets. Moreover,

[1] http://www.itl.nist.gov/iad/mig/tests/ace/2004/doc/ace04-evalplan-v7.pdf
[2] http://www.itl.nist.gov/iad/mig/tests/ace/2005/doc/ace05-evalplan.v2a.pdf

relations types/subtypes from both datasets with a number of examples less than or equal to 15 are considered to be outliers and are then filtered out. [Hachey *et al.*, 2011] gives more detail about the entire reannotation process. The reACE datasets is available for redistribution through the Linguistic Data Consortium (LDC)[3].

Table 6.1. Relation distribution of the reACE 2004/2005 datasets.

reACE 2004 (a)	
Relation **Type**/Subtype	Freq
Employee-Membership-Subsidiary (EMP_ORG)	
Employee-Staff	303
Employee-Executive	220
Member-of-Group	80
Employ-Undetermined	13
Partner	3
General-Affiliation (GEN_AFF)	
Located	352
Citizen-Resident-Religion-Ethnic	98
Part-Whole (PRT_WHOLE)	
Part-Whole	174
Subsidiary	100
Personal-Social (PER_SOC)	
Business	35
Family	15
Agent-Artifact	
User-Owner-Inventor-Manufact	6
Total	**1399**

reACE 2005 (b)	
Relation **Type**/Subtype	Freq
Organization-Affiliation (ORG_AFF)	
Employment	228
Membership	36
Sports-Affiliation	14
Founder	8
Investor-Shareholder	7
Ownership	3
Student-Alumnus	3
General-Affiliation (GEN_AFF)	
Located	280
Citizen-Resident-Religion-Ethnic	39
Part-Whole (PRT_WHOLE)	
Geographical	119
Subsidiary	47
Personal-Social (PER_SOC)	
Business	16
Family	42
Lasting-Personal	10
Agent-Artifact	
User-Owner-Inventor-Manufact	15
Total	**867**

Table 6.2. Some examples of the reACE 2004 relations

Relations: *type.subtype(arg1, arg2)*	*Sentences or phrase examples*
PER-SOC.business(John, superiors)	*John´s superiors ...*
EMP-ORG.employ-exec(Investors, Wall Street)	*Investors on Wall Street...*
EMP-ORG.employ-staff(ABC, John Martin)	*Here's ABC's John Martin.*
GPE-AFF.citizen/resident(voters, Missouri)	*Some Missouri voters ...*

TREC Dataset [Roth and Yih, 2004**].** The experiments reported in this section are based on the Text Retrieval Conference (TREC) dataset[4], which is composed of articles from the WSJ (Wall Street Journal). This dataset has been annotated for named entities and relations, containing 1,441 sentences with 5,349 entities, namely, 1,691 people, 1,968 locations, 984 organizations, and 706 miscellaneous names. Each one of the 1,441 sentences has at least one active relation. Among those sentences, there are 19,080 possible binary relations with the frequency distribution of the positive ones as shown in Tab. 6.3. This table also shows an example of each relation and the constraints with respect to its two arguments. Most of the candidate binary relations have no active relations at all; this results in an unbalanced distribution between positive and negative examples. Fig. 6.1 depicts the domain ontology created for storing the instances extracted in OntoILPER.

[3] https://www.ldc.upenn.edu/
[4] http://cogcomp.cs.illinois.edu/Data/ER/conll04.corp

Table 6.3. Binary relations and their arguments types

Relation	Arg-1	Arg-2	Example	# of relations
located_in	LOC	LOC	(Toledo, Ohio)	405
work_for	PER	ORG	(Winter, Court)	401
orgBased_in	ORG	LOC	(HP, Palo Alto)	452
live_in	PER	LOC	(Tvazir, Israel)	521
kill	PER	PER	(Oswald, JFK)	268

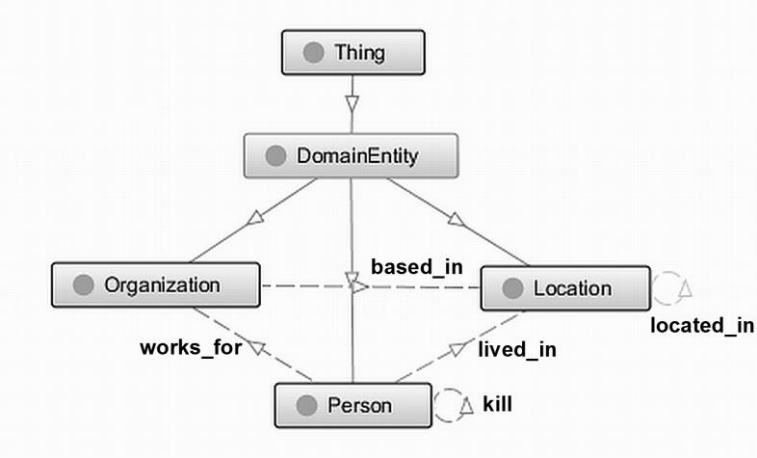


Figure 6.1. TREC underlying domain ontology with entities and relations.

Biomedical Corpora.

Three standard PPI datasets on biomedical domain were selected.
These corpora are considered more complex than normal English text used in news domain.

- *Learning Language in Logic (LLL)* [Nedellec, 2005]. This dataset was proposed to the genic interaction task from a set of sentences concerning Bacillus subtilis transcription.
- *HPRD50* [Fundel *et al.*, 2007]. It consists of a randomly selected subset of 50 abstracts referenced by the Human Protein Reference Database (HPRD).
- *Interaction Extraction Performance Assessment (IEPA)* [Ding *et al.*, 2002]. It consists of a corpus containing 303 abstracts from PubMed, each containing a specific pair of co-occurring chemicals.

Tab. 6.4 summarizes the basic statistics on these datasets.

Table 6.4**.** Basic statistics of three corpora for RE

Corpus	#Sentences	#E+	#E-
LLL	77	164	166
HPRD50	145	163	270
IEPA	486	335	482

6.1.2. One vs. All Learning Technique and Automatic Generation of Negative Examples

In GILPS, the task of inducing target predicates requires that positive and negative examples be explicitly indicated before the generation of the classification model. Thus, negative examples are artificially created as the complement of the positive ones, according to the *one vs. all class binarization* technique when building the rule models. In short, the underlying idea of the one vs. all strategy consists in producing several 2-class datasets by discriminating each class against the union of all the other classes. Thus, given the set of N possible entity classes C_i, $i = 1..N$, for each positive instance c_i of a given class C_i in the training set, a negative example is created for each one of the other $N - 1$ classes. Thereby, a multiclass learning problem is reduced to several binary classification problems, one for each class.

On the other hand, for building rule models for relations, the step of generating negative examples is somewhat different. For that, the same technique as proposed in [Airola, *et al*., 2008] was adopted, in which the RE extraction task is regarded as a binary classification problem, where interacting argument pairs are positive examples, and the other pairs of co-occurring entities in the same sentence are negative ones. As a result, for each sentence, $C_{n,2} = n! / 2*(n - 2)!$ examples are generated, where n is the total number entities in a given sentence.

6.1.3. Evaluation Metrics, Cross-Validation, and Statistical Significance Test

Aiming at assessing the effectiveness of the proposed approach, several experiments on the TREC dataset were conducted. The performance evaluation is based on the IR classical measures, i.e., Precision *P*, Recall *R*, and *F1*-measure [Baeza-Yates and Ribeiro-Neto, 1999]. Besides that, Airola *et al*. (2008) suggest to use the Area Under Curve measure.

Area Under Curve. In the evaluations, the F1-measure, used in most of previous work in RE, is fairly sensitive to ratio of the underlying positive/negative distribution of examples in a training set. Thus, as an alternative to F1-measure, some authors have proposed to use the *Area Under the Receiver Operating Characteristics Curve (AUC)* [Hanley and Mcneil, 1982] as a performance measure for classification systems.

One of the most important properties of the AUC measure is that it is invariant to the class distribution in a training set of examples. Because of that and other beneficial properties, especially for comparative evaluation, the usage of AUC for performance evaluation has been recently advocated in the machine learning community [Airola, 2008] [Tikk *et al*., 2010].

The AUC is defined as

$$AUC = \frac{\sum_{i=1}^{m_+} \sum_{j=1}^{m_-} H(x_i - y_j)}{m_+ . m_-},$$

where m_+ and m_- are the numbers of positive and negative examples, respectively, and $x_1,...,x_{m+}$ are outputs of the system for the positive, and $y_1,...,y_{m-}$ for the negative examples, and

$$H(r) = \begin{cases} 1, \text{if } r > 0 \\ 0.5, \text{ if } r = 0 \\ 0, \text{otherwise} \end{cases}$$

Cross-Validation (CV). A well-known method of measuring a classifier´s performance is in terms of the error rate. The classifier prediction of a given instance, for each class, may

be correct, which is counted as a success; or it is an error. Thus, the error rate is just the proportions of errors made over a whole set of instances, and it can be used for measuring the overall performance of the classifier. The standard way of prediction the error rate of a machine learning technique, having a limited amount of data available, consists in use a *holdout* procedure where part of the data is used for training the classifier, and the remainder part, for testing it.

As all datasets chosen for the experimental evaluation have a limited amount of data for training and testing, a simple variant form of the holdout procedure was chosen because of its important statistical basis: the *cross-validation* technique. In cross-validation, one must decide a fixed number *n* of folds, or partitions, of the available data. Then, the data is split into *n* partitions of approximately the same size: 1 partition is used in turn for testing, while the other *n-1* partitions are used for training.

Previous work on extensive tests using several machine learning techniques where conducted on many datasets of different kinds using the 10-fold cross validation. The results suggested that 10 was the right number of folds to obtain the best estimate of the error rate [Witten and Frank, 2005]. However, these results are by no means conclusive, and debates continue to appear in the machine learning and data mining community about what would be the best technique for evaluation.

Depending upon the corpora evaluated, the experiments reported in this thesis will adopt either 5- or 10-fold cross validation. This fairly guarantees the maximal use of the available data, and allows comparison with relevant related work.

Statistical Significance Testing. Statistical significance tests in all comparative evaluation in this work are used, either *intrinsic* (between different models using the same algorithm) or *extrinsic* (between different classification algorithms or systems).

An example of the former test is the *paired t-Student* test [Witten and Frank, 2005]. Here, it is assumed that difference between performance scores follows a normal distribution, which is the case for all experiments evolving different induced models, but just varying some parameters of the classifiers. The latter, the Wilcoxon *signed- rank* test [Cooligam, 2004] enables to check for significant differences between the average performance of two *distinct* algorithms or classification systems. This is a non-parametric test analogue of the paired t-Student test. This test ranks the absolute value of the differences observed in performance of the pair of algorithms. Ties are discarded and the ranks are then given signs depending on whether the performance of the first algorithm is higher or lower than that of the second one. If the null hypothesis holds, the sum of the signed ranks should be approximately equal to zero.

It is worth mentioning that the comparison of between two distinct classification algorithms using the Wilcoxon test is equivalent to determining if the AUC of the algorithms differ significantly [Hand, 1997].

In both significance tests, the *null hypothesis* is that the two populations from which the scores are sampled are *identical*. Usually, the difference between means is taken, e.g., $\mu_1 \neq \mu_2$. Following convention, the null hypothesis is rejected for values of *p* less than or equal 0.05, which corresponds to a 95% of confidence interval for the difference of the two means μ_1, and μ_2.

6.1.4. Measure for Assessing the Generalization Degree of Theories

The *theory ratio* is defined as the measure of the generalization degree of the learned theories in the context of this work.

This measure is defined by:

$$\text{Theory Ratio } (TR) = \frac{\text{number of clauses in the final theory}}{\text{number of positive examples in the training set}}$$

TR measures the quality of the final theories, along with the typical measures of *precision*, *recall*, and *F-measure*, which produces a quantifiable measure of its performance on a given dataset.

Note that since each rule must cover at least one positive example, the number of rules learned for a given training set of examples will always be less than or equal to the total number of positive examples, and, as a result, the *TR* will always be between 0 and 1, with a lower value indicating a more general theory. Moreover, the *TR* is a well-motivated and meaningful measure of the quality of a final learned theory in the context of this work for two reasons: First, it is a relative measure with respect to the number of examples in a training dataset, so one can use it to compare regularity of small and large number of training sets. Second, it is rather general and objective, since it consists of an unbiased measure of how many non-overlapping rules it would be required to cover every positive example without covering any negative examples.

6.2. Optimal Parameters in Rule Learning Phase (EQ 1)

The main goal of the learning component GILPS in the proposed framework consists of building models from a training set of examples. However, for building models of high classification accuracy and, at the same time, understandability, it is necessary to find optimal parameters at first place.

The best possible model requires determining optimal values for some parameters of the ProGolem ILP system according to [Santos, 2010]. The best values are estimated by applying the method proposed in [Kohavi and John, 1995] which recommends the following steps:

i. separating the most relevant parameters;
ii. obtaining, using the training set only, unbiased estimates of the classification accuracy of the models built after a systematic variation across some small number of values for the parameters chosen in the previous step;
iii. taking the values that yielded the best average predictive accuracy across all target predicates.

Accordingly, several preliminary experiments were performed for determining the best parameters in the OBIE scenario. Tab. 6.5 shows the best setting for the ILP parameters according to two criteria: achieving high-quality results, and preventing model *overfitting*. These parameters were obtained using a separate test dataset, i.e., a dataset with unseen examples. Unless contrarily indicated, for all the experiments reported in this section, the aforementioned parameter setting will be adopted.

Table 6.5. Optimal ProGolem parameters

ILP Parameters	
Parameter	**Value**
theory_construction	global
evalfn	coverage
i	3
minprec	0.0
minpos	5
noise	0.2

Overfitting Control. Overfitting is a major problem for all machine learning techniques. In ILP in particular, overfitting causes the effect that a very complex theory would be induced, and the size of this theory (in number of individual clauses) would increase according to the size of the training set. Overfitting is also caused by poor example representations that do not capture relevant domain characteristics, and noisy or erroneous instances that could be derived from inconsistency in human annotation, for example.

In the OBIE scenario, the overfitting problem is addressed by biasing the learner towards simple concept descriptions to explicitly control the size of the final induced theories. The three last parameters from Tab. 6.5 consists of the simplest form of avoiding overfitting by imposing some *stopping criteria* that requires that (i) a rule covers a certain minimum number of examples (*minpos*), or a minimum precision constraint (*minprec*). Finally, the *noise* parameter allows the selection of rules that cover a few false positives. Actually, it is accepted rules that has covered a minimum of five examples (*minpos = 5*) with 20% of noisy (*noise = 0.2)*, i.e., 1 of 5 examples can be erroneous.

6.3. Results and Discussion on the reACE Corpora (News Domain)

This section answers the Experimental Questions 2-5 on the reACE corpora.

6.3.1. Experiments on Features (EQ 2-5)

In OntoILPER first experimental results on relation subtypes (9 in total), an analysis of the features was carried out by gradually incorporating them in the training phase. Tab. 6.6 reports the results of the combinations of features including structural and attributive predicates that correspond to nine feature subspaces of interest (Lines 1-9 in Tab. 6.6).

The first feature subspace (*Line 1*) constitutes the *baseline*, that is, the smallest feature subspace with only morphological features plus the structural predicate *next*/2. The other features combinations are further distinguished into four features categories: structural (in italics), attributive (normal font), and semantic and NER (in bold).

The performance improves as more features are applied, starting with the F-measure of 53.40% and reaching 81.40% in the reACE 2004. In the reACE 2005, the best overall F1 performance (71.80%) may indicates that this dataset is a more difficult than the reACE 2004. Actually, this may due to the fact that, in reACE 2005 dataset, some relations (specially *bussiness*) are very poor represented with only 16 positive examples, which used to hamper the overall scores because OntoILPER was not able to induce any rule for the business relation in all combinations above.

Table 6.6. Contribution of different features over relation subtypes in reACE 2004/2005 datasets

		reACE 2004			reACE 2005		
ID	**Features**	**P**	**R**	**F1**	**P**	**R**	**F1**
1	Morphological + *Next* = Baseline	81,09	39,81	53,40	60,53	25,12	35,52
2	+ *Chunks*	80,17	47,13	59,36	75,05	34,03	46,80
3	+ *Dep*	81,01	46,93	59,43	72,91	36,51	48,65
4	+ *Dep* + *Chunks*	89,01	54,40	67,53	74,81	38,14	50,48
5	+ *Dep* + *Chunks* + POS	91,16	62,04	73,83	81,75	44,24	57,37
6	+ *Dep* + *Chunks* + POS + Chunk related	93,30	66,68	77,77	83,68	50,43	62,91
7	+ *Dep* + *Chunks* + POS + Chunk related + **NER types**	93,04	67,12	77,99	80,59	51,39	62,68
8	+ *Dep* + *Chunks* + POS + Chunk related + **ACE types**	92,20	71,13	80,31	83,03	63,38	**71,86**
9	+ *Dep* + *Chunks* + POS + Chunk related + **ACE types + NER**	92,91	73,07	**81,80**	82,30	61,85	70,62

Discussion. *Lines 2-4* demonstrate the usefulness of structural features in the generated models. Thus, the system achieved more than 14% of improvement in F1 when comparing the baseline (*Line 1*) with the other structural predicates in the model (*Line 4*), i.e., chunks

and dependencies features. Taken separately (*Lines 2 and 3*), chunk and dependencies features practically brought the same boost in performance in reACE 2004, which was around 6%. On the other hand, in reACE 2005, chunk information was even more effective, as it increases F1 score by 11.28% against only almost 5% due to dependencies information.

The above results agree with the ones obtained in previous work [Zhou *et al.*, 2005], [Zhou *et al.*, 2007], [Jiang and Zhai, 2009], in which these authors reported that, for these datasets, chunk analyses or syntactic parsing tree information reflects the more effective structural feature for relation extraction. Inspecting the reACE 2004/2005 datasets, it was found that most of the candidate entities are encapsulated by nominal chunks, which indicates the quite local characteristic of the semantic relations occurring in them. Indeed, the distance distribution (in number of tokens) between the two related entity instances was analysed for both datasets (Fig. 6.2). The results revealed that there exists about 67% (in reACE 2004) and 60% (in reACE 2005) of relation instances in which their arguments are separated by at most two tokens.

In addition, this suggests that each feature subspace alone already captures most of the useful structural information between tokens for relation extraction in these experiments. Due to the locality of semantic relations in reACE 2004/2005, more complex features like dependency trees can only take effect in the remaining minority of long-distance relations. Furthermore, as previously demonstrated, the full parsing of sentences, e.g., dependency parsing, is more susceptible to parsing errors than chunking analysis. Consequently, for such kinds of short-distance relations, sequence information may be even more reliable than syntactic or dependency information.

It should be pointed out that the number of chosen features has a direct impact on the size of training examples, as more features are added as background knowledge. This would certainly require more computational resources. As a result, depending on either the domain or the application, one should take into account this interesting trade-off in feature selection for RE.

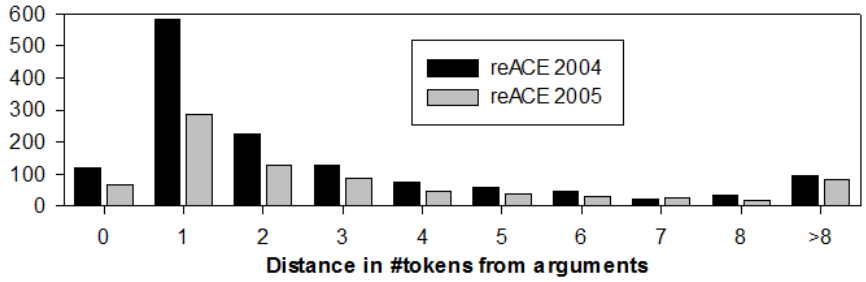


Figure 6.2. Distribution of the number of words between the arguments of relations in reACE 2004/2005.

Incorporating both POS and chunking-related features (*Lines 5 and* 6 from Tab. 6.6) respectively, also contribute to performance improvement. In particular, POS information (*Line 5*) increases the F1 score by more than 6.0 units on both datasets but, contrarily to the reACE 2005 in which these units were evenly distributed between precision and recall, the performance in reACE 2005 increased due to the much higher recall score compared to the precision.

The last 3 lines of Tab. 6.6 show the contribution of attaching the semantic type information to the extraction models. Concerning the impact of the semantic-related features, such as NER and ACE types in Line 7-9 of this same table, ones can notice that, these features lead to another significant performance increases in both datasets.

For the reACE 2004 dataset, a tiny improvement (0.22%) was achieved on using named entities as semantic features (compare Line 6 with 7). Surprisingly, these same entities had a converse effect, decreasing the final F1 score in 0.23 points for the reACE 2005. A closer look at the results reveals that the applied NER component conflicts with the entity types provided by the annotations in datasets, generating more false positives.

However, both entity type and subtype information together with mention level features (like the binary predicate *m_type*) consistently enhanced the final F1 score by significant increasing recall in both datasets.

To conclude, more accurate semantic information of entity instances can contribute a great deal in reACE corpora. As already mentioned, these results have also agreed with previous findings in RE. This is not surprising, given that semantic information, e.g., classes and subclasses from an ontology, typically impose strong constraints on the types of the entities participating in a given relation, indicating that such kind of feature has an crucial discriminative power in RE.

6.3.2. Experiments on Hierarchical Classification (EQ 6)

Since reACE corpora provide both entity and relation hierarchies (type and subtype), hierarchical classification was accomplished on these *corpora* in order to assess OntoILPER when it takes into account both entity and relation taxonomical information from the input domain ontology. Here, the focus is on the relation hierarchy with three levels, from the more specific to general one [Zhou *et al.*, 2007]:

- *subtype classification* at leaf level, consisting of 9 relations in reACE 2004, and 8 relations in reACE 2005, respectively;
- *type classification*, which denotes 4 middle level relation for both corpora;
- *relation detection*, which denotes the classification task of predicting if a relation holds between two entity instances. This last task can also be considered as a simple binary classification.

Tables 6.7, 6.8, and 6.9 summarizes the classification results of the subtype and type classification aforementioned. Besides *P*, *R*, and *F1*, Tab. 6.8 and 6.9 display the number of positive examples (E^+) by type, the number of rules occurring in the final theory (*#Rules*), and the theory ratio (*TR*).

Discussion. From a broad view, the average results shown in Tables 6.7, 6.8, and 6.9 reveal that it is more difficult to classify on hierarchy's deeper levels for both corpora, being the reACE 2005 dataset the one that most profit of the hierarchical classification on a shallow level. Indeed, comparing the average results reported on Tab. 6.7 (left part) with those ones in Tab. 6.8, it is clear that the type classification on reACE 2004 yielded a significant improvement compared to its subtype classification. The analogous comparison between subtype classification (Tab. 6.7 right part) and type classification (Tab. 6.9) on the reACE 2005 portraits that the overall improvement was even more substantial.

On the other hand, a comparison of type/subtype classification results of these tables put in evidence that:

- The performance scores for the PER_SOC on the type classification level of both corpora rank best among the 4 relations types. Moreover, the type classifier yielded the highest possible precision on both corpora, also with a boost on recall.
- The GEN_AFF type classification achieves a small improvement in recall for both corpora.
- EMP_ORG and PARTWHL type results produced comparable results with their corresponding subtype classification on reACE 2004 corpus.

- ORG_AFF and PART WHL relations in reACE 2005 did not benefit from the more abstract level of type classification.

The above results are in accordance with the previous ones reported in [Zhou *et al.*, 2005] and [Zhou *et al.*, 2007], as they reveal that is more difficult to classify on deeper levels of the hierarchy because there are less examples per class and the classes are getting more similar as the classification level gets deeper. The authors in [Zhou *et al.*, 2007] also argued that the closer distance among the classes at subtype level generally causes the performance decreasing, since this might let the classifiers at deeper levels more unstable.

Table 6.7. Performance results of relation subtypes on both datasets

	reACE 2004				reACE 2005			
Rel. Type	**Rel. Subtype**	**P**	**R**	**F1**	**Rel. Subtype**	**P**	**R**	**F1**
EMP_ORG	Employ-Staff	78,10	86,90	82,27	Employ	89,60	86,22	87,88
	Employ-Exec	95,49	77,00	85,25	-	-	-	-
	Member	92,18	76,82	83,80	Member	94,30	71,03	81,03
GEN_AFF	Citizen-Resident	98,81	69,58	81,66	Citizen-	100,00	61,10	75,85
	Located	83,28	80,09	81,65	Located	86,00	84,10	85,04
PERS_SOC	Business	100,00	69,42	81,95	Business	0,00	0,00	0,00[5]
	Family	100,00	39,11	56,23	Family	92,70	57,70	71,13
PRT_WHL	Part-Whole	93,20	83,38	88,02	Geo	100,00	62,10	76,62
	Subsidiary	95,10	75,30	84,05	Subsidiary	95,80	72,51	82,54
	Avg	**92,91**	**73,07**	**81,80**	**Avg**	**82,30**	**61,85**	**70,62**

Table 6.8. Classification results of relation types in the reACE 2004 dataset

Type	**#E+**	**#Rule**	**TR**	**P**	**R**	**F1**
EMP_OR	603	65	0,11	86,00	84,0	84,9
GEN_AF	450	51	0,11	86,90	78,9	82,7
PER_SO	50	18	0,36	100,0	64,4	78,3
PRT_WH	274	33	0,12	91,00	81,6	86,0
Total	**137**	**167**	**0,12**			
Avg				**90,98**	**77,2**	**83,5**

Table 6.9. Classification results of relation types in the reACE 2005 dataset

Type	**#E**	**#Rule**	**TR**	**P**	**R**	**F1**
ORG_AF	26	38	0,14	88,70	77,8	82,8
GEN_AF	31	60	0,19	94,40	70,4	80,6
PER_SO	58	19	0,33	100,0	58,3	73,6
PRT_WH	16	35	0,21	87,60	72,3	79,2
Total	**80**	**152**	**0,19**			
Avg				**92,68**	**69,7**	**79,5**

Theory Ratios. In Tables 6.8 and 6.9, further results concerning the theory ratio are reported.

Comparing the overall result in terms of TR, on both corpora, one draws the conclusion that more rules were necessary to cover the examples on reACE 2005, with an average of 8.24 rules per examples, against 5.3 rules per examples on reACE 2004. Yet, these TR scores are also reflected in the overall F-measure, significantly lower on the reACE 2005 dataset. The reason may be the presence of more complex examples that are only covered by more lengthy rules, or rules with lower generalization degree, in reACE 2005.

For the three relations types that both datasets have in common, i.e., GEN_AFF, PER_SOC and PRT_WHL, the number of final rules in their respective models are approximately equal, particularly for the last two relation types. However, considering the different proportion of examples of the PRT_WHL type relation in both corpora, this relation was responsible for the major gap in the TR score, which was approximately 0.9 units.

The bottom line with respect to the theory ratio is that the proposed solution generates theories of tractable size in the experiments of both datasets.

[5] The zero result for the *business* relation in reACE 2005 is due to very few instances available for training.

6.3.3. Experiments on Composite Model and Cross Corpus Learning (EQ 7)

Cross-validation has been applied as the facto standard for information extraction evaluation. However, as mentioned by Airola *et al.*, (2008), it is also somewhat biased because both training and test datasets share very similar characteristics.

In order to differently assess the level of generalization of the induced rules in OntoILPER, in the scenario when more than one dataset are available for training, two experiments were conducted: (i) *composite model*, when both datasets are integrated in the training step in order to investigate how they differ; (ii) *cross-corpus learning*, in which one trains the system with one corpus and tests it using the other one.

The former experiment may also confirm the existence of overlapping rules or patterns in both datasets. Thus, focusing on their inner differences, customized background knowledge can be provided by the expert domain which will certainly take profit of this valuable information. The latter mainly concerns the relevant research question concerning the training in the proposed ILP-based learning component: *Would the final induced rules generalize beyond the specific characteristics of the data they were trained on*?

In view of the very plausible existence of difference in both the types of named entities annotated in the corpus, and the relative positive-negative ratio of pairs, it is not obvious how the learning component, trained on a given corpus, would perform on data which was not used in its previous training phase.

In the following discussion, the equivalence of relation subtypes shown in Tab. 6.10 was taken into consideration, with their respective number of positive examples.

Table 6.10. Equivalence of relation subtypes between the reACE datasets

reACE 2004	#E+	reACE 2005	#E+
Business	35	Business	16
Citizen-Resident	98	Citizen-Resident	39
Employee-Executive + Employee-	523	Employement	228
Family	15	Family	42
Part-Whole	174	Geographical	119
Located	352	Located	280
Member of Group	80	Membership	36
Subsidiary	100	Subsidiary	47

Table 6.11. Performance results on composite and cross-corpus learning.

Model	P	R	F1
reACE 2004 + reACE 2005 = Composite Model (CV*)	93,28	71,21	80,76
Composite Model applied on reACE 2004	96,18	85,55	**90,55**
Composite Model applied on reACE 2005	96,81	83,61	89,73
Cross-corpus Learning: reACE 2004 ► reACE 2005	42,54	55,25	48,07
Cross-corpus Learning: reACE 2005 ► reACE 2004	48,34	49,14	48,73

* 5-fold cross-validation

Discussion. The first row of the Tab. 6.11 reports the experimental results on the composite model built using all instances of both corpora. The achieved performance, in terms of P/R/F1 (93.28/71.21/80.76), did not vary significantly compared to the best individual model (reACE 2004) (92.91/73.07/81.80) evaluated with the same 5-fold cross-validation.

On the contrast, compared to the reACE 2005 individual cross-validation, the gain was in almost 10 percentage points in overall performance (83.03/63.3/71.86). This means that, to some extent, a larger amount of training data can compensate for the differences between corpora, even with different annotation strategies.

In the second group of experiments, the aim is to evaluate the performance of the composite models applied on each dataset individually. Unsurprisingly, one can observe that the more training data is available, the better the performance is. Actually, the combined positive effect of more available data for training was the reason for an increase in terms of F1-measure of almost 10% for the reACE 2004, and 20% for the reACE 2005.

The last group of experiments on cross-corpus learning, as expected, did not perform well for both datasets, despite both corpora have been originated from the same news broadcast domain. These results are in accordance with the findings on biomedical domain reported in [Airola *et al*., 2008] in which five corpora were evaluated for cross-corpus learning.

In this case, F1-measure score dropped around 30% when the reACE 2004 model is applied on the reACE 2005. For the opposite direction, applying the reACE 2005 model on the reACE 2004 dataset caused the F1 score decreased in 20%. A closer look at Tab. 6.10 reveals that the reduction in performance was mainly due to:

- the fact that the datasets has different positive-negative distribution of examples;
- some relations that are not really equivalent, i.e., they contain different named entities as arguments.

In the above mentioned study in [Airola *et al*., 2008], the authors dealt with several cases where the performance was quite low, with a decreasing in F1 measure more than 50%. Moreover, as discussed in [Caporaso *et al*., 2008], applying text mining tools beyond the development data may lead to disappointing results.

It worth noting that the negative difference found in the cross-corpus learning experiment, compared to the results obtained in the cross-validation, seems to break the well-accepted assumption made by the majority of machine learning methods that both training and test examples are evenly distributed. A further observation, according to the statistics presented in Tab. 6.10, is that the examples are clearly not evenly distributed over the corpora. As explained before, in cross-validation all characteristics of the test corpus are also present in the training corpus, and such corpus peculiarities are thus somehow learned by the current machine learning algorithms

The conclusion is that the relatively large differences in the obtained performance scores indicate that different datasets, even belonging to the same domain, may have notably different characteristics that are more perceptible when performing cross-corpus experiments, as the ones discussed above. Therefore, the cross-corpus learning results not only support the assumption that the learned models generalize beyond the corpora they were trained on, but also have the potential to indicate the best resource for training from a generalization perspective, that is, they can reveals the distinguish characteristics of a particular corpus.

6.3.4. Qualitative Analysis of the Induced Rules (EQ 8)

One of the main advantages of the learned classifiers in the proposed approach, is that the final models, or the set of extraction rules, are expressed in symbolic form. As a result, this enabled us to further inspect several characteristics of them. Note that this would not be possible using statistical or numerical-based classifiers that only return us a numerical value denoting the likelihood of an example belonging to a given class. It follows a discussion about (i) the rule size distribution, and (ii) the contribution of feature types in the final induced theories.

Fig. 6.3 displays the rule size distribution in the induced set of rules.

Discussion. For both corpora, one can see that the great majority of learned rules have less than 12 ground BK predicates in their body part. Actually, for both datasets, more than 80% of the found rules have up to 11 predicates. This leads to other desired qualities of these rules: they are more suitable and easier to be understood by a human domain expert. Another interesting finding revealed by Fig. 6.3 concerning both datasets is that, the shorter the final rules, the better the level of rule generalization of the training examples. In other words, the rule size distribution can inform how general or, contrarily, how specific the final rules are. In addition, it also indicates the overfitting degree of the final set of induced rules.

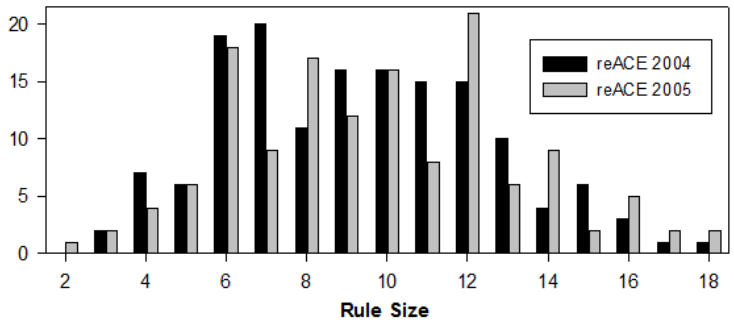


Figure 6.3. Rule size distribution in reACE 2004/2005.

Second, the objective is to investigate the contribution of different features, or ground predicates that form the hypothesis space for rule induction on the final learned models of the reACE corpora. Fig. 6.4 shows the contribution of five group of features (in percentage points). Not surprisingly, due to the same domain and source of both datasets, the higher level view of each group of features shows that their learned models are very similar. Yet, in both datasets, the structural and semantic group of features were predominant, as these two group of features were responsible for almost 60% of participation in the final theories. Interestingly, this confirms the usefulness of the proposed graph-based model which highlights the importance of the relational and semantic aspects of the underlying model.

On the other side, the detailed view (Fig. 6.5) of the contribution of each individual feature reveals that:

- the chunking-related features in both corpora were equally exploited, except for the *ck_dist_root* predicate that was much more used by the rules of the reACE 2005;
- the very poor participation of the *stem* predicate in the models of both datasets denotes, in fact, a positive effect on the extraction rules, since they do not need to match the exact stem of the words in the examples.
- Among the lexical features, the orthographical features, and mainly word length contributed most.
- POS-related features were evenly used in both corpora. Notice that any POS trigram appeared in the extraction rules.
- the extraction rules relied more on *next* and *dependency* structural features for the reACE 2004.
- As already mentioned, the semantic features provided by both corpora consistently appeared in the extraction rules, which indicates that the entity type information has a high discriminative effect in RE.

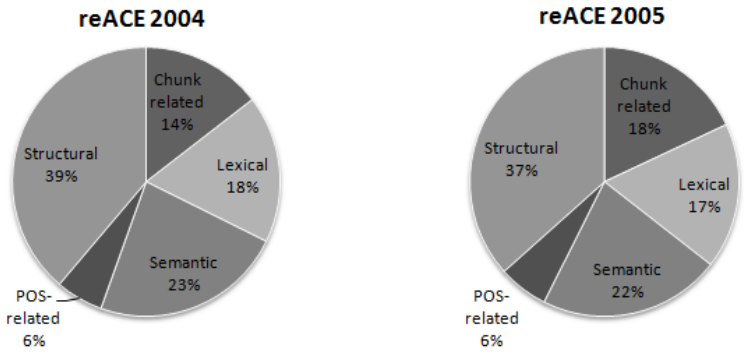


Figure 6.4. Ratios of the BK predicate types in final theories.

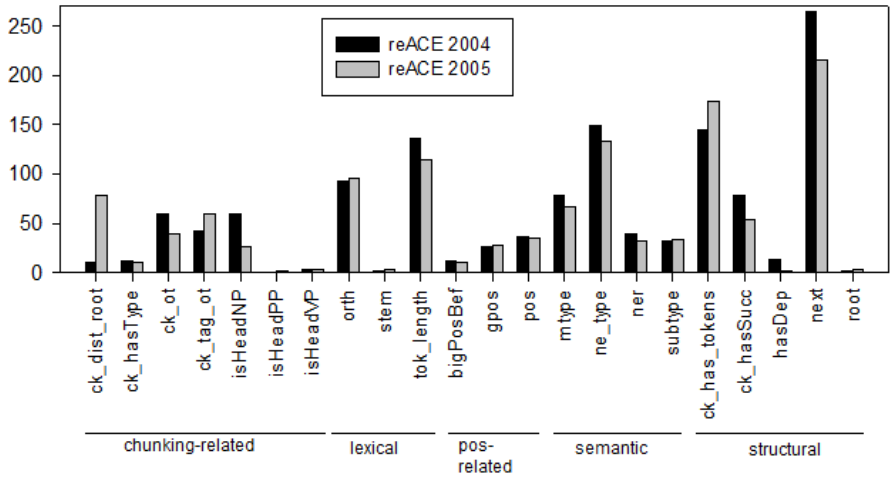


Figure 6.5. Frequencies of the BK ground predicates in final induced theories.

6.3.5. Discussion on Error Analysis and Remarks on the reACE Corpora

In all experiments discussed in this section, the typical trade-off between precision and recall was again verified by the proposed OBIE framework preferring higher precision than recall[6]. In order to obtain a clear understanding of the precision-recall trade-off in the experiments, the distributions of errors from misclassification were evaluated across relation subtypes (Tab. 6.12) for both corpora using all training data.

According to the Tab. 6.12, the number of *false negatives* (*FN*) overtakes the number of *false positives* (*FP*) for both corpora. A possible reason for that may be due to the unbalanced class distribution in data. In this case, the number of negative examples greatly outnumbers positive examples, with a positive-negative ratio around 10% in both datasets. Thus, for the reACE datasets, in which very few of the training examples are relation instances, the classifier is less likely to identify candidate instances as actual relations.

In fact, the impact of class imbalance on the performance results was also reported on the original ACE datasets by [Cullota and Sorensen, 2004], [Zhou *et al.*, 2007]. In [Chawla, 2002)], it was pointed out that datasets with unbalanced class distributions present a number of problems for all machine learning algorithms. In addition, the domains addressed by RE systems tend to have a large number of objects and relations, in which just a few are positive examples.

This clearly directs the future efforts for a possible solution to the problem of unbalanced class distribution. The most straightforward way of dealing with this problem

[6] The ILP learning component in OntoILPER can be biased either towards precision or recall by means of the appropriate parameter setting.

consists in employing sampling techniques, such as *under-sampling* [Chawla, 2002], which selects a subset of negative examples during training. This filtering technique not only allows for the creation of a balanced training dataset by considerably reducing the number of negative examples, but also enables faster rule learning.

Table 6.12. Distribution of errors in reACE corpora

	#FP	#FN	Total
reACE 2004	109	145	254
reACE 2005	34	87	121[7]

Another source of errors is related to the introduction of parsing errors and their propagation in the text processing pipeline. As the experimental evaluation on reACE corpora suggests, shallow natural language processing (tokenization and chunk analysis) is more accurate and may also complement, with useful information, what was missed by deeper text preprocessing techniques, such as dependency parsing. Indeed, related work on RE only based on deeper representations inevitably suffered from errors introduced by such kinds of representations.

To alleviate the impact of parsing errors in the proposed method, the graph-based model of sentences representations that guides the generation of relational features allows for an effective exploration of the highly reliable syntactic features as well as other useful deep natural language subtasks. In this light, another hypothesis of this thesis is that the proposed graph-based model of sentence representation could be fully exploited in the ideal scenario where the text preprocessing errors at each step in the pipeline would be independent, i.e., if there were no dependency between the preprocessing modules performed in OntoILPER.

6.3.6. Conclusions on the reACE Corpora

The overall conclusion concerning the experimental assessment on the reACE corpora discussed so far validates the effectiveness of the proposed hypothesis space which, contrary to the trend of using propositional features or single table-based representations in RE community, relies on a set of relational features as the formalism for representing the examples (*Research Questions 2-5*).

In the OBIE scenario, one can also note that the hierarchal classification results obtained by OntoILPER (*Research Question 6*) agreed with related work in the sense that the classification on deeper hierarchical levels in an ontology.

The performance results on cross corpus learning also showed that, even belonging to the same domain, there may exists notable peculiarities in these datasets that, not only reveal the distinguish characteristics of each dataset in particular, but also can significantly influence in the accuracy results (*Research Question 7*).

Finally, the results of the qualitative analysis demonstrated that the combination of structural, lexical, syntactical, and semantic features allows for a synergic cooperation that seemed to be critical for relation extraction in this research work (*Research Questions 2-5, 8).*

[7] Due to the insufficient number of examples, the *business* relation was not included.

6.4. Results and Discussion on the TREC Dataset (News Domain)

This section evaluates and discusses the experimental results on both NER and RE using the TREC dataset. For all experiments, it was adopted the 5-fold cross-validation that provides unbiased performance estimates of the learning algorithms, and also enables the comparison with other IE systems evaluated using the same corpus. Moreover, although OntoILPER provides a named entity tagger in its preprocessing component, it was decided that, for having a fair experimental setup for all experiments conducted on the TREC dataset, this named entity tagger should not be used.

6.4.1. Assessment Scenarios (EQ 9)

In all experiments reported in this section, the same assumption made in [Roth and Yih, 2007] [Giuliano *et al*., 2007] was adopted in which the problem of *phrase detection* is already solved, and the entity boundaries are provided by the dataset as input. Thereby, OntoILPER only needs to concentrate on the NER and RE classification tasks.

Several experiments were conducted for evaluating the effectiveness of OntoILPER taking into consideration three different kinds of extraction models for classifying instances of classes and relations. For the sake of convenience, these extractions models (classifiers) were named using the same name convention proposed in [Roth and Yih, 2007].

These models are used in three different assessment scenarios, and they are described as follows:

- **Separate Model**. The separate models for entities E_S and relations R_S are constructed by training entities and relation classifiers separately or independently, i.e., the entity classifier E_S does not know the labels of the relations in the sentence; while the relation classifier R_S is not aware of the labels of its entity arguments neither. In other words, the E_S and R_S classifiers are build using no information from each other. These models are generated by OntoILPER as illustrated in Fig. 6.6 part (a).
- **Pipeline Model**. The pipeline model for entities, denoted as E_P (for entity classifiers) is obtained by first training a separate relation classifier R_s, in which its output is then used as additional features for training the E_P classifier. Analogously, the R_P model uses the predictions on its two entity arguments, given by a separate entity classifier (E_S), as additional features in its learning process. This model construction process is displayed by Fig. 6.6 part (b).
- **Omniscient Model**. In the omniscient model, it is assumed that the entity classifier E_O knows the correct relation labels given from the annotated corpus. This model takes advantages of such labels as additional input features in its learning process. Similarly, the relation classifier R_O knows the correct entity labels, available from the annotated corpus as well. These additional features are then used in training. Although this assumption may appear unrealistic, it may reveal how accurate the classifiers can be without this information. Fig. 6.6 part (c) illustrates the construction process of the omniscient models in OntoILPER.

All classification models for the named entities Location (LOC), Organization (ORG), and Person (PER) obtained high overall accuracy in all models (see Tab. 6.13). All these models (E_O, E_P, E_S) are highly precise, with precision values ranging from 93.5 (obtained by the PER entity) to 98.7 (obtained by the ORG entity).

Still considering the entity models, the same analysis of their performance results in terms of recall values (ranging from 74.4 to 92.4) shows that more instances were not considered during classification, than the number of instances with false entity class predictions. The results in Tab. 6.13 also reveal the balanced nature between precision and recall in all classification models for the LOC and PER entities. On the contrary, the ORG models obtained the highest precision among all entities, but with the lowest performance, in terms of recall.

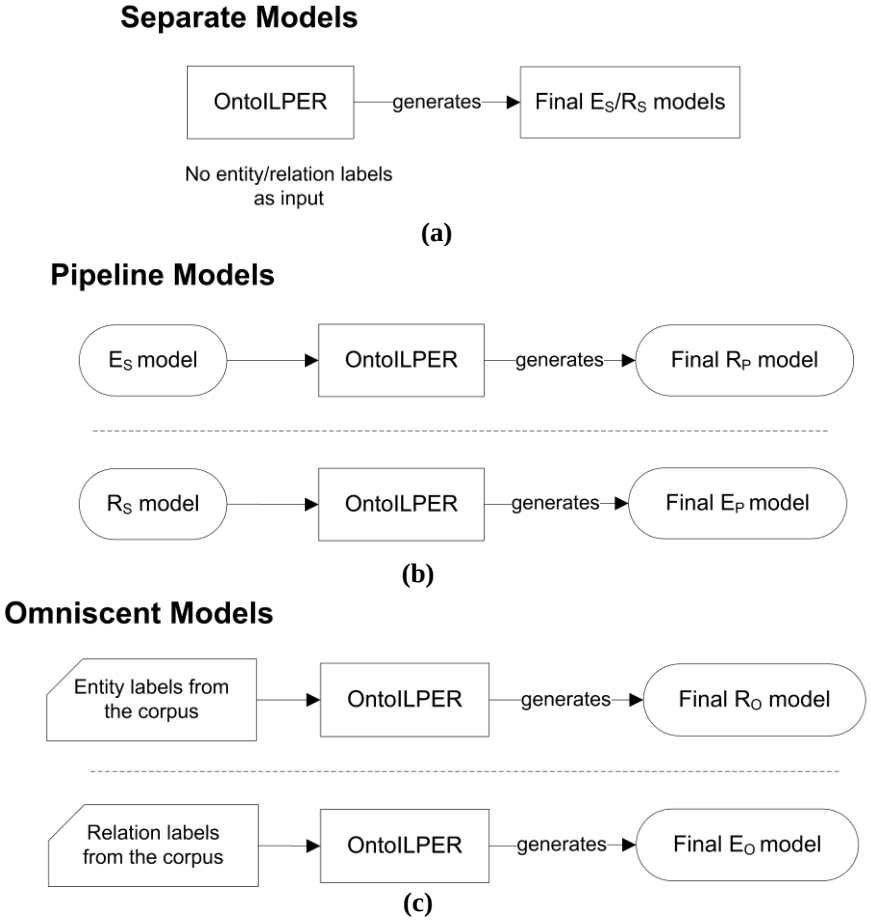


Figure 6.6. Entity and relation extraction models built using the TREC dataset. The name convention for the models was the same in (Roth and Yih, 2007)

Tables 6.13 and 6.14 report the classification results achieved by all three aforementioned models for entities and relations, respectively.

As already discussed, the RE task has been a more difficult task than NER. This was once more confirmed by the performance results achieved by the relation models (R_O, R_P, and R_S) in all results shown in Tab. 6.14. Similarly to the entity extraction models, the relation extraction models preferred to have more precision than recall: precision scores range from 85.7 to 93.1, while recall ones range from 72.1 to 86.1.

Although the results in Tab. 6.13 and 6.14 suggest that OntoILPER had always the preference of precision over recall, that is not a right conclusion because, in reality, OntoILPER can use other evaluation functions that would prefer recall than precision, such as the recall evaluation function (Santos, 2009).

Table 6.13. Results for Entity Classification (All Models)

NER Model	LOC			ORG			PER		
	P	*R*	*F1*	*P*	*R*	*F1*	*P*	*R*	*F1*
E_O	95.9	92.4	**94.1**	98.7	79.2	**87.8**	93.7	91.2	**92.4**
E_P	95.2	92.0	93.5	97.5	76.5	85.7	93.5	89.0	91.3
E_S	96.0	88.4	92.0	97.0	74.4	84.3	94.8	87.5	91.0

Table 6.14. Results for Relation Classification (All Models)

RE Model	located_in			work_for			orgBased_in			live_in			kill		
	P	*R*	*F1*	*P*	*R*	*F1*	*P*	*R*	*F1*	*P*	*R*	*F1*	*P*	*R*	*F1*
R_O	90.5	78.6	**84.0**	85.7	86.1	**85.8**	88.7	82.5	85.4	87.4	76.9	**81.7**	92.3	78.0	**84.3**
R_P	91.1	78.0	83.9	87.2	80.8	83.8	91.5	84.0	**87.5**	85.7	72.1	78.2	91.5	77.6	83.9
R_S	91.2	75.9	82.6	93.1	72.9	81.7	88.4	77.0	82.2	92.5	67.4	78.0	97.5	73.7	83.8

Discussion. The classification results achieved by the richer models for entities (E_O and E_P), compared to the E_S model as the comparison baseline, had a little improvement for all entities both in precision and recall scores. These results were expected as the E_O and E_P models are richer, i.e., more informed models than the E_S model. Surprisingly, for the PERSON entity, the entity labels caused a drop of 1.3% in precision, contrarily to our intuition. This can be originated from noisy ORG class examples in the dataset.

The results in Tab. 6.14 shows that, for almost all relation models (except for the ORGBASED_IN relation) the entity labels provided to the R_O model, decrease the precision of the classifiers, but contribute to improving the recall scores in all relation classifiers. This can be explained by the fact that the noisy information about entities in the dataset itself can be mitigated by these further clues to the classifiers.
Thus, the correct entity labels enable the classifiers to cover more examples in this case.

Indeed, an interesting question to be answered concerns the case of having a new dataset with all its entities annotated. In this case, what is the better way of using OntoILPER for RE? Or putting differently, what is the best relation extraction model to learn in OntoILPER: R_P or R_S?

According to the results summarized in Tab. 6.13 and 6.14, the pipeline models outperformed the separate ones on both NER and RE tasks. But, specially for the relation extraction models, the overall F1-measure results showed a significant statistical difference between the RP and RS models. Therefore, the distinct characteristic of the learning process in OntoILPER, i.e., its capability to employ rules learned in a previous learning stage as additional BK predicates at a posterior learning stage can be very useful, as suggested by the above results in the TREC corpus.

Example of an induced rule in OntoILPER. In the following, an induced rule from the R_P model for the *located_in* relation is shown. This rule is expressed in terms of (*number of literals*), (*positive examples covered*), (*negative examples covered*), and the (*rule precision P*):

Rule: #Literals=4, PosScore = 187, NegScore = 19, Prec = 90.8%
located_in(A,B):- t_class(A,loc), t_next(A,B), t_class(B,loc).

The above rule is in Prolog syntax. One can see that this rule classifies an instance of the *located_in* relation in which its high precision score is mainly due to the presence of several phrases similar to "Perugia, Italy" in the learning corpus, indicating that the first argument (*A*) "Perugia" is followed by (predicate *next*) the second argument (*B*) "Italy", not considering the punctuation symbol between them. This rule belongs to the R_P relation model in OntoILPER.

6.4.2. Learning Curves

A further evaluation of the E_S and R_S classifiers were performed in which aims at investigating the effect of limited training examples in the learning process. This is done by incrementally adding subsets of examples as training data to OntoILPER.

For that, nine experiments were conducted in which incremental parts of the training data, corresponding to 10% of the total number of examples each one, are added to a previous subset of training data at a time. Therefore, starting from a training set with only 10% of the total training examples, one generated other training sets of 20%, 30%, 40%, and so on, from the total number of available examples in the corpus.

The learning curves in Fig. 6.7 relate the F1-measure score for each portion of the training dataset. From the left part of Fig. 6.7, one can observe that for LOCATION and PERSON entities, their classifiers yielded a reasonable F1 score (around F1 = 70%) with just 20% of the total number of training examples. That corresponds to 30 and 26 extraction rules in the final induced model for LOCATION and PERSON entities, respectively. In contrast, for the ORGANIZATION entity classifier, more learning examples were necessary for attaining the same performance. Indeed, 70% of the total number of examples were used. A possible explanation of this phenomenon may be related to the more representative number of ORGANIZATION entities in the TREC corpus. Actually, the number of examples instances in the PERSON and LOCATION classes are almost the double of the ORGANIZATION entity class.

In Fig. 6.7 (right part), almost all relations obtained better performance relations as more and more training data are available, which explains the steadily increasing rate in terms of F1 for all relation curves. Particularly for the ORG_BASED_IN and LIVE_IN relations, this trend was also verified, but with the initial lower F1 scores for the 10%-40% of the training corpus, and becoming rapidly higher for the rest of the corpus.

The fact that the ORG_BASED relation has as one of its arguments, an ORGANIZATION entity, may also explain the lower learning performance curve for the ORGBASED_IN relation.

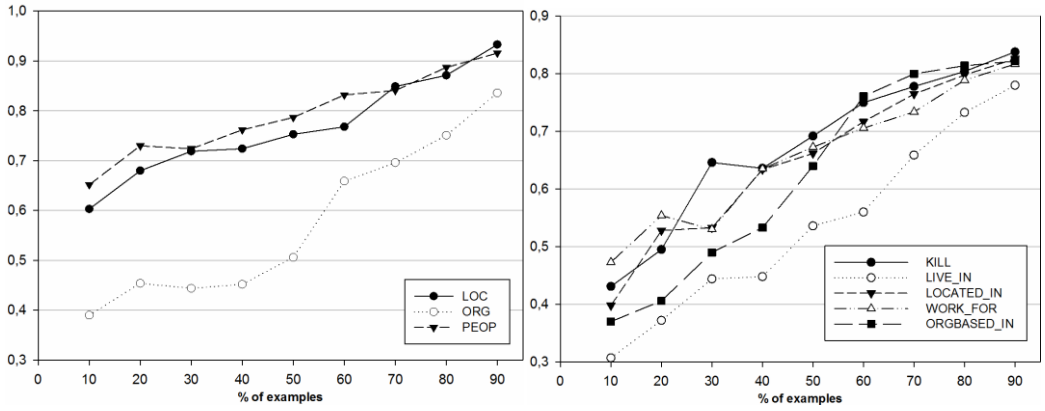


Figure 6.7. Learning curves for entity (a) and relations (b) classifiers: E_S and R_S, respectively.

6.4.3. Comparative Evaluation (EQ 10)

This section provides a comparative evaluation of the NER and RE classification models generated by OntoILPER with the best ones presented in [Roth and Yih, 2007] and [Giuliano *et al*., 2007] on the TREC dataset. To the best of our knowledge, these are the only works that used this dataset for evaluating both NER and RE tasks.

For the *entity classification* comparison, the two best classification models found in [Giuliano *et al*., 2007] and [Roth and Yih, 2007] were chosen, which are described as follows:

- the M_C model [Giuliano *et al*., 2007]. Giuliano *et al*. (2007) also assume that entity boundaries have already been determined. This *MC* corresponds to the E_S model generated by OntoILPER.
- the *Separate with Inference* (Separate w/inf) model [Roth and Yih, 2007]. This NE classification model uses an additional global inference procedure to produce the final decision. The *Separate w/Inf* equally relates to the ES model.

The comparative assessment involving these NER classifiers are summarized in Tab. 6.15, while 6.16 shows the comparative results on the RE task.

The second comparison relates the best RE classifiers proposed by the same aforementioned works.

In Tab. 6.16, the R_O model consists of the best relation extraction model discussed in Section 6.4.1. The two other RE models in this comparison are:

- The $M_O|K_{SL}$ model [Giuliano *et al*., 2007] similarly corresponds to the R_O model, in the sense that the former also uses the corpus entity labels during learning as it is done in OntoILPER.
- The *Omniscient w/Inf* model consists of the best informed RE classifier reported in [Roth and Yih, 2007]. This classifier also employs a global inference procedure to produce its final decision on the relation classification.

Discussion. The results on NER shown in Tab. 6.15 suggest that the M_C model has superior performance in terms of F1 score. However, statistical significance tests (*paired Student t-test*) for the difference between F1 scores of the E_S model and the M_C model revealed that there is no significant difference at $\alpha = 0.05$ (95% confidence interval) between them. The same result occurs when the E_S model is compared with the *Separate w/Inf*. A further look at Tab. 6.15 shows that the E_S model is more precise than the other ones, but has a lower recall performance. For those applications in which precision is more desirable than recall, the E_S model could be a good alternative, as it avoids overloading end-users with too many false positives.

On the other hand, considering the relation classifiers, the results in Tab. 6.16 show that OntoILPER outperforms the others. The main reason for that probably relies on the better sentence representation model employed by OntoILPER.

In fact, in the proposed graph-based model, any kind of relationships between terms in a sentence are represented using a richer formalism of representation (first order logic) which is more expressive than the feature-based representation used by the related work evaluated. In other words, such statistical extraction models cannot effectively capture the structural information derived from parsed sentences [Zhou *et al* 2005]. Contrary to that, OntoILPER employs a first-order model for representing examples and, as the experiments reported here have demonstrated, this structural information is critical for further performance improvement in RE.

Table 6.15. Comparative Results for Entity Classification (Separate Models)

NER Model	LOC			ORG			PER		
	P	*R*	*F1*	*P*	*R*	*F1*	*P*	*R*	*F1*
E_S	96.0	88.4	92.0	97.0	74.4	84.3	94.8	87.5	91.0
M_C	94.2	94.4	**94.3**	91.9	88.5	**90.2**	94.8	96.6	**95.7**
Separate w/Inf	91.8	88.6	90.1	91.2	71.0	79.4	90.6	90.5	90.4

Table 6.16. Comparative Results for Relation Classification (Best Models)

RE Model	located_in			work_for			orgBased_in			live_in			kill		
	P	*R*	*F1*	*P*	*R*	*F1*	*P*	*R*	*F1*	*P*	*R*	*F1*	*P*	*R*	*F1*
R_O	90.5	78.6	**84.0**	85.7	86.1	**85.8**	88.7	82.5	**85.4**	87.4	76.9	**81.7**	92.3	78.0	**84.3**
$M_O \mid K_{SL}$	79.6	76.0	77.8	76.8	80.0	78.4	74.3	77.2	75.7	78.0	65.8	71.4	82.8	81.0	81.9
Omniscient	61.9	62.9	59.1	79.2	50.3	61.4	81.7	50.9	62.5	63.9	57.3	59.9	79.9	81.4	79.9

The OntoILPER experimental results on the TREC corpus point in the direction of being more effective on RE than NER. Actually, OntoILPER relies on the Stanford CoreNLP NER, which has demonstrated state-of-the-art performance in comparison to other NER systems [Dlugolinsky *et al*., 2013]. However, even without using the Stanford NER in its preprocessing step on the TREC corpus, OntoILPER outperformed the feature-based NER method proposed by Roth and Yih (2007).

On the other side, the CRF-based NER system proposed by Giuliano *et al*. (2007) uses a gazetteer of location, people's names and organizations in its preprocessing phase, which certainly has a boosting impact on the NER performance results. Despite that, the statistical significance tests showed that OntoILPER is comparable to NER system proposed by Giuliano *et al*. (2007).

Although the IE systems chosen for this comparative assessment may be a little outdated, these results provide a baseline performance evaluation aiming at validating the OntoILPER effectiveness when compared to other NER systems. In addition, future work on feature engineering, specially for NER, can contribute to further improve OntoILPER results on this information extraction task.

A final remark concerns the NLP preprocessing component in OntoILPER which is based on supervised trained models on the generic *News* domain. Therefore, this fact might raise the following question: "does OntoILPER have state-of-art performance on other different domains?"

In fact, due to its extensive range of relational features easily integrated by a hypothesis space carefully tailored, OntoILPER has equally outperformed other very recent state-of-the-art feature- and kernel-based methods for RE on biomedical domain, as reported in (Lima *et al*., 2014).

6.5. Results and Discussion on PPI Corpora (Biomedical Domain) (EQ 11)

This section details the experiments and results related to the PPI extraction task using several combinations of the simplification rules.

For the experiments reported in this section, the AUC as recommended by Airola *et al*. (2008)] was also used in the evaluations.

In addition, the best setting for the GILPS parameters was established, according to the following two criteria: maximizing accuracy performance and mitigating model overfitting. Such parameters were obtained using a separate test dataset, i.e., a dataset with no previously used examples. The parameter setting in all experiments reported in this section are summarized in Tab. 6.17.

Table 6.17. ProGolem parameters for the biomedical experiments.

ILP Parameters	
Parameter	**Value**
theory_construction	global
evalfn	coverage
i	3
minprec	0.0
minpos	3
noise	0.35

6.5.1. Results and Discussion on Graph Transformations

Four different combinations of simplification rules (see Section 4.5.2), or *Filters* were used in all experiments reported in this section. The set of rules that compose each filter is shown in Tab. 6.18. As it can be noticed, they gradually become more restrictive, i.e., the filter in the last row of the table is composed of the preceding filters. Besides the *baseline filter* (*No Filter* in Tab. 6.18), denoting the case where no transformation rule is applied, four different combinations of simplification rules (see Section 4.5.2), hereafter *filters*, were provided.

Table 6.18. Filters used in the experiments.

Filter	Rules composing the filter
No Filter	No rule application
Filter 1	R4 + R5 + R6 on dependencies in {*det, aux, auxpass*}
Filter 2	R4 + R5 + R6 on dependencies in {*det, aux, auxpass, amod, predet, advmod, partmod, tmod, mark* }
Filter 3	Filter 2 + R1 + R2
Filter 4	Filter 3 + R3

The analysis here starts with the overall results of the transformation/simplification process itself with its basic statistics taken into consideration that are reported in Tab. 6.19. The average number of tokens and dependencies per sentence, compared to the original version of the dataset (with no reduction) and the final version applying all simplification rules are shown in Tab. 6.19. As expected, for all datasets, the number of nodes or tokens in the graph-based representation of sentences, and dependencies, the edges in this same graph are considerably reduced by the proposed simplification rules. One should note that the average number of nodes is always slightly less than the number of the edges in the representation graphs. This can be explained by the fact that several tokens or nodes of the sentences are removed, e.g., determiners and prepositions, contrarily to the dependencies (edges) that still remains in the simplified version of the graph. For instance, if one considers the following prepositional phrase “tears in heaven”, where the token denoting the preposition is removed from the graph and it is created the edge *prep_in(tears, heaven)* in the graph.

Table 6.19**.** Average number of nodes and dependencies per sentence before/after transformation

	LLL		**HPDR50**		**IEPA**	
	No filter	**All filters**	**No filter**	**All filters**	**No filter**	**All filters**
Average #nodes/sent	20.3	13.3	18.4	13.1	22.0	15.4
Average #dep/sent	20.4	13.2	18.8	13.4	22.9	16.0

Tab. 6.20 shows the simplification ratio, in terms of the number of edges, for all datasets and all filters. Again, the more restrictive filters further reduce the number of edges in the graphs. According to the results reported in this table, the minimum (12%) and the maximum reduction ratio (35%) for the IEPA and the LLL datasets were obtained, respectively.

Besides the direct effect on the extraction results, the reduced version of the graphs also has a significant impact on reducing the training time. In reality, a reduction up to 15% in training time was obtained using the simplified versions for the datasets in all experiments in this section. The experiments were performed on an Intel Core 2 Duo, 2.30 GHz with 6 GB of RAM on Linux. GILPS[8] was executed on YAP Prolog 6.2.2[9].

Table 6.20**.** Reduction of the number of dependencies per filter in percentage points (%)

	LLL	**HPD**	**IEPA**
Filter 1	17.31	12.50	12.08
Filter 2	19.29	14.74	14.20
Filter 3	28.26	24.01	25.03
Filter 4	35.07	28.75	29.91

Going deeper into the types of the dependencies reduced from the IEPA dataset using Filter 4, i.e., the more restrict, it is obtained the results shown in Tab. 6.21. This filter removes all of the determiners, verb auxiliaries and markers (like "that") from the IEPA corpus. The modifiers including adjective, temporal, and quantifiers are severely reduced from the original version of the dataset. The same occurs to some dependencies representing the subject and clausal complement of the sentences, due to the clausal-level simplification rules defined in Section 4.5.2, which only removes the non-informational clauses in sentences, leaving untouched the other more informational ones.

Table 6.21. Number of the typed dependencies removed from the IEPA corpus using the Filter 4

Dep.	*det*	*aux*	*mark*	**mod*	*advcl*	*nsubj*	*ccomp*
No filter	956	593	169	1632	79	919	179
Filter 4	0	0	0	113	72	829	60

6.5.2. Results and Discussion on PPI Extraction

The OntoILPEr framework was used in the experiments to evaluate the proposed method for simplifying the graph-based representations of sentences. In the experiments on PPI extraction, the filters showed in Tab. 6.18 were applied on the original version of each dataset described in Section 5.1. The resulting simplified versions of the PPI were employed by OntoILPER for training. Table 6.22 reports on the results obtained for each modified versions of the datasets. The baseline for comparisons in this experiment was established as indicated by "No Filter*"*.

A first look at Tab. 6.22 reveals that the overall average result in terms of *P*, *R*, and consequently *F1*, was consistently higher than the baseline score for all datasets. On the other hand, taking the filters individually, one can see that the only case where a filter hampered a little bit the recall score was for the *Filter 2* of the HPRD50 dataset, but for all the filters in the other datasets, the simplified versions of the dataset achieved better results. Such results suggest that all evaluated corpora may profit from the application of the transformation/simplification rules proposed in the present work.

[8] General Inductive Logic Programming System (GILPS). http://www.doc.ic.ac.uk/~jcs06/GILPS
[9] Yet Another Prolog. http://www.dcc.fc.up.pt/~vsc/Yap

The best filter seems to be *Filter 2* on the LLL dataset and *Filter 1* for the other datasets. Interestingly, *Filters 3* and *Filter 4* practically had the same performance in terms of *P/R/F1* on the LLL dataset. An explanation for such results is probably due to the size of such dataset. Indeed, the LLL dataset has less than 100 sentences, and the system could not found any sentence for applying the *R3* rule using this dataset. On the contrary, with the other larger datasets, especially for the IEPA dataset which contains 480 sentences, the difference between the results of *Filter 3* and *Filter 4* is more noticeable.

The last row of Tab. 6.22 shows the difference in performance in percentage points between the average F1 score of the filters, and its related baseline score. According to the results reported in Tab. 6.22, one can expect an improvement in terms of both precision and recall, since the difference between the average of all filters and the baseline performance (*No Filter*) is always positive. In fact, the extraction rules that were induced using the simplified versions of the graphs were slightly more precise (in average 2.9 points) than the same rules induced without any simplification. Particularly to the IEPA dataset, the same line of this table shows a significant improvement in precision of 4.8 percentage points.

Table 6.22. Performance results on PPI extraction using the filters 1-4.

	LLL			HPRD50			IEPA		
Filter	P	R	F	P	R	F	P	R	F
No Filter	*80.9*	*67.9*	*73.8*	*68.4*	*68.4*	*68.4*	*64.1*	*75.5*	*69.3*
Filter 1	80.8	74.0	77.3	**72.6**	**78.2**	**75.3**	**71.7**	**81.0**	**76.1**
Filter 2	**84.2**	**76.1**	**79.9**	69.3	66.9	68.1	68.7	76.2	72.3
Filter 3	82.8	75.3	78.9	70.2	73.7	71.9	68.6	71.7	70.1
Filter 4	82.8	75.3	78.9	69.9	73.2	71.5	66.6	77.5	71.6
F1 average of all	82.7	75.2	78.8	70.5	73.2	71.7	68.9	76.6	72.53
Diff. to baseline	1.8	**7.3**	5.0	2.1	**4.60**	3.3	**4.8**	1.1	3.3

Considering the difference between averaged recall scores and their corresponding baseline performance scores shown in the last line of Tab. 6.22, the results were even more promising (up to 7% to on the LLL dataset). Indeed, the improvement in recall on all datasets in this experiment supports the working hypothesis raised here that a previous transformation step, carried out before the relation extraction task, can improve overall extraction results, mainly alleviating overfitting of the extraction rules automatically induced by OntoILPER. In other words, the rules induced using the simplified versions of the graphs were able to generalize more examples, as demonstrated by the consistent higher recall values for all dataset, and notably for the LLL and HPRD50 datasets. However, contrarily to what was expected, it was on precision, instead of recall, that the filters contributed more for the IEPA dataset.

Tab. 6.23 summarizes the detailed results of the Tab. 6.22 showing the percentage gain relative to the baseline performance of each filter.

Table 6.23. Relative gain (in F1) of the filters to the baseline.

Relative Gain	**LLL**	**HPRD50**	**IEPA**
Filter 1	3.50	**6.90**	**6.80**
Filter 2	**6.10**	-0.30	3.00
Filter 3	5.10	3.50	0.80
Filter 4	5.10	3.10	2.30
Average relative gain	4.95	3.30	3.23

The highest relative gain in terms of *F1* was obtained on the LLL dataset. Interestingly, the only case in which the *Filter 2* was not good for the PPI extraction occurred on the

dataset HPRD50. By contrast, *Filter 1* achieved the best *F1* score on the HPRD50 and IEPA datasets.

One can draw to the conclusion that, the proposed filters fit well to relation extraction as they significantly improve the relation extraction results in both precision (2.9 in average) and recall (4.36 in average) when evaluated on three PPI corpora used in this study. In addition, *Filter 1* is the best filter to choose in future applications, as they do not remove too many nodes from the graphs, while improving the overall extraction results. However, if one wants to reduce the size of the training data to a minimum possible, while keeping the overall accuracy of the extraction rules, *Filter 4* seems to be the right choice.

Comparative Assessment. A comparative assessment was conducted with the aim of positioning the contribution of this thesis against related work. For these experiments, the same PPI datasets evaluated in the previous section were used. All compared classification models, excluding the ones proposed by us, are based on Support Vector Machines, a state-of-the-art in ML research community. The evaluation is carried out using 10-fold cross-validation.

Besides the traditional evaluation measures of *precision P*, recall *R*, and F1-measure, and the results in terms of AUC are also reported. AUC is as alternative to *F1* score, which is invariant to the class distribution of the test dataset. In other words, AUC is not biased in the case of unbalanced dataset a major difference in positive/negative ratio in the test set [Airola *et al.*, 2008]. In addition, all comparative test to determine whether an improvement of performance is statistically significant or not is based on the t-Student paired test in which null hypothesis is rejected for values of $p <= 0.005$. These comparative results are reported in Tab. 6.24, where the highest scores for *F1* are in bold, whereas the best AUC score are in italics.

The original models without any filter and the best filtered models discussed in the previous section are indicated by the column entitled *New Model (no simplification)* and New *Simple Model* in Tab. 6.24, respectively. As already discussed in the previous section, the former statistically outperforms the latter. Table 6.24 also indicates that the model proposed here ("Simplified Model") is the best on the HPRD50 dataset. On the other hand, the PPI extraction method recently introduced by Quian and Zhou (2012) achieved the best performance on the LLL corpus.

On the IEPA dataset, the simplified model takes the second position in the rank of the best models (*F1*) assessed on the IEPA dataset, being the first position occupied by the model proposed in [Miwa *et al.*, 2010]. It worth stressing that the boost in performance was due to the transformation rules that significantly contributed to the overall second position. Actually, that is the only system analyzed that performs a simplification step somewhat similar to the one proposed in this paper.

Reference [Miwa *et al.*, 2010] proposes a set of sentence simplification rules focuses on entities. Their method for sentence simplification consists of two groups of rules: clause-selection which constructs a simpler sentence by removing noisy information before the relevant clause; and the entity-phrase rules that simplifies an entity-containing region. More concretely, the clause-selection rules remove some marker of relative clauses, copula phrases, and some non-information clauses, while the entity-phrase rules removes coordination, parenthesis involving entities, and appositions.

The work presented here differs from the proposed by Miwa and colleagues [Miwa *et al.*, 2010] mainly with respect to the type of rules, and the target syntactical constructions of the sentences to be simplified. Furthermore, the method presented here is based on the typed-dependencies given by a dependency parser, while in [Miwa *et al.*, 2010] a constituent parser was used. As a result, the rules presented here tend to be simpler in the

sense that the graphs derived by a dependency par. Besides that, they tend also to be more robust to the order of the target entities and clauses in the sentence. It is a well-known fact that contrarily to dependency parsers, rules based on the output of constituent parsers have to consider the exact position of the target elements in a sentence [Buyko *et al*., 2011].

Table 6.24. Comparative evaluation of the RE systems tested on three PPI corpora.

Corpus	**New Model (no simplif.)**		**New Simple Model**		**Miwa *et al.* (2010)**		**Quian/Zhou (2012)**		**Tikk *et al.* (2010)**		**Airola *et al.* (2008)**	
	F1	AUC	F1	AUC	F1	AUC	F1	AUC	F1	AUC	F1	AUC
LLL	73.8	82.0	79.9	85.2	82.9	*90.5*	**84.6**	89.9	79.1	86.8	76.8	83.4
HPRD50	68.4	86.8	**75.3**	*87.6*	75.0	86.6	68.8	83.7	69.7	84.0	63.4	79.7
IEPA	69.3	84.9	76.1	87.2	**77.8**	*88.7*	69.8	82.8	70.7	81.0	75.1	85.1

The bottom line is that the overall results on the simplified sentence representations achieved an average increase of almost 4%, in terms of recall, compared to their original versions. The proposed simplification rules also proved very effective in the sense that they allowed for a statistically significant boost in performance when tested using three biomedical corpora. The obtained results also outperformed some of the state-of-the-art systems evaluated using the same datasets.

6.6. Results and Discussion on Domain Adaptability (EQ 12)

In this experiment, the goal is to investigate to what extent the accuracy performance of OntoILPER can vary among different domains. More precisely, the learned models in OntoILPER were analysed considering the full set of features produced by the text preprocessing step.

Table 6.25 summarizes the result of applying several combinations of features on 2 news domains (reACE 2004 and 2005) and biomedical domain (IEPA dataset).

Starting with the Line 1 in Tab. 6.25, features are incrementally added to the baseline set of features. The legend of the Tab. 6.25 describes what means each component in terms of features types.

Table 6.25. Contribution of different combination of features over relation subtypes in reACE 2004/2005 datasets and over IEPA corpus

		reACE 2004			**reACE 2005**			**IEPA**		
ID	**Combination of features**	**P**	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**
1	Baseline	90.68	66.68	77.77	83.68	50.43	62.91	60.22	72.33	65.72
2	Baseline + *NER*	92.32	67.12	77.99	80.59	51.39	62.68	-	-	-
3	Baseline + *Corpus types*	90.30	71.13	80.31	83.03	63.38	71.86	64.40	75.21	69.39
4	Baseline + *Corpus types* + *NER*	92.23	73.07	81.80	82.30	61.85	70.62	-	-	-
5	Baseline + **Semantic**	93.30	70.68	79.44	82.86	59.71	69.41	62.83	74.70	68.25
6	Baseline + *NER* **+ Semantic**	93.04	75.49	83.06	83.30	60.95	70.40	-	-	-
7	Baseline + *Corpus types* **+ Semantic**	92.20	77.53	83.43	82.29	63.90	71.94	68.91	79.35	73.76
8	Baseline + *Corpus types* + *NER* **+ Sem.**	92.91	79.50	85.39	94.24	62.99	75.10	-	-	-

Baseline = Dep + Chunk + POS + Chunk related
NER = Named entity recognition
Corpus types = golden standard labels of the corpus (reACE and IEPA)
Semantic = SRL, WN synonyms/hypernyms (baseline sense), mapping to WN Domains and SUMO ontology, normalization, similar words, selectional preferences.

Table 6.26. Performance difference between using corpus-based NE and OntoILPER Semantic features

	reACE 2004			reACE 2005			IEPA		
Difference	**Δ P**	**Δ R**	**Δ F**	**Δ P**	**Δ R**	**Δ F**	**Δ P**	**Δ R**	**Δ F**
Line 5 - Line 1	2.62	4.00	1.67	-0.82	9.28	6.50	2.60	2.40	2.54
Line 6 - Line 2	0.72	8.37	5.07	2.71	9.56	7.72	-	-	-
Line 7 - Line 3	1.90	6.40	3.12	-0.74	0.52	0.08	4.50	4.10	4.35
Line 8 - Line 4	0.68	6.43	3.59	11.94	1.14	4.48	-	-	-
Average	1.48	**6.30**	3.36	3.27	5.13	**4.69**	**3.55**	3.25	3.45

First, the baseline group of features (*Baseline*) is composed of syntactical dependencies (*Dep*), chunk information (*Chunk*), POS tagging, and *chunk related* features. NER denotes the typical named entity feature given by the Stanford CoreNLP tools. The other features correspond to corpus type and the semantic features. *Corpus types* denote the golden standard annotations already available in the corpus. For instance, as shown in Section 6.1.1, the reACE corpora provides four types of named entity annotations: ORG, PER, FVW, and GPL, while the IEPA corpus only assigns the label "protein" to each term associated with a protein in the corpus. Finally, the *semantic* features denote the group of semantic related-features available in OntoILPER.

The main goal of this experiment is to compare the several combinations of the above features as shown by the first column of the Tab. 6.25. This table details all results in terms of P/R/F1 for all of the pertinent combinations.

Note that some entries in the IEPA are not available because NER in OntoILPER was trained on news corpora and, the use of NER case for identifying named entities in the IEPA biomedical corpus, is useless.

For the sake of the analysis, Tab. 6.26 summarizes the result in terms of the difference between each pair of lines displayed in its first column. Basically, one want to answer the following question: “can OntoILPER achieve state-of-the-art performance in more than one domain”?

To answer it, the news and biomedical domains were chosen for an initial comparison. The results shown in Tab. 6.26 reveal then some interesting findings.

First, note that adding the semantic features into the learning process improves overall performance in terms of F1 measure for all datasets evaluated. The boost in F1 measure attain almost 5% for the reACE 2005 corpus, and more than 3% for reACE 2004 and IEPA corpus. However, the contribution in terms of precision and recall were unbalanced for the reACE corpora, since the semantic features contributed relatively more in recall than in precision. This contrast with the yielded results on the IEPA corpus, in which precision score was almost equal.

Second, the highest difference in performance was achieved in the reACE 2005 corpus, as the semantic features obtain a gain of almost 12% in precision. However, for the two other combinations in this dataset, adding the semantic features in fact hampers precision.

Third, for both reACE datasets, one can expect a significant increase in recall. This fact confirmed our expectations because the semantic features can be regarded as a mechanism for providing OntoILPER with an extended categorization of named entities in the corpus. In other words, the mapping to semantic recourses like WordNet and SUMO ontology enriches the terms in the corpus with a layer of annotation relating them to senses in WordNet and classes in the SUMO ontology. Thus, it is expected that the term can be categorized into several classes. As a result, the learning component in OntoILPER is more likely to generalize a term given that it has several semantic features attached to it.

In contrast, for the IEPA corpus, the use of semantic features increased precision more than recall. This result led us to inspect the final induced rules to figure this was mainly due to the semantic role features. Indeed, several verbs used to indication an interaction between two proteins in IEPA corpus were correctly annotated along with the roles of its arguments.

Discussion. The problem of using a supervised NER system on data from a different source than the training data was investigated before [Ciaramita *et al*., 2005] [Pyysalo, 2008].

Ciaramita *et al*. (2005) studied the effects of applying a NER system in a novel domain with external lexical knowledge. The authors argued that even working with the same domain, the classification results can be discouraging. For instance, the goal could be to find, people, organizations, and locations in the Wall Street Journal with a tagger trained on the manually annotated portion of the Reuters newswire corpus. Unfortunately, it turns out that even for such similar types of texts the performance of a supervised classifier degrades significantly.

The authors trained several NER classifiers on the CONLL 2003 dataset and evaluated them on a manually annotated section from the Wall Street Journal portion of the Penn Treebank. They demonstrated that the performance of the novel data can be improved by coupling the NER system with a domain-independent dictionary, and simple string similarity features [Ciaramita, 2005].They concluded that the model supported by the domain-independent dictionary was more accurate than the unsupported model in terms of F-score by almost 5%.

Similar results were reported in [Pysalo, 2008] in the biomedical domain. The author evaluated three aspects of the effect of the adaption of a general POS tagger to the biomedical domain with respect to vocabulary coverage, ambiguity, and parsing performance. For that, they provided a morphological extension to the POS tagger by means of rules relating morphological features found in biomedical domain, such as "-ase" (for a the noun kinase). They achieved a relative gain in recall of 10,1% comparing the enhanced version of the POS tagger with the original version.

In an experiment employing and “oracle” knowledge of named entities in support of parsing, the author obtained a relative gain even more interesting of 16.8%.

Comparing the results of OntoILPER with those above mentioned, one can conclude that the average relative gain up to 4% of the former, achieved in two distinct domains, is very encouraging. Actually, statistical significant tests showed that the learned models trained with the semantic features are significantly better than the versions without using them.

6.7. Conclusion

According to the obtained results and the answers to the experimental questions raised at the beginning of this chapter and investigated through several experiments reported here, one can draw to the conclusion that OntoILPER contributes to the state-of-the-art in IE.
This claim is justified by the fact that its unique way of combining a graph-based model for sentence representation, which implies in a highly expressive hypothesis space for extraction rule learning; with multiple ontologies guiding the its IE process.

The solution to IE offered by OntoILPER address some of the open problems in IE area discussed in Section 3.5.6 (final discussion on related work).

In that section, the following major issues for IE were addressed:

- the integration and effective exploitation of rich linguistic knowledge into the IE process (Sections 6.3.1 and 6.6);
- the integration of ontologies for formalizing the expressive relational model of the examples (Section 6.6);
- the legibility of the induced rule model by means of a symbolic extraction rule learning component(Section 6.3.4);
- experimental evaluation with encouraging results on more than one domain of interest (Section 6.4 and 6.5);
- the exploitation of relevant and diverse background knowledge.

Another crucial aspect investigated in the experiments reported in this chapter concerns the portability of OntoILPER to other domains.

Indeed, Section 6.6 was devoted to this particular adaptability problem, in which one can conclude that the rich set of linguistic-based features (syntactic, semantic and structural attributes), properly represented and unified by the Annotation ontology, can effectively contribute to the two most important issues in IE at present time: improving state-of-the-art performance, and promoting a portable solution to IE.

Chapter 7

Conclusions

This thesis focused on the design and implementation of OntoILPER, an ontology- and ILP-based method for extracting instances of entities and relations from textual sources. The proposed solution, in the form of both a symbolic supervised method and framework, contributes to the IE field by offering a unified method for two major IE subtasks: NER and RE.

OntoILPER is the outcome of an investigation aiming to prove the hypothesis that, with an automatic acquisition of a substantial body of linguistic knowledge from textual data and its formalization using ontologies; in combination with an expressive inductive learning technique, it is possible to automatically generate effective models for information extraction, which present the desirable features of being domain-independent as well.

The core learning approach in OntoILPER is guided by semantic structures defined by two ontologies: the domain ontology, and a domain-independent task ontology (Annotation Ontology) that fully characterized OntoILPER as an OBIE method. The rational for using multiple ontologies is that they not only enable an inexpensive adaptation to a new domain, but they also improve the IE process.

Additionally, the inductive learning component in OntoILPER allows prior knowledge about the domain to be easily integrated in the construction process of classification models.

In OntoILPER, the choice for adopting the ILP technique is justified by the following reasons:

- both BK and examples are expressed at the same symbolic level, allowing to enrich the IE process by integrating additional semantic resources, such as a thesaurus or ontologies, without modifying the core of the IE process. For instance, any constraint to the problem can be expressed in the form of auxiliary predicate definitions provided by the user as additional BK. Moreover, the first-order formalism is able to take into account the structural information (relation features) of the examples.
- The final induced extraction rules are expressed in human-readable format, facilitating rule inspection and understanding.

In OntoILPER, examples are classified by reasoning over the lexical, syntactic, semantic, and structural feature dimensions of the training examples. Such rich set of features are formalized by the formal and expressive representation formalism provided by the ontologies.

The IE tasks performed in OntoILPER can be viewed as a form of restricted text understanding, since OntoILPER takes into account the meaning of the instances to be extracted by connecting the candidate instances to ontological structures.

The features that distinguish OntoILPER from related work which are based on the propositional representation formalism of examples are:

- It overcomes the representational limitations of propositional (attribute-value) learning systems that employ a table-based example representation. In such formalism, the representation model of the training examples corresponds to rows in a table, and the features to columns, in which a single value is assigned to each one of the attributes.

– The final induced set of symbolic extraction rules in OntoILPER maps several features (lexical, syntactic, semantic, and structural) of the examples to a predefined set of entity or relation labels.

The goal of this chapter is to answer the research questions raised in the introductory part of this thesis (Section 7.1), to emphasize the scientific contributions made to the IE research are (Section 7.2), to discuss the OntoILPER limitations (Section 7.3), and to outline directions for future work (Section 7.4).

7.1. Answers to the Research Questions

As an attempt to contribute to the IE field, this thesis have investigated the NER and RE tasks under various settings, such as different datasets from two distinct domains, different learning scenarios, and various entity and relation types.

By varying the use of several NLP annotations, and the availability of training data in more than one domain, one could understand how NLP tools and training data influenced the performance of both NER and RE in OntoILPER.

In the following section, the research questions raised in Chapter 1 are revisited, summarizing the findings that answer them.

1. *How can ontologies and semantic resources be exploited to address the challenge of IE systems concerning better overall performance?*

The answer the above question is structured into two parts, one for each question component:

Overall performance with respect to the ontologies. According to what is reported in the Section 6.6, OntoILPER takes advantage of two ontologies:

– the annotation ontology has the explicit role of converting lexical, syntactic, semantic, and structural dependency predicates to BK. This leads to a gain in performance reflected in the experimental results. In fact, the results reported in Section (4.6.2) compares the two versions of OntoILPER with and without such ontology. The enhanced version of the system outperforms the simpler one by a margin up to 5% in the two distinct domains evaluated.

– the domain ontology provides the extraction template and a class hierarchy that enables the system with a coarser granularity of what is being extracted. Its presence also yields an increase in performance due to the latter feature. By analyzing the results of the experiment discussed in Section 6.3.2 which compares the classification results in one level against a two-level hierarchy, the latter outperformed the former, suggesting that it is easier for the system to “typify” more specific rules.

Accuracy performance with respect to the semantic resources. WordNet, WordNet Domains, and VerbNet are used as means to enhance the traditional broad classification of named entities in NER. These resources improve recall, since they can generalize terms, thereby mitigating the problem of term variability (e.g., the word "happiness" is recognized as an “abstraction”, and can be labeled as such instead of having no label). Consequently, the resulting rule generalization can cover more examples and therefore can achieve better recall. One must also be aware that recall is a recurrent problem for many machine learning algorithms, as they tend to construct classification models with

high precision and comparatively lower recall. However, this problem is minored in OntoILPER thanks to the use of the semantic resources.

2. What is the influence of syntactic, semantic, and relational features on RE?

To answer this question, each kind of features will be considered individually:

Influence of syntactic features. Syntactic information is widely used in IE, and RE in special. As already mentioned in Chap. 3, the state-of-the-art RE systems employ different levels of linguistic information varying from POS tags to structures, such as dependency parse trees.

Current RE methods follow the trend of combining syntactic information provided by NLP tools. This is based on the hypothesis that such kind of information can eventually improve performance.

In general, syntactic information is useful for RE as it has been shown earlier [Jiang, 2012] [Zouaq, 2010]. However, it is still unclear which types of syntactic information are best suited for the relation extraction tasks.

In OntoILPER, syntactic features are mainly derived from dependency parsing which provides relations between two entities in an abstract form, including subject-verb-object and noun phrase.

In Sections 6.3.1 and 6.3.4, the contribution of such kind of features was discussed in the context of a RE task concerning nine relations types from the reACE corpora. The conclusion was that the dependency features and noun phrases (or nominal chunks) contributed to raise 6% in overall F1-measure over a model without such features. Combining these two, this figure increases to more than 14%. Therefore, one can conclude that removing syntactic features from the extraction modes would deteriorate performance substantially.

The experimental evaluation on the biomedical corpora displayed very similar results. However, one cannot always expect such similar results when the domain changes. Moreover, even though the Stanford parser in charge of the dependency parsing in OntoILPER preprocessing stage was not trained on the biomedical data, it is quite robust for parsing biomedical corpora.

Influence of semantic features. This question was addressed in Section 6.6 by exploring various semantic resources incorporated into the OntoILPER learning process. It was demonstrated that the combination of syntactic information with semantic knowledge can significantly outperform the experiment baselines. An interesting finding is that, for both domains studied in this thesis, OntoILPER achieved a boost in overall extraction performance (*F1*) around 5%. This clearly indicates that prior knowledge in the form of semantic features constitutes a crucial factor in RE.

Influence of structural features. The distribution of the features (or predicates) in the final induced extraction rules on the reACE datasets revealed that structural information is very useful (see Section 6.3.4).

Detailed examination of the rules revealed that relations can be easily recognized by placing constraints on their arguments, i.e., on their semantic features, while others cannot. Thus, semantic constraints alone do not guarantee accurate relation extraction, and it is widely accepted that the context of the entities in a sentence has to be considered [Roxana *et al.*, 2006]. OntoILPER takes a more realistic and pragmatic approach of integrating syntactic and semantic features into the same model. Thereby, these two kind of features, semantic and structural, are complementary to each other.

3. *What is the role of the ILP-based learning component to both NER and RE tasks?*

The NER and RE tasks can be seen as complementary to each other when aided by ILP, which induces declarative rules referring to predicates:

- NER helps RE (see the reasons presented in the last paragraph concerning the Question 1), and;
- RE supplies NER with previously unrecognized instances of a given class. The reason is that the rule that extracted the named entity instances specifies their role either in the domain or in the range of the relation. Therefore, the argument types are determined by the relation. Put it differently, adding specialized relations with restrictions on the domain and ranges of the class instances, implies in the restriction on types of such instances. This interesting result was also discussed in Section 6.4.1.

4. *What are the benefits of the graph-based model of sentence representation in the final induced classification models?*

This model for sentence representation encompasses relational, i.e., structural features that capture dependencies and structural information that other models face difficulties or simply cannot capture. Comparative assessments reported in Section 6.4.3 and 6.5.2 with other state-of-the-art systems corroborates such statement.

5. *Concerning the graph-based model of sentence representation mentioned above, would it be useful to adopt a simplification strategy before the learning phase?*

The simplification strategy reduces the size of graphs. The ILP learning component then handles a shorter search space without losing its coverage of the most important entities present in a sentence represented by the graph. In other words, due to the proposed transformation/simplification rules presented in Section 4.5, much of the misleading information, in terms of nodes and edges, is eliminated from the graphs. As a result, the reduced graphs, being smaller, can be generalized by rules with less literals in their body part. Indeed, as shown in the experimental assessment reported in Sections 6.5.1 and 6.5.2 the use of the simplified graphs in OntoILPER enhance recall in 4% in most of the cases.

6. *To what extent is OntoILPER portable to other domains?*

In order to port the OntoILPER Framework across domains, the only resource to be replaced is the domain ontology. Then, the ILP-based learning process has to be re-executed with the same BK expressed in the annotation ontology. The new rules concerning the new domain are generated accordingly.

OntoILPER adaptability is easier to achieve than all state-of-the-art OBIE systems based on the manual development of the extraction rules because, in the latter, a human knowledge engineer has to adapt the extraction rules using a given rule representation language, while the extraction rules are automatically learned in OntoILPER. This is justified by the fact that ontologies can be used to represent the context in which the relevant kind of information is naturally embedded. Thus, they can serve both as a specification of relevant information the system has to look for and a conceptualisation of the domain of interest. Moreover, by putting the specification and domain knowledge in an ontology, which can be seen as an external and independent component from the system, future changes in the specification will only require changes in the domain ontology.

Still more relevant is the fact that this new OntoILPER formula of joining morphological, syntactic, and semantic features can improve up to 5% in overall F-1 performance in both of the domains of concern in this thesis. More surprisingly, this

increment in performance is achieved without using the named entities already annotated in the input corpus. The experimental results reported in Section 6.6 demonstrated that.

7.2. Contributions

The research activities conducted in this work resulted in the following contributions to the OBIE area:

On a theoretical point of view, this thesis contributes to the IE area with:

- An OBIE method, called OntoILPER that automatically extracts instances of classes and relations from text sources. OntoILPER benefits from several linguistic-related knowledge sources and additional ontologies (in OWL/DL) that formalize the background knowledge to achieve state-of-the-art performance. OntoILPER distinguishing features, compared to related work, consist in its higher expressivity of extraction rules and the rich set of features that are utilized by an ILP-based learning component for inducing symbolic extraction rules.

  The induced extraction rules map linguistic expressions, or actual linguistic structures delivered by NLP tools to the target entity and relations. The learning component is driven by the target semantic structure of the examples and enables inexpensive adaptation to new domains of interest. Another OntoILPER advantage is that OntoILPER has the potential to extract implicit information.

- A graph-based model for sentence representation that encompasses several types of features, including lexical, syntactic, semantic, and structural ones. This model also takes into account mapping of terms onto linguistic semantic resources and ontologies in a rich and unified model.

- A method for transforming and simplifying graph-based representations of sentences. This simplification method allows improving the performance of overall extraction in terms of precision and recall.

At a more technical level, this thesis made the following contributions:

- OntoILPER Framework, the implementation of the OntoILPER as a framework for ontology population. This framework was implemented as a modular, pipelined architecture that integrates all of the models proposed in this thesis.

- A comprehensive NLP component that integrates various NLP subtasks/tools as a unique component able to perform a comprehensive natural language preprocessing of textual corpora. The current implementation of this component is modular, and extensible to new languages and additional language analysis.

- A hybrid XML-based model for linguistic annotation representation that combines the advantageous aspects of two standard linguistic annotation formats: inline and stand-off annotation.

- The design and implementation of a domain-independent and expressive annotation ontology in OWL/DL, the Annotation Ontology, that formalizing BK during the induction of extraction rules. This ontology formalizes the resultant analysis carried out in the preprocessing stage, in which various kinds of features were mapped to formal structures. Such structures are defined by an expressive knowledge representation formalism that defines all types of background knowledge used by the rule learning component in the proposed solution.

Taken together, the experimental results reported in Chap. 6 confirmed the working hypotheses that:

- given an automatic acquisition of a substantial body of linguistic knowledge from textual data and its formalization using ontologies, in combination with an expressive inductive learning technique, it is possible to automatically generate effective information extraction models

  Actually, the experimental results reported and discussed in Sections 6.3 and 6.4 on three news domain corpora showed that OntoILPER significantly outperforms related work due to its more expressive graph-based sentence representation that not only takes into account the lexical, syntactic, and semantic features (as used by most of the related work), but also considers the *structural* information of the examples. These results were published in reference [*Lima et al.*, 2013].

- a previous transformation/simplification step of the graph-based representations of sentences, carried out before the relation extraction task, can improve overall extraction results, specially alleviating overfitting of the extraction rules.

  Indeed, the overall results on the simplified sentence representations achieved an average increase of almost 4%, in terms of recall, compared to their original versions. The proposed simplification rules also proved very effective in the sense that they allowed for a statistically significant improvement in performance when evaluated on three biomedical corpora. The obtained results also outperformed other state-of-the-art systems that used the same datasets. These results were published in [Lima *et al.*, 2014].

Finally, the experimental assessment also revealed the following findings:

- incorporating ontologies into the IE enables the extraction models to take advantage of additional background knowledge, especially by extending the entity types recognized by traditional NER.
- the information extraction process can be improved by using multiple ontologies.
- the performance of the proposed IE models is consistent not only across both NER and RE tasks, but also across two distinct domains: news and biomedical.
- the graph-based representation model proposed in this thesis leads to significant improvements in extraction performance, outperforming state-of-the-art related work.
- The deep natural language analysis helped to identify the semantic constraints that can be imposed on the relation arguments during RE. Such constraints proved their usefulness by enabling that the overall extraction results can achieve state-of-the-art performance even when the golden standard labels of the named entities present in the corpus are not used.

7.3. OntoILPER Limitations

This section presents the OntoILOPER limitations and briefly discusses strategies to overcome or alleviate them.

Error Propagation in Text Preprocessing Stage. OntoILPER adopts the pipelined architecture commonly used by NLP-based applications. The main problem concerning the

natural language processing stage in OntoILPER is that it is prone to parsing errors. Such errors might hamper the representation of examples and, consequently, the extraction performance by means of noisy patterns introduced in the graph-based representation of the sentences

Indeed, for some domains, specially the biomedical one, the complexity of sentences poses a challenge to natural language parsers, which are commonly trained on large scale corpora of non-technical text (news broadcast texts). Such complexity may be caused by: high average sentence length [Jonnalagadda et al., 2009], inconsistent use of nouns and partial words [Tateisi and Tsujii, 2004], greater lexical density, and increased number of relative clauses and prepositional phrases [Gemoets, 2004]. Pyysalo et al. (2007) demonstrated that, even the simplest simplification steps to counter such problems, may be quite useful. In fact, they showed that by truncating a few dependencies related to noun phrases and also removing non-traditional characters could potentially improve parsing. More recently, [Jonnalagadda et al., 2009] proved the utility of sentence simplification to improve the accuracy of well-known parsers in more than 4%.

Therefore, an obvious idea to be pursed in OntoILPER consists in simplifying the sentences by removing the aforementioned complexities aiming at increasing the performance of different NLP tools.

High Computation Cost of Learning Stage. When applying OntoILPER to large relational datasets, one major problem with using a covering algorithm approach is the amount of time needed to generate a theory. As already stated, searching process for good rules is time intensive, mainly due to the repeated sequential examination of hundreds, or even thousands of clauses to find the best one to be added to the final theory. This is especially pronounced in larger datasets of training examples.

A first approach to alleviate this problem consists in sampling the training example in order to reduce the space to a reasonable size. For that, several sampling techniques have already been proposed [Chawla, 2005], [Liu et al., 2009]. In particular, undersampling techniques [Chawla, 2005] which consists of reducing the number of examples (most commonly negative ones) should be explored. In addition, sample size selection methods [Byrd et al., 2012] for optimizing the ILP-based learning component can also be considered.

Overfitting of the Learned Rules. Machine-learning algorithms often have a tendency to overfit or memorize their training data.

In particular, the standard ILP approach is biased toward producing many high-precision, low-recall clauses, which when combined typically create a high-recall, low-precision theory.

In fact, for several experiments reported in Section 6, the rule inspection revealed that 15 to 20% of the rules were very specialized, i.e., covering at most 2 positive examples.

For tackling this problem, *ensemble methods* [Dietterich, 2000], which are combinations of simpler classifiers, can be focused within ILP on learning diverse (with respect to precision and recall) internal classifiers with for increased quality. Ensemble methods construct multiple classifiers and merge them to provide a consensus prediction for each example, often with higher accuracy. The key idea is that each classifier can compensate for the deficiencies in training of the other classifiers, as they can be combined.

In OntoILPER scenario, the ensemble classifiers can be constructed in many ways:

- **by manipulating the training set**. In this approach, multiple training sets are created by resampling the original data according to some sample distribution. This sampling introduces a bias in each learned hypothesis toward its particular

training set. Thus, these classifiers then vote on the classification of test examples, usually with the majority class being selected as the output classification. A candidate ensemble technique is *bagging* [Breiman, 1996], a popular ensemble method that manipulates the training sets.

- **by manipulating the input features.** Here, a subset of input features is chosen to form each training set. The subset can be either chosen randomly or based on the recommendation of a domain expert. This approach is particularly successful with datasets that contain highly redundant features**.**
- **by manipulating the ILP-based learning algorithm.** The ILP-based learning component in OntoILPER can be manipulated in such a way that it can result in different models by changing some of its parameters. For all approaches seen above, the base classifiers should be generated in parallel, for the sake of the reducing the amount of time in training phase.

7.4. Future Work

The IE methods presented in this thesis can be further explored in several aspects.

First, from the application point of view, OntoILPER can be employed to perform the following text mining tasks:

– **Event Extraction.** The overwhelming majority of IE papers published in the biomedical domain focus on binary interactions between biological named entities. However, with the emergence of corpora providing complex and typed event annotations, more complex IE tasks have also been proposed [Björne *et al.,* 2009].

An example of such complex corpora is the GENIA Event corpus [Kim *et al.,* 2008], which annotates events and static relationships using a more expressive formalism. In this corpus, the type, direction, and the trigger statement in the text stating the relationship (often a verb) are annotated. In addition, events can have more than two participants whose roles are specified, allowing the accurate representation of statements such as "proteins *A*, *B* and *C* form a complex".

In this context, OntoILPER can be applied to other domains and even more complex relation extraction tasks, such as Event Extraction. Event Extraction can be defined as follows: given a set of candidate trigger words, it is necessary to associate them with appropriate entities participating in a given event. Thus, event extraction involves extracting relations having more than two arguments. This intent can be justified thanks to the graph-based model for sentence representation in OntoILPER can be employed to event extraction with very little effort because, as shown in Section 4.4, these graphs capture various distinct forms of annotations from natural language corpora in a unified, yet expressive format. Actually, the proposed unified graph-based model abstracts from the various information extraction tasks and defines a shared representation for the domain-independent layers of linguistic annotation.

For performing event extraction, one should follow the same steps proposed by [Bjorne *et al.,* 2008] and [Heimonen *et al.,* 2008] in representing the semantic annotations as graphs whose nodes correspond to entities and events, and labelled directed edges to their relationships.

The relationship edges describe *themes* and *causes* of events, structural relations between physical entities, such as substructures, and coreferences. To sum up, in order to cast OntoILPER to event extraction, one only needs to generate the edges of the relationship graphs given its nodes, i.e. events and entities.

- **Automatic Summarization.** Automatic summarization of documents aims at producing a shorter version of a source document text, while retaining its main semantic content [Ferreira *et al.*, 2013] [Ferreira *et al.*, 2014].

  Automatic summarization is motivated by the increasing size and availability of digital documents, and the necessity for more efficient methods of information retrieval [Gagnon and Sylva, 2006].

  Methods of automatic summarization include extractive, in which text are summarized by only selecting a limited number of sentenced extracted from the original text; and abstractive, which produces a new shorter text but changing the original text.

  Thanks to the great flexibility of the graph-based sentence representation introduced in Section 4.4, which is able to represent annotated documents at several levels of granularity (e.g., at sentence and token levels), one could easily adapt OntoILPER to perform the automatic extractive summarization task. In this case, it will only be necessary to integrate new features describing sentences, including sentence length, sentence position in the text, number of named entities in a sentence, etc.

The following aspects are related to the way positive and negative training examples are handled in OntoILPER.

- **Active Learning.** The well-known bottleneck of the supervised machine learning approach is the lack of reliable annotated training data, or the need to annotate them. The reason is that manual corpus annotation is a time-consuming process. Since OntoILPER is based on the supervised approach, it shares the same concerns stated above. Therefore, alternative annotation scenarios should be considered as potential solutions to this problem.

  Active Learning [Settles, 2012] is a valuable option because it offers mechanisms to accelerate corpus creation: it only selects examples that are useful for learning. Therefore, OntoILPER could benefit from an additional component for suggesting useful examples based on the active learning strategy.

- **Unbalanced Datasets.** Due to the unbalanced distribution between positive and negatives examples found in the majority of available RE corpora, one possible line of investigation concerns the impact of sampling techniques, mainly undersampling techniques [Chawla, 2005] which would allow speeding up the learning task in OntoILPER framework by reducing the number of negative examples automatically generated.
- **Evaluation Metrics for Unbalanced Datasets.** Another problem, strictly related to the previous one, is related to the evaluation metrics used for ILP systems, especially for unbalanced datasets.

  The most common way to measure performance in large highly-skewed domains is to use precision and recall [Manning & Schutze, 1999], two evaluation metrics which focus on the correct classification of the positive examples. As suggested by Goadrich et al. (2006) a more useful evaluation would be to create a recall-precision curve, to illustrate the trade-off between these two measurements

Other new ideas for follow-up research in OntoILPER include:

- **Inclusion of Additional Simplification Rules.** Despite the encouraging results achieved by the simplification of graph-based sentence representation reported in [Lima *et al.*, 2014], there is room for further work and improvements in different

aspects. First, the evaluation of the simplification rules on other biomedical corpora is an important issue. Second, the inclusion of additional rules to deal with other sentence constructions, including the removal of irrelevant noun appositive, gerundive, and nonrestrictive clauses is also planned.

- **Semantic Data Mining.** It has been widely stated that one of the most important and challenging problems in data mining is incorporation of domain knowledge. Fayyad *et al.* (1996) argue that the use of domain knowledge is important in all stages of the knowledge discovery process. When both data and domain knowledge are available, it is worthwhile to explore their fusion. From this perspective, OntoILPER can be applied to perform Semantic Annotation [Novak *et al.,* 2009] a crucial step towards realizing semantic data mining by bringing meaning to data. Accordingly, future work in this line of research should study the quality of the final extraction rules with respect to the semantic features they contain, and the related classes and relations described by the rules.

At last but not least, future work should address the scalability of OntoILPER Framework. As a proof-of-concept, OntoILPER relies on OWL/DL ontologies for formalizing the background knowledge used in both training and application modes.

Pure OWL/DL reasoners may present scalability problems with large ontologies or, in OntoILPER, with large datasets composed of hundred of documents. However, a valuable alternative consists in taking advantage of ontologies in RDF, even though some expressiveness can be lost in the process.

Contrarily, RDF ontologies are lightweight graph databases that can offer high query performance and the further support SWRL. Therefore, OntoILPER framework should profit of high performance RDF triplestores like Stardog[1] and OpenRDF[2] Sesame to be able to scale up to large input corpora.

[1] Stardog. The enterprise graph database. http://www.stardog.com
[2] OpenRDF Sesame. http://www.openrdf.org

References

[ACE, 2007] Automatic content extraction (ACE) evaluation. http://www.itl.nist.gov/iad/mig/tests/ace.

[ACE, 2004] ACE (2004). Automatic Content Extraction 2004 Evaluation. Available from http://www.itl.nist.gov/iad/mig/tests/ace/2004.

[Adrian *et al.*, 2009] B. Adrian, J. Hees, L. van Elst and A. Dengel, iDocument: using ontologies for extracting and annotating information from unstructured text. In: Proceedings of the 32nd Annual German Conference on AI, Springer-Verlag, Heidelberg, 2009.

[Agichtein and Gravano, 2000] Agichtein E. and Gravano L. Snowball: Extracting relations from large plain-text collections. In Proceedings of the 5th ACM Conference on Digital Libraries, pages 85–94, 2000.

[Airola *et al.*, 2008] Airola A, Pyysalo S, Björne J, Pahikkala T, Ginter F, Salakoski T, All-paths graph kernel for protein–protein interaction extraction with evaluation of cross corpus learning, BMC Bioinformatics 9:S2, 2008.

[Alphonse and Rouveirol, 2000] Alphonse E and Rouveirol C. Lazy propositionalisation for relational learning. In Horn W. (ed.). 14th European Conference on Artificial Intelligence (ECAI'2000), Berlin, Germany, pp. 256-260, IOS Press, 2000.

[Ananiadou and John, 2006] Ananiadou S and John M. Text Mining for Biology and Biomedicine. Artech House, 33:300, 2006.

[Appelt and Israel, 1999] Appelt D. E. and Israel D. J. Introduction to information extraction technology. In Proceedings of the 16th International Joint Conference on Artificial Intelligence (IJCAI), 1999.

[Appelt, 1993] Appelt D. E., Hobbs J. R. , Bear J. , Israel D. J. , and Tyson M. . FASTUS: A finite-state processor for information extraction from real-world text. In IJCAI, pages 1172-1178, 1993.

[Baader *et al.*, 2008] Baader, F. Horrocks, I. and Sattler, U. (2008). Description Logics. In van Harmelen, F., Lifschitz, V., and Porter, B., editors, Handbook of Knowledge Representation, chapter 3, pages 135-180. Elsevier, 2008.

[Bach and Badaskar, 2007] N. and Badaskar S.. A Survey on Relation Extraction. Language Technologies Institute, Carnegie Mellon University, 2007.

[Badica et al, 2005] Badica C., Badica A., and Popescu E. Tuples extraction from HTML using logic wrappers and inductive logic programming. In Proceedings of the Third international conference on Advances in Web Intelligence (AWIC'05), Piotr S. Szczepaniak, Janusz Kacprzyk, and Adam Niewiadomski (Eds.). Springer-Verlag, Berlin, Heidelberg, 44-50, 2005.

[Baeza-Yates and Ribeiro-Neto, 1999] R. A. Baeza-Yates, and B. Ribeiro-Neto, Modern Information Retrieval: Addison-Wesley Longman Publishing, Boston, USA, 1999.

[Banko and Etzioni, 2008] Banko M. and Etzioni O. The trade-off between open and traditional relation extraction. In Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics, pages 28–36, 2008.

[Banko *et al.*, 2006] Banko M., Cafarella M. J., Soderland S., Broadhead M., and Etzioni O. Open information extraction from the Web. In Proceedings of the 20th International Joint Conference on Artificial Intelligence, pages 2670–2676, 2007.

[Beisswanger *et al.*, 2008] E. Beisswanger, S. Schulz, H. Stenzhorn, U. Hahn, BioTop: an upper domain ontology for the life sciences. A description of its current structure, contents and interfaces to OBO ontologies, Applied Ontology 3 (4), 205–212, 2008.

[Bender *et al.*, 2003] Bender O., Och F. J., and Ney H.. Maximum entropy models for named entity recognition. In Proceedings of the 7th Conference on Natural Language Learning, 2003.

[Ben-Dov and Feldman, 2010] Ben-Dov M. and Feldman R.. Text Mining and Information Extraction. In Data Mining and Knowledge Discovery Handbook. pages 809-835, editors Oded Maimon and Lior Rokach, Springer, 2010.

[Bentivogli *et al.*, 2004] Bentivogli L., Forner P., Magnini B. and Pianta E.. "Revising WordNet Domains Hierarchy: Semantics, Coverage, and Balancing", in COLING 2004 Workshop on "Multilingual Linguistic Resources", Geneva, Switzerland, August 28, pp. 101-108, 2004.

[Berardi and Malerba, 2006] Berardi, M. e Malerba D. Learning Recursive Patterns for Biomedical Information Extraction. In: S. Muggleton, R. Otero and A. Tamaddoni-Nezhad (Eds.): Inductive Logic Programming, 16th International Conference on Inductive Logic Programming, ILP 2006, Santiago de Compostela, Spain, August 24-27, 2006. Series: Lecture Notes in Artificial Intelligence 4405 Springer, 79-93.2006.

[Berners-Lee *et al.*, 2001] Berners-Lee, T., Hendler, J., and Lassila, O. The Semantic Web - a new form of Web content that is meaningful to computers will unleash a revolution of new possibilities. In Scientific American, vol. 284, issue 5, pp. 34-43, 2001.

[Bikel *et al.*, 1997] Bikel D. M., Miller S., Schwartz R., and Weischedel R.. Nymble: a high-performance learning name-finder. In Proceedings of the 5th Conference on Applied Natural Language Processing, pages 194–201, 1997.

[BioCreative, 2006] BioCreative - Critical Assessment for Information Extraction in Biology. http://biocreative.sourceforge.net.

[BioNLP, 2009] BioNLP'09 Shared Task on Event Extraction.http://www-tsujii.is.s.u-tokyo.ac.jp/GENIA/SharedTask.

[Byrd *et al.*, 2012] Byrd R. H., Chin G. M., Nocedal J. and Wu Y. Sample size selection in optimization methods for machine learning. Math. Program. 134, 1, August 2012.

[Björne *et al.*, 2008] Björne J., Pyysalo S., Ginter F., and Salakoski T. How complex are complex protein-protein interactions? In Proc. of SMBM'08, pages 125–128, 2008

[Björne *et al.*, 2009]. Björne J. Ginter F., Heimonen J. Pyysalo S., Salakoski T., Learning to Extract Biological Event and Relation Graphs. In: Proceedings of the 17th Nordic Conference of Computational Linguistics NODALIDA 2009, NEALT Proceedings Series Vol. 4 (2009), 18-25, Northern European Association for Language Technology (NEALT), 2009.

[Björne, *et al.*, 2010] Björne J., Ginter F., Pyysalo S., Tsujii J., Salakoski T. Complex event extraction at PubMed scale. Bioinformatics, 2010 June 15; 26(12).

[Blohm, 2010] Blohm S. Large-Scale Pattern-Based Information Extraction from the World Wide Web. Phd. Thesis, Karlsruher Institut of Technology, 2010.

[Bock *et al.*, 2008] Bock J., Haase P., Ji Q. Volz R. Benchmarking OWL Reasoners.. 1FZI Research Center for Information Technologies, Karlsruhe, Germany, 2008.

[Bollacker *et al.*, 2008] Bollacker K., Evans C., Paritosh P., Sturge T., and Taylor J. Freebase: a collaboratively created graph database for structuring human knowledge. In Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data, pages 1247–1250, 2008.

[Bonial *et al.*, 2010] Bonial C., Babko-Malaya O., Choi J. D., Jena Hwang, and Martha Palmer. PropBank Annotation Guidelines. Technical report, University of Colorado at Boulder, 2010.

[Bonial *et al.*, 2011] Bonial C., Corvey W., Palmer M., Petukhova V., and Bunt H. A Hierarchical Unification of LIRICS and VerbNet Semantic Roles. In Proceedings of the 2011 IEEE 5th International Conference on Semantic Computing, ICSC'11, pages 483–489, 2011.

[Borst, 1997] Borst, P. Construction of engineering ontologies for knowledge sharing and reuse. Doctoral Dissertation, Universiteit , Twente, the Netherlands, 1997.

[Bouillon *et al.*, 2002] Bouillon P., Claveau V., Fabre C., Sébillot P.. Acquisition of Qualia Elements from Corpora - Evaluation of a Symbolic Learning Method, Proceedings of the Lexical Ressources and Evaluation Conference (LREC), Las Palmas de Gran Canaria, Spain, May, 2002.

[Bratko, 2000]. Bratko, I. Prolog (3rd Ed.): Programming for Artificial Intelligence. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2000.

[Breiman, 1996] Breiman, L. Bagging Predictors. Machine Learning, 24, 123–140, 1996.

[Brickley and Guha, 2000] Brickley, D., Guha, R.V. Resource Description Framework (RDF) Schema Specification 1.0.W3C Candidate Recommendation, 27 March, 2000.

[Brin, 1998] Brin S. Extracting patterns and relations from the World Wide Web. In Proceedings of the 1998 International Workshop on the Web and Databases, 1998.

[Brown and Kros, 2003] Brown, M. L. and Kros, J. F. Data Mining and the Impact of Missing Data. Industrial Management and Data Systems, 103(8): 611-621, 2003.

[Bui, 2012] Bui, Q. C. Relation Extraction Methods for Biomedical Literature. Faculty of Science, University of Amsterdam, 2012.

[Buitelaar and Siegel, 2006] Buitelaar P. and Siegel M., Ontology-based Information Extraction with SOBA. In: Proceedings of the Fifth International Conference on Language Resources and Evaluation, European Language Resources Association, Genoa, Italy, 2006.

[Buitelaar *et al.*, 2008] Buitelaar, P., Cimiano, P., Frank, A., Hartung, M., Racioppa, S. Ontology-based Information Extraction and Integration from Heterogeneous Data Sources. International Journal of Human Computer Studies (JHCS), 66 (pp. 759–788), 2008.

[Bunescu and Mooney, 2005] Bunescu R. C. and Mooney R. J. A shortest path dependency kernel for relation extraction. In Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing (HLT '05). Association for Computational Linguistics, Stroudsburg, PA, USA, 724-731, 2005.

[Bunescu and Mooney, 2006] Bunescu, R. C. and Mooney, R. Subsequence kerne ls for relation extraction. In Advan ces in N eural In formation Processing S ystems 1 8 (NIPS'06) , pages 171–178, 2006.

[Buyko *et al.*, 2011] Buyko, E., Faessler, E., Wermter, J., and Hahn, U. 2011. Syntactic Simplification and Semantic Enrichment: Trimming Dependency Graphs for Event Extraction. Computational Intelligence, 27(4): 610-644, 2011.

[Califf and Mooney, 1998] Califf, M. E., and Mooney, R. J. Relational learning of pattern-match rules for information extraction. In Working Notes of AAAI Spring Symposium on Applying Machine Learning to Discourse Processing. pp.6-11, 1998.

[Califf and Mooney, 2003] Califf, M. E., and Mooney, R. J. Bottom-up relational learning of pattern matching rules for information extraction. Journal of Machine Learning Research. Vol.4, pp.177-210, 2003.

[Caporaso *et al.*, 2008] Caporaso, J. G., Deshpande, N., Fink, J. L., Bourne, P. E., Cohen, K. B., Hunter, L. Intrinsic evaluation of text mining tools may not predict performance on realistic tasks. In Proceedings of Pacific Symposium on Biocomputing 2008, pp. 640-651, 2008.

[Carroll *et al.* 1998] Carroll, J., Minnen, G., Canning, Y., Devlin, S., and Tait, J. Practical simplification of English newspaper text to assist aphasic readers. In Proceedings. of AAAI, 7–10, 1998.

[Carston, 2002] Carston, R. Thoughts and Utterances: The Pragmatics of Explicit Communication. Oxford: Blackwell, 2002.

[Castano *et al.*, 2008] Castano, S., Peraldi, I.S.E., Ferrara, A., Karkaletsis, V., Kaya, A., Möller, R., Montanelli, S., Petasis, G., Wessel, M. (2008). Multimedia Interpretation for Dynamic Ontology Evolution. Journal of Logic and Computation, 2008.

[Celjuska and Vargas-Vera, 2004] Celjuska D. and Vargas-Vera M. Ontosophie: A semi-automatic system for ontology population from text, Proc. of International Conference on Natural Language Processing ICON, vol.4, 2004.

[Chambers and Jurafsky, 2011] Chambers, N. and Jurafsky, D. (2011). Template-based information extraction without the templates. In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, pp. 976–986, 2011.

[Chan and Roth, 2010] Chan, Y.S. and Roth D. Exploiting background knowledge for relation extraction. In Proceedings of the 23rd International Conference on Computational Linguistics, pp. 152–160, 2010.

[Chandrasekar and Srinivas, 1997] Chandrasekar, R., and Srinivas, B. Automatic induction rules for text simplification. Knowledge-Based Systems, vol. 10, 183–190, 1997.

[Charniak *et al.*, 1993] Charniak, E., Hendrickson, C, Jacobson, N., and Perkowitz, M. Equations for part-of-speech tagging. In Proceedings of the 11th National Conference on Artificial Intelligence (AAAI), pages 784-789, 1993.

[Chawla, 2005] Chawla, N. V. Data mining for imbalanced datasets: An overview. In Data mining and knowledge discovery handbook (pp. 853-867). Springer US, 2005.

[Chieu and Ng, 2003] Chieu H. L. and Ng H. T. Named entity recognition with a maximum entropy approach. 7th Conference on Natural Language Learning, pages 160–163, 2003.

[Chieu *et al.*, 2003] Chieu, H., H. Ng, and Y. Lee. Closing the Gap: Learning-Based Information Extraction Rivaling Knowledge-Engineering Methods. In Proceedings of the 41th Annual Meeting of the Association for Computational Linguistics, 2003.

[Choi *et al.*, 2009] Choi S-P, Jeong C-H, Choi Y-S, Myaeng S-H. Relation extraction based on extended composite kernel using flat lexical features. JKIISE: Software Application, 36:8, 2009.

[Choi *et al.*, 2013] Choi, S. P., Lee. S., Jung, H., Song, S. (2013). An intensive case study on kernel-based relation extraction. Proceedings of Multimedia Tools and Applications, Springer, US, (pp. 1-27), 2013.

[Church, 1988] Church, K. A stochastic parts program and noun phrase parser for unrestricted text. In Proceedings of the Applied Natural Language Processing Conference (ANLP), pages 136-143, 1998.

[Ciaramita and Altum, 2005] Ciaramita M., Altun Y.. Named-entity recognition in novel domains with external lexical knowledge. Advances in Structured Learning for Text and Speech Processing Workshop (NIPS). 2005.

[Ciaramita and Altun, 2006] Ciaramita M., Altun Y.: Broad-Coverage Sense Disambiguation and Information Extraction with a Supersense Sequence Tagger. EMNLP 2006: 594-602, 2006.

[Cimiano *et al.*, 2004] Cimiano, P., Handschuh, S., and Staab, S. (2004). Towards the self-annotating web. WWW 2004, pp. 462–471, 2004.

[Cimiano *et al.*, 2005] Cimiano, P., Saric, J., and Reyle, U. Ontology-driven discourse analysis for information extraction. Data and Knowledge Engineering, 55(l):59-83, 2005.

[Cimiano *et al.*, 2005] Cimiano P., Ladwig G., and Staab S.. Gimme' the context: context-driven automatic semantic annotation with C-PANKOW. In: Proceedings of the 14th International Conference on World Wide Web, ACM, New York, 2005.

[Cimiano, 2006] Cimiano, P. Ontology Learning and Population from Text: Algorithms, Evaluation and Applications. Springer-Verlag, New York, 2006.

[Ciravegna, 2001] Ciravegna, F. $(LP)^2$, an adaptive algorithm for information extraction from Web-related texts. In Proceedings of the IJCAI-2001 Workshop on Adaptive Text Extraction and Mining held in conjunction with 17th International Joint Conference on Artificial Intelligence (IJCAI), Seattle, USA, 2001.

[Ciravegna, 2001] Ciravegna F. Adaptive information extraction from text by rule induction and generalisation. In Proceedings of the 17th In- ternational Joint Conference on Artificial Intelligence - Volume 2, pages 1251–1256, 2001.

[Collier et al., 2000] N. Collier, C. Nobata, and J.-i. Tsujii. Extracting the names of genes and gene products with a hidden Markov model. In Proceedings of the 18th Conference on Computational Linguistics Volume 1, pages 201–207, 2000.

[Collins and Dufft, 2001] Collins, M. and Duffy, N. (2001). Convolution Kernels for Natural Language. NIPS'2001, (pp. 625-632). Cambridge, MA. 2001.

[Cullota and Sorensen, 2004] Culotta, A. and Sorensen, J. (2004). Dependency tree kernels for relation extraction. ACL'2004. (pp. 423-429). 21-26 July 2004. Barcelona, Spain. 2004.

[Cunningham, 2006] Cunningham H.. Information Extraction, Automatic. Encyclopedia of Language and Linguistics, 2nd Edition, 5:665-677, November, 2006.

[Curran and Clark, 2003] James R. Curran and Stephen Clark. Language independent NER using a maximum entropy tagger. In Proceedings of the 7th Conference on Natural Language Learning, 2003.

[Cussens and Dzeroski, 2000] Cussens J., Dzeroski S. (Eds.): Learning Language in Logic. Springer Lecture Notes in Computer Science, 2000.

[Cussens and Pulman, 2000] Cussens J. and Pulman S. Experiments in Inductive Chart Parsing. In J. Cussens and S. Dzeroski, Eds., Learning Language in Logic , volume 1925 de Leture Notes in Computer Science, p. 143-156. Springer-Verlag, 2000.

[Daellemans *et al.*, 1999] Daelemans, W., Buchholz, S. and Veenstra, J. Memory-based Shallow Parsing. In Proceedings of the Conference on Computational Natural Language Learning, CoNLL-99, 1999.

[De Marnerffe and Manning, 2008] De Marneffe, M. C. and Manning, C. D. (2008). Stanford Dependencies Manual, 2008.

[De Raedt, 2010] De Raedt, L. Inductive Logic Programming. Encyclopedia of Machine Learning, 2010, pp. 529-537, 2010.

[Dedek, 2012] Dedek J. Semantic Annotations. PhD thesis, Charles University in Prague, Czech Republic, 2012.

[Dehasp and Toironen, 2000] Dehasp L. and Toironen H.. Relational Data Mining, chapter Discovery of relational association rules, pages 189208. Springer Verlag, 2000.

[DeRaedt, 2010] Raedt L. D.: Inductive Logic Programming. Encyclopedia of Machine Learning 2010: 529-537, 2010.

[Dietterich, 2000] Dietterich, T. G. Ensemble Methods in Machine Learning. Lecture Notes in Computer Science, 1857, 1–15, 2000.

[Ding *et al.*, 2002] Ding J., Berleant, D., Nettleton, D., and Wurtele, E. Mining MEDLINE: abstracts, sentences, or phrases? Proc. of the Pacific Symposium on Biocomputing, 326-337, 2002.

[Dipper *et al.*, 2007] Dipper, S., Götze, M., Küssner, U., and Stede, M. (2007). Representing and querying standoff XML. In Proceedings of the GLDV-Frühjahrstagung, Tübingen, Germany, 2007.

[Dlugolinsky *et al.*, 2013] Dlugolinský, S., Ciglan, M., Laclavík, M. Evaluation of Named Entity Recognition Tools on Microposts. In INES 2013 : 17th IEEE International Conference on Intelligent Engineering Systems 2013. Budapest : IEEE Industrial Electronic Society, p. 197-202, 2013.

[Drozdzynski *et al.*, 2005] Drozdzynski W, Krieger H.-U., Piskorski J., Schäfer U.. SProUT - a General-Purpose NLP Framework Integrating Finite-State and Unification-based Grammar Formalisms Proceedings of the 5th International Workshop on Finite-State Methods and Natural Language Processing, Lecture Notes in Artificial Intelligence number 4002, Pages 302-303, Helsinki, Finland, Springer - Lecture Notes in Artificial Intelligence, Berlin/Heidelberg, 9,2005.

[Dzeroski *et al.*, 2001] Džeroski S., Cussens J., and Manandhar S. 2001. An introduction to inductive logic programming and learning language in logic. In Learning language in logic, James Cussens and Sašo Džeroski (Eds.). Springer-Verlag New York, Inc., Secaucus, NJ, USA 3-35, 2001.

[Dzeroski, 2010] Dzeroski, S. Relational Data Mining. Data Mining and Knowledge Discovery Handbook. Maimon, Oded and Rokach, Lior (eds.), Springer, US, pp. 887-911, 2010.

[Elmasri and Navathe, 2010] Elmasri R. and Navathe S., 2010. Fundamentals of Database Systems (6th edition). Addison-Wesley Publishing Company, USA, 2010.

[Embley, 2004] D.W. Embley, Toward semantic understanding: an approach based on information extraction ontologies. In Proceedings of the 15th Australasian database conference, (Australian Computer Society, Darlinghurst, Australia, 2004.

[Erk and Pado, 2004] Erk, K. and Pado, S. A Powerful and Versatile XML Format for Representing Role-semantic Annotation. LREC 2004, 2004.

[Fader *et al.*, 2011] Fader A., Soderland S., and Etzioni O. Identifying relations for open information extraction. In Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, pages 1535–1545, 2011.

[Faure and Nedellec, 1999] Faure D. and Nedellec C. Knowledge Acquisition of Predicate Argument Structures from Technical Texts using Machine Learning: the System ASIUM. InD. F. R. Studer, Ed., Proceedings of the 11th European Workshop EKAW'99, Dag-stuhl, Allemagne : Springer-Verlag, 1999.

[Fayyad, 1996] U. Fayyad, G. Piatetsky-shapiro, and P. Smyth. From data mining to knowledge discovery in databases. AI Magazine , 17:37–54, 1996.

[Feldman and Sanger, 2007] Feldman, R., and Sanger, J. The Text Mining handbook – advanced approaches in analyzing unstructured data, 1st ed., Cambridge, Cambridge University Press, 2007.

[Fellbaum, 1998] Fellbaum, C. D. WordNet – An Electronic Lexical Database. Language, Speech and Communication. MIT Press, 1998.

[Fensel *et al*., 2002] Fensel, D., Bussler, C., Ding, Y., Kartseva, V., Klein, V., Korotkiy, M., and Omelayenko, B. (2002). Semantic web application areas. In Proceedings of the 7th International Workshop on Applications of Natural Language to Information Systems (NLDB), Stockholm, Sweden, 2002.

[Ferreira *et al*., 2013] Ferreira, R., Cabral, L. S., Lins, R. D., Silva, G. F., Freitas, F., Cavalcanti, G. C., Lima, R., Simske, S. J., Favaro, L. 2013. Assessing sentence scoring techniques for extractive text summarization, Expert Systems with Applications, Volume 40, Issue 14, (October), 5755-5764, 2013.

[Ferreira *et al.*, 2014] Ferreira. R., Cabral, L. S., Freitas, F., Lins, R. D., Silva, G. F., Simske, S. J., Favaro, L. 2014. A multi-document summa-rization system based on statistics and linguistic treatment, Expert Systems with Applications, Volume 41, Issue 13, (Oc-tober), 5780-5787, 2014.

[Fillmore, 1968] Fillmore, C. J. (1968). The Case for Case. In Bach, E. and Harms, R., editors, Universals in Linguistic Theory. Holt, Rinehart, and Winston, New York, 1968.

[Fillmore, 1976] Fillmore C. J., Frame semantics and the nature of language, Annals of the New York Academy of Sciences 280 (1) (1976) 20–32, 1976.

[Finn, 2006] Finn, A. (2006). Multi-Level Boundary Classification Approach to Information Extraction, Phd thesis, University College Dublin, 2006.

[Fiorentini *et al.*, 2010] Fiorentini, X., Sudarsan, R., Suh, H., Lee, J., Sriram, R.: An analysis of description logic augmented with domain rules for the development of product models. Journal of Computing and Information Science in Engineering 10, 1–13, 2010.

[Fonseca, 2006]. Nuno A. F. Parallelism in Inductive Logic Programming Systems. PhD thesis, University of Porto, 2006.

[Fortineau *et al.*, 2012] Fortineau V., Paviot T., Louis-Sidney L., and Lamouri S. SWRL as a rule language for ontology-based models in power plant design. International Conference on Product Lifecycle Management (PLM 12) IFIP, Montréal, Canada, 2012.

[Frasconi *et al.*, 2012] Frasconi, P., Costa, F., Raedt, L. D. and Grave, K. D. (2012). kLog: A Language for Logical and Relational Learning with Kernels. CoRR, abs/1205.3981, 2012.

[Freitag, 1998] Freitag, D. Machine learning for information extraction in informal domains. Ph.D. thesis, Computer Science Department. Carnegie Mellon University, 1998.

[Freitag, 2000] Freitag,D. and Kushmerick, N. Boosted wrapper induction. In Proceedings of the ECAI Workshop on Machine Learning for Information Extraction, 2000.

[Freitas, 2002] Freitas, F. Sistemas multiagentes cognitivos para a recuperação e extração integradas de informação da web. Tese de Doutorado, Programa de pós-graduação em Engenharia Elétrica da Universidade Federal de Santa Catarina (UFSC), Florianópolis, 2002.

[Freitas, 2003] Freitas, F. Ontologias e a web semântica. In Proceedings of the XXIII Congresso da Sociedade Brasileira de Computação, pp. 1-52, Campinas, São Paulo, Brasil, 2003.

[Fukuda *et al.*, 1998] Fukuda K., Tamura A., Tsunoda T., and Takagi T. Toward infor mation extraction: Identifying protein names from biological papers. In Pacific Symposium on Biocomputing, pages 707–718, 1998.

[Fundel *et al.*, 2007] Fundel, K., Kuffner, R., and Zimmer, R. RelEx Relation extraction using dependency parse trees. Bioinformatics, 23(3), 365-371, 2007.

[Fürnkranz *et al.*, 2012] Fürnkranz, J., Gamberger, D. and Lavrac, N. (2012). Foundations of Rule Learning, Springer-Verlag, 2012.

[Gagnon and Sylva, 2006] Gagnon M., Sylva L. Text Compression by Syntactic Pruning. 19th Canadian Conference on Artificial Intelligence, Québec, Canada, 2006.

[Gangemi *et al.*, 2001] Gangemi A., Guarino N., and Oltramari A. Conceptual analysis of lexical taxonomies: the case of WordNet top-level. In Proceedings of the international conference on Formal Ontology in Information Systems - Volume 2001 (FOIS '01), Vol. 2001. ACM, New York, NY, USA, 285-296, 2001.

[Gaspar *et al.*, 2012] Gaspar P., Carbonell, J. and Oliveira, J. L. On the parameter optimization of Support Vector Machines for binary classification. Journal of Integrative Bioinformatics, 9(3):201, 2012.

[Gemoets, 2004] Gemoets, D. Assessing Readability of Consumer Health Information: An Exploratory Study. MEDINFO, 2004.

[Genesereth and Fikes, 1992]. M. R. Genesereth, R. E. Fikes (Editors). Knowledge Interchange Format, Version 3.0 Reference Manual. Computer Science Department, Stanford University, Technical Report Logic-92-1, June,1992.

[Gerdes *et al.*, 2014] K. Gerdes, Hajicova E., and Wanner L. Computational Dependency Theory. IOS Press, Amsterdam, The Netherlands, 2014.

[Gildea and Jurafsky, 2002] Gildea D. and Jurafsky D. Automatic Labeling of Semantic Roles. Computational Linguistics, 28(3), 2002.

[Giuliano and Lavelli, 2007] Giuliano, C., A. Lavelli and L. Romano. Relation Extraction and the Influence of Automatic NER, ACM Transactions on Speech and Language Processing, vol 5, no.1, ACM, 2007.

[Goadrich *et al.*, 2006] Goadrich M., Oliphant L., Shavlik J. W.: Gleaner: Creating ensembles of first-order clauses to improve recall-precision curves. Machine Learning 64(1-3): 231-261, 2006.

[Grau et al, 2008] OWL2: The next step for OWL. Grau B. C, Horrocks I., Motik, B., Parsia B., Patel-Schneider P., Sattler U. In Journal of Web Semantics: Science, Services and Agents on the World Wide Web. Volume 6, Issue 4, November 2008, Pages 309–322, 2008.

[Grefenstette, 1994] Grefenstette, G. Explorations in Automatic Thesaurus Construction. Kluwerm 1994.

[Grenon *et al.*, 2004] Grenon, P., B. Smith, and L. Goldberg. Biodynamic ontology: Applying bfo in the biomedical domain. In D.M. Pisanelli, editor, Ontologies in Medicine, Studies in Health Technology and Informatics,102:20–38. IOS Press, Amsterdam, the Netherlands, 2004.

[Grishman and Sundheim, 1996] Grishman R. and Sundheim B. Message understanding conference-6: A brief history. In Proceedings of the 16th International Conference on Computational Linguistics, pages 466–471, 1996.

[Grishman and Yangarber, 2000] R. Grishman and R. Yangarber. Issues in corpus-trained information extraction. In In Proceedings of International Symposium: Toward the Realization of Spontaneous Speech Engineering, pages 107–112, February, 2000.

[Gruber, 1993] Gruber, T. Towards Principles for the Design of Ontologies used for Knowledge Sharing. Int. Workshop on Formal Ontology in Conceptual Analysis and Knowledge Representation. Kluwer Academic Publishers, Deventer, The Netherlands, 1993.

[Guarino, 1998] Guarino, N. Formal ontology and information systems. In Proceedings of Formal Ontology in Information Systems (FOIS), Trento, Italy, pp. 3–15, 1998.

[Hachey et al, 2011] Hachey, B., Grover, C. and Richard Tobin, R. (2011). Datasets for Generic Relation Extraction. Journal of Natural Language Engineering. Cambridge University Press. 2011

[Hahn and Wermter, 2006] Hahn, U., Wermter, J. "Levels of Natural Language Processing for Text Mining". In: Ananiadou, S., McNaught, J. (eds.) Text Mining for Biology and Biomedicine. Norwood, MA: Artech House, pp. 13-41, 2006.

[Hakenberg, 2009]. Hakenberg J: Mining Relations from the Biomedical Literature. PhD Thesis 2009:179, 2009.

[Hand, 1997] Hand, D.J. onstructing and assessment of classification rules. Chichester: Wiley Series in Probability and Statistics. 1997.

[Hanley and Mcneil, 1982] Hanley, J. A., and McNeil, B. J. 1982. The meaning and use of the area under a receiver operating characteristic (ROC) curve. Radiology, 143(1), 29-36, 1982.

[Hao *et al.*, 2005] Hao Y, Zhu X, Huang M, Li M: Discovering patterns to extract protein-protein interactions from the literature: Part II. Bioinformatics 2005, 21:3294-300, 2005.

[Harabagiu *et al.*, 2005] Sanda Harabagiu, Cosmin Adrian Bejan, and Paul Morarescu. 2005. Shallow semantics for relation extraction. In Proceedings of the 19th international joint conference on Artificial intelligence (IJCAI'05). Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1061-1066, 2005.

[Harris, 1968] Harris, Z. Mathematical Structures of Language. Wiley, 1968.

[Hearst, 1992] M. A. Hearst, Automatic acquisition of hyponyms from large text corpora. In: Proceedings of the 14thConference on Computational linguistics, ACM, New York, 1992.

[Heimonen *et al.*, 2008] J. Heimonen, S. Pyysalo, F. Ginter, and T. Salakoski. Complex-to-pairwise mapping of biological relationships using a semantic network representation. In Proc. of SMBM'08, 2008.

[Hilbert *et al.*, 2006] Hilbert M, Lobin H, Bärenfänger M, Lüngen H, Puskas C. A text-technological approach to automatic discourse analysis of complex texts. In: Proceedings of KONVENS 2006, Konstanz, 2006.

[Hirschman, 1998] Hirschman, L., "The Evolution of Evaluation: Lessons from the Message Understanding Conferences," Computer Speech and Language, Vol. 12, 1998, pp. 281–305, 1998.

[Hitzler and Parsia, 2009]. Hitzler, P., and Parsia, B. Ontologies and Rules. Handbook on Ontologies, pp. 111-132, 2009.

[Hitzler *et al.*, 2009] Hitzler, P., Krötzsch, M., Parsia, B., Patel-Schneider, P.F., Rudolph, S. (editors) (2009). OWL 2 Web Ontology Language Primer. W3C Working Draft, Jun 11 2009. http://www.w3.org/TR/owl2-primer

[Hobbs *et al.*, 1997] J Hobbs. R., Appelt D., Bear J., Israel D., M., M. Stickel, and Tyson M. Fastus: A cascaded finite-state transducer for extraction information from natural language text. In E Roche and Y Schabes, editors, Finite-State Language Processing, chapter 13, pages 383–406. MIT Press, 1997.

[Horrocks et al, 2010] Horrocks, I., Patel-Schneider, P., Boley, H., Tabet, S., Grosof, B., Dean, M.: SWRL: A Semantic Web Rule Language combining OWL and RuleML, 2010.

[Horvath *et al.*, 2009] Horváth, T., Paass, G. , Frank Reichartz, Stefan Wrobel: A Logic-based Approach to Relation Extraction from Texts. ILP 2009: 34-48, 2009.

[Hoste et al, 2002] Hoste, V., Hendrickx, I., Daelemans, W., van den Bosch, A. Parameter Optimization for Machine-Learning of Word Sense Disambiguation. Natural Language Engineering, Cambridge University Press, 8:311-325,2002.

[Humphreys *et al.*, 2000] Humphreys K., Demetriou G., and Gaizauskas R.. Two applications of information extraction to biological science yournal articles: Enzyme interactions and protein structures. In Pacific Symposium on Biocomputing, pages 502–513, 2000.

[Horrocks and Patel-Schneider, 2009] Horrocks I. and Patel-Schneider P. F. A proposal for an OWL rules language. In Proc. of the Thirteenth International World Wide Web Conference (WWW 2004). ACM, New York, 2004, pages 723–731, 2009.

[Ide *et al.*, 2003] Ide,N.,Romary,L., and de la Clergerie, E. (2003). International standard for a linguistic annotation framework. In Proceedings of HLT-NAACL03 Workshop on The Software Engineering and Architecture of LanguageTechnology, 2003.

[Isozaki and Kazawa, 2002] Isozaki, H. and Kazawa, H. (2002). Efficient support vector classifiers for named entity recognition. 19th International Conference on Computational Linguistics, 2002.

[Jang *et al.*, 2006] Jang H, Lim J, Lim J-H, Park S-J, Lee K-C, Park S-H: Finding the evidence for protein-protein interactions from PubMed abstracts. Bioinformatics 2006, 22:e220-6, 2006.

[Jiang and Zhai, 2007] Jiang, J. and Zhai, C. X. (2007). A systematic exploration of the feature space for relation extraction. Annual Conference of the North American Chapter of the Association for Computational Linguistics, NAACL-HLT'2007, Rochester, NY, USA, 2007 pp. 113–120,2007.

[Jiang, 2012] Jiang, J. Information Extraction from Text, in C.C. Aggarwal and C.X. Zhai (eds), Mining Text data, Chap. 2 (pp. 11-41), 2012.

[Joachims, 1999] Joachims, T. Making large-Scale SVM Learning Practical. Advances in KernelMethods - Support Vector Learning, B. Schölkopf and C. Burges and A. Smola (ed.), MIT-Press, 1999.

[Johansson and Nugues, 2008] Johansson R. and Nugues P. Dependency-based Semantic Role Labeling of PropBank. In Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing (EMNLP'08), pages 69–78, 2008.

[Jonnalagadda and Gonzalez, 2009] Jonnalagadda, S. and Gonzalez, G. Sentence Simplification Aids Protein-Protein Interaction Extraction. In Proc. of the 3rd International Symposium on Languages in Biolo-gy and Medicine (LBM'09), (November), 8-10, 2009.

[Jonnalagadda *et al.*, 2009] Jonnalagadda, S., et al. Towards effective sentence simplification for automatic processing of biomedical text. NAACL HLT, 2009.

[Junker *et al.*, 2000] Junker M., Sintek M. and Rink M. Learning for Text Categorization andInformation Extra tion with ILP. In J. Cussens and S. Dzeroski, Eds., LearningLanguage in Logic, volume 1925 de Leture Notes in Computer Science, p. 247{258.Springer-Verlag, 2000.

[Jurafsky and Martin, 2009] Jurafsky, D., and Martin J. H. Speech and Language Processing: An Introduction to Natural Language Processing, Speech Recognition, and Computational Linguistics. 2nd edition. Prentice-Hall, 2009.

[Kambhatla, 2004] Kambhatla, N. Combining lexical, syntactic and semantic features with Maximum Entropy models for extracting relations. ACL'2004 (Poster), 21–26 July 2004, Barcelona, Spain, 2004 (pp. 178–181), 2004.

[Karkaletsis *et al.*, 2011] Karkaletsis V., Fragkou P., Petasis G., and Iosif E. (2011). Ontology Based Information Extraction from Text. Paliouras G. *et al.* (Eds.) Multimedia Information Extraction, LNAI 6050 (pp. 89–109), 2011.

[Kavurucu, 2011] Kavurucu Y., Senkul P., Toroslu I. H.: A comparative study on ILP-based concept discovery systems. Expert Syst. Appl. 38(9): 11598-11607, 2011.

[Kazakov and Manandhar, 2001] Kazakov D. and Manandhar S. Unsupervised Learning of Word Segmentation Rules with Genetic Algorithms and Inductive Logic Programming. Machine Learning, 43, 121-162.Kietz J.-U. 2001.

[Kazama *et al.*, 2002] Kazama J., Makino T., Ohta Y., and Tsujii J.. Tuning support vector machines for biomedical named entity recognition. In Proceedings of the ACL-02 Workshop on Natural Language Processing in the Biomedical Domain - Volume 3, pages 1–8, 2002.

[Kietz *et al.*,2000] Kietz J., Maedche A., and Volz R., A method for semi-automatic ontology acquisition from a corporate intranet. In: Proceedings of the EKAW'00 Workshop on Ontologies and Text, (Springer, Berlin, 2000.

[Kim *et al.*, 2002] Kim, S., Alani, H., Hall, W., Lewis, P., Millard, D., Shadbolt, N., Weal, M.: Artequakt: Generating Tailored Biographies from Automatically Annotated Fragments from the Web. In: Proceedings of Workshop on Semantic Authoring, Annotation and Knowledge Markup (SAAKM 2002), the 15th European Conference on Artificial Intelligence (ECAI 2002), Lyon, France, pp. 1–6, 2002.

[Kim *et al.*, 2007] Kim, J.-H., Mitchell, A., Attwood, T. K. and Hilario, M. Learning to extract relations for protein annotation.. ISMB/ECCB, Supplement of Bioinformatics, pp. 256-263 2007.

[Kim *et al.*, 2008] Kim J-D., Ohta T., and Tsujii J. Corpus annotation for mining biomedical events from literature. BMC Bioinformatics, 9(1):10, 2008.

[King *et al.*, 1992] King R.D, Muggleton S., and Sternberg M.J.E.. Drug design by machine learning: The use of inductive logic programming to model the structure-activity relationships of trimethoprim analogues binding to dihydrofolate reductase. In Proceedings of the National Academy of Sciences, volume 89, pages 1132211326, 1992.

[King, 2004] King R. D. Applying inductive logic programming to predicting gene function. AI Magazine, 25(1):57-68, 2004.

[Kinoshita and Cohen, 2005] Kinoshita S., Cohen K. B., Ogren P., and Hunter L.. BioCreAtIvEtask 1A: Entity identification with a stochastic tagger. BMC Bioinformatics, 6(Suppl 1):S4, 2005.

[Kipper *et al.*, 2006] Kipper K., Korhonen A., Ryant N., and Palmer M.. Extensive Classifications of English verbs. Proceedings of the 12th EURALEX International Congress. Turin, Italy. September, 2006.

[Kipper-Schuler, 2005] Karin Kipper-Schuler, K. VerbNet: A broad-coverage, comprehensive verb lexicon. PhD. Thesis. Computer and Information Science Dept., University of Pennsylvania. Philadelphia, PA, 2005.

[Klein *et al.*, 2003] Klein D., Smarr J., Nguyen H., and Manning C. D. Named entity recognition with character-level models. In Proceedings of the 7th Conference on Natural Language Learning, 2003.

[Kohavi and John, 1995] Kohavi, R. and John, G. H. Automatic parameter selection by minimizing estimated error. 12th International Conference on Machine Learning. San Francisco: Morgam Kaufman, 1995.

[Koike *et al.*, 2005] Koike A, Niwa Y, Takagi T: Automatic extraction of gene/protein biological functions from biomedical text. Bioinformatics 2005, 21:1227-36, 2005.

[Koivunen and Miller, 2001] Koivunen, M., and Miller E. W3C Semantic Web activity. In Semantic Web Kick-Off in Finland: Vision, Technologies, Research, and Applications, Helsinki Institute for Information Technology (HIIT), pp. 27-43, Helsinki, Finland, 2001.

[Konstantopoulos, 2003] Konstantopoulos S. Using ILP to Learn Local Linguistic Structures. PhD Dissertation, Faculty of Mathematics and Natural Sciences, Rijksuniversiteit Groningen, 2003.

[Kordjamshidi *et al.*, 2011] Kordjamshidi P., Frasconi P., Otterlo M. V., Moens M.F., and Raedt L. De. Relational learning for spatial relation extraction from natural language. In Proceedings of the 21st international conference on Inductive Logic Programming (ILP'11), Stephen H. Muggleton, Alireza Tamaddoni-Nezhad, and Francesca A. Lisi (Eds.). Springer-Verlag, Berlin, Heidelberg.204-220, 2011.

[Kowalski and Kuehner, 1971] Kowalski R. A. and Kuehner D. Linear resolution with selection function. Artificial Intelligence, 2(3/4):227–260, 1971.

[Kowalski, 1974] Kowalski R. A. Predicate logic as programming language. In Proceedings of IFIP Congress 74, pages 569–574. North Holland Publishing Co., Amsterdam, 1974.

[Kramer *et al.*, 2001] Kramer S., Lavrac N.; and Flach P. Propositionalization approaches to relational data mining. In Relational Data Mining, Saszo Dzeroski (Ed.). Springer-Verlag New York, Inc., New York, NY, USA 262-286, 2001.

[Krause *et al.*, 2014] Krause S., Li, H., Xu F., Uszkoreit, H., Hummel, R., Spielhagen, L. Language Resources and Annotation Tools for Cross-Sentence Relation Extraction. LREC 2014: 4320-4325, 2014.

[Kübler, 2009] Kübler, S., McDonald, R., and Nivre, J. Dependency Parsing. Editors: Graeme Hirst, Synthesis Lectures on Human Language Technologies, University of Toronto, 2009.

[Kushmerick, 1997] Kushmerik, N.Wrapper induction for information extraction. Ph.D. thesis, University of Washington, 1997.

[Lavrac and Dzeroski, 1994] Lavrac, N. and Dzeroski S. Inductive Logic Programming: Techniques and Applications. Ellis Horwood, New York, 1994.

[Lavrac *et al.*, 2011] Lavra N., Vavpetic A., Soldatova L., Trajkovski I., and Novak P. K. Using ontologies in semantic data mining with SEGS and g-SEGS. In Proceedings of the 14th international conference on Discovery science (DS'11), Tapio Elomaa, Jaakko Hollmén, and Heikki Mannila (Eds.). Springer-Verlag, Berlin, Heidelberg, 165-178, 2011.

[Lehman, 2009] Lehmann, J. DL-Learner: Learning Concepts in Description Logics. Journal of Machine Learning Research 10: 2639-2642, 2009.

[Levin, 1993] Levin, B. English Verb Classes and Alternations: A Preliminary Investigation. University of Chicago Press, Chicago, 1993.

[Lewis, 1992] Lewis, D. D. Representation and learning in information retrieval. PhD thesis, 1992.

[Lima *et al.*, 2014] Lima, R., Batista, J., Ferreira, R., Freitas, F., Lins, R., Simske, S., Riss, M. Transforming Graph-based Sentence Representation to Alleviate Overfitting in Relation Extraction. To appear in Proceedings of the 14th ACM Symposium on Document Engineering (DocEng 2014), September 16-19, Denver, Colorado, USA, 2014.

[Lin *et al.*, 2003] Lin D., Zhao S., Qin L., Zhou M.: Identifying Synonyms among Distributionally Similar Words. IJCAI 2003: 1492-1493, 2003.

[Lin, 1998] Lin, D. Dependency-based Evaluation of MINIPAR. Proc. Workshop on the Evaluation of Parsing Systems, 1998.

[Lisi and Esposito, 2009] Lisi, and Esposito, F. On Ontologies as Prior Conceptual Knowledge in Inductive Logic Programming. In Knowledge Discovery Enhanced with Semantic and Social Information, Springer Berlin Heidelberg, 2009, pp. 3.17, 2009.

[Liu *et al.*, 2009] Liu X-Y., Wu J., and Zhou Z-H. Exploratory undersampling for class-imbalance learning. Trans. Sys. Man Cyber. Part B 39, 2 (April 2009), 539-550, 2009.

[Liu *et al.*, 2011] Liu, K., Hogan, W., Crowley, R. Natural Language Processing Methods and Systems for Biomedical Ontology Learning. In Journal of Biomedical Informatics, Elsevier, vol. 44, pp. 163-179, 2011.

[Lowe *et al.*, 1997] Lowe, John B., Collin F. Baker, and Charles J. Fillmore. A Frame-Semantic Approach to Semantic Annotation." Tagging Text with Lexical Semantics: Why, What, and How? Proceedings of the Workshop. Special Interest Group on the Lexicon, 1997. 18-24, 1997.

[Maeche *et al.*, 2002] Maedche, A., Neumann, G., Staab, S.: Bootstrapping an Ontology-Based Information Extraction System. In: Szczepaniak, P.S., Segovia, J., Kacprzyk, J., Zadeh, L.A. (eds.) Intelligent Exploration of the Web Series. Studies in Fuzziness and Soft Computing. Springer, Heidelberg, 2002.

[Maedche, 2002] A. D. Maedche. Ontology Learning for the Semantic Web. Kluwer Academic Publishers,Norwell, MA, USA, 2002.

[Maedche, A., Staab, 2004] Maedche, A., Staab, S.: Ontology Learning. In: Handbook on Ontologies, 2004.

[Malerba *et al.*, 2003] Malerba, D. Learning Recursive Theories in the Normal ILP Setting. Fundamental. Informatics. 57 (1), 39-77, 2003.

[Manning and Schutze, 1999] Manning, C., and Schutze, H. (1999). F oundations of statistic al natural language processing. MIT Press, 1999.

[Manynard *et al.*, 2008] Maynard, D., Li, Y., and Peters, W. NLP techniques for term extraction and ontology population. In Bridging the Gap between Text and Knowledge - Selected Contributions to Ontology Learning and Population from Text, IOS Press, 2008.

[Marcus *et al.*, 1993] Marcus M., Santorini B., and Marcinkiewicz, M.A.. Building a large annotated corpus of English: the Penn Treebank. Computational Linguistics, 19(2):313–330, 1993.

[Marneffe and Manning, 2008] M-C de Marneffe and C. D. Manning, Stanford Dedendencies Manual, 2008.

[Marsh and Perzanowski, 1998] Marsh, E. and Perzanowski, D.. MUC-7 Evaluation of IE Technology: Overview of Results. 29 April 1998 - http://wwwnlpir.nist.gov/related_projects/muc/proceedings/muc_7_proceedings/marsh_slides.pdf

[Maslennikov and Chua, 2007] Maslennikov, M. and T. Chua A Multi-Resolution Framework for Information Extraction from Free Text. In Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics, 2007.

[Maynard *et al.*, 2006] Maynard D., Peters W., and Li Y., Metrics for evaluation of ontology-based information extraction. In: Proceedings of the WWW 2006 Workshop on Evaluation of Ontologies for the Web, (ACM, New York, 2006.

[McDowell and Cafarella, 2006] McDowell L. and Cafarella M. J., Ontology-driven information extraction with OntoSyphon. In: Proceedings of the 5th International Semantic Web Conference, Springer, Berlin, 2006.

[Melli, 2007] Melli G.. (2007). Inductive Approaches to the Detection and Classification of Semantic Relation Mentions. Depth Report, Simon Fraser School of Computing Science. August 27, 2007.

[Michie *et al.*, 1994] Michie, D., Muggleton, S. H., Page, D., Srinivasan, A. To the international computing community: a New East-West challenge (Tech. Rep.) Orford, UK: Oxford University Computing Laboratory, 1994.

[Mihalcea and Faruque, 2004] Mihalcea R. and Faruque E. SenseLearner: Minimally supervised word sense disambiguation for all words in open text. In Proceedings of ACL/SIGLEX Senseval-3, Barcelona, Spain, July, 2004.

[Miller and Charles, 1991] Miller, G. and Charles, W. Contextual correlates of semantic similarity. Language and Cognitive Processes, 6:1-28, 1991.

[Miller *et al.*, 1997] D.M., Miller, Bikel, D.M., Miller, S., Schwartz, R., and Weischedel, R. (1997). Nymble: a high-performance learning name-finder. 5th Conference on Applied Natural Language Processing (pp. 194–201), 1997.

[Miller, 2005] Miller G. A. WordNet: A Lexical Database for English. Communications of the ACM Vol. 38, No. 11: 39-41, 1995.

[Mintz *et al.*, 2009] Mintz M., Bills S., Snow R., and Jurafsky D.. Distant supervision for relation extraction without labeled data. In Proceedings of the Joint Conference of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th Inter- national Joint Conference on Natural Language Processing of the AFNLP, pages 1003–1011, 2009.

[Mitchel, 1982] Mitchel, T. Generalization as Search. Artificial Intelligence 18, (pp. 203-226), 1982.

[Miwa *et al.*, 2010] Miwa M., Saetre, R., Miyao, Y. and Tsujii, J. (2010). Entity-focused sentence simplification for relation extraction. COLING '10. Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 788-796, 2010.

[Miyao *et al.*, 2009] Miyao Y, Sagae K, Saetre R, Matsuzaki T, Tsujii J. (2009). Evaluating contributions of natural language parsers to protein-protein interaction extraction. Bioinformatics 2009, 25:394- 400, 2009.

[Mooney, 1999] Mooney R. J. Learning for Semantic Interpretation: Scaling Up WithoutDumbing Down. In Proeedings of the Learning Language in Logic Workshop, LLL'99,Bled, Slovenie. Morin E., 1999.

[Motik *et al.*, 2004] Motik B., Sattler U., and Studer R. Query answering for OWL-DL with rules. In International Semantic Web Conference, 2004, pages 549–563, 2004.

[Muggleton and Fen, 1990] Muggleton, S. and Fen, C. (1990). Efficient induction of logic programs. 1st Conference on Algorithmic Learning Theory (pp. 368-381), Tokyo, 1990.

[Muggleton and Raedt, 1994] Muggleton S. , Raedt L. De . Inductive logic programming: Theory and methods. Journal of Logic Programming 19/20: 629-679, 1994.

[Muggleton and Santos, 2009] Muggleton S., Santos J., and Tamaddoni-Nezhad A. ProGolem: a system based on relative minimal generalisation. In Proceedings of the 19th International Conference on ILP, Springer, pages 131{148, Leuven, Belgium, 2009.

[Muggleton *et al.*, 2009] Muggleton, S., Santos, J. and Tamaddoni-Nezhad, A. (2009). ProGolem: a system based on relative minimal generalisation. 19th International Conference on ILP, Springer (pp. 131-148), Leuven, Belgium, 2009.

[Muggleton, 1990] Muggleton S. and Feng C. Efficient induction of logic programs. In Proceedings of the 1st Conference on Algorithmic Learning Theory, pages 368{381, Tokyo, 1990.

[Muggleton, 1991] Muggleton, S. (1991). Inductive Logic Programming. New Generation Computing 8 (4): 29, 1991.

[Muggleton, 1995] Muggleton, S. (1995). Inverse entailment and Progol. New Generation Computing, 13 (pp. 245-286), 1995.

[Nassif, 2012] Nassif, H., Differential Relational Learning. PhD dissertation, University of Winsconsin, 2012.

[Navigli and Velardi, 2006] Navigli R. and Velardi P., Enriching a formal ontology with a thesaurus: An application in the cultural heritage domain, Proc. of the 2nd Workshop on Ontology Learning and Population: Bridging the Gap between Text and Knowledge, Sydney, Australia, pp.1-9, 2006.

[Nazerenko *et al.*, 2006] Nazarenko A., C. Nédellec, Alphonse E., Aubin S., Hamon T., and Manine A.-P.. Semantic annotation in the Alvis project. In W. Buntine and H. Tirri, editors, Proceedings of the International Workshop on Intelligent Information Access, pages 40–54, Helsinki, Finlande, 2006.

[Nédellec and Nazarenko, 2005] Nédellec, C. and Nazarenko, A. Ontologies and Information Extraction. 2005. http://arxiv.org/abs/cs.AI/0609137

[Nédellec et al., 2008] Nédellec,C., Nazarenco, A. Bossy, R.: Information Extraction. In: Staab, S., Studer, R. (editors). Ontology Handbook. Springer, Heidelberg , 2008 .

[Nedellec, 2005] Nédellec C: Learning language in logic – genic interaction extraction challenge. Proceedings of the 4th Learning Language in Logic Workshop 2005:31-37, 2005.

[Nederstigt *et al.*, 2014] Nederstigt, Aanen S. S., Vandic D., Frasincar F., FLOPPIES: A Framework for Large-Scale Ontology Population of Product Information from Tabular Data in E-commerce Stores, Decision Support Systems, Volume 59, March 2014, Pages 296-311, ISSN 0167-9236, 2014.

[Ngyyen and Moschiti, 2011] Nguyen T., and Moschitti A. End-to-end relation extraction using distant supervision from external semantic repositories. In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics, pages 277–282, 2011.

[Nile and Pease, 2001] Nile I. and Pease A.. Towards a Standard Upper Ontology. In Chris Welty and Barry Sm ith, editors, Proceedings of the 2nd International Conference on Formal Ontology in Info rmation Systems (FOIS-2001), 2001.

[Niles and Pease, 2003] Niles, I. and Pease, A. Linking Lexicons and Ontologies: Mapping WordNet to the Suggested Upper Merged Ontology. In Proceedings of the 2003 International Conference on Information and Knowledge Engineering (IKE 03), Las Vegas, Nevada, June 23-26, 2003.

[Nitesh *et al.*, 2002] Nitesh, V., Chawla, Kevin W., Bowyer, Lawrence, O. Hall, and Philip Kegelmeyer, W. SMOTE: synthetic minority over-sampling technique. Journal of Artificial Intelligence Research, 16, 1, June 2002, pp. 321-357), 2002.

[Novak *et al.*, 2009] Novak P., Vavpetic A., Trajkovski I., and Lavrac N. Towards semantic data mining with g-segs. In Proceedings of the 11th International Multiconference Information Society IS 2009, 2009.

[Ogden and Richards, 1923] Ogden, C.K. and Richards, I.A. The meaning of meaning: a study of the influence of language upon thought and of the science of symbolism. 8th edition. Reprint New Youk, 1923.

[Okanohara *et al.*, 2006] Okanohara D., Miyao Y., Tsuruoka Y., and Tsujii J.. Improving the scalability of semi-Markov conditional random fields for named entity recognition. In Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics, pages 465–472, 2006.

[Ono *et al.*, 2001] Ono T, Hishigaki H, Tanigami A, Takagi T: Automated extraction of information on protein-protein interactions from the biological literature. Bioinformatics 2001, 17:155-61, 2001.

[Page and Srinivasan, 2003] Page D. and Srinivasan A.. ILP: A short look back and a longer look forward. Journal of Machine Learning Research, 4:415–430, 2003.

[Palmer *et al.*, 2005] Palmer M., Gildea D., and Kingsbury P. The Proposition Bank: An Annotated Corpus of Semantic Roles. Comput. Linguist. 31, 1 (March 2005), 71-106,2005.

[Patel *et al.*, 2010] Patel, A., Ramakrishnan, G. and Bhattacharya, P. Incorporating Linguistic Expertise Using ILP for Named Entity Recognition in Data Hungry Indian Languages, LNCS, vol. 5989 (pp. 178-185), Springer Berlin Heidelberg, 2010.

[Petasis *et al.*, 2011] G. Petasis, V. Karkaletsis, G. Paliouras, A. Krithara, and E. Zavitsanos, Ontology Population and Enrichment: State of the Art, in G. Paliouras *et al.* (Eds.): Multimedia Information Extraction, LNAI 6050, pp. 134–166, 2011.

[Plotkin, 1971a] Plotkin, G. A note on inductive generalization. Machine Intelligence 5 (pp. 153-163), 1971.

[Plotkin, 1971b] Plotkin, G.D. Automatic Methods of Inductive Inference. PhD thesis, Edinburgh University, 1971.

[Popel and Zabokrtsky, 2010] Popel M., Žabokrtský Z.: TectoMT: Modular NLP Framework. In Proceedings of IceTAL, 7th International Conference on Natural Language Processing, Reykjavík, Iceland, August 17, 2010, pp. 293–304, 2010.

[Popov et al, 2004] B. Popov, A. Kiryakov, D. Ognyanoff, D. Manov and A. Kirilov, KIM - a semantic platform for information extraction and retrieval, Journal of Natural Language Engineering 10(3-4) (2004) 375-392, 2004.

[Pyysalo *et al.*, 2007] Pyysalo S, Ginter F, Heimonen J, Björne J, Boberg J, Järvinen J, Salakoski T:Bioinfer: a corpus for information extraction in the biomedical domain. BMC Bioinformatics 2007, 8(50), 2007.

[Pyysalo, 2008] Pyysalo S. A dependency parsing approach to biomedical text mining, Ph.D. thesis, Department of Information Technology, University of Turku, 2008.

[Qian and Zhou, 2012] Qian, L., and Zhou, G. Tree kernel-based protein-protein interaction extraction from biomedical literature. Journal of Biomedical Informatics, 45(3), 535-543, 2012.

[Quinlan and Cameron-Jones, 1993] Quinlan, J. and Cameron-Jones, R. FOIL: A midterm report. In Proceedings of the European Conference on Machine Learning (ECML). Vienna, Austria, 3–20, 1993.

[Quinlan, 1990] Quinlan, J. R. Learning Logical Definitions From Relations. Machine Learning, 5:239-266, 1990.

[Raedt, 2008] Raedt, L. D. Logical and Relational Learning. Springer, 2008.

[Rabiner, 1989 Rabiner] L. R.. "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition," Proceedings of the IEEE, vol 77, no 2, 257—287, 1989.

[Raghavan *et al.*, 2012] Raghavan S, Mooney R. J., and Ku H.. Learning to "read between the lines" using Bayesian logic programs. In Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers - Volume 1 (ACL '12), Vol. 1. Association for Computational Linguistics, Stroudsburg, PA, USA, 349-358, 2012.

[Ramakrishnan *et al.*, 2008] Ramakrishnan, G., Joshi, S., Balakrishnan, S. and Srinivasan, A. Using ILP to Construct Features for Information Extraction from Semi-structured Text. In Proceedings of the 17th International Conference on Inductive Logic Programming, LNAI 4894 (pp. 211-224), Berlin, 2008. Springer, 2008.

[Rilloff, 1993] Riloff, E. Automatically Constructing a Dictionary for Information Extraction Tasks. In Proceedings of the Eleventh National Conference on Artificial Intelligence. pp.811-816, 1993.

[Rosenfeld and Feldman, 2007] Rosenfeld, B. and Feldman, R. (2007). Clustering for unsupervised relation identification. In Proceedings of the 16th ACM conference on Conference on Information and Knowledge Management, (pp. 411–418), 2007.

[Roth and Yih, 2007] Roth, D. and Yih,W. Global Inference for entity and relation identification via a linear programming formulation. In Introd. to Statistical Relational Learning, L. Getoor and B. Taskar, Eds. MIT Press, 2007.

[Roth and Yih, 2009] Roth, D. and Yih, W. Linear Programming Formulation for Global Inference in Natural Language Tasks. CoNLL (2009) pp. 1-8, 2009.

[Roxana *et al.*, 2006] Roxana G., Badulescu A., and Moldovan D.. Automatic discovery of part-whole relations. Computational Linguistics, 32(1):83–135 , 2006. (Cited on pages 30, 133 , 134 , 175, and 177, 2006.

[Ruiz-Martinez et al, 2011] Ruiz-Martinez J-M, Minarro-Gimenez J-A, Castellanos-Nieves D., Garcia-Sanchez F., Valencia-Garcia R. Ontology Population: an Application for the E-Tourism Domain, International Journal of Innovative?Computing, Information and Control, Volume 7, Number 11, November 2011.

[Ruiz-Martínez *et al.*, 2010] Ruiz-Martínez J.M., Valencia-García R., Martínez-Béjar, R. A. Hoffmann, Populating biomedical ontologies from natural language texts, in: Proceedings of the International Conference Knowledge Engineering and Ontology Development (KEOD), pp. 27–36, 2010.

[Ruiz-Martínez *et al.*, 2012] Ruiz-Martínez J. M., Valencia-García R., Martínez-Béjar R,, Hoffmann A., BioOntoVerb: A top level ontology based framework to populate biomedical ontologies from texts, Knowledge-Based Systems, Volume 36, December 2012, Pages 68-80, ISSN 0950-7051, 2012.

[Saggion *et al.*, 2007] Saggion, H., Funk, A., Maynard, D. and Bontcheva, K. 2007. Ontology-based Information Extraction for Business Intelligence, ISWC'07/ASWC'07 , pp. 843-856, 2007.

[Saint-Dizier and Viegas, 1994]. Saint-Dizier P. and Viegas E. (Eds.). Computational Lexical Semantics. Cambridge University Press, New York, NY, USA ,1994.

[Sanchez, 2007] Sanchez D. Domain Ontology Learning from the Web. Phd. Thesis. Technical University or Catalonia (UPC), Spain, 2007.

[Santos *et al.*, 2009] Santos, J., Tamaddoni-Nezhad, A., Muggleton, S. An ILP System for Learning Head Output Connected Predicates. In Progress in Artificial Intelligence, LNCS, Heidelberg, Berlin, Springer, pp. 150-159, 2009.

[Santos, 2010] Santos, J. Efficient Learning and Evaluation of Complex Concepts in Inductive Logic Programming, Ph.D. Thesis, Imperial College, 2010.

[Sarawagi, 2008] Sarawagi S. 2008. Information Extraction. Found. Trends databases 1, 3 , March, 2008.

[Schmidt, 2005] Thomas Schmidt. Heterogeneity in Focus: Creating and Using Linguistic Databases, ISIS Interdisciplinary Studies on Information Structure), Working Papers of the SFB 632, Potsdam University, 2005.

[Schuhma *et al.*, 2010] Rebholz-Schuhmann D, Jimeno-Yepes A, Arregui M, Kirsch H: Measuring prediction capacity of individual verbs for the identification of protein interactions. Journal of biomedical informatics 2010, 43:200-7, 2010.

[Seneviratne and Ranasinghe, 2011] Seneviratne, M. D. S. and Ranasinghe, D. N., Inductive Logic Programming in an Agent System for Ontological Relation Extraction, International Journal of Machine Learning and Computing vol. 1, no. 4, pp. 344-352 , 2011.

[Settles, 2004] Settles, B. Biomedical named entity recognition using conditional random fields and rich feature sets. International Joint Workshop on Natural Language Processing in Biomedicine and Its Applications (pp. 104–107), 2004.

[Settles, 2012] Settles, B., Active Learning. Synthesis Lectures on Artificial Intelligence and Machine Learning, June 2012, Vol. 6, No. 1 , Pages 1-114, 2012.

[Seymore *et al.*, 1999] Seymore, K., McCallum A., and Rosenfeld, R. Learning Hidden Markov Model Structure for Information Extraction. AAAI 99 Workshop on Machine Learning for Information Extraction, 1990.

[Shapiro, 1981] Shapiro, E. Inductive inference of theories from facts. Technical Report 624, Dept. of Computer Science, Yale University, 1981.

[Shapiro, 1983] Shapiro E.Y. Algorithmic Program Debugging. The MIT Press, 1983.

[Shen *et al.*, 2003] Shen D., Zhang J., Zhou G., Su J., and Tan C.-L.. Effective adaptation of a hidden markov model-based named entity recognizer for biomedical domain. In Proceedings of the ACL 2003 Workshop on Natural Language Processing in Biomedicine - Volume 13, pages49–56, 2003.

[Shinyama and Sekine, 2006] Shinyama Y, and Sekine S. Preemptive information extraction using unrestricted relation discovery. In Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics, pages 304–311, 2006.

[Siddharthan, 2001] Siddharthan, A. Text simplification using typed de-pendencies: a comparison of the robustness of different gen-eration strategies. In Proc. of the 13th European Workshop on Natural Language Generation (ENLG '11). Association for Comp. Linguistics, Stroudsburg, PA, USA, 2-11, 2001.

[Siddharthan, 2011] Siddharthan, A. Text simplification using typed dependencies: a comparison of the robustness of different generation strategies. ENLG '11. ACL, Stroudsburg, PA, USA , pp. 2-11, 2011.

[Sinha and Mihalcea, 2007] Sinha R. and Mihalcea R.Unsupervised Graph-based Word Sense Disambiguation Using Measures of Word Semantic Similarity, in Proceedings of the IEEE International Conference on Semantic Computing (ICSC 2007), Irvine, CA, September 2007.

[Sirin *et al.*, 2007] Sirin E., Parsia B., Grau B. C. Kalyanpur A. and Katz Y.. Pellet: A practical OWL-DL reasoner, Journal of Web Semantics, 5(2), 2007.

[Smole *et al.*, 2011] Smole, D., Ceh, M. and Podobnikar, T. Evaluation of inductive logic programming for information extraction from natural language texts to support spatial data recommendation services.. International Journal of Geographical Information Science, 25, 1809-1827, 2011.

[Soderland *et al.*,1995] Soderland, S., Fisher, D., Aseltine, J., and Lehnert, W. CRYSTAL: Inducing a conceptual dictionary. In Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence (IJCAI'95). pp.1314-1319., 1995.

[Soderland, 1999] Soderland S.. Learning information extraction rules for semi- structured and free text. Machine Learning, 34(1-3):233–272, February, 1999.

[Sowa, 2000] Sowa, J. F. Ontology, metadata, and semiotics. In Ganter, B. and Mineau, G. W., editors, Conceptual Strutuctures: Logical, Linguistic, and Computacional Issues, Proceedings of the 8th International Conference on Conceptual Structures (ICS), vol 1867 of Lecture Notes in Computer Science, pp. 55-81.Springer, 2000.

[Spasic *et al.*, 2005] Spasic,I., Ananiadou,S., McNaught,J. *et al.* Text mining and ontologies in biomedicine: making sense of raw text. Brief. Bioinformatics, 6, 239–251, 2005.

[Specia et al, 2006] Specia L., Srinivasan A., Ramakrishnan G., Volpe M. das G., Word Sense Disambiguation using Inductive Logic Programming, 16th International Conference on Inductive Logic Programming (ILP 2006), Santiago, Spain, August 24-27, 2006.

[Specia *et al.*, 2009] Specia L., Srinivasan A., Joshi S., Ramakrishnan G., and Nunes. M. G.V. An investigation into feature construction to assist word sense disambiguation. Mach. Learn., 76(1):109–136, Kluwer Academic Publishers, Hingham, MA, USA, 2009.

[Sperberg and Bernard, 1994] C. M. Sperberg and Lou Bernard. Guidelines for Electronic Text Encoding and Interchange (TEI P3). Text Encoding Initiative, Chicago, Oxford, 1994.

[Srinivasan, 2007] Srinivasan A. The Aleph Manual, Version 4 and above. http://www.cs.ox.ac.uk/activities/machlearn/Aleph/aleph.html

[Studer *et al.*, 1998] Studer, R., Benjamins, V.R., Fensel, D.: Knowledge engineering: principles and methods. IEEE Trans. Knowl. Data Eng. 25 (1–2), 161–197, 1998.

[Choi *et al.*, 2013] Choi, S., Lee. S., Jung, H., Song, S. An intensive case study on kernel-based relation extraction. Proceeding s of Multimedia Tools and Applictions, Springer, US, p. 1-27, 2013.

[Tanev and Magnini, 2006] H. Tanev and B. Magnini, Weakly supervised approaches for ontology population, Proc. of EACL- ?2006, Trento, pp.3-7, 2006.

[Tang *et al.*, 2007] Tang J., Hong M., Zhang D., Liang B., and Li J.. Information Extraction: Methodologies and Applications. In the book of Emerging Technologies of Text Mining: Techniques and Applications, Hercules A. Prado and Edilson Ferneda (Ed.), Idea Group Inc., Hershey, USA, 2007. pp. 1-33, 2007.

[Tateisi and Tsujii]Tateisi, Y., and Tsujii, J. Part-of-speech annotation of biology research abstracts. LREC, 2004.

[Tausend, 1994] B. Tausend. Biases and their effects in inductive logic programming. In F. Bergadano and L. De Raedt, editors, Proceedings of the 7th European Conference on Machine Learning, volume 784 of LNAI, pages 431-434. Springer-Verlag, 1994.

[Tikk *et al.*, 2010] Tikk, D., Thomas, P. E., Palaga, P., Hakenberg, J., Leser, U. 2010. A Comprehensive Benchmark of Kernel Methods to Extract Protein-Protein Interactions from Literature., PLoS Computational Biology, 6 (7), 2010.

[Tsuruoka *et al.*, 2005] Y. Tsuruoka, Y. Tateishi, J.D. Kim, T. Ohta, J. McNaught, S. Ananiadou, J. Tsujii,Developing a robust part-of-speech tagger for biomedical text, Advances in Informatics (2005) 382–392, 2005.

[Turcotte *et al.*, 1998] Turcotte M., S. Muggleton H., and Sternb M. J. E. Application of inductive logic programming to discover rules governing the three-dimensional topology of protein structure. In D. Page, editor, Proceedings of the 8th International Conference on Inductive Logic Programming , volume 1446, pages 5364. Springer-Verlag, 1998.

[Turmo *et al.*, 2006] Turmo J., Ageno A., and Català N. Adaptive information extraction. ACM Comput. Surv. 38, 2, Article 4, July, 2006.

[Vavpetic and Lavrac, 2013] Vavpetic, A. and Lavrac, N. 'Semantic Subgroup Discovery Systems and Workflows in the SDM-Toolkit, Computional Journal 56 (3) , 304-320, 2013.

[Vickrey, D. and Koller] Vickrey, D., and Koller, D. Sentence simplification for semantic role labeling. In Proceedings of ACL-08: HLT, 344–352. Association for Computational Linguistics, 2008.

[Völker *et al.*, 2008] Völker J., Haase P., and Hitzler P. 2008. Learning Expressive Ontologies. In Proceedings of the 2008 conference on Ontology Learning and Population: Bridging the Gap between Text and Knowledge, Paul Buitelaar and Philipp Cimiano (Eds.). IOS Press, Amsterdam, The Netherlands, The Netherlands, 45-69, 2008.

[Wang and Yu, 2011] Wang, Q., Yu, X.: Reasoning over OWL/SWRL ontologies under CWA and UNA for industrial applications. Advances in Artificial Intelligence 7106 (2011) 789 -798, 2011.

[Wang *et al.*, 2005] Wang, T., Bontcheva K., Li Y., Cunninghan, H. (2005). Ontologoy-based Information Extraction. SEKT project deliverable D2.1.2, 2005.

[Wimalasuriya and Dou, 2009] Wimalasuriya D. C., D. Dou, Ontology-Based Information Extraction: An Introduction and a Survey of Current Approaches, Journal of Information Science, 2009, pp. 1–20, 2009.

[Wimalasuriya and Dou, 2010] Wimalasuriya, D. C., Dou, D. Components for Information Extraction: Ontology-Based Information Extractors and Generic Platforms. CIKM'10, October 26–30, 2010, Toronto, Ontario, Canada, 2010.

[Witten and Frank, 2005] Witten, I. H., and Frank, E. Data Mining: Practical machine learning tools and techniques. Morgan Kaufmann, 2005.

[Wrobel, 1997] Stefan Wrobel. An algorithm for multi-relational discovery of subgroups.In Proceedings of the First European Symposium on Principles of Data Mining and Know ledge Discovery, pages 7887, London, UK, 1997. Springer-Verlag, 1997.

[Wu and Weld, 2008] Wu, F., and Weld, D. (2008) Automatically refining the Wikipedia infobox ontology. In Proceedings of the 17th International Conference on World Wide Web, New York, USA, 2008.

[Wu *et al.*, 2008] F. Wu, R. Hoffmann, and D. S. Weld. Information extraction from Wikipedia: moving down the long tail. In KDD, pages 731–739, NY, USA, 2008. ACM, 2008.

[Yeh *et al.*, 2005] A. Yeh, A. Morgan, M. Colosimo, and L. Hirschman. BioCreAtIvE task 1A: Gene mention finding evaluation. BMC Bioinformatics, 6(Suppl 1):S2, 2005.

[Yildiz and Miksch, 2007] Yildiz B. and Miksch S. OntoX - a method for ontology-driven information extraction. In: Proceedings of the 2007 International Conference on Computational Science and Its Applications, Springer, Berlin, 2007.

[Yildiz, 2007] B. Yildiz. Ontology-Driven Information Extraction, Phd. Thesis, Institute of Software Technology and Interactive Systems. Faculty of Informatics. Vienna, March, 2007.

[Xavier *et al.*, 2013] Xavier C.C., Lima V. L. S., Souza M. "Open Information Extraction Based on Lexical-Syntactic Patterns", BRACIS, 2013, 2013 Brazilian Conference on Intelligent Systems (BRACIS), 2013 Brazilian Conference on Intelligent Systems (BRACIS) 2013, pp. 189-19, 2013.

[Zajic *et al.*, 2007] Zajic D., Dorr J., Lin J., and Schwartz, R. Multi-candidate reduction: Sentence compression as a tool for document summarization tasks. In Information Processing and Management Special Issue on Summarization, Vol. 43, No. 6, 1549-1570, 2007.

[Zelenko *et al.*, 2003] Zelenko D., Aone C. and Richardella A.. (2003). Kernel methods for relation extraction. Journal of Machine Learning Research. 3, 2003, pp. 1083-1106, 2003.

[Zelle, 1995] Zelle J. M. Using Inductive Logic Programming to Automate the Construction of Natural Language Parsers. PhD thesis, Department of Computer Sciences, University of Texas, Austin, Texas, USA, 1995.

[Zhang *et al.*, 2006] Zhang M, Zhang J, Su J, and Zhou G. A composite kernel to extract relations between entities with both flat and structured features," in 21st International Conference on Computational Linguistics and 44th Annual Meeting of the ACL, pp. 825–832, 2006.

[Zhang and Zhou, 2008] Zhang, M., Zhou, G.D. and Aw, A.T. Exploring syntactic structured features over parse trees for relation extraction using kernel methods, Information Processing and Management, 44, 2008 (pp. 687–701, 2008.

[Zhao and Grisman, 2005] Zhao, S. B., and Grisman, R. Extracting Relations with Integrated Information using Kernel Methods. ACL'2005, 25–30 June 2005, Ann Arbor, USA, pp. 419–426, 2005.

[Zhou and He, 2008] Zhou D and He, Y. Extracting interactions between proteins from the literature. Journal of Biomedical Informatics 2008, 41:393-407, 2008.

[Zhou *et al.*, 2008] Zhou D, He Y, Kwoh CK: From Biomedical Literature to Knowledge: Mining Protein-Protein Interactions. In Comp. Intel. in Biomedical and Bioinformatics Springer; 2008:397-421, 2008.

[Zhou *et al.*, 2005] Zhou, G. D., Su, J., Zhang, J. and Zhang, M. Exploring various knowledge in relation extraction, in: Proceedings of the Annual Meeting of the Association for Computational Linguistics - ACL'2005, 25–30 June 2005, Ann Arbor, Michigan, USA, 2005 (pp. 427–434), 2005.

[Zhou *et al.*, 2007] Zhou, G., Zhang, M., Ji, D-H., Zhu, Q. Tree Kernel-based Relation Extraction with Context-Sensitive Structured Parse Tree Information. 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, Prague, June 2007 (pp. 728–736),2007.

[Zhilla and Gelbukh, 2014] Zhilla A. and Gelbuk A. Comparison of open information extraction for engish and spanish. In Dialogue 2014, 2014.

[Zouaq *et al.*, 2010] Zouaq, A., Gagnon, M. and Ozell, B.: Semantic Analysis using Dependency-based Grammars and Upper-Level Ontologies, International Journal of Computational Linguistics and Applications, 1(1-2): 85-101, Bahri Publications, 2010.

Appendix A - List of POS tags used in the Penn Treebank Project

Tag	Description
CC	Coordinating conjunction
CD	Cardinal number
DT	Determiner
EX	Existential *there*
FW	Foreign word
IN	Preposition or subordinating conjunction
JJ	Adjective
JJR	Adjective, comparative
JJS	Adjective, superlative
LS	List item marker
MD	Modal
NN	Noun, singular or mass
NNS	Noun, plural
NNP	Proper noun, singular
NNPS	Proper noun, plural
PDT	Predeterminer
POS	Possessive ending
PRP	Personal pronoun
PRP$	Possessive pronoun
RB	Adverb
RBR	Adverb, comparative
RBS	Adverb, superlative
RP	Particle
SYM	Symbol
TO	*to*
UH	Interjection
VB	Verb, base form
VBD	Verb, past tense
VBG	Verb, gerund or present participle
VBN	Verb, past participle
VBP	Verb, non-3rd person singular present
VBZ	Verb, 3rd person singular present
WDT	Wh-determiner
WP	Wh-pronoun
WP$	Possessive wh-pronoun
WRB	Wh-adverb

Appendix B - Hierarchy of the Stanford Dependency Labels [De Marneffe & Manning, 2008]

- *root* - root
- *dep* - dependent
  - *aux* - auxiliary
    - *auxpass* - passive auxiliary
    - *cop* - copula
  - *arg* - argument
    - *agent* - agent
    - *comp* - complement
      - *acomp* - adjectival complement
      - *ccomp* - clausal complement with internal subject
      - *xcomp* - clausal complement with external subject
      - *obj* - object
        - *dobj* - direct object
        - *iobj* - indirect object
        - *pobj* - object of preposition
    - *subj* - subject
      - *nsubj* - nominal subject
        - *nsubjpass* - passive nominal subject
      - *csubj* - clausal subject
        - *csubjpass* - passive clausal subject
- *cc* - coordination
- *conj* - conjunct
- *expl* - expletive (expletive "there")
- *mod* - modifier
  - *amod* - adjectival modifier
  - *appos* - appositional modifier
  - *advcl* - adverbial clause modifier
  - *det* - determiner
  - *predet* - predeterminer
  - *preconj* - preconjunct
  - *vmod* - reduced, non-finite verbal modifier
  - *mwe* - multi-word expression modifier

mark - marker (word introducing an *advcl* or *ccomp*

advmod - adverbial modifier

neg - negation modifier

rcmod - relative clause modifier

quantmod - quantifier modifier

nn - noun compound modifier

npadvmod - noun phrase adverbial modifier

tmod - temporal modifier

num - numeric modifier

number - element of compound number

prep - prepositional modifier

poss - possession modifier

possessive - possessive modifier (*'s*)

prt - phrasal verb particle

parataxis - parataxis

punct - punctuation

ref - referent

sdep - semantic dependent

xsubj - controlling subject

Appendix C - PropBank semantic role labels [Bonial et al. (2010)]

Label	Description
ARG0	Agent
ARG1	Patient, theme
ARG2	Instrument, benefactive, attribute
ARG3	Staring point
ARG4	Ending point
ARGA	External causer
ARGM-ADJ	Adjectival
ARGM-ADV	Adverbial
ARGM-CAU	Cause
ARGM-COM	Commutative
ARGM-DIR	Direction
ARGM-DIS	Discourse
ARGM-GOL	Goal
ARGM-EXT	Extent
ARGM-LOC	Location
ARGM-MNR	Manner
ARGM-MOD	Modal
ARGM-NEG	Negation
ARGM-PRD	Secondary predication
ARGM-PRP	Purpose (previously, ARGM-PNC)
ARGM-REC	Reciprocal
ARGM-TMP	Temporal

Appendix D - VerbNet Semantic Role Labels [Bonial et al. (2011)]

Label	Description
actor1,2	Pseudo-agents, used for some communication classes
agent	animate subject, volitional or internally controlled
asset	Currency, used for Build/Get/Obtain Classes
attribute	Changed quality of patient/theme
beneficiary	Entity benefiting from action
cause	Entity causing an action, used for psychological/body verbs
destination	End point/target of motion
experiencer	Participant that is aware of experiencing something
extent	Range or degree of change
instrument	Objects/forces that come into contact and cause change in another object
location	Underspecified destination/source/place
material	Starting point of transformation
patient1,2	Affected participants, used for some combining/attaching verbs
predicate	Predicative complement
product	End result of transformation
recipient	Target of transfer
source	Spatial location, starting point
stimulus	Events/objects that elicit a response from an experiencer
theme	Participants in/undergoing a change of location
theme1,2	Indistinct themes, used for differ/exchange classes
patient	Affected participants undergoing a process
time	Class-specific, express temporal relations
topic	Topic of conversation, message, used for communication verbs
proposition	Complement clause indicating desired/requested action, used for order class

Appendix E – Excerpt of an XML File Generated by the Text Preprocessing Stage in OntoILPER

Excerpt of the XML file for the input sentence: “I think he has been in Washington too long”.

```
<sentence has_ne="true" has_ref="false" is_mention="false" s_id="s_27" text="I think he 's been in Washington too
long ." voice="passive"
  <tokens>
    <token ck_ot="NP" ck_tag_ot="B-NP" head="true" length="1" normalized="person" orth="UpperCase"
      pos="PRP" s_id="s_27" similar="persons,man,individuals,guy,people"  stem="I" string="I" t_id="t_339"
      type="Word"  />
    <token bl_domain=”null” bl_hypernyms="[evaluate#v#2,think#v#3]" bl_sense_id="think#v#1"  bl_sumo=""
      bl_supersense_id="verb.cognition" bl_synset_id="think%2:31:01::" head="true" length="5" normalized="think"
      orth="lowercase" pos="VBP" s_id="s_27" similar="believe,know,expect,imagine,say" stem="think" tring="think"
      t_id="t_340" type="Word" wsd_domain=”” sd_hypernyms="[evaluate#v#2,think#v#3]"
      wsd_sense_id="think#v#1" wsd_sumo=”” wsd_supersense_id="verb.cognition" sd_synset_id="think%2:31:01::"
    />
    <token ck_ot="NP" ck_tag_ot="B-NP"  head="true" length="2" normalized="person" orth="LowerCase" pos="PRP"
      s_id="s_27" similar="persons,man,individuals,guy,people" stem="he" string="he" t_id="t_341" type="Word" />
  </tokens>
```

```
 <chunkings>
   <chunking ck_id="ck_185" s_id="s_27" text="I" type="np">
     <tokens>
       <token ck_id="ck_185" head="true" s_id="s_27" string="I" t_id="t_339"/>
     </tokens>
   </chunking>
   <chunking ck_id="ck_186" s_id="s_27" text="think" type="vp">
     <tokens>
       <token ck_id="ck_186" head="true" s_id="s_27" string="think" t_id="t_340"/>
     </tokens>
   </chunking>
   <chunking ck_id="ck_187" s_id="s_27" text="he" type="np">
     <tokens>
       <token ck_id="ck_187" head="true" s_id="s_27" string="he" t_id="t_341"/>
     </tokens>
   </chunking>
   <chunking  ck_id="ck_188" s_id="s_27" text="'s been" type="vp">
     <tokens>
       <token ck_id="ck_188" s_id="s_27" string="'s" t_id="t_342"/>
       <token ck_id="ck_188" head="true" s_id="s_27" string="been" t_id="t_343"/>
     </tokens>
 </chunkings>
```

```
<dependences>
  <pair rel="root">
    <arg1 id="t_338" seq="0" string="ROOT"/>
    <arg2 id="t_340" seq="2" string="think"/>
  </pair>
  <pair rel="advmod">
    <arg1 id="t_347" seq="9" string="long"/>
    <arg2 id="t_346" seq="8" string="too"/>
  </pair>
  <pair rel="advmod">
    <arg1 id="t_343" seq="5" string="been"/>
    <arg2 id="t_347" seq="9" string="long"/>
  </pair>
</dependences>
```

<srl>
<roleset id="think.01" name="think" t_id="t_340" vncls="29.4">
<arg name="Thinker" role_id="A0" t_id="t_339" text="I" vncls="29.4" vntheta="Agent"/>
<arg name="Thought" role_id="A1" t_id="t_343" text="been" vncls="29.4" vntheta="Theme"/>
</roleset>
<roleset id="be.01" name="copula" t_id="t_343" vncls="">
<arg name="topic" role_id="A1" t_id="t_341" text="he" vncls="" vntheta=""/>
<arg name="comment" role_id="A2" t_id="t_344" text="in" vncls="" vntheta=""/>
<arg role_id="AM-TMP" t_id="t_347" text="long"/>
</roleset>
</srl>

<instances>
<nes>
<ne ne_id="ne_81" ne_is_rel_arg="true" ne_nam="none" ne_string="he" ne_subtype="none" ne_type="PER" s_id="s_27" t_id="t_341"/>
<ne ne_id="ne_82" ne_is_rel_arg="true" ne_nam="yes" ne_string="Washington" ne_subtype="Population- Center" ne_type="GPL" s_id="s_27" t_id="t_345"/>
</nes>
<rels>
<rel is_ex_pos="false" rel_id="r_179" rel_subtype="located" rel_type="gen-aff" arg1="t_345" arg2="t_341"/>
<rel is_ex_pos="true" rel_id="r_180" rel_subtype="located" rel_type="gen-aff" arg1="t_341" arg2="t_345"/>
</rels>
</instances>

</sentence>

Appendix F - Closed vs. Open World Assumptions

In OntoILPER, the conversion from Prolog rules into rules in SWRL entails a key aspect related to the differences between these two rule formalisms, namely *Closed vs. Open World Assumptions*.

One can view Logic Programming as a logical theory expressing knowledge about the world. In certain situations, one might assume that the program contains *complete information* about the objects (facts) and relations between these facts. It is also possible, in a logical program, to make further inferences about the world based on the assumptions that the knowledge is complete. This characterizes the *Closed World Assumption* (CWA) which makes the assumption that the logical program contains complete knowledge in which ground atomic formulas are true; and under the CWA, if a fact is not in the database then it is not true [Fortineau *et al*., 2012].

On the other side of the coin, OWL/DL and SWRL work under the *Open World Assumption* (OWA), mainly because they originally focus on the Semantic Web that deals with an unlimited knowledge resource (Internet). In OWA, when an assertion is not found as a known fact, the assertion truth-value is unknown [Fortineau *et al*., 2012].

In this respect, it should be examined the impact that the difference between OWA and CWA can have on results of the extraction rules in OntoILPER framework. First, related directly to the CWA, the *Unique Name Assumption* (UNA) states that individuals with different names are *different*. This implies that the relational model underlying the Prolog database of facts in OntoILPER, i.e., the graph-based model for representing sentences, makes two simplifying assumptions:

- it is assumed that the only objects and relationships that exist in the domain are those that are explicitly represented in the database. Therefore, this characterizes the CWA.
- identifiers uniquely identify objects in the domain (UNA). As a result, one obtains a single canonical model, where objects and relationships are in a one to one correspondence with the data in the Prolog factual database.

Second, the interest of using ontologies is the extendibility of the model based on OWA. Indeed, the two simplifying assumptions seen earlier do not favour the richness of expressivity needed for the Web, and thus they should be rejected in that context. This represents an important paradox that constraints model design, as also remarked in [Fortineau et. al, 2012], i.e., the same query, on the same model, can give different results regarding if the reasoning is made under a closed or an open world.

Given that the OntoILPER Framework performs the translation of the learned rules under the CWA, but the final repository of these rules is under the OWA, one could expect a "semantic mismatch problem" because the facts that can be inferred in the CWA, may not be true in OWA. In fact, this same problem is discussed in [Wang and Wu, 2011] where control rules were converted to SWRL. The authors propose a new reasoner called *BCAR* for reasoning on SWRL-based models under the CWA, achieving very encouraging results.

Simulating UNA with HasKey Axiom. As seen earlier, the semantic mismatch problem must be carefully addressed in view of the difference between the CWA vs. OWA paradigms. Such a difference may directly interfere on the results of inference rules when changing from one formalism to another. In the Prolog database of facts, unique identifiers are employed to uniquely identify individuals of a given class. Similarly, the OWL/DL version of the same Prolog database in OntoILPER has to be handled appropriately in order to maintaining the same semantics regarding the UNA assumed in Prolog.

To accomplish this, OntoILPER takes advantage of the *HasKey axiom* in OWL 2 version. The HasKey axiom was included by the World Wide Web Consortium (W3C) working group[1] as a restricted variant of keys commonly known as *easy keys* [Grau *et al*., 2008]. Such an axiom, a kind of a DL-Safe rule has the form *HasKey(CP1..Pn)*, which states that each named instance of a class is uniquely identified by a (data or object) property, i.e., if two named instances of the class coincide on values for each of key properties, then these two individuals are the same. Hermit, Pellet and KAON2 are some of the reasoners that have implemented the HasKey axiom.

For instance, the OWL axioms in the box below states that a person in the domain is uniquely identified by his/her social security number. Such OWL/DL axioms are expressed in functional notation[2].

```
Declaration( DataProperty(:hasSSN) )      // hasSSN data property assertion
FunctionalDataProperty(:hasSSN)           // hasSSN is  a functional property
HasKey( :Person :hasSSN  )                  // assign the hasSSN to the class Person

DataPropertyAssertion(:hasSSN :PSmith "123-45-6789")
        // PSmith´s social security number is "123-45-6789"

DataPropertyAssertion(:hasSSN :PaulSmith "123-45-6789")
        // PaulSmith´s social security number is "123-45-6789"
```

Use of the HasKey axiom for simulating UNA in OWA.

Given that the individuals “PSmith” and “PaulSmith” are named instances, the above axioms entail that “PSmith’ and “PeterSmith” are the same individuals, that is, *SameIndivividual*(*PSmith PaulSmith)* is true. Conversely, the following assertion is false: *DifferentIndividuals(:PSmith a:PaulSmith)*.

It should be highlighted that, one could declare the named individuals in the above example as being all different to each other by means of the *DifferentIndividuals* axiom:

DifferentIndividuals(:id_1 :id_2, ..., :id_n)

However, this way should cause performance problems for reasoners when the number of individuals is large.

When converting Prolog rules to SWRL rules, after that all facts in the Prolog database are converted to OWL/DL, it is also associated a *HasKey* axiom to each one of the classes in the working ontologies. With this, one guarantees the semantic equivalence of unique identifiers, provided that each new instance in the working ontologies is assigned a unique name.

As recommended by W3C, the semantics of the hasKey axioms is specific in the sense that this axiom applies *only* to individuals explicitly introduced in the ontology by name, and not to *unnamed* individuals[3]. Indeed, being equivalent to a variant of DL-safe rules, the *HasKey* axiom will usually not influence class-based inferences, such as the computation of the subsumption hierarchy. On the other hand, the *HasKey* axiom can play a crucial role in answering queries about individuals [Grau, 2008]. Other benefits of using the *HasKey* axiom is that this axiom performs more efficiently by the SW reasoners than the

[1] http://www.w3.org

[2] OWL 2. Structural Specification and Functional-Style Syntax (Second Edition). http://www.w3.org/TR/owl2-syntax/

[3] Unnamed individuals consist of the individuals whose existence is implied by existential quantification.

AllDifferent axiom, normally used in OWL/DL for distinguishing each one of the ontological instances.

In the context of the present thesis, the target corpora to be processed can be very large containing hundreds or even thousands of documents. This could result in a huge number of graphs. Consequently, in order to cope with a great number of possible candidate instances to be inferred in the domain ontology, the non-functional requirement of a rapid response time when classifying instances in the ontology is crucial. Furthermore, it is intended to use the most prominent OWL/DL reasoners to assess the runtime performance of the inference task on the classification of instances of classes and relations in the ontologies.