National Distance Learning University

Computer Science School

Natural Language Processing & Information Retrieval Research Group

Computer Science

Master's Thesis

Knowledge Capture and Textual Inference

Bernardo Cabaleiro Barciela

Madrid, February 2014

National Distance Learning University

Computer Science School

Natural Language Processing & Information Retrieval Research Group

Computer Science

Master's Thesis

Knowledge Capture and Textual Inference

Author: Bernardo Cabaleiro Barciela
Advisor: Anselmo Peñas Padilla

Madrid, February 2014

Master's Thesis
KNOWLEDGE CAPTURE AND TEXTUAL INFERENCE

Author
BERNARDO CABALEIRO BARCIELA

Advisor
ANSELMO PEÑAS PADILLA

The defense of the present Master's thesis has been done on February 27th 2014, being evaluated from the next tribunal:

PRESIDENT:

SECRETARY:

VOCAL:

and having obtained the next QUALIFICATION:

MADRID, 27TH FEBRUARY 2014

A mi familia.

Content

Acknowledgments **xi**

Preface **xiii**

List of Figures **xv**

List of Tables **xvii**

Alphabetic Index **1**

1 Introduction **1**
1.1 Text, Knowledge and Inference 1
1.2 General Hypothesis 3
1.3 Research Questions 3
1.4 Objectives 4
1.5 Contributions 5
1.6 Outline 6

2 Background and Related Work **7**
2.1 Gap between Language and Knowledge Bases 9
2.2 General Picture 9
2.3 Deep Processing of Text 10
2.3.1 Part of Speech Tagging 12
2.3.2 Lemmatization 12
2.3.3 Parsing 12
2.3.4 Named Entity Recognition 13
2.3.5 Event Recognition 14
2.3.6 Coreference Resolution 14
2.3.7 Semantic Role Labeling 14

2.3.8 Extraction of Temporal Expressions 15
2.3.9 Temporal Linking 15
2.4 Information Extraction and Relation Extraction 16
2.5 Acquisition, Representation and Uses of Knowledge 17
2.6 Relational Knowledge 18
2.7 Open Information Extraction 19
2.8 Propositional Knowledge 22
2.9 Textual inferences and Applications 24
2.9.1 Type Coercion 24
2.9.2 Logical Metonymy 25
2.9.3 Implicit Semantic Role Labeling 26
2.9.4 Paraphrase and Textual Entailment 26
2.10 Evaluation 26

3 Graph Based Representation of Text 29
3.1 Why a Graphical Representation at Document Level? 29
3.2 From Plain Text to Document Graphs 30
3.3 Representations 31
3.3.1 Initial Representation 31
3.3.2 Enriched Representation 32
3.4 Methodology 34
3.5 Implementation 36
3.5.1 Data 38
3.5.2 Output 38

4 Automatic Capture of Semantic Classes 39
4.1 A Brief Overview on Semantic Classes 39
4.2 Methodology 40
4.3 Implementation 41

5 The Effect of Graphical Representation on Relation Extraction 45
5.1 The Temporal Slot Filling Task 45
5.2 Research Questions 47
5.3 Experimental design 48
5.4 Results 50
5.5 Conclusions 52

6 The Role of Semantic Classes in Parsing Appositions. 55
6.1 About the Relation between Appositions and Semantic Classes . 55
6.2 Parsing Appositives 56
6.3 Methodology 57
6.4 Cases 58
6.4.1 One Valid Candidate 58
6.4.2 Several Valid Candidates 58
6.4.3 Undecidable Candidates 59
6.4.4 No-apposition Case 59
6.4.5 Summing up 60
6.5 Experiment Design 61
6.5.1 Method 61
6.5.2 Measures 61
6.5.3 Test Set 62
6.5.4 Baseline 62
6.5.5 Configurations 63
6.6 Results 63
6.6.1 Evidence 64
6.6.2 Measures 64
6.6.3 Best Configuration 65
6.6.4 Examples 65

7 Conclusions and Future Work 67
7.1 Summary 67
7.2 Looking Forward 71

8 Author's publications related to this work 73

Bibliography 87

Appendix

A Graph transformation rules 91
A.1 Rules for semantic class extraction 91
A.2 Rules for genitive normalization 95
A.3 Rules for naïve semantic role labeling 97
A.4 Rules for semantic edges addition 101

A.5 Rules for delete spurious relations in the enriched representation 103

B Graph Representation 109

B.1 Initial Representation 109

B.1.1 DOT output 109

B.1.2 JSON output 111

B.2 Enriched Representation 115

B.2.1 DOT output 115

B.2.2 JSON output 117

ACKNOWLEDGMENTS

First I really would like to thank my advisor Anselmo Peñas for his guidance through the writing of this master's thesis. I am deeply grateful to him for his support and for taking the time to have some enriching talks about language.

Many thanks to my colleagues of the Natural Language Processing & Information Retrieval Group, not only for the many research discussions, but also for their friendship.

I would also want to express my gratitude to the people from the University of Edinburgh, specially Mirella Lapata and Oier López de Lacalle, for their support and help during my stay there.

This work has been partially supported by the Spanish Ministry of Science and Innovation, through the project Holopedia (TIN2010-21128-C02) and by the Spanish Government (MINECO) in the framework of CHIST-ERA program (READERS project).

PREFACE

The present and future information needs of the society rely on the ability of computers to understand and manage knowledge. The lack of this mechanism explains the problems of knowledge driven systems to effectively perform tasks as question answering and machine reading.

One of the biggest bottlenecks is the automatic knowledge acquisition problem. In the actual stage of development, it seems obvious that only semi-supervised or unsupervised techniques can scale to deal with large corpora of natural language like the Web. The trend has evolved from populating a predefined ontology to expressing knowledge through either unconstrained relations or propositions.

The arrival of new deep language processing technologies let us think that we can annotate large collections of text with accurate predicates that can be used to extracting knowledge from text without tying it to any predefined logical schema.

On the other hand, it is not clear which tasks can harness this knowledge and how it can be done.

This master's thesis proposes a new method of knowledge capture and textual inference based on three cornerstones: (1) First, we develop a procedure to turn plain text into a graph based representation taking advantage of existing tools. (2) Second, we develop a proposition extraction system. (3) Lastly, we study an unsupervised method for correction of appositive dependencies, as an example of the textual inferences that the generated proposition store enables.

In addition, we generate two useful resources for future tasks of natural language processing: A corpus of 7 million documents represented as semantically enriched graphs and a proposition store of semantic classes with 8 million instances of entity-class relations.

List of Figures

2.1 General picture of the knowledge bases and its population techniques. 11

3.1 Transformation from initial representation, G_D, to enriched representation, G_C, for the example document: “David’s wife, Julia, is celebrating her birthday. She was born in September 1979". . . . 36

List of Tables

3.1 Hardware used to represent the collection of documents as graphs. 37

4.1 Most frequent classes associated to the named entity type *person*. 41
4.2 Patterns for the assignment of semantic classes. Each entry belongs to a tuple $governor- \ dependency- \ dependant$ where NN is a common name and NE is a named entity. 42
4.3 Hardware used to generate the knowledge base and retrieve the probabilities associated to each instance. 43

5.1 Slot names for the two entity types in the RSF task 46
5.2 Slot names for the two entity types in the TSF task 47
5.3 Temporal relations in the TSF training data 48
5.4 Features included in the model for RSF. 49
5.5 Features included in the model for TSF. 50
5.6 Regular Slot Filling task results 50
5.7 Final results on the Temporal Slot Filling task 51
5.8 Temporal Slot Filling task results until the relation extraction phase 51

6.1 Classes of appositions. 60
6.2 Baseline: Selection of candidates by the parsers. 62
6.3 Results considering the entity name as evidence: Configurations 1, 2 and 3. 64
6.4 Results considering the entity type as evidence: Configurations 4, 5 and 6. 64
6.5 Results considering both entity name and entity type as evidence: Configurations 7 and 8. 65
6.6 Example of apposition corrected. 66
6.7 Example of apposition incorrectly changed. 66

CHAPTER 1
INTRODUCTION

Humans rely on their knowledge to understand language, but also they are able to acquire knowledge from the language. However, despite of the years that have passed from the initial days of artificial intelligence, even now it is not clear how we can provide this behavior to computers.

1.1 Text, Knowledge and Inference

In the current stage of development there is a renewed interest on natural language understanding. There are three main reasons: there are plenty of data available, tools and methods of natural language processing are more refined and mature, and crucially, the computational power is bigger and cheaper.

The range of tasks that can potentially benefit from a machine able to acquire knowledge from language is very wide; especially those that need some kind of inference, in particular, to interpret language. For instance, question answering or text summarization.

Compile large databases of knowledge is a hard task: Knowledge is changing continuously, and it is expensive and time-consuming to get experts to annotate every little piece of data. Some domains are just too wide to completely write down all the available information. Moreover, to define a structure to store

and use knowledge is not trivial, and this produces formalisms far from natural language. Thus, there is a gap too wide between natural language and knowledge representations formalisms, being difficult to automatically learn the mapping from one to the other.

Related to this, there is a long tradition of research on automated knowledge acquisition. These techniques can take advantage of the bulk of documents available nowadays, either with structured or unstructured text.

There are two main perspectives. On the one hand, it is possible to start from a predefined logical model and then populate it with data found in texts through supervised or semi-supervised methods. This approach is known as information extraction, and it is a very popular research field since the 80's.

Although this approach is very useful for a wide range of tasks, it is limited because the discovered data are tied to the predefined model and thus they are domain-dependent. To solve this issue, it has been proposed the Open Information Extraction paradigm.

This technique is able to acquire knowledge free of any model. However it has some important drawbacks: Often the knowledge extracted is incomplete and noisy. Therefore it is necessary to filter and validate the obtained data, complete missing data and deal with uncertainty in some degree. Furthermore, there are very few works trying to explore what kind of inferences can be done with such resources.

This work aims at exploring the process of knowledge acquisition in the paradigm of the Open Information Extraction and its use for textual inference. It is structured in three major points:

1. Document representation: Our first step to gather knowledge is to represent documents in a machine friendly format that expresses the information contained on plain text. In particular we explore a graph-based representation obtained after deep language processing.

2. Knowledge acquisition: The second step is to extract knowledge from the document representation. In our case, knowledge is expressed using natural language through propositions, which are predicate structures that express open relations.

3. Textual inference: There are many open research questions about the use of this kind of knowledge and the inferences that can be performed with

it. We explore here this issue, in particular, if knowledge about semantic classes is useful to improve dependency parsing. Specifically, we use this knowledge for the correction of appositive dependencies.

1.2 General Hypothesis

There are many approaches to acquire and use common sense knowledge. Our starting point is grounded by the next hypotheses:

The first hypothesis is that common sense knowledge can be obtained after processing large collections of text. We use a knowledge base close to natural language as we aim to extract open relations. Propositions based on predicate structures can serve as basis for some textual inferences such as: recovery of implicit predicates, logical metonymy, implicit semantic role labeling, etc.

The second hypothesis is related to the representation needed to obtain the knowledge. We expect to gather more knowledge by capturing long distance relations present on the text. A graph-based representation can be suitable for this task. Moreover, this kind of representation allows to include extra information. Enriched graphs with semantic information can help to acquire better propositional knowledge and at the same time reduce the sparsity.

Lastly, we expect that propositional knowledge can be used to perform textual inference for tasks related to language interpretation. Parsing is a knowledge-dependent task that can use textual inferences.

1.3 Research Questions

In the process of defining and creating the knowledge acquisition and textual inference system we formulate the next research questions.

- What problems arise from developing an automatically acquired propositional knowledge base?
- Is it feasible to build the knowledge base from documents represented as graphs? What does this kind of representations provide?
- How can we perform textual inference using the propositions?

We have decided to create a compact representation at document level that aggregates morphosyntactic and semantic information. Regarding this representation:

- What are the main problems in the design of the representation?
- Which are the benefits and the drawbacks?
- In the context of automatic relation extraction, is it useful?

Our last objective is to use the extracted knowledge to perform textual inference, specifically to improve parsing in appositive dependencies. We tackle the problem of grammatical ambiguity when there are several candidates to govern an apposition with a named entity as a dependent. We assume that the common noun with higher semantic compatibility with the named entity is the best choice to govern the apposition. In this context, we formulate the next research questions:

- How do parsers behave when they have to process appositions and what kind of errors do they commit?
- Is it possible to overcome these errors considering semantic information captured previously from text collections? What evidence can it provide to characterize the named entity?
- What is the most effective way to measure the semantic compatibility between the candidates and the named entity?
- What configuration of evidences and measures achieves the best results?

1.4 Objectives

In order to answer our research questions, we aim to complete the next objectives:

1. Obtain a large collection of documents represented as graphs. The representation has to fulfill the next requisites:
    - Different mentions of the same named entity have to be clustered together.

- Syntactic relations have to be simplified to normalize the predicates in the text.
- Implicit semantic information of the text has to be aggregated to the representation.

2. Use the collection to compile a large knowledge base. Knowledge has to be structured in propositions.

3. Evaluate the contribution of the graph-based representation in the KBP2011 Temporal Slot Filling Task.

4. Use the knowledge base to perform textual inference. We use it for the correction of appositive dependencies.

1.5 Contributions

As a result of this work we obtain several useful resources:

1. On the one hand we define and implement a procedure to turn plain text into a graph-based representation using free distributions tools. We develop two different representations, initial and enriched (Section 3.3).

2. The next contribution is the methodology and the implementation of a proposition extraction system.

3. We also study the cases of the ambiguous appositive relations according to the semantic compatibility of the candidates to governor and the dependent part.

4. We design and implement an unsupervised method for the correction of appositive dependencies. This method finds grammatical ambiguous cases where there are two or more candidates to be the governor of an apposition. Then, it decides which candidate is better according to the semantic compatibility with the dependent part. We compare several sources of evidence, different semantic compatibility measures and its combinations, concluding which is the best approach.

5. We produce three collections of documents represented as graphs. The first collection is the corpora of the KBP 2011 Temporal Slot Filling Task. It contains about 1.7 million documents. The second one is the Gigaword Corpus. We have processed about 5 million documents of this collection. The third one is the Webtext Corpus of the KBP 2012 Slot Filling Task. It contains 1 million documents. In total, we represent 7.7 million documents as graphs. We produce three different outputs for these graphs, DOT, JSON and Java serialized objects (Section 3.5.2).

6. The last contribution is a knowledge base of semantic classes that contains near 8 million instances of entity-class relations.

In the context of this work, we publish several articles (See Section 8).

1.6 Outline

Chapter 2 shows the relevant work done in related areas of natural language processing, including deep processing of text, information extraction, knowledge capture and relational knowledge.

Chapters 3 and 4 describe the method and implementation of the tools to acquire the knowledge. In Chapter 3 we define a graphical representation at document level, which is used as a base for the knowledge capture in futures stages. In Chapter 4 we show a method to automatically acquire semantic class knowledge.

Chapters 5 and 6 evaluate the resources created by applying them to natural language processing tasks that involve language understanding. In Chapter 5 we extrinsically evaluate the utility of the graphical representation through a participation in the KBP Temporal Slot Filling Task. In Chapter 6 we study the relation between semantic classes and apposition dependencies, and provide a method to improve decision making of parsers in cases with grammatical ambiguity.

Finally in Chapter 7 we provide some conclusions and state our future directions for research.

CHAPTER 2
BACKGROUND AND RELATED WORK

There is a renewed interest in knowledge driven systems, with many attempts to advance the state of the art in natural language understanding and reasoning.

Knowledge Bases (KB) are a kind of databases that are used for store and manage knowledge, where manage means the capacity to organize and serve that knowledge. They contain information that describes a domain, employing for it a representation vocabulary, which is known as representation language.

Some sources of knowledge are oriented to direct exploitation by people (e.g. Wikipedia). They provide an efficient access to the users that search for texts, usually formative, as manuals, articles or encyclopedias. These resources are written in natural language, and therefore intelligent systems cannot use them easily. Although they are really useful, even for knowledge acquisition, they fall outside the scope of the current work.

We focus instead on computer-oriented databases. They contain information organized for other systems, named knowledge based systems (KBS) that use artificial intelligence techniques to solve complex problems.

Unfortunately, these systems depend on big amounts of data. There are two main reasons: In the first place, extensive knowledge over the domain is often crucial for understanding and disambiguation. Second, knowledge should be able to express the concepts of the domain discourse.

Another crucial part in a KBS is the inference capability, which is the ability

to create new knowledge from data. It heavily depends on the language used to represent knowledge.

How to gather and represent data has been a research topic for many years. One problem comes with the scale: high quality data leads to small databases, whereas big databases are often noisy. The other comes with the representation: higher structured languages have less representative power, but provide easier inferences, whereas open languages are able to express more knowledge but they are more difficult to use for inference.

There are three main methods to build such databases: pay professionals, ask volunteers or acquire it automatically. Generally speaking, these methods go from high quality and low size to low quality and high size.

The two first methods cannot scale to web size. As a result, the only feasible method to build a large knowledge base is to acquire it from text.

The information contained can be divided in two types:

- Factual knowledge provides knowledge about the objective realities in the interest domain (objects, relations, events, states, causal relations, etc.) mainly because it is particularly useful for plenty of problems, but also because it is easy to acquire.

- Solving-problems knowledge provides knowledge about how to get different objectives. For example, methods to solve problems across different domains.

This chapter is structured as follows: First of all, we describe why it is important to close the gap between language and knowledge bases, and then we show the general picture of the knowledge acquisition problem. Then, we describe a set of techniques for extracting an expressive representation from text, known as deep processing of text. Next section resumes the research on the field of relational knowledge. We understand relational knowledge as data structured in relations. A close research field is propositional knowledge, which we explain right after. Propositional knowledge differs from relational knowledge in that propositions do not have to express necessarily a predefined relation, but any fact in the world defined by a predicate and its arguments. Then we address some of the most interesting applications of this kind of textual knowledge bases. Finally we explain some methods to evaluate knowledge bases.

2.1 Gap between Language and Knowledge Bases

The first problem that arises when building a knowledge database is data representation. [Hobbs, 1985] argues that a representation language has to be defined according two criteria:

1. *The notation should be as close to natural language as possible*. This criterion refers to the ability of a language of encoding every fact. Therefore, according only with this criterion, the ideal choice would be the natural language itself.

2. *The notation should be syntactically simple*. These criterion aims to obtain a representation that simplifies the manipulation of the facts extracted. Unfortunately, it invalidates natural language as a choice, and favors representations as ontologies.

Natural language understanding systems need to chain complex sequences of inferences to achieve its goals. These systems need to have reasoning strategies to choose between different paths of reasoning.

Highly structured bases, like ontologies, are easy to understand, and it is also easy to build a inference engine on top of them. However, they need to be defined by hand, and every piece of information that does not fit in the general schema will be lost.

When dealing with web-size knowledge extraction, it is hard to find a enough detailed schema to represent every data that can potentially be gathered. Moreover, different domains could need different schemas.

On the other hand, natural language allows us to express all the knowledge, as it is flexible enough to be used in every domain. However, automatic systems struggle to handle it to perform inference.

As a result, in order to perform textual inference it is necessary to find a compromise solution that balances the trade-offs between natural language and structured ontologies.

2.2 General Picture

There are a set of techniques that may be considered part of the automatic knowledge base generation, such as information extraction, relation extraction,

entity linking, open information extraction, semantic parsing, etc.

All of them provide a method to turn text into some sort of structured representation. However, there are many important concept differences between them:

- The representations can be classified according to the presence or absence of a predefined schema where the information obtained is bounded.
- The extracted information can be structured in languages with a variable degree of flexibility, from ontologies to what is called now Linked Data.

Figure 2.1 shows a general picture of these different kinds of knowledge databases and the acquisition techniques used to populate them. There are two main ways to structure text in the first place:

- Through information extraction and relation extraction, which leads to knowledge bases with a predefined schema.

  In this context, the flexibility of the knowledge representation ranges from structured specific domain ontologies to general purpose linked data bases such as Freebase [Bollacker et al., 2008], DBpedia [Auer et al., 2007] or Yago [Suchanek et al., 2007], which are closer to natural language.

- Through semantic representation of text, including deep processing, that produces knowledge without a predefined schema.

  In this case knowledge representation is always flexible, because it has to provide a mechanism to express unknown relations. Even so, we can see two approaches: (1) Knowledge expressed by binary relations, although there is not a particular set of relation labels. This is usually called Open Information Extraction. (2) Knowledge expressed by predicates, with a free number of arguments (usually from one to three).

2.3 Deep Processing of Text

This section explains some of the techniques commonly used to enrich text. We understand deep processing as the task of transform a plain text to a representation which includes structured syntactic and semantic features in a machine

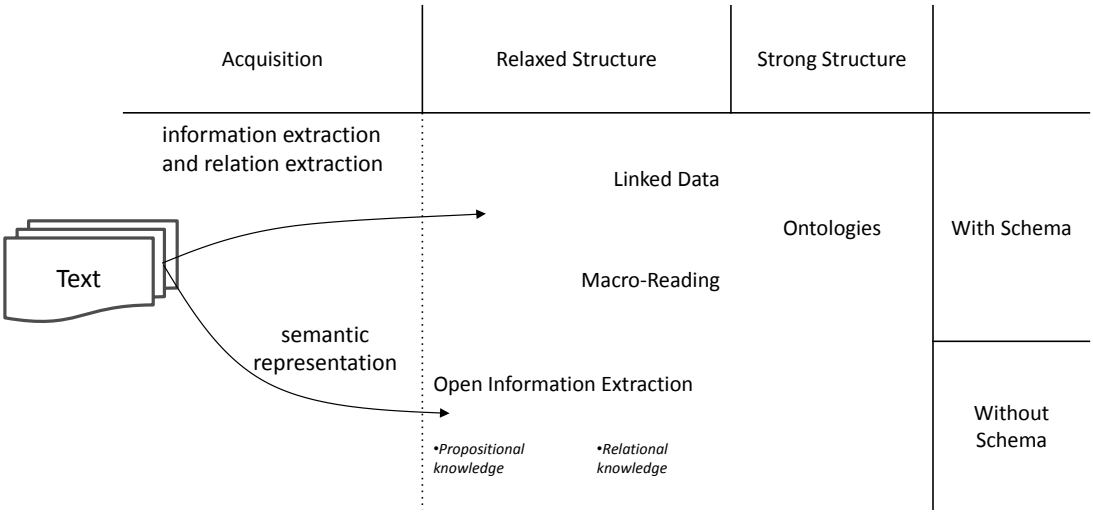

Figure 2.1. General picture of the knowledge bases and its population techniques.

readable format that allow further inference. It can be seen as finding a computational representation closer to a logical representation.

There are two general ways to process text: shallow processing and deep processing.

Shallow processing is commonly referred as a bag-of-words approach, where different statistical processes are used to characterize text. It is fast and cheap, but usually can only get the surface semantics of sentences.

In contrast with shallow processing, deep linguistic processing provides a rich, expressive, and structured representation which capture long-distance dependencies [Baldwin et al., 2007].

The major drawback of deep linguistic processing is the amount of computational effort that it takes. As a result, many processes cannot be applied to large text corpora. However, some of the processing techniques, such as the ones used in this work, are affordable with the actual resources.

In the next sections, we are going to describe the techniques used in the context of this work. They are part of speech tagging, lemmatization, parsing, named entity recognition, event recognition, coreference resolution, semantic role labeling, extraction of temporal expressions and temporal linking.

2.3.1 Part of Speech Tagging

Part of speech tagging, or POS tagging, is one of the most common and studied process to enrich a text. POS taggers assign to each word in a sentence the contextually appropriate grammatical descriptor [Voutilainen, 2003].

In many cases, POS taggers also provide inflectional and lexico-semantic information, such as the distinction between common and proper nouns, or verb tenses.

The main problem is that, in most languages, each word can play different roles depending on the semantics of the sentence. Moreover, this task is very language-dependent, so generally a different POS tagger has to be used for each language.

There are two common approaches to develop POS taggers, one is rule-based, where some hand-defined or automatically acquired rules are used to tag the words, and the other is stochastic, where statistical techniques are used. The latter outperforms the former, as it is reported 97% or more accuracy [Toutanova et al., 2003, Shen et al., 2007, Hajič et al., 2009]. However, this problem is not solved yet, as some authors claim that the sentence accuracy is only 56% [Manning, 2011].

2.3.2 Lemmatization

A lemmatizer extracts the lemma for a word. This is the canonical form of a word, which is the particular form that, chosen by convention, represents the lexeme. It is useful because it allows grouping all the different inflected forms of a word under the same representation.

In some occasions, it is important to disambiguate the word according to the context, and in this case the task is not trivial because it requires understanding the context that surrounds the word.

2.3.3 Parsing

Parsing is the process of analyzing a sentence to annotate the components with some relevant information.

There are different kinds of parsers. Here, we distinguish between two: constituent parsing and dependency parsing.

Constituent Parsing

A constituency parser splits sentences into constituents, which are usually the types of phrases. Relations between constituents are unlabeled.

Most of the parsers are built with a probabilistic model trained on a hand-labeled corpora. There are two main approaches for statistical parsing, the Head-driven Phrase Structure Grammar (HPSG) [Pollard, 1994] and the Lexical Functional Grammar (LFG) [Kaplan and Bresnan, 1982].

Some of the most popular parsers are the Charniak parser [Charniak and Johnson, 2005], Collings parser [Collins, 2003], Stanford parser [Klein and Manning, 2003, De Marneffe et al., 2006], Minipar [Lin, 2003] and English Slot Grammar parser [McCord et al., 2012].

Dependency Parsing

A dependency parser is a system that annotates the grammatical structure of a sentence. It is a critical part in many NLP systems, and consequently it is a very active area of research.

In dependency parsing, the output is a tree whose nodes are words, and edges are the syntactic relations between them.

Some examples of dependency parsers are: FANSE parser [Tratz and Hovy, 2011], (extended) Stanford Parser [De Marneffe et al., 2006], MaltParser [Nivre et al., 2006], Ensemble Parser [Surdeanu and Manning, 2010].

2.3.4 Named Entity Recognition

Named entity recognition is, as defined at the Sixth Message Understanding Conference [Grishman and Sundheim, 1996], the task to identify the names of all the people, organizations and geographic locations in a text. This is frequently extended to a few other entities such as times, currencies and percentage expressions, but it is possible to find systems that distinguish between many more categories [Sekine and Nobata, 2004]. NER is an “easy” task, and it is considered as solved.

There are two main approaches to solve this task, handcrafted rule-based algorithms and machine-learning approaches. The latter achieve better results, and are commonly used, for example in [Finkel et al., 2005,

Ratinov and Roth, 2009].

2.3.5 Event Recognition

Event recognition is a close field of named entity recognition. In this case, the goal is to find events in the text, which consists in states, actions or properties. It is a very important task, especially for question-answering systems, as events are frequently a key part of the input question, but also for tasks related to natural language understanding.

This task is usually performed with a pre-defined set of relation patterns [Brill et al., 2002, Hovy et al., 2002], but there are some other approaches that combine linguistic and statistic knowledge to improve the results [Saurí et al., 2005, Bethard and Martin, 2006].

2.3.6 Coreference Resolution

Coreference relations occur between different instances in a discourse that refer to the same real world entity. These instances can be written with different linguistic expressions, so relate them is not a trivial process.

However, knowing that two instances refer to the same entity is very important to correctly comprehend a text. This makes coreference resolution a very wide research field, but despite all efforts the performance of the coreference resolvers in the state of the art is not good enough [Ng, 2010].

In the first place, most of the coreference research tries to apply heuristic approaches, but from the 1990s most of the systems use machine learning techniques [Raghunathan et al., 2010, Lee et al., 2011].

2.3.7 Semantic Role Labeling

Semantic roles are the part that each of the syntactic constituents plays in the predicate of a sentence. These constituents are the phrases that point the agent (who), the patient (whom), place (where), cause (why), time (when) and others. Semantic role labeling (SRL) is the task to find and tag them. It is a key process for many NLP applications, and therefore there is a long research tradition on this field.

Some SRL systems are the Illinois Semantic Role Labeler

[Punyakanok et al., 2008], the ClearNLP package[1] and the MateTools package, which includes the SLR system of [Björkelund et al., 2009].

Semantic role labeling is closely related to predicate-argument identification, where roles are substituted by arguments that lack of a specific semantic as agent or patient, and instead they perform a different undefined action depending on the predicate. The predicate-argument structure builder [McCord et al., 2012] is an example of this kind of systems.

2.3.8 Extraction of Temporal Expressions

Temporal expressions determine when does the events in the text occur, and thus they are important to anchor them to a timeline. However there is many ways to express a temporal expression, and some of them require a reference time to be computed (i.e. today, tomorrow, yesterday, next month, etc.).

In order to perform this task some systems were developed to locate and normalize temporal expressions. For example, the TempEx tagger [Mani and Wilson, 2000], GUTime tagger [Verhagen et al., 2005] or SUTime [Chang and Manning, 2012]. All of them are rule-based systems.

Moreover, it is useful to obtain all the dates and intervals in the same representation. The most accepted standard is TIMEX3 representation, contained within the TimeML language [Pustejovsky et al., 2003].

2.3.9 Temporal Linking

In many NLP tasks it is important to know which events occur after or before a concrete date (or other event). Here we denominate Temporal linking to the task of relate events to temporal expressions and infer the relative order of the events that appear in the text.

Unlike other systems discussed here, there is not much research on this area. Some of the most notable systems rely on hand-written rules, and their performance is not as good as desirable. The Tarsqi Toolkit [Verhagen and Pustejovsky, 2008] includes some tools to perform temporal linking, for example GUTenLINK [Mani et al., 2003] anchors events to time expressions and S2T creates temporal links from modal relations in the text.

[1] https://code.google.com/p/clearnlp/

2.4 Information Extraction and Relation Extraction

There are multiple approaches to structure data, depending on the target to be structured. Semantic parsers structure data at document level, and dependency and constituent analysis work at sentence level. On the other hand, information extraction systems try to retrieve relevant pieces of information, for example named entities in case of Named entity recognition systems.

Relation extraction is a subtask inside information extraction. Roughly it consists on identify semantic relations between entities. For example, in the sentence *John is married to Mary*, we can extract that there is a relation between two persons, and we can label it as *marry*.

More formally, a relation is an ordered binary relation $r(e1, e2)$, where r is the relation name, or label, $e1$ is the target entity (i.e. the entity about we are querying) and $e2$ is the value, the entity that holds the relation with the target entity. A relation instance is a single assignment of entities and relation. In the past example, the relation instance would be $marry(John, Mary)$.

Information extraction as we understand was first studied in the Message Understanding Conference (MUC) [Beth, 1995, Grishman and Sundheim, 1996, Chinchor, 1998] and continued with the program Automatic Content Extraction (ACE) [Maynard et al., 2003] until 2008. These were the first quantitative evaluation of systems of this kind, and they were a major factor for the advance of the state of the art in automatic text processing.

From 2009, the Text Analysis Conference (TAC) [McNamee and Dang, 2009] took over the ACE in the evaluation of information retrieval systems. The task Knowledge Base Representation (KBP) was proposed in this conference. It may be seen as a combination between information extraction and question answering, where the task is harder because the information must be gathered across different documents. A detailed explanation of this competition is shown in Chapter 5. This task contains the subtasks Regular Slot Filling (SF) and Temporal Slot Filling (TSF).

As it can be seen in [Ji et al., 2011], SF and TSF remain as open research problems because competitors could not get close to manual annotation results. Participants latter editions use similar representation systems, for example, the systems of CUNY [Artiles et al., 2011] and Stanford [Surdeanu et al., 2011] use

tokenization, segmentation, named entity detection, coreference resolution and syntactic dependency parsing.

This research topic is very popular, and it has many small competitions related. As an example, the DDI Extraction 2011 challenge[2] is a Spanish competition that focuses in the biological relation extraction.

2.5 Acquisition, Representation and Uses of Knowledge

Knowledge bases can be divided in three main groups depending on how the knowledge is acquired: manual, semiautomatic and automatic.

Some examples of KBs manually generated are WordNet [Miller, 1995], Cyc [Lenat, 1995] and SUMO [Pease et al., 2002]. These KBs are widely used in research, as manual annotations provide accurate data with almost no noise. However, generate a manual KB is very expensive, and therefore they are not updated or the updates are slow and difficult. Moreover, they cannot get the width and the recall of semi-automated process.

As a result some KBs were developed with information extraction techniques. Some of them are built extracting information from semi-structured text, such as Wikipedia infoboxes. These KBs have a precision around 95%. Some examples of these KBs are Yago [Suchanek et al., 2007], Kylin/KOG [Wu and Weld, 2007], and DBpedia [Auer et al., 2007].

Despite the good precision, these approaches depend on sources of semi-structured text, which is not as rich and abundant as natural language.

As a result, other approximations tried to extract the information from natural language directly. Most of the information available is presented in this format, and therefore this systems are potentially capable of obtain the most complete KB. The problem with natural language documents is how hard is to learn knowledge from them.

There are two main techniques to obtain this knowledge: semi-supervised and unsupervised techniques. Semi-supervised techniques consist in defining by hand a set of patterns or correct instances. For example, supposing a system that starts with a set of patterns, it will use them to collect a set of instances from the document collection. Then, it will search other matches

[2] http://labda.inf.uc3m.es/DDIExtraction2011/

of these instances in different sentences that will be used to define new patterns. This process is executed iteratively to increase the recall of the system. However, each iteration introduces more errors, decreasing the precision. DIPRE [Brin, 1999], SnowBall [Agichtein and Gravano, 2000] and KnowItAll [Etzioni et al., 2005] are some examples of this kind of KB.

These systems are limited because the learning depends on the initial seeds. Moreover they can acquire only a set of predefined relations.

Unsupervised systems were developed to solve this problem, as they do not need the initial seeds nor the expected relation types.

There are two major paradigms within the unsupervised acquisition systems: relational knowledge (Section 2.6) and propositional knowledge (Section 2.8).

The major drawback of unsupervised information extraction is that it has a high error rate compared to semi-supervised extraction. To avoid this problem it is common to use statistical techniques that search for redundancies in the obtained data for estimate the confidence in each one of the instances.

2.6 Relational Knowledge

This section shows the two main paradigms to extract relations according to a predefined schema. They are Preemptive information extraction and Macro-reading.

Preemptive Information Extraction [Shinyama and Sekine, 2006] first groups documents based on pairwise vector clustering, and then applies additional clustering to group entities based on document clusters. As it depends on cluster algorithms, this approach is difficult to scale to large corpora.

Macro-reading [Mitchell et al., 2009] is a different paradigm that aims to extract information at collection level, instead of sentence level. Their main idea is that it is not important to extract every little piece of information from a document, but that important facts will be stated many times on a large corpora with many rewordings, thus making easier the extraction task. They also use a pre-defined ontology to drive the extraction, so the task is actually to populate it. With this schema, inferences would be easier, however, it lacks the expressiveness to represent all the knowledge of the collection. ReadTheWeb is the first system based on the Macro Reading paradigm.

Never Ending Language Learning (NELL) [Carlson et al., 2010] is a related system. It learns extractors for a set of predefined relations through distant supervision. To do so, it uses a bootstrapping approach, it learns extraction patterns from extraction instances, and then runs the new patterns to extract new instances. These kind of systems tend to degrade after few iterations, but NELL uses different heuristics to reduce the noise.

2.7 Open Information Extraction

The Open Information Extraction (OIE) paradigm aims to extract high quality relations from natural language in Web size scale. OIE supposes a new point of view over the traditional information extraction. Whereas the first one generally learns an extractor for each target relation from labeled training examples, OIE aims to learn *relation phrases*, which are phrases that express arbitrary relations. The main advantage of this paradigm is that it is completely unsupervised, as it is independent of any dictionary of relations or instances.

Generally, OIE systems extract tuples *Arg1, predicate, Arg2* in four steps:

1. Label a set of sentences using heuristics or distant supervision.
2. Learn a relation phrase extractor using some probabilistic model based on sequences (Naïve Bayes, CRF, Markov Logic Network, etc.).
3. Once the sentence is identified, the system selects two candidates to arguments and selects which words in the path between them form the relation.
4. Finally some systems do an assessment of the instances in base of the redundancy found.

To avoid using lexical features, the extractors are trained using POS and NP-chunk features from examples heuristically generated from the Penn Treebank.

DIRT [Lin and Pantel, 2001a] (Discovering Inference Rules from Text) automatically identifies rules of inference from paths in dependency trees. Basically, a dependency parser is applied to a text collection, and then, the different paths between the arguments are gathered and finally the similarity between different

paths is calculated. This KB has been used in textual entailment among others [Marsi et al., 2007].

KnowItAll KnowItAll [Etzioni et al., 2004, Etzioni et al., 2005] can be considered the predecessor of the Open Information Extraction paradigm. It starts from a set of predefined relations, manually introduced by the user. Then it is able to extract information without labeled training examples using syntactic patterns. After the extraction phase, it refines the instances obtained using pointwise mutual information on information gathered on search engines.

This system only uses a part of speech tagger and dispense with named entity recognizers or dependency parsers. However, it relies on several search engine queries. As a result, it is not scalable and depends on search engines.

TextRunner TextRunner [Banko et al., 2007] is the first Open Information Extraction system. The relation phrase extractor uses a Naive Bayes model with unlexicalized POS and NP-chunk features. The examples for the training are generated through heuristics from the Penn Treebank.

In the labeling phase, TextRunner uses a parser to label a small set of training examples, restricted by three heuristics: there is a dependency chains between the arguments shorter than certain threshold, the paths between the arguments does not contain any sentence boundaries and the arguments are not just a pronoun.

These examples are used to generate features to train a Naive Bayes classifier. Some examples of features are number of tokens in the relation are: number of stopwords, part of speech tags, etc.

Then, the system makes a single pass over its corpus to identify entities and tags words with the part of speech. Each entity that is a suitable candidate is introduced in the classifier to label it as trustworthy or not.

Finally, the tuples extracted are normalized and aggregated to assign a probability for each one.

The experiments reported in [Banko et al., 2007] use a corpus of 9 million web pages, and claim to extract a total of 60.5 million tuples.

As stated in [Etzioni et al., 2011], the main drawbacks of this approach are the incoherent extractions and uninformative extractions. Incoherent extractions are relations that are not meaningful because the extractor chose the

wrong words to describe it. Uninformative extractions are cases where the tuple, besides being informative, misses critical information.

Resolver Resolver [Yates and Etzioni, 2007] is an extension of TextRunner. Its goal is to merge different syntactical variances of the same meaning. To do so, they use the data collected by TextRunner and iteratively use cluster algorithms to merge clusters of co-referential names.

WOE The main advantage of WOE [Wu and Weld, 2010] is that it extract its features from Wikipedia. They run the general schema where they start with a set of seeds, and then retrieve many examples through bootstrapping, to finally learn a set of patterns.

ReVerb ReVerb [Etzioni et al., 2011] addresses the problem of incoherent and uninformative extractions. To do so, it uses a shallow NLP to tag the words with its part of speech, and then apply a set of simple syntactic and semantic constraints defined by regular expressions. These techniques provided a significant advance compared to TextRunner and WOE.

OLLIE OLLIE [Mausam et al., 2012] addresses two important flaws in the previous OIE systems. First, it deals with relations not mediated by verbs, mainly those that occur in nominal compounds. Second, it takes into account the context of the sentences to avoid extracting relations that are not asserted as factual.

To do so, they start with the popular approach of starting with a set of seeds, in their case, 110.000 tuples and then build a bootstrapping set searching for these seeds in a Web corpus. Once they retrieve a set of sentences, they apply the Malt Dependency Parser [Nivre et al., 2006] and learn a set of syntactic and semantic patterns that encode different ways of expressing the relations. This process is called syntactic scope expansion.

Then they apply what they call the context analysis component. This component detects when a sentence does not assert a fact, but only states a conditional truth or an attribution.

A conditional truth is a case where the sentence states that something is true if some condition happens. OLLIE uses the dependence relations to ex-

tract the condition and annotate it in an additional field. In the attribution case, OLLIE adds an extra field that indicates who said, suggested, believes, hopes or doubts the information extracted. To do so they train a classifier to assign a confidence to each extraction using a different training set of 1000 instances.

CSD-IE One of the most recent systems of OIE is CSD-IE [Bast and Haussmann, 2013]. The main idea behind this system is to extract facts that semantically "belong together".

They illustrate this property through an example: *Ruth Gabriel, daughter of the actress and writer Ana Maria Bueno, was born in San Fernando.*,

In this sentence, they aim to identify that the sentence contains two entities, each one with its own semantic classes and relations. Identify, for example, that *Ruth Gabriel* is a *writer* would be an error, even if those words are in a close context. They denote the process of identify the context of each entity as contextual sentence decomposition. This process is similar to the techniques used in [Mausam et al., 2012].

2.8 Propositional Knowledge

In contrast with relational knowledge, propositional knowledge is not limited to extract relations. Instead, it is structured in propositions, which are a realization of a predicate and its arguments.

This approach is more general, as it allows expressing n-ary relations. Also, it provides a more fine-grained representation, where a relation is not limited by an extracted tag, but it can be any word read in the text.

These kind of specific, large sources of knowledge have the potential to help many issues related to natural language understanding. However, how to use propositional knowledge is a research line on an early stage of development.

The main drawback of propositions is their sparsity, as semantically similar propositions cannot be clustered without further processing.

KNEXT [Schubert, 2002] argues that text contain background knowledge in form of assertions that can be exploited by processing large quantities of text. In this way, they focus more in finding redundancies than in refining the intra-

document analysis. Their goal is to extract general relationships from texts, instead of predetermined kinds of facts. For achieve this, they use information about the phrase structure and also compositional interpretive rules. With these tools, they build a large lexical semantics knowledge base.

They follow the next algorithm:

1. Preprocess the input tree (e.g., mark infinitives, passives, temporal noun phrases and prepositional phrases, categorize prepositional phrases, etc.)

2. Apply a set of ordered patterns to the tree recursively; these amount to phrase structure rules allowing for regular expressions (including negation) on the right-hand side

3. For each successfully matched subtree, abstract the interpretations of the essential constituents, and combine the abstracted interpretations in accord with the semantic rule linked to the pattern that matched the subtree

4. In processing the tree recursively, collect interpretations of phrases expected to provide general "possibilistic" propositions

5. Formulate possibilistic propositions from the collected phrasal interpretations, and output these along with simple English verbalizations.

DART Closest to our job, DART (Discovery and Aggregation of Relations in Text)[Clark and Harrison, 2009] is a work based on the idea of Len Schubert that tries to exploit the redundancy of instances acquired from patterns in a corpora.

In this project, they obtain world knowledge in a semi-formal notation. They have a database with 23 million propositions. The main contributions ahead Schubert's work are the kind of tuples extracted, the amount of data collected and a way to evaluate the results. Their method uses a hand-built parser called SAPIR, which extracts the phrase structure, and then extracts tuples from the parse tree. To test that Dart database was useful, it was used in a parsing test and in recognizing textual entailment.

Prismatic PRISMATIC is also a technique to extract knowledge from large datasets. However the knowledge representation is more complex, as it is

based on frames instead on binary relations or triplets. Besides using a dependency parser, it also uses a coreference resolver and an entity recognizer. PRISMATIC is one of the bases over which the question answering system Watson was built [Ferrucci et al., 2010].

BKB Anselmo Peñas and Eduard Hovy [Peñas and Hovy, 2010] suggest a new approach where the data would be more structured, and where it would be possible to differentiate classes and instances of those classes.

For achieve this goal, they try to make explicit the implicit or missed information in the text. This is done relying on Hobbs's theory about abduction [Hobbs et al., 1988], so they try to find the proposition which best explains the knowledge already extracted, and then add this proposition to the knowledge base.

The first propositions are acquired through syntactic dependencies, which may be interpreted as semantic relations. Then, new semantic relations are imported trying to fit them to other syntactic dependencies already contained in the database.

To test this system, they do a comparison with DART and TextRunner, and they find that they add new capabilities as management of instances, discovering new relations and being able to fit the knowledge base to a specified domain.

2.9 Textual inferences and Applications

Despite the great effort in compile large knowledge databases, it is still not clear how to take advantage of the information extracted. The main questions are which tasks can be targeted and how can this knowledge be used.

Several tasks related to language interpretation are heavily knowledge dependent. For example, those tasks which include inferring implicit knowledge contained in the text. Here we provide some examples:

2.9.1 Type Coercion

A precise definition of type coercion is provided by [Pustejovsky, 1991]: *Type coercion is a semantic operation that converts an argument to the type that is*

expected by a function, where it would otherwise result in a type error

Recently it has been redefined as a task [Pustejovsky et al., 2010], where type coercion is *the task of characterizing the type of compositional operation that exist between a predicate and the argument that it selects. Specifically, the goal is to identify whether the type that a verb select is satisfied directly by the argument, or the argument must change type to satisfy the verb typing*.

This definition is based in that, when a predicate takes an argument, this argument may fit in that role only through an adjustment or coercion that makes it belong to the type that the predicate expects.

For example, *enjoy the book* actually means *enjoy reading the book*. Broadly speaking, type coercion would be to identify the accepted argument of *enjoy* would be an event like *reading* instead of an object like *book*, thus tagging the change as *event-for-object*.

An example of a work that address this issue is [Roberts and Harabagiu, 2011]

2.9.2 Logical Metonymy

According to the Oxford Dictionary, metonymy is *the substitution of the name of an attribute or adjunct for that of the thing meant, for example suit for business executive, or the turf for horse racing.*

This process is used very frequently by humans to optimize their communication, as it provides a fast way to translate concepts from a sender to a receiver.

However, this use implies that systems are not able to retrieve the full meaning of a sentence. To do so, they have to convert the concept stated on the text for the original meaning.

From the point of view of natural language processing, a logical metonymy occurs when a logical argument (i.e. subpart) of a semantic type that is selected by some function denotes the semantic type itself [Pustejovsky, 1991].

In fact, metonymy is a process very related to type coercion. Regarding the example used before *enjoy the book*, one can understand that being readable is part of the semantics of the book, thus inferring that the natural way of enjoy a book is to read it.

Several approaches have been proposed to solve this issue [Fass, 1991, Utiyama et al., 2000, Lapata et al., 2003, Lapata and Lascarides, 2003].

2.9.3 Implicit Semantic Role Labeling

Semantic Role Labeling systems deal with the problem of tagging the arguments of a predicate. However, this approach is limited in cases where the arguments are implicit in the sentence, especially in cases where they appear in a long distance context. As a result, predicates obtained with these systems are incomplete, and eventually can produce noisy predicates.

Implicit Semantic Role Labeling, also known as null-instantiation resolution, aims to recover these predicates. Whereas it has a long standing problem [Palmer et al., 1986], the popularity of this task has been rising recently due to a task proposed at SemEval-2010 [Ruppenhofer et al., 2010] and the availability of newer knowledge resources.

However, this is a very hard challenge, as the first ranked system on the competition achieved $F1 = 0.014$ [Chen et al., 2010].

2.9.4 Paraphrase and Textual Entailment

The process where speakers reword a sentence with different syntactic and lexical variations is called paraphrasing. This phenomenon results in big difficulties for machines to assert that two phrases mean the same. This problem is called textual entailment.

There is a long tradition of research in textual entailment. One newer approach, Paralex [Fader et al., 2013] tries to take advantage of knowledge bases to associate natural language patterns to database concepts, solving the problem of linking input strings with entities, relations and question patterns.

2.10 Evaluation

Knowledge bases evaluation is a difficult task, and it is easy to get into subjective measures. Ideally, it will consist in building a standard with all the information that should be available in the knowledge base and compare KB and standard with precision and recall.

However, this standard is not easy to build, in first place it will be a very intensive task, but moreover it is not trivial to decide in which point the knowledge base is complete, or when the granularity is enough.

Once is stated that the ideal evaluation is unfeasible, it is necessary to distinguish if the knowledge base uses a schema or it does not [Mayfield and Finin, 2012].

The first problem is easier. The most common method to evaluate is to select a small corpus of documents, run a system and then manually judge if each extraction is correct or incorrect. This is a feasible method, however, for some tasks it does not allow to compare different systems and moreover, it is expensive and depends on the judgment of people. For example, it is used in [Etzioni et al., 2011]

Also, as missing extractions are not tagged, recall only can be estimated. One way to do it is to use the total number of extractions from all systems compared labeled as correct by the judges.

In [Mayfield and Finin, 2012]a new evaluation is proposed. Instead of evaluating if every fact is present, it defines a set of evaluation queries, and check if the system is able to answer them. To solve the problem of align the KB entities with the gold standard it uses entry points. An entry point is defined by a document and an entity mention string. The entry points should be aligned with the constructed KB, and then is compared with the gold standard.

[Lawrie et al., 2013] introduces a new approach to KB comparison the exploiting of provenance, which is the link between the entities and the strings that mention them in a document. In this way, they can compare where and why two KBs differ.

An objective way of evaluate a knowledge base, either with or without predefined schema, is to include it in a system that solves a different task, this is, to perform an extrinsic evaluation. In this way, it can be measured the performance of the system using and without using the knowledge base, in order to compute the difference between both of them.

A related but different way of evaluation is to measure its utility in a concrete task. Here, the goal is to check if the knowledge base allows solving a problem that without it or with a different one is not possible.

CHAPTER 3
GRAPH BASED REPRESENTATION OF TEXT

In this chapter we explain the characteristics of the representation that we use as the base for the knowledge acquisition. This is a representation at document level, evolved from plain text to a graph representation through deep processing techniques.

3.1 Why a Graphical Representation at Document Level?

Computers lack of the human ability to read and comprehend natural language. This issue is due to their problems to build a conceptual model that expresses the semantics behind the text. Thus, a good text representation is a conceptual representation derived from text that expresses the semantics in a machine-readable format.

There is a long tradition in text representation research. First approaches are based on conceptual dependency theories from [Schank, 1972], and meaning-text theory from [Mel'cuk and Polguère, 1987]. Later some text representation techniques were used in large datasets, i.e. in the work from [Pradhan et al., 1994] applied to the medical domain. Despite the large number of representations proposed, each NLP task has different knowledge representation needs, so this is still an active research field.

One of the most widely used models is bag-of-words. In this model, a text is represented through the frequencies of each word contained. The reasons behind this popularity are that is computationally very cheap and allows expressing a brief notion of context. However, this model is very limited, as it loses many semantics encoded on the original text because it does not represent the original concepts, and the relation between them.

Many graph representations have been proposed to improve the document representation. The intuition is that these representations are capable of capturing the relations of the elements present on the original text. However, they often are built at sentence-level, thus they lack of cohesion between the elements on distant contexts.

Our hypothesis is that a graph representation at document level will solve this issue. We will build this representation by transforming plain text using tools of deep language processing.

Our objective is to provide a base for techniques of information extraction. If this base is appropriate, the classifiers could use new features that would allow the capture of context that might be distant in text.

3.2 From Plain Text to Document Graphs

The main goal of our representation is to express the whole semantics of one document in a simplified, condensed way.

To do so, we take advantage of coreference, which is a linguistic relation that is established between two or more expressions that refer to the same entity, being or not isomorphic. We use the term *discourse referent* coined by [Karttunen, 1968] to refer to the set of all of these expressions for an entity. We denote the process of creating a discourse referent by grouping the mentions as *collapse*. This procedure is similar to *file cards* used by [Heim, 1983] or *baskets* in [Recasens, 2010].

This process by itself would create a big graph with many different grammatical relations between its components. This will produce features with high sparsity, which is not desirable for automatic learning techniques. Our solution is to simplify the morphosyntactic relations in some sort of naïve semantic role labeling. We normalize different expressions that are semantically equal. We consider grammatical voice, genitives expressed in different ways, nominal

compounds, etc.

On top of that, we will enrich our graphs with extra semantic information extracted with simple rules. We assign semantic classes to named entities through a set of patterns that detect some structures with genitives, nominal compounds and appositions.

3.3 Representations

We want to measure the contribution of our techniques over the gain that the external tools provide. To do so, we will evaluate two different representations: Initial representation and enriched representation.

3.3.1 Initial Representation

In this configuration we use all the external tools to compose an initial representation. It contains nodes annotated with morphological information, edges with syntactic information, coreferences and temporal relations.

Specifically each document D is represented by a graph, G_D, with a set of nodes N_D and a set of edges A_D.

Nodes

Each node represents a unit of information, generally a word, except in two cases: a multi-word named entity or a verb and its auxiliaries. Nodes have attributes that in some cases are specific of some special nodes.

Attributes Nodes are tagged with a set of attributes, some of them are common for every node:

- Words
- Lemmas
- Morphosyntactic tag
- Descriptor

A descriptor is a representative string. For the nodes that are not entities the string is composed by the lemmas of the words. Otherwise we choose the string just as it is found on the text. Besides it, every other attribute is obtained from the dependency parser.

Special Nodes Moreover, some nodes are tagged with one or more special types that include extra attributes.

- Events: Verbs or actions that describe an event. Annotated with time, aspect and polarity. A detailed explanation of the events can be found in [Saurí et al., 2005].
- Temporal Expressions: Used to identify words or multi-words that refer to a concrete moment or to a temporal period. It has a normalized temporal value according to the TimeX3 standard.
- Named entities: Entities recognized in text. Annotated with entity type, for example *organization*, *location*, *person*, etc.

Edges

Edges represent three types of relations between the nodes:

- Syntactic: Indicates that there is a syntactic relation between two nodes. We maintain the Stanford dependencies tags [De Marneffe and Manning, 2008].
- Coreference: Indicates that two nodes are mentions of the same discourse referent.
- Temporal: Shows that there is a temporal relation between an event and a temporal expression. The relations can be of the following types: before, after, within, throughout, beginning and ending.

3.3.2 Enriched Representation

Graphs with initial representation G_D are transformed to create enriched representation graphs G_C.

One of the major changes is that we collapse the nodes in discourse referents. Collapsing provokes that nodes tagged with different attributes group.

These attributes can reinforce an evidence or supply with extra data. However, occasionally contradictory data is grouped. We make different decisions depending on the attribute.

Recall that the nodes prone to group are mainly named entities. However, as the result of inconsistencies in the processing tools, we consider that any node could group.

Discourse Referents

Each group of nodes related by coreferences $n_0...n_k \in N_D$ are grouped in a discourse referent $r = \cup n_0...n_k$, thus creating a new set of discourse referents R_C. Discourse referents are in fact the new nodes of the graph, but to avoid confusion, we simply call discourse referents to the nodes of the enriched representation.

Attributes Discourse referent's attributes are slightly different from the ones of the nodes. They keep the descriptor and the morphosyntactic tag, but lose the lemmas and the words.

With descriptors, the objective is to find the most representative string of the node. To do so, we take the longest string of every original node. In the future, it would be a better idea to create a knowledge base with every coocurrence of the descriptors, and from it to select the right one. It would be as a disambiguation system.

For the morphosyntactic tag, we assign to every named entity the value *N* and to every event *V*, while for the rest of the words we maintain the same of the original node. This is a simple approximation that takes advantage of the fact that named entities and events are the nodes that can have coreference relations. With this decision we seek to normalize the nodes that usually contain the most important information.

Special Nodes Discourse referents keep the types of the original nodes (events, temporal expressions and named entities). Collapsing may cause that one referent contains several annotations of the same type. We denote as *attribute collapsing* to the process of choosing the final attributes of the discourse referents.

- Events: When several events collapse in a single discourse referent, although infrequent, we keep all the annotations.
- Temporal Expressions: The same occurs with the temporal expressions.
- Named Entities: Annotations of named entity nodes often are contradictory. In these cases, we choose the most frequent types and discard the others. As with the descriptors, this process could be improved by storing every coocurrence in a database and using it as a feedback for the type assignment. *Persons* can be also tagged with gender and age.

Enriched Edges

Given other discourse referent, $r' = \cup n'_0 ... n'_k$, we tie both referents with an edge if any of their nodes were tied, that is: $\exists a(n_i, n'_k) \Longrightarrow \exists a'(r, r')$

We keep the same types of edges of the initial representation. On top of them we add edges with semantic information. As a result, we get three types of edges:

- Syntactic: Syntactic dependencies simplified through syntactic patterns. We add the tags *arg0*, *arg1* and *arg2*, which roughly correspond to subject, direct object and indirect object.
- Temporal: Same edges that are in the initial representation
- Semantic: Indicates that there is a semantic relation between two nodes. We distinguish between four subspecified semantic tags: *is*, *has*, *hasClass* and *hasProperty*. Semantic relations are obtained through a set of syntactic patterns.

3.4 Methodology

Our process is divided in the next steps:

1. Tokenization: The first step is dividing the sentences in tokens.
2. Morphosyntactic analysis: This step annotates each word with its part of speech and lemma.

3. Dependency parsing: Then are parsed to find the syntactic dependencies.

4. Named entity recognition: In this step named entities are recognized, and annotated with its named entity type.

5. Coreference resolution: This step annotates the relations of coreference of named entities and pronouns.

6. Extraction of temporal expressions: In this step temporal expressions are found and annotated with is value according to the standard TimeX3.

7. Event recognition: This step tags events with the tense, aspect and polarity.

8. Temporal linking: In this step temporal expressions and events are linked, and these relations are annotated with temporal information.

9. Collapsing nodes into discourse referents: Then we collapse nodes into discourse referents. To do so we group all nodes related by coreference.

10. Naïve semantic role labeling: This step simplifies a set of syntactic dependencies into three new dependencies, *arg0*, *arg1* and *arg2* that correspond to subject, direct object and indirect object.

11. Normalization of dependencies: This step normalizes some dependencies without semantic difference, such as passive voice and some genitives.

12. Semantic edges addition: We add some edges that denote a semantic relation between two nodes. This semantic relations can be *is*, *has*, *hasClass* and *hasProperty*.

13. Attribute collapsing: Finally we pick the representative named entity type and descriptor for the discourse referents.

Figure 3.1 shows the transformation from initial representation to enriched representation. In the initial representation (top graph) steps 1 to 8 have been done, while the enriched representation (bottom graph) is completed with steps 9 to 13.

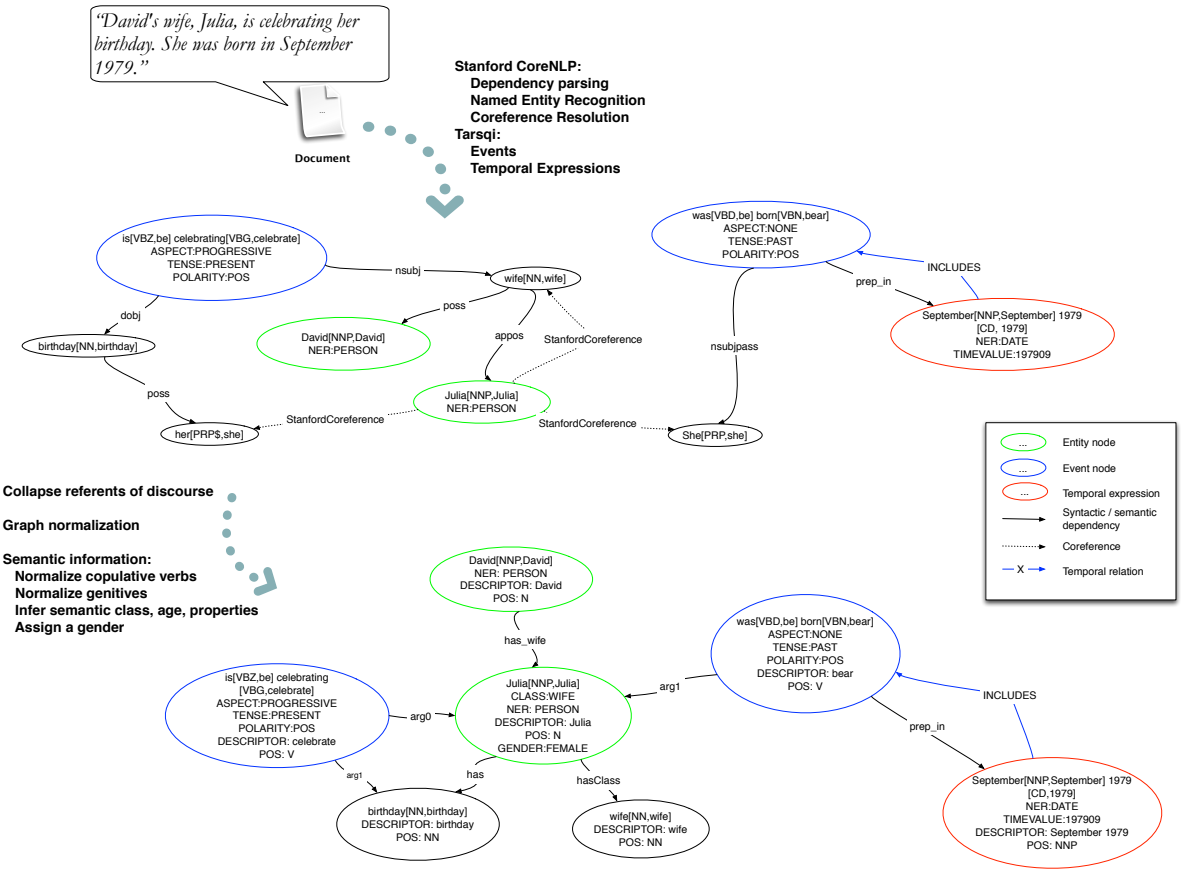

Figure 3.1. Transformation from initial representation, G_D, to enriched representation, G_C, for the example document: "David's wife, Julia, is celebrating her birthday. She was born in September 1979".

3.5 Implementation

In order to build the graphs, we process text with several tools of deep language processing. We rely on external exchangeable tools to make our representation as independent as possible.

We have developed a Java application that integrates these tools. We start applying the Stanford CoreNLP package [Klein and Manning, 2003] to perform steps 1 to 6. To avoid memory issues, we limit the length of the documents to 10000 characters. This process takes as input the file "document.txt" and outputs the file "document.stanford".

Then we apply the Tarsqi Toolkit [Verhagen et al., 2005] to perform steps 5 to 8. This tool has its own tokenizer, but to avoid inconsistencies, we create an input document with the tokenization given by the Stanford CoreNLP. This takes

as input the file “document.stanford” and outputs the files “document.tarsqi_in” and “document.tarsqi_out”.

Then we use JESS[1], a Java rule engine, to perform steps 9 to 12. Rules are divided in the next types:

- Semantic class extraction
- Genitive normalization
- Naive semantic role labeling
- Semantic edges addition
- Delete spurious relations

Rules are described in deep in the Appendix A.

Once the rules are applied, the final outputs are generated. This process takes as input the files “document.stanford” and “document.tarsqi_out” and outputs “document.dot”, “document.json” and “document.serialized”.

We have divided the number of documents of each collection in 10 subsets, which we launch independently. Each process uses 8 GB of memory, and contains its own instance of Stanford CoreNLP and Tarsqi Toolkit.

Table 3.1 shows the hardware where we run our process. The hard drives column shows the number of hard drives of each computer, along with the effective space after using the Raid5 Standard for data distribution.

Server	Operating System	Cores	Hard Drive	RAM
2 x 2,66Ghz Xeon X5650 95W CISCO C250	Centos 6	2 x 6	3x300GB(raid5)=600GB	16x8GB=128 GB
2 x 2.0Ghz Pentium Xeon 5130 DELL	RHEL5	2 x 2	3x73GB(raid5)=140GB	4x2GB=8GB
2 x 2.0Ghz Pentium Xeon 5130 DELL	RHEL5	2 x 2	3x73GB(raid5)=140GB	4x2GB=8GB
2 x 2.0Ghz Pentium Xeon 5130 DELL	RHEL5	2 x 2	3x73GB(raid5)=140GB	8x2GB=16GB
2 x 2.0Ghz Pentium Xeon 5130 DELL	RHEL5	2 x 2	3x73GB(raid5)=140GB	8x2GB=16GB
2 x 2.0Ghz Pentium Xeon 5130 DELL	RHEL5	2 x 2	3x146GB(raid5)=280GB	4x8GB=32GB
2 x 2.0Ghz Pentium Xeon 5130 DELL	RHEL5	2 x 2	3x146GB(raid5)=280GB	4x8GB=32GB

Table 3.1. Hardware used to represent the collection of documents as graphs.

[1] http:www.jessrules.com

3.5.1 Data

We have processed three different collections of documents:

- KBP 2011: 1.7 million documents
- Gigaword: 5 million documents
- Webtext: 1 million documents

3.5.2 Output

We produce three different outputs: Java serialized objects, JSON and DOT.

The first output is the Java serialized objects. These objects correspond to the classes DiscourseRefent, Relation and AbstractNode.

The second output is a JSON-like representation, where we annotate the relevant attributes for each node and relation. We have used this representation for the relation extraction task (see Chapter 5).

Finally we use DOT[2], which is a graph description language. It is useful because there are many tools to draw the graphs. For example, Graphviz[3], Canviz[4] or Gephi[5].

[2]http://www.graphviz.org/doc/info/lang.html
[3]http://www.graphviz.org/
[4]http://code.google.com/p/canviz/
[5]https://gephi.org/

CHAPTER 4
AUTOMATIC CAPTURE OF SEMANTIC CLASSES

In this chapter we show our method to extract knowledge from the graph representation. Specifically, we extract instance-class relations with an associated probability. This knowledge will be the base for the textual inference step.

4.1 A Brief Overview on Semantic Classes

Semantic classes are a very popular topic of research in natural language. Determining that *wife* is a class of *person*, whereas *Spanish* is a class of either *person*, *organization* or *location*, is a very valuable knowledge for many tasks, including information extraction, question answering and many others.

There are multiple ontologies or dictionaries with information about semantic classes that can be used as background knowledge, either created by hand as WordNet [Miller, 1995] or in semi-supervised ways, such as DBPedia [Mendes et al., 2012]. The biggest problem with these databases is that its coverage is insufficient, especially for open classes. Moreover it is hard to include new knowledge, and often the granularity provided does not fit the needs of the task.

To solve this problem, there were proposed multiple unsupervised methods of semantic class acquisition, which is known between other names as *semantic class learning*, *semantic class induction* or *hyponym acquisition* [Lin and Pantel, 2001b]. The most popular approach is to process the text with a syntactic parser and select one or more surface patterns to extract a set of semantic classes, and then refine the results [Hearst, 1992, Kozareva et al., 2008]. The results obtained tend to be noisy, so is common to apply a filter to clean the results.

In our case, we want to extract relations between classes and instances with an associated probability. Our hypothesis is that we can get it from large sources of text following the Open Information Extraction paradigm [Banko et al., 2007]. For example, we expect to see frequently the instance *Obama* associated with the semantic classes *president* or *democrat*. Likely we will see a long tail of classes associated with instances with less frequency. For example, *Obama* could be found as *husband*, *citizen*, etc. Unfortunately, some errors will arise, and we could retrieve examples like *Obama* related to the class *daughter*. We expect to overcome these mistakes because its frequency would be low.

As instances are stored with its entity type, we also have the relation between entity types and classes. For example, *persons* are more likely to have classes as *teacher*, *player* or *mother*, whereas organizations would have classes as *company*, *holding*, *library*, etc.

4.2 Methodology

Our process is divided in the next steps:

1. In the first place we process a large number of documents using the representation proposed in Chapter 3

2. Then we obtain a set of classes from the graphs using very simple patterns based on syntactic dependencies where a common noun is a class for a named entity. Named entities have a named entity type associated (*person*, *organization* or *location*) given by the named entity recognizer. When we find a match, we assign the common noun as semantic class of the named entity and we get an instance $NE- \ Class- \ Type$.

We do not pretend to obtain all the relations instance-class expressed in the collection with these patterns, but to acquire a representative and sufficient number of classes to evaluate their compatibility with named entities.

3. After obtaining the semantic class assignments, we aggregate the information from all the collection to obtain the frequencies of named entities, classes and types. As a matter of example, Table 4.1 contains the 5 most common classes associated to the entity type *person*.

4. Finally, we convert the frequencies in probabilities and get the joint probability between the classes and entity names, and also between classes and entity types. To get them we use a maximum likelihood estimator and marginalize:

$$p(c, en) = \sum_{t} p(en, c, t) \tag{4.1}$$

$$p(c, t) = \sum_{en} p(en, c, t) \tag{4.2}$$

Where c is the class, ne the entity name and t is the entity type.

Class	NE type	Frequency
spokesman	person	140229
president	person	102877
director	person	98182
leader	person	79716
coach	person	55511

Table 4.1. Most frequent classes associated to the named entity type *person*.

4.3 Implementation

First we use the graph representation shown in Chapter 2. Summing up, we start from the documents parsed with Stanford Parser [Klein and Manning, 2003]. We also use other features of The Stanford

CoreNLP package: POS tagger [Toutanova and Manning, 2000], named entity recognizer [Finkel et al., 2005] and the coreference resolution system [Lee et al., 2011]. Then we represent the documents as graphs, collapse the nodes in discourse referents, and enrich the representation with extra semantic information.

Once the graphs are generated, we use a Java application to apply a set of patterns (See Table 4.2). The output of this step is a set of tab separated values (TSV) files with instances *Document ID - Number of sentence - Named entity - Named entity type - Relation - Semantic class*

- *Document ID*: Identifier of the document where the instance was found.
- *Number of sentence*: Number of the sentence where the relation takes place.
- *Named entity*: Named entity descriptor.
- *Named entity type*: Named entity type, i.e. *person*, *organization* or *location*.
- *Relation*: Indicates what the type of the relation is. In this case, *hasClass*.
- *Semantic class*: Semantic class gathered through the patterns.

Syntactic pattern		
NE	nn	NN
NE	appos	NN
NE	abbrev	NN
NN	appos	NE
NN	abbrev	NE
NE	such_as	NN
NE	like	NN
NE	be	NN

Table 4.2. Patterns for the assignment of semantic classes. Each entry belongs to a tuple $governor - \ dependency - \ dependant$ where NN is a common name and NE is a named entity.

Then we use a bash script to get the frequency of each instance deleting the fields *Document Id* and *Number of sentence*, getting a set of documents with instances *Named entity - Named entity type - Relation - Semantic class - Frequency*

Table 4.3 shows the hardware where we run our process. The hard drives column shows the number of hard drives of each computer, along with the effective space after using the Raid5 Standard for data distribution.

Server	Operating System	Cores	Hard Drive	RAM
2 x E5-2650 UCS CISCO C240 M3	Centos 6.3	2 x 8	10x900GB(raid6)=6,5 TB	12x8GB=96GB
2 x 2,66Ghz Xeon X5650 95W CISCO C250	Centos 6	2 x 6	3x300GB(raid5)=600GB	16x8GB=128 GB

Table 4.3. Hardware used to generate the knowledge base and retrieve the probabilities associated to each instance.

CHAPTER 5
THE EFFECT OF GRAPHICAL REPRESENTATION ON RELATION EXTRACTION

This chapter describes how we took advantage of the graphical representation for the Temporal Slot Filling task in the TAC Knowledge Base Population evaluation.

5.1 The Temporal Slot Filling Task

The main objective of the transformation of the documents in graphs is to improve the performance of the systems that actually use the representation. Accordingly, we perform an extrinsic evaluation. To do so, we have chosen the 2011 Slot Filling task of the Knowledge Base Population track [Ji et al., 2011].

The Slot Filling task is composed by two subtasks: Regular Slot Filling (RSF) and Temporal Slot Filling (TSF). The goal of RSF is, given a set of entities and a set of slots, extract from a corpus of millions of documents the correct values for each combination of entity and slot. This is, for a set of queries composed by the pair *(entity, attribute)* the systems have to answer with a tuple *(entity, attribute, value)*. For instance, for the query *(Barack Obama, spouse)* the answer should

be *(Barack Obama, spouse, Michelle Obama)*.

The correct answer to a slot may consist in a list of values. Only the right values are scored, and the redundant answers are ignored.

Entities are divided in two categories, *person* and *organization*. Each category has its own set of slots (See Table 5.1).

The organizers distribute a knowledge database built from the Wikipedia

Person	Organization
per:alternate_names	org:alternate_names
per:date_of_birth	org:political/religious_affiliation
per:age	org:top_members/employees
per:country_of_birth	org:number_of_employees/members
per:stateorprovince_of_birth	org:members
per:city_of_birth	org:member_of
per:origin	org:subsidiaries
per:date_of_death	org:parents
per:country_of_death	org:founded_by
per:stateorprovince_of_death	org:founded
per:city_of_death	org:dissolved
per:cause_of_death	org:country_of_headquarters
per:stateorprovinces_of_residence	org:city_of_headquarters
per:cities_of_residence	org:shareholders
per:schools_attended	org:website
per:title	
per:member_of	
per:employee_of	
per:religion	
per:spouse	
per:children	
per:parents	
per:siblings	
per:other_familiy	
per:charges	

Table 5.1. Slot names for the two entity types in the RSF task

Infoboxes. It contains noisy information about entities, values and relations, useful to train the systems. Moreover a 1.7 million documents collection is provided as solutions search space. It contains documents from different sources, such as newswire, web and others.

The TSF adds the temporal component. In this subtask, the tuples *(entity, relation, value)* have to be anchored in temporal interval. A fuzzy temporal interval is defined as a tuple of four values (t_1, t_2, t_3, t_4), that denote that the slot started in a point between t_1 and t_2, and it ended between t_3 and t_4. If the value of t_1 or t_3 is not defined it means that the value is $-\infty$, while for t_2 or t_4 is $+\infty$.

Furthermore, the number of target slots is reduced for this subtask. Table 5.2 shows the TSF slots.

Person	Organization
per:spouse	org:top_employees/members
per:title	
per:employee_of	
per:member_of	
per:cities_of_residence	
per:stateorprovinces_of_residence	
per:countries_of_residence	

Table 5.2. Slot names for the two entity types in the TSF task

The organization provides with training data annotated with temporal anchors, and also intermediate local information regarding temporal constraints. This information consists on a temporal expression located on a document and the temporal relation (See Table 5.3) between the slot value and the temporal expression.

5.2 Research Questions

The research questions tackled in this chapter are:

1. What features of the classifier are different because of the graph representation?

2. What features are provided exclusively by the representation?

3. Which is the performance of a classifier trained with those features in a task of automatic relation extraction?

4. In the same task, once generated the graphs, is it an improvement to add semantic information?

5.3 Experimental design

In order to answer these questions we have participated in the RSF and TSF tasks [Garrido et al., 2011]. We have used a single multi-class classifier and a battery of binary classifiers for each task respectively. Regardless, in each task we follow the next steps:

1. First we retrieve the relevant documents given for the given queries, up to 100 documents per query.

2. Then, we transform the documents into the initial and enriched representation

Relation	Role of temporal expression	**Example**
Beginning	the start time for the slot value	Rob joined GE in 1999
Ending	the end time for the slot value	Rob left GE in 1999
Beg_and_end	the slot value is true exactly for the specified time	Rob was named linguist of the month for June 1999
Within	the slot value is true for at least some portion of the specified time	Rob worked for GE in 1999
Throughout	the slot value is true for all of the specified time	Rob commuted to work from his home in Denver for all of the 1999
Before_start	a moment before the start time for the slot value	In 1999, before Rob joined GE, . . .
After_end	a moment after the end time for the slot value	By 1999 Rob had already left GE

Table 5.3. Temporal relations in the TSF training data

3. Within the graphs, we search for the node or nodes that match the query entity.

4. Finally, each of the nodes at 10 or less length from the node is a candidate to be the slot value. With this information we produce an unlabeled example *(entity node, value node, document)*, which is processed to the classifier to label it as either positive for any of the slot type or with the negative label.

Furthermore, the classifier takes advantage of the representation to extract the features to characterize each example. Table 5.4 shows the features used in RSF. X stands both for *entity* and *value*.

TSF uses three extra features detailed in Table 5.5. These are Verb features, which are generated from the verbs, V, identified in the path between *entity* and *value*. Features provided by our representation are underlined.

Feature family	Feature name	Description
Syntactic dependency	path	path between *entity* and *value* in the sentence [represented with the unigrams and bigrams of dependency labels, POS tags and NE tags]
Placeholders	X-annotation	NE annotations for the sentence fragment X
	X-pos	Part-of-speech annotations for the sentence fragment X
Lexical context	X-gov	Governor of X in the dependency path
	X-mod	Modifiers of X in the dependency path
Properties	X-has_age	X is a NE, and we have identified it has an age attribute.
	X-has_class-C	X is a NE, and we have identified it has a class C.
	X-has_property-P	X is a NE, and it has a property P
	X-has-Y	X is a NE, and it is in a relationship to-have with another NE, Y
	X-is-Y	X is a NE, and it is in a relationship to-be with another NE, Y
	X-gender-G	X is a NE, and it has gender G

Table 5.4. Features included in the model for RSF.

Feature family	Feature name	Description
Properties of the verb V	V-tense	Tense of the verb V in the path.
	V-aspect	Aspect of the verb V in the path.
	V-polarity	Polarity (positive or negative) of the verb V in the path.

Table 5.5. Features included in the model for TSF.

We submitted two different runs to evaluate the performance of the enrichment and collapsing. Both follow the process described, but the first one uses the initial representation, whereas the second uses the enriched representation.

5.4 Results

Table 5.6 shows the results of the Regular Slot Filling Task, compared with the manual annotation (LDC) and the top teams in the competition. Our system clearly underperforms compared with the other systems. These results were expected as many teams have mature systems whereas our system is in an early stage of development. However, it is notable how results improve using the enriched representation.

	LDC	Top-1 Team	Top-2 Team	Median Team	Initial	Enriched
Precision	0.8618	0.3502	0.4917	0.1031	**0.0259**	**0.0343**
Recall	0.7259	0.2550	0.1259	0.1650	**0.0455**	**0.0529**
F1	0.7880	0.2951	0.2005	0.1269	**0.0330**	**0.0416**

Table 5.6. Regular Slot Filling task results

Table 5.7 shows the general results of the Temporal Slot Filling Task. It can be observed that the performance of the system is similar to other systems on the state of the art, and achieving the highest precision among all the participants. Moreover, despite the low recall, the system gets the third best F_1 measure.

This implies that the graphical representation is a promising chance in the task of automatic relation extraction, because, despite being in the first steps of development, it allows to compete with the systems on the state of the art.

System	Precision	Recall	F1
BLENDER2	0.1789	0.3030	0.2250
BLENDER1	0.1796	0.2942	0.2231
BLENDER3	0.1744	0.2976	0.2199
IIRG1	0.2457	0.1194	0.1607
Initial	**0.2996**	**0.0703**	**0.1139**
Enriched	**0.2596**	**0.0609**	**0.0986**
Stanford 12	0.0233	0.1680	0.0409
Stanford 11	0.0238	0.1453	0.0408
USFD20112	0.0152	0.0070	0.0096
USFD20113	0.0079	0.0014	0.0024

Table 5.7. Final results on the Temporal Slot Filling task

The system is made by several components pipelined: Information retrieval, document representation, semisupervised learning, relation extraction and in the case of TSF, temporal anchoring. This kind of system is very sensitive to error propagation, and therefore is interesting to study precision and recall taking into account the maximum bound imposed by the information retrieval component.

Table 5.2 shows the results in the Temporal Slot Filling task until the relation extraction phase. It contains the precision and recall taking into account the number of tuples *(entity, attribute, value)* successfully obtained.

Representation	Initial	Enriched
Recall	0.08	0.08
Precision	0.42	0.45
F1	0.14	0.14

Table 5.8. Temporal Slot Filling task results until the relation extraction phase

Data shows that both precision and recall obtain very similar results in both types of representations, being slightly better in the enriched graphs case.

A manual inspection of the results shows that the generated graphs contain many mistakes. After the participation in the task we have identified many programming mistakes that possibly worsen the results, especially in the enriched representation. However, the enriched graphs represent a slight improvement against the initial graphs.

5.5 Conclusions

In this phase, we have evaluated the document level representation as semantically enriched morphosyntactic graphs in the temporal slot filling task. This representation allows performing multiple tasks such as information aggregation, semantic class assignment and temporal information representation in documents.

1. What features are provided exclusively by the representation?

   For RSF, we have used a set of unique features of this representation: *X-has_age*, *X-has_class-C*, *X-has_property-P*, *X-has-Y*, *X-is-Y* and *X-gender-G*. In TSF we also added *V-tense*, *V-aspect* and *V-polarity*.

2. What features of the classifier are different because of the graph representation?

   Path features are modified because of the graph representation. The naive role labeling and the normalization aim to simplify the paths extracted, thus the learning should be easier.

3. Which is the performance of a classifier trained with those features in a task of automatic relation extraction?

   Results show that this representation is promising for feature extraction for classifiers in the automatic relation extraction task. The early stage of development of the representation leads us to think that there are many alternatives to explore and that future versions will achieve better results.

4. In the same task, once generated the graphs, is it an improvement to add semantic information?

We have got slightly better results using the enriched graphs. With these data we can say that the enrichment phase is useful for the automatic relation extraction task, as the balance between the gain of collapsing the graphs and the loss of the added errors introduced by the process is positive.

Moreover, the representation system is based on interchangeable external tools. As a result, we have a flexible and modular application that fits for future changes.

CHAPTER 6
THE ROLE OF SEMANTIC CLASSES IN PARSING APPOSITIONS.

In this chapter we define our method to exploit the semantic class knowledge to improve the parsing on appositive structures. To do so, we select the grammatical option that is semantically more compatible. We define a set of measures of compatibility and evaluate its performance.

6.1 About the Relation between Appositions and Semantic Classes

There appears to be an interesting paradox when we want to build semantic representations from syntactic structures: In many cases, these syntactic structures require semantics to solve structural ambiguities.

This is particularly true for dependency analysis, where making explicit the dependencies between words allows expressing some semantics. However, the correct syntactic dependence between two words might require in turn considering possible semantic relationships.

An example of this is the appositive structure[1]. The structure where one of

[1]We follow the definition of apposition as the grammatical construction with two contiguous

the noun phrases is governed by a named entity and the other is governed by a common noun is used very frequently to denote the semantic class of the entity. Consider the following example:

David supports the team of his wife, ***Julia***.[2]

Dependency parsers often determine that there is an apposition between the common noun *wife* and the named entity *Julia*, leading to interpret that *Julia* belongs to the class wife. People would accept this interpretation using their background knowledge where *Julia* is a common name for a female person. Now consider the following example with exactly the same syntactic structure:

David supports the team of his wife, ***The Vikings***.

Without semantic information it is not possible to determine the scope of the apposition. However we know that *The Vikings* is not a usual name for a *wife*, whereas in the close context there is a different candidate, the common noun *team*, which is semantically more compatible. Thus, we want parsers to reproduce this behavior and link *The Vikings* and *team*.

6.2 Parsing Appositives

Our goal is to study appositions where dependency parsers have a chance to make a mistake selecting a governor because there are many grammatically correct candidates.

In particular, we focus on cases where errors come from not taking into account the semantic compatibility between the two parts involved.

One structure where these errors can be found is a phrase where one side of the apposition is a named entity, and the other one has two or more common nouns that can be the nucleus of the noun phrase, and therefore a semantic class of the named entity. We refer to those common nouns as *candidates*.

We search for this structure in a large collection of textual data belonging to the TAC KBP task [Ji et al., 2011], composed by around 1.5 million documents that belong to different categories including newswire, blogs and phone calls transcriptions. We parse the collection with The Stanford Parser [Klein and Manning, 2003].

---

noun phrases where one defines or modifies the other

[2]We denote in bold the named entity and underline the candidate common nouns.

In this collection we obtain a total of 41,285,844 sentences, with 691,394 apposition dependencies, where 240,392 (34.7%) have more than one candidate.

We took a sample of 300 sentences to study the behavior of the parser in those cases (See Section 6.4.

6.3 Methodology

We have followed the next steps:

1. First, we have parsed a large text corpus with our graphical representation (See Chapter 3). Among other tools, this representation uses a dependency parser that can be used to locate the apposition dependencies.

2. Then, we get a sample of the appositions where we compare the behavior of two different dependency parsers in the state of the art.

3. With this sample, we also study the source of errors in parsing, and provide with the cases of the ambiguous appositive relations according to the semantic compatibility of the candidates to governor and the dependent part.

4. Then, we face the problem of design and implement a method to correct appositions. In particular, we focus on cases where there are many candidates to govern the apposition. Our hypothesis is that we can overcome these errors by considering some background knowledge automatically extracted from large text collections.

5. We use as background knowledge our proposition stores (See Chapter 4). These propositions contain information about instances relating named entities with classes with an associated probability. Named entities have also its named entity type.

6. We assume that the most semantically compatible candidate is the correct one. To measure this compatibility we study different configurations to combine the available evidences with three different measures.

6.4 Cases

We classify the sentences according to the semantic compatibility of the candidates and the named entity.

6.4.1 One Valid Candidate

In the simplest case, such as the examples in the introduction, the common noun that acts as nucleus of the first noun phrase should be the governor of the apposition and it is more suitable than the other.

. . . the leader of its largest rebel group, ***Manuel Marulanda****, . . .*

We target these cases as the ones to solve. Ideally, our background knowledge should be able to discriminate which candidates are suitable and those which are not. For example, it should be clear that a person as *Manuel Marulanda* is more frequently a *leader* than a *group*.

6.4.2 Several Valid Candidates

However, in some occasions there are multiple common nouns that are valid candidates to be the governor of the relation. One case is where there are noun compounds whose common nouns act as a modifier, like in the next examples:

. . . used by its domestic subsidiary airline, ***Eagle Air****, . . .*

. . . by the IOC 's chief Beijing organizer, ***Hein Verbruggen****, . . .*

In other cases the noun phrase includes a subordinated sentence, which also has a common noun candidate to govern the apposition, for example:

Another passenger who gave only his surname, ***Chen*** *. . .*

Or even when the first noun phrase contains one or more conjunctions, so that there are more than one valid common nouns to govern the apposition relation. For example:

. . . a prominent Jewish writer and Holocaust survivor, ***Ralph Giordano*** *. . .*

What we assume in these cases is that there are different simultaneous classes of the named entity. In some cases it could be interesting to add an apposition for each of the candidates.

Note that in some cases, sentences with the same grammatical structure do not lead to multiple appositions. For example:

. . . which Singapore founding father Lee Kuan Yew and his son, Prime Minister ***Lee Hsien Loong*** *. . .*

A special case of the sentences with two or more suitable candidates are the ones with a preferred candidate. In these sentences the nucleus, even if it is suitable, it is less discriminative as semantic class than the other common noun, for example:

Henry is grounded by his illustrator partner, ***Rudy*** *. . .*

6.4.3 Undecidable Candidates

In some sentences there are two classes referred to two different entities, but some extra-linguistic knowledge is needed to decide how they are related. For example:

. . . at least one brother of another defendant, ***Ali Dayeh Ali****.*

In the previous example there are two classes, *brother* and *defendant*, that refer to two different entities, *Ali Dayeh Ali* and an unknown entity. Without external knowledge it cannot be decided if *Ali Dayeh Ali* belongs to the class *brother* or to the class *defendant*.

6.4.4 No-apposition Case

In the manual inspection we have found multiple examples where the parser detects an incorrect apposition relation. To correct these errors is a line of work by itself, but here we limit ourselves to show what cases we have found.

- Conjunction between two sentences. For example:

  Guo Wenjun won the women's 10-meter air pistol, ***Guo Jingjing*** *and Wu Minxia the women's synchronized 3-meter springboard, . . .*

- Wrong apposition relation between a noun phrase and a verbal phrase. In many cases, the right analysis would be to consider the noun phrase as object of the verbal phrase. For example:

  . . . will serve as the incoming president's chief of staff, President-elect ***Ma Ying-jeou****'s office announced.*

- Structures that denote the relation location-region. For example:

  . . . ESA mission control in Toulouse, southwestern ***France****, . . .*

- Enumerations. For example:

  *A spa tucked away in the basement includes a large pool, a **whirlpool**, workout room, saunas, a solarium . . .*

- Nonsense text. Example:

  *. . . Anti-Muslim Bigots, V- for-Vendicar, fruitella, Zionism equal Racism, **The Chemical Oil Nazi**, LORD RAMA RANTER, Muslim With Mission, . . .*

6.4.5 Summing up

In this section we have shown some cases where parsers depend on semantic knowledge to solve apposition dependencies with several grammatical alternatives. We have classified the sentences according to the semantic compatibility of the candidates and the named entity. Table 6.1 shows the number and percentage of appositions in each case.

In general, there are two main sources of errors in apposition parsing: When there is actually no apposition and the parser makes a bad choice of the dependency type, and when there are several candidates to govern the apposition and the parser makes a bad identification of the governor of the apposition. We tackle the latter, which includes cases with one valid candidate, several valid candidates and undecidable candidates. These cases represent the 78.6% of the appositions of the sample. We compose a test set from those appositions, where there is a margin of improvement of 19.1% (see Sections 6.5.3 and 6.5.4).

One valid candidate	212 (70.6%)
Several valid candidates	22 (7.3%)
Undecidable candidates	2 (0.6%)
Total	236 (78.6%)
No apposition	64 (21.3%)

Table 6.1. Classes of appositions.

6.5 Experiment Design

In this section, we explain our approach to identify and select the candidates to govern an apposition relation.

We assume that the correct candidate is the most semantically compatible with the named entity.

In the phase of identification and correction of appositions we need the joint probability between the classes and entity names, and also between classes and entity types (see Section 6.2).

6.5.1 Method

We gather dependencies formed by two noun phrases that fulfill two premises. The first one is that the first noun phrase must have more than one common noun. Each one is a candidate to be the governor of the apposition dependence. We denote the set of candidates as $c_0, \ldots, c_n$.

The second premise is that the nucleus of the second noun phrase has to be a named entity. The named entity is the dependent part of the apposition. It is defined by two aspects, the entity name ne, which is the proper string that forms the named entity, and its entity type t (*person*, *organization* or *location*). We will refer to these values as evidence (e).

Once we have candidates to govern an apposition relation, we decide which one is more suitable by comparing its semantic compatibility with the named entity.

6.5.2 Measures

We use three measures of compatibility: normalized pointwise mutual information between candidate and evidence $npwmi(c_i, e)$; conditioned probability of the class given the evidence $p(c_i|e)$; and smoothed conditioned probability $p_{JM}(c_i|e)$.

The formula that describes the normalized pointwise mutual information we use is:

$$npmi(x;y) = \frac{pmi(x;y)}{-\log p(x,y)} \tag{6.1}$$

Where:

$$pmi(x;y) = log\frac{p(x,y)}{p(x)p(y)} \tag{6.2}$$

$p(x, y)$ is the probability estimated in the previous phase, and $p(x)$ and $p(y)$ is the result of marginalization. Using the normalized pointwise information puts the results in a range of $(-1, 1)$, where -1 is the value where there are not joint observations and 1 is the value where the observations are always together.

To calculate conditional probabilities we apply:

$$p(x|y) = \frac{p(x, y)}{p(y)} \tag{6.3}$$

In the third measure probabilities are smoothed through the Jelinek-Mercer smoothing. We use as default $\alpha = 0.99$.

$$p_{JM}(e|c) = \alpha p(e|c) + (1 - \alpha)p(e) \tag{6.4}$$

6.5.3 Test Set

We have built a Gold Standard from the sample where we annotate the right apposition dependence. When there are several valid candidates, or it is undecidable which candidate is valid, we will consider all valid choices as correct. For our experimentation, we will ignore the non-apposition cases. The result is a Gold Standard of 236 appositions (see Section 6.4.5).

6.5.4 Baseline

We will use as baselines the results of considering the governor of the dependence given by two parsers, Stanford Parser [Klein and Manning, 2003] and Fanse Parser [Tratz and Hovy, 2011].

Table 6.2 shows the performance of each parser over the 236 appositions in the Gold Standard. Stanford Parser has better results, but still there is a margin of improvement of 19.1%. We estimate that in the whole collection there are around 30.000 mistakes that can be solved with our method.

	correct	incorrect
Stanford Parser	191 (80.9%)	45 (19.1%)
Fanse Parser	167 (70.7%)	69 (29.3%)

Table 6.2. Baseline: Selection of candidates by the parsers.

6.5.5 Configurations

Once we have the set of candidates $c_0, \ldots, c_n$, and the evidences $e = ne$, $e = t$ and $e = ne, t$ we will select the candidate c_f that maximizes the compatibility score:

$$c_f = \arg\max_c(f) \tag{6.5}$$

Where f corresponds to the following configurations:

- Entity name as evidence
  - — Configuration 1: $npwmi(ne; c)$
  - — Configuration 2: $p(c|ne)$
  - — Configuration 3: $p_{JM}(c|ne)$
- Named entity type as evidence
  - — Configuration 4: $npwmi(t; c)$
  - — Configuration 5: $p(c|t)$
  - — Configuration 6: $p_{JM}(c|t)$
- Both, entity name and entity type as evidence. We combine the evidences through a conditioned distribution. We assume conditional independence between ne and t given c.
  - — Configuration 7: $p(c|ne, t) \propto p(ne|c) * p(t|c) * p(c)$
  - — Configuration 8: $p_{JM}(c|ne, t) \propto p_{JM}(ne|c) * p_{JM}(t|c) * p(c)$

When $npwmi(e; c) = -1$ for each candidate $c_0, \ldots, c_n$ we say that we have no evidence. We deal in the same way with $p(c|e) = 0$ and $p_{JM}(c|e) = 0$. In these cases, we respect the choice made by the parser in the first place.

6.6 Results

Tables 6.3, 6.4 and 6.5 show the results obtained according to the evidence considered: ne, t or both. For each evidence we show the performance with the measures considered.

We split the results of each configuration according to the behavior of our method with respect to the baseline. The row *Change* shows the number of appositions that have been changed with respect to the baseline. The row *No change*, shows the appositions that remain as in the baseline, and may be for two reasons: either our evidence points at the same noun than the parser or we have no evidence and we leave the original candidate (see Section 6.5.5).

Last row shows the difference between our method and the two baselines proposed along with the relative improvement.

		$npwmi(ne;c)$			$p(c\|ne)$			$p_{JM}(c\|ne)$		
		Correct	Incorrect	Total	Correct	Incorrect	Total	Correct	Incorrect	Total
Change		30 (12.7%)	13 (5.5%)	43 (18.1%)	32 (13.5%)	5 (2.1%)	37 (15.6%)	47(19.8)	21 (8.8%)	69 (29.2%)
No change	Coincidence	119 (50.4%)	10 (4.2%)	129 (54.6%)	129 (54.6%)	6 (2.5%)	135 (57.2%)	161 (68.2%)	7 (2.9%)	168 (71.1%)
	No evidence	49 (20.7%)	15 (6.3%)	64 (27.1%)	49 (20.7%)	15 (6.3%)	64 (27.1%)	0 (0%)	0 (0%)	0 (0%)
Total		198 (83.8%)	38 (16.2%)	236 (100%)	210 (88.9%)	26 (11.1%)	236 (100%)	208 (88.1%)	28 (11.9%)	236 (100%)
Improvement	Stanford Parser	7 (3.6%)			19 (9.9%)			17 (8.9%)		
	Fanse Parser	21 (22.6%)			43 (25.7%)			41 (24.5%)		

Table 6.3. Results considering the entity name as evidence: Configurations 1, 2 and 3.

		$npwmi(t;c)$			$p(c\|t)$			$p_{JM}(c\|t)$		
		Correct	Incorrect	Total	Correct	Incorrect	Total	Correct	Incorrect	Total
Change		54 (22.8%)	24 (10.1%)	78 (32.1%)	50 (21.1%)	20 (8.4%)	70 (29.6%)	50 (21.1%)	20 (8.4%)	70 (29.6%)
No change	Coincidence	155 (65.6%)	3 (1.2%)	158 (66.9%)	163 (69%)	3 (1.2%)	168 (71.1%)	163 (69%)	3 (1.2%)	168 (71.1%)
	No evidence	0 (0%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)
Total		209 (88.5%)	27 (11.5%)	236 (100%)	213 (90.2%)	23 (9.8%)	236 (100%)	213 (90.2%)	23 (9.8%)	236 (100%)
Improvement	Stanford Parser	18 (9.4%)			22 (11.5%)			22 (11.5%)		
	Fanse Parser	42 (24.1%)			46 (27.5%)			46 (27.5%)		

Table 6.4. Results considering the entity type as evidence: Configurations 4, 5 and 6.

6.6.1 Evidence

We can see across the three different tables how, when considering the same measure, combining both evidences performs better than the use of entity types alone, which in turn performs better that considering only the entity name. This confirms the intuition that entity types are useful to generalize entity names, but combining both data is even better to measure semantic compatibility.

6.6.2 Measures

Relating to the measures, when comparing the results within each table, it is noticeable how conditional probability outperforms normalized pointwise mutual

		$p(c\|ne,t)$			$p_{JM}(c\|ne,t)$		
		Correct	Incorrect	Total	Correct	Incorrect	Total
Change		33 (13.9%)	3 (1.2%)	36 (15.2%)	46 (19.4%)	9 (3.8%)	55 (23.3%)
No change	Coincidence	131 (55.5%)	5 (2.1%)	136 (57.6%)	170 (72%)	6 (2.5%)	176 (74.5%)
	No evidence	49 (20.7%)	15 (6.3%)	64 (27.1%)	4 (1.6%)	1 (0.4%)	5 (2.1%)
Total		213 (90.2%)	23 (9.8%)	236 (100%)	220 (93.2%)	16 (6.8%)	236 (100%)
Improvement	Stanford Parser	22 (11.5%)			29 (15.1%)		
	Fanse Parser	46 (27.5%)			53 (31.7%)		

Table 6.5. Results considering both entity name and entity type as evidence: Configurations 7 and 8.

information regardless the evidence considered.

Smoothed conditioned probabilities are tricky. Intuitively, smoothing should be useful in cases where there is no evidence, because it will favor the most probable candidates.

However, when considering the entity name as evidence, smoothing worsens the results. This is because in the sample there is 64 instances with no evidence, but 49 (76.5%) of them are correctly classified by the parser. Smoothing these instances divides them between correct and incorrect, reducing the number of correct appositions.

When considering the entity type as evidence, smoothing is not useful because in the sample considered there are no instances without evidence. Thus, the results remain equal.

Finally, considering both entity name and entity type, smoothing becomes useful. It allows considering 59 instances that previously had no evidence and classify them better than the baseline.

6.6.3 Best Configuration

The best performance is reached using both sources of evidence in a smoothed conditional probability (Configuration 8 $p_{JM}(c|ne,t)$). With this configuration results rise from 80.9% accuracy in the baseline to 93.2%. This represents an improvement of 15% with respect to the parser.

6.6.4 Examples

Table 6.6 shows a case where every evidence and measure considered chooses the correct candidate. In this case, *Yuval Diskin* is the named en-

tity and its entity type is *person*. Candidates are *agency*, *chief* and *security*. Both Stanford parser and Fanse parser have chosen *agency* in the first place, but the correct candidate is *chief*. In bold we denote the candidate chosen by the method for each configuration.

... the chief of the Shin Bet security agency, Yuval Diskin, ...

Configuration	$npwmi(ne;c)$	$p(c\|ne)$	$p_{JM}(c\|ne)$	$npwmi(t;c)$	$p(c\|t)$	$p_{JM}(c\|t)$	$p(c\|ne,t)$	$p_{JM}(c\|ne,t)$
agency	0.13	0.02	0.02	-0.22	4.2E-4	4.5E-4	5.5E-8	5.8E-8
chief	**0.28**	**0.33**	**0.32**	**0.09**	**0.01**	**0.01**	**8.9E-6**	**8.8E-6**
security	-1	0	1.4E-6	-0.25	1.1E-4	1.1E-4	0	2.0E-11

Table 6.6. Example of apposition corrected.

Table 6.7 shows one case where our method worsens the baselines. In this case, *Virginia Casey* is the named entity and its entity type is *person*. Candidates are *daughter* and *cousin*. Both Stanford parser and Fanse parser have chosen *cousin* in the first place, and it is the correct candidate. In bold we denote the candidate chosen by the method for each configuration.

The reason of the failure is that we have gathered more evidence of persons being *daughter* than *cousin*, and we have no evidence of the name *Virginia Casey* with these classes.

... daughter Kim Boyer and Boyer's cousin, Virginia Casey ...

Configuration	$npwmi(ne;c)$	$p(c\|ne)$	$p_{JM}(c\|ne)$	$npwmi(t;c)$	$p(c\|t)$	$p_{JM}(c\|t)$	$p(c\|ne,t)$	$p_{JM}(c\|ne,t)$
daughter	-1	0	**2.7E-5**	**0.06**	**0.003**	**0.003**	0	**5.7E-12**
cousin	-1	0	4.9E-6	0.05	7.2E-4	7.2E-4	0	1.0E-12

Table 6.7. Example of apposition incorrectly changed.

CHAPTER 7
CONCLUSIONS AND FUTURE WORK

In this chapter we state the main conclusions obtained and we discuss some interesting lines of research.

7.1 Summary

In this work we have tackled the problem of knowledge acquisition and textual inference. To do so, we have represented a large collection of documents as graphs, we have extracted knowledge structured as propositions and lastly we have used textual inference to perform a task of dependency parsing, specifically apposition dependence correction.

We have started from a set of research questions:

- What problems arise from developing an automatically acquired propositional knowledge base?

  Propositional knowledge bases rely on several steps, each one of them introduces some limitations in the acquisition. The conceptual representation of text greatly influences the extracted propositions. In our case we have chosen to use syntactic patterns to extract semantic class knowledge. These syntactic patterns, although being very productive, impose

a limit on the relations that can be extracted. Moreover, they have to be defined by hand.

The last problem that we have faced is how to model a named entity through its semantic classes. In our case we use frequencies estimated through a maximum likelihood estimator.

- Is it feasible to build the knowledge base from documents represented as graphs? What does this kind of representations provide?

  The graph-based representation directly affects to which propositions can be extracted. A long-distance dependence successfully captured means an interesting proposition acquired.

  Besides, other features included such as temporal and semantic relations allow extracting newer and unique propositions. Unfortunately, also each mistake of the representation is transferred to the knowledge base.

- How can we perform textual inference using the propositions?

  The textual inference depends on the target task. In our case, we have used the propositions to characterize appositive structures in the context of dependency parsing. Our textual inference in this case is to measure the semantic compatibility between an entity and a class through the knowledge gathered in the proposition store.

Regarding the graph-based representation:

- What are the main problems in the design of the representation?

  We have faced several problems in the design of the representation. First, different mentions of the same entity should be bounded. We have taken advantage of the coreference relation to group them in discourse referents.

  Second, we had to deal with the sparsity that would produce a graph with many different grammatical relations between its components. To do so, we have simplified the morphosyntactic relations through a naive semantic role labeling, and normalized different semantically equal expressions.

- Which are the benefits and the drawbacks?

The graph representation is designed to capture long-range context, even those that happen across different sentences. Moreover, the simplification of the relations allows extracting cleaner features, in the sense that, when the occurrences are aggregated, sparsity is reduced.

This representation comes with a high computational cost, as it uses several deep language processing tools. Moreover, these tools introduce errors in the process, which are transferred into the subsequent systems.

- In the context of automatic relation extraction, is it useful?

  To answer this question, we have participated in the Regular Slot Filling and Temporal Slot Filling tasks. Even if the results are discrete, there is evidence that enriched graphs improve the results on these tasks. We expect that a refinement of the representation allows us to perform precise relation extraction.

In the context of the correction of appositive dependencies:

- How do parsers behave when they have to process appositions and what kind of errors do they commit?

  In general, there are two main sources of errors: (1) When actually there is no apposition and the parser makes a bad choice of the dependency type, and (2) when there are several candidates to govern the apposition and the parser makes a bad identification of the governor of the apposition.

  We have done a qualitative analysis of appositions types according to the semantic compatibility between a governor and its grammatically correct candidates. The 21.3% are no-appositive cases and correspond to the first source of errors. The remaining 78.6% of appositions belong to the second source, and are divided in three cases: appositions with one valid candidate, with several candidates or with undecidable candidates. We have focus on solving these three cases.

- Is it possible to overcome these errors considering semantic information captured previously from text collections? What evidence can they provide to characterize the named entity?

  We have automatically acquired semantic classes from large text collections by using very simple syntactic patterns. Then, we have used them

as background knowledge to measure the semantic compatibility of candidates and named entities.

This knowledge was divided in two different evidences, one that relates semantic classes with named entities and other that relates semantic classes with named entity types.

The results obtained reinforce our hypothesis that considering semantic compatibility between the two parts of the apposition can help to overcome parsing errors.

- What is the most effective way to measure the semantic compatibility that allows a better identification of the dependencies?

  We have used two different evidences (entity name and named entity type) and three different measures (normalized pointwise mutual information, conditional probabilities and smoothed conditional probabilities).

  Using entity name as evidence does not improve the results, as the sparseness of the instances class-entity makes that the estimated probability of coocurrence is not robust. This issue could be improved replacing the maximum likelihood estimators with others, for example models based on conditional entropy or mutual information. It is notable that in the test set there is not a single example that is correctly answered only with the evidence class-entity name, as either it matches the evidence of the entity type or with the solution given by the parser in the first place.

  Moreover, the main problem of the evidence of the class-entity type is that some classes are very dominant (*chief*, *business*), and tend to be overassigned.

  Within these evidences, it is clear that is more effective to combine both entity names and entity types to get an accurate measure of semantic compatibility.

- What configuration of evidences and measures achieves the best results?

  Regarding the configurations tested, the best results are obtained when combining both sources of evidence with smoothed conditioned probabilities. We reach a 93.2% of correct appositions which is a 15% of relative improvement with respect to the best baseline (80.9%).

7.2 Looking Forward

In this work we have tackled the problem of the knowledge acquisition and textual inference using simple techniques in each step. Besides the interesting results obtained, this works opens many opportunities of improvement in all the stages of the system.

As we explained in the Chapter 6, we have used semantics to solve structural ambiguities after using syntactic dependencies to build the semantic representation. This is a recursive problem: the more we improve the syntactic correction, the better semantic representation. One future line of research is, once we get the new apposition dependencies for a large corpus, we can repeat the process of knowledge acquisition, creating a bootstrap method that iteratively improves the dependence analysis and semantic class acquisition.

Although this work focuses on textual inference on apposition dependencies, it would be interesting to study whether this technique could be extrapolated to solve other types, such as abbreviations, copulative verbs, or even coreference resolution.

In this work we have captured information about semantic classes. This information has been useful for our purposes, but is in fact limited. Our representation allows selecting subgraphs to obtain morphosyntactic or semantic structures that can be useful for other tasks. We can also take advantage of the temporal edges to compile databases of temporal relations.

Regarding the representation used, we have in mind several ideas to improve it. First, we can take advantage of the information that is hidden on the text, but that people recover effortlessly [Peñas and Ovchinnikova, 2012]. Authors do not include information that they assume that their readers know, because its inclusion would mean an extra importance. The problem is that automatic systems cannot recover this information because they lack of the background knowledge necessary and the inference capabilities to use it.

One way to do it is to use type coercion to substitute general syntactic edges of structures that tend to contain this information, such as genitives or nominal compounds, for other edges more specific that point to the nature of the relation between the components could help to improve the information extraction systems.

Other opportunity of improvement is to include a system of event corefer-

ence [Humphreys et al., 1997, Hasler et al., 2006], and perform a collapsing to the one of the nominal compounds. This technique aims to improve the cohesion of the information.

Regarding the method for apposition correction, it could benefit from adding extra semantic information. For example, we could characterize entities with gender or age, to get more accurate measures of semantic compatibility. For example:

David meets a friend of his wife, ***Peter****.*

Knowing that *wife* is a class used with females and *Peter* is usually a male name leads us to consider the link between *friend* and *Peter* more appropriate.

We believe that including this method as a feature of current dependency parsers would lead to better performance.

CHAPTER 8
AUTHOR'S PUBLICATIONS RELATED TO THIS WORK

Author's publications related to this work:

- B. CABALEIRO AND A. PEÑAS, "Representación Gráfica de Documentos para Extracción Automática de Relaciones", in *Procesamiento del Lenguaje Natural*, 49(0), 2012, URL `http://journal.sepln.org/sepln/ojs/ojs/index.php/pln/article/view/4562`

- G. GARRIDO, B. CABALEIRO, A. PEÑAS, A. RODRIGO AND D. SPINA, "Distant supervised learning for the TAC KBP Slot Filling and Temporal Slot Filling Tasks", in *Text Analysis Conference*, TAC 2011, Workshop, Notebook Papers, 2011.

- G. GARRIDO, A. PEÑAS, B. CABALEIRO AND A. RODRIGO, "Temporally anchored relation extraction", in *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1,* pags 107-116, Association for Computational Linguistics, 2012.

- B. CABALEIRO AND A. PEÑAS, "Corrección no Supervisada de Dependencias Sintácticas de Aposición mediante Clases Semánticas", in *Procesamiento del Lenguaje Natural*, 51(0), 2013, URL

http://journal.sepln.org/sepln/ojs/ojs/index.php/pln/article/view/4873

- B. CABALEIRO AND A. PEÑAS, "The Role of Semantic Classes in Parsing Appositives", forthcoming.

This work has been used to support the research on the next publications:

- H. ANAYA-SANCHEZ, A. PEÑAS AND B. CABALEIRO, "UNED-READERS: Filtering Relevant Tweets using Probabilistic Signature Models", in P.FORNER, R. NAVIGLE AND D. TUFIS (Eds.), *CLEF 2013 Evaluation Labs and Workshop. Online Working Notes.,* 2013.
- A. PEÑAS, B. CABALEIRO AND M. LAPATA, "Unsupervised Interpretation of Eventive Propositions", in (Ed.): *CICLing 2014*, forthcoming, 2014.

Bibliography

[Agichtein and Gravano, 2000] E. AGICHTEIN and L. GRAVANO, "Snowball: Extracting relations from large plain-text collections", in *Proceedings of the fifth ACM conference on Digital libraries*, pgs. 85–94, ACM, 2000.

[Artiles et al., 2011] J. ARTILES, Q. LI, T. CASSIDY, S. TAMANG and H. JI, "CUNY BLENDER TAC-KBP2011 Temporal Slot Filling System Description", in *Proceedings of the Text Analysis Conference*, 2011.

[Auer et al., 2007] S. AUER, C. BIZER, G. KOBILAROV, J. LEHMANN, R. CYGANIAK and Z. IVES, "Dbpedia: A nucleus for a web of open data", in *The semantic web*, pgs. 722–735, Springer, 2007.

[Baldwin et al., 2007] T. BALDWIN, M. DRAS, J. HOCKENMAIER, T. H. KING and G. VAN NOORD, "The impact of deep linguistic processing on parsing technology", in *Proceedings of the 10th International Conference on Parsing Technologies*, pgs. 36–38, Association for Computational Linguistics, 2007.

[Banko et al., 2007] M. BANKO, M. J. CAFARELLA, S. SODERLAND, M. BROADHEAD and O. ETZIONI, "Open Information Extraction from the Web.", in *IJCAI*, volume 7, pgs. 2670–2676, 2007.

[Bast and Haussmann, 2013] H. BAST and E. HAUSSMANN, "Open information extraction via contextual sentence decomposition", in *Semantic Computing (ICSC), 2013 IEEE Seventh International Conference on*, pgs. 154–159, IEEE, 2013.

[Beth, 1995] S. BETH, "Proceedings of the Sixth Message Understanding Conference", MUC-6, Columbia, MD, 1995.

[Bethard and Martin, 2006] S. BETHARD and J. H. MARTIN, "Identification of event mentions and their semantic class", in *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pgs. 146–154, Association for Computational Linguistics, 2006.

[Björkelund et al., 2009] A. BJÖRKELUND, L. HAFDELL and P. NUGUES, "Multilingual semantic role labeling", in *Proceedings of the Thirteenth Conference on Computational Natural Language Learning: Shared Task*, pgs. 43–48, Association for Computational Linguistics, 2009.

[Bollacker et al., 2008] K. BOLLACKER, C. EVANS, P. PARITOSH, T. STURGE and J. TAYLOR, "Freebase: a collaboratively created graph database for structuring human knowledge", in *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pgs. 1247–1250, ACM, 2008.

[Brill et al., 2002] E. BRILL, S. DUMAIS and M. BANKO, "An analysis of the AskMSR question-answering system", in *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*, pgs. 257–264, Association for Computational Linguistics, 2002.

[Brin, 1999] S. BRIN, "Extracting patterns and relations from the world wide web", in *The World Wide Web and Databases*, pgs. 172–183, Springer, 1999.

[Carlson et al., 2010] A. CARLSON, J. BETTERIDGE, B. KISIEL, B. SETTLES, E. R. HRUSCHKA JR and T. M. MITCHELL, "Toward an Architecture for Never-Ending Language Learning.", in *AAAI*, 2010.

[Chang and Manning, 2012] A. X. CHANG and C. MANNING, "SUTime: A library for recognizing and normalizing time expressions.", in *LREC*, pgs. 3735–3740, 2012.

[Charniak and Johnson, 2005] E. CHARNIAK and M. JOHNSON, "Coarse-to-fine n-best parsing and MaxEnt discriminative reranking", in *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pgs. 173–180, Association for Computational Linguistics, 2005.

[Chen et al., 2010] D. CHEN, N. SCHNEIDER, D. DAS and N. A. SMITH, "SEMAFOR: Frame argument resolution with log-linear models", in *Proceedings of the 5th international workshop on semantic evaluation*, pgs. 264–267, Association for Computational Linguistics, 2010.

[Chinchor, 1998] N. CHINCHOR, "Overview of MUC-7", in *Proceedings of the Seventh Message Understanding Conference (MUC-7)*, pgs. 178–185, 1998.

[Clark and Harrison, 2009] P. CLARK and P. HARRISON, "Large-scale extraction and use of knowledge from text", in *Proceedings of the fifth international conference on Knowledge capture*, K-CAP '09, pgs. 153–160, ACM, New York, NY, USA, 2009, URL `http://doi.acm.org/10.1145/1597735.1597763`.

[Collins, 2003] M. COLLINS, "Head-driven statistical models for natural language parsing", *Computational linguistics*, 29(4):589–637, 2003.

[De Marneffe et al., 2006] M.-C. DE MARNEFFE, B. MACCARTNEY, C. D. MANNING and OTHERS, "Generating typed dependency parses from phrase structure parses", in *Proceedings of LREC*, volume 6, pgs. 449–454, 2006.

[De Marneffe and Manning, 2008] M.-C. DE MARNEFFE and C. D. MANNING, "Stanford typed dependencies manual", *URL http://nlp. stanford. edu/-software/dependencies manual. pdf*, 2008.

[Etzioni et al., 2004] O. ETZIONI, M. CAFARELLA, D. DOWNEY, S. KOK, A.-M. POPESCU, T. SHAKED, S. SODERLAND, D. S. WELD and A. YATES, "Web-scale Information Extraction in Knowitall: (Preliminary Results)", in *Proceedings of the 13th International Conference on World Wide Web*, WWW '04, pgs. 100–110, ACM, New York, NY, USA, 2004, URL `http://doi.acm.org/10.1145/988672.988687`.

[Etzioni et al., 2005] O. ETZIONI, M. CAFARELLA, D. DOWNEY, A.-M. POPESCU, T. SHAKED, S. SODERLAND, D. S. WELD and A. YATES, "Unsupervised named-entity extraction from the web: An experimental study", *Artificial Intelligence*, 165(1):91–134, 2005.

[Etzioni et al., 2011] O. ETZIONI, A. FADER, J. CHRISTENSEN, S. SODERLAND and M. MAUSAM, "Open information extraction: The second generation", in *Proceedings of the Twenty-Second international joint conference on Artificial Intelligence-Volume Volume One*, pgs. 3–10, AAAI Press, 2011.

[Fader et al., 2013] A. FADER, L. ZETTLEMOYER and O. ETZIONI, "Paraphrase-Driven Learning for Open Question Answering", in *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pgs. 1608–1618, Association for Computational Linguistics, Sofia, Bulgaria, August 2013, URL http://www.aclweb.org/anthology/P13-1158.

[Fass, 1991] D. FASS, "met*: A method for discriminating metonymy and metaphor by computer", *Computational Linguistics*, 1991, URL http://dl.acm.org/citation.cfm?id=971741.

[Ferrucci et al., 2010] D. FERRUCCI, E. BROWN, J. CHU-CARROLL, J. FAN, D. GONDEK, A. A. KALYANPUR, A. LALLY, J. W. MURDOCK, E. NYBERG, J. PRAGER and OTHERS, "Building Watson: An overview of the DeepQA project", *AI magazine*, 31(3):59–79, 2010.

[Finkel et al., 2005] J. R. FINKEL, T. GRENAGER and C. MANNING, "Incorporating non-local information into information extraction systems by Gibbs sampling", ACL '05, pgs. 363–370, Association for Computational Linguistics, Stroudsburg, PA, USA, 2005, URL http://dx.doi.org/10.3115/1219840.1219885.

[Garrido et al., 2011] G. GARRIDO, B. CABALEIRO, A. PEÑAS, Á. RODRIGO and D. SPINA, "Distant supervised learning for the TAC KBP Slot Filling and Temporal Slot Filling Tasks", in *Text Analysis Conference, TAC 2011,Workshop, Notebook Papers*, 2011.

[Grishman and Sundheim, 1996] R. GRISHMAN and B. SUNDHEIM, "Message Understanding Conference-6: A Brief History.", in *COLING*, volume 96, pgs. 466–471, 1996.

[Hajič et al., 2009] J. HAJIČ, J. RAAB, M. SPOUSTA et al., "Semi-supervised training for the averaged perceptron POS tagger", in *Proceedings of the*

12th Conference of the European Chapter of the Association for Computational Linguistics, pgs. 763–771, Association for Computational Linguistics, 2009.

[Hasler et al., 2006] L. HASLER, C. ORASAN and K. NAUMANN, "NPs for events: Experiments in coreference annotation", in *Proceedings of the 5th edition of the International Conference on Language Resources and Evaluation (LREC2006)*, pgs. 1167–1172, Citeseer, 2006.

[Hearst, 1992] M. A. HEARST, "Automatic acquisition of hyponyms from large text corpora", in *Proceedings of the 14th conference on Computational linguistics - Volume 2*, COLING '92, pgs. 539–545, Association for Computational Linguistics, Stroudsburg, PA, USA, 1992, URL `http://dx.doi.org/10.3115/992133.992154`.

[Heim, 1983] I. HEIM, *File Change Semantics and the Familiarity Theory of Definiteness*, pgs. 164–189, Walter de Gruyter, 1983.

[Hobbs, 1985] J. R. HOBBS, "Ontological promiscuity", in *Proceedings of the 23rd annual meeting on Association for Computational Linguistics*, pgs. 60–69, Association for Computational Linguistics, 1985.

[Hobbs et al., 1988] J. R. HOBBS, M. STICKEL, P. MARTIN and D. EDWARDS, "Interpretation as abduction", in *Proceedings of the 26th annual meeting on Association for Computational Linguistics*, pgs. 95–103, Association for Computational Linguistics, 1988.

[Hovy et al., 2002] E. HOVY, U. HERMJAKOB and D. RAVICHANDRAN, "A question/answer typology with surface text patterns", in *Proceedings of the second international conference on Human Language Technology Research*, pgs. 247–251, Morgan Kaufmann Publishers Inc., 2002.

[Humphreys et al., 1997] K. HUMPHREYS, R. GAIZAUSKAS and S. AZZAM, "Event coreference for information extraction", in *Proceedings of a Workshop on Operational Factors in Practical, Robust Anaphora Resolution for Unrestricted Texts*, pgs. 75–81, Association for Computational Linguistics, 1997.

[Ji et al., 2011] H. JI, R. GRISHMAN and H. DANG, "Overview of the TAC2011 Knowledge Base Population Track", in *TAC 2011 Proceedings Papers*, 2011.

[Kaplan and Bresnan, 1982] R. M. KAPLAN and J. BRESNAN, "Lexical-functional grammar: A formal system for grammatical representation", *Formal Issues in Lexical-Functional Grammar*, pgs. 29–130, 1982.

[Karttunen, 1968] L. KARTTUNEN, *What do referential indices refer to?*, Rand Corporation: [Paper], Rand Corp., 1968.

[Klein and Manning, 2003] D. KLEIN and C. D. MANNING, "Accurate unlexicalized parsing", in *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1*, pgs. 423–430, Association for Computational Linguistics, 2003.

[Kozareva et al., 2008] Z. KOZAREVA, E. RILOFF and E. HOVY, "Semantic Class Learning from the Web with Hyponym Pattern Linkage Graphs", in *Proceedings of ACL-08: HLT*, pgs. 1048–1056, Association for Computational Linguistics, Columbus, Ohio, 2008, URL http://www.aclweb.org/anthology/P/P08/P08-1119.

[Lapata et al., 2003] M. LAPATA, F. KELLER and C. SCHEEPERS, "Intra-sentential context effects on the interpretation of logical metonymy", *Cognitive Science*, 2003, URL http://www.sciencedirect.com/science/article/pii/S0364021303000351.

[Lapata and Lascarides, 2003] M. LAPATA and A. LASCARIDES, "A probabilistic account of logical metonymy", *Computational Linguistics*, 2003, URL http://dl.acm.org/citation.cfm?id=873747.

[Lawrie et al., 2013] D. LAWRIE, T. FININ, J. MAYFIELD and P. MCNAMEE, "Comparing and Evaluating Semantic Data Automatically Extracted from Text", in *AAAI 2013 Fall Symposium on Semantics for Big Data*, volume 25, pgs. 16–33, AAAI Press, 2013.

[Lee et al., 2011] H. LEE, Y. PEIRSMAN, A. CHANG, N. CHAMBERS, M. SURDEANU and D. JURAFSKY, "Stanford's multi-pass sieve coreference resolution system at the CoNLL-2011 shared task", CONLL Shared Task

’11, pgs. 28–34, Association for Computational Linguistics, Stroudsburg, PA, USA, 2011, URL http://dl.acm.org/citation.cfm?id=2132936.2132938.

[Lenat, 1995] D. B. LENAT, “CYC: A large-scale investment in knowledge infrastructure”, *Communications of the ACM*, 38(11):33–38, 1995.

[Lin, 2003] D. LIN, “Dependency-based evaluation of MINIPAR”, in *Treebanks*, pgs. 317–329, Springer, 2003.

[Lin and Pantel, 2001a] D. LIN and P. PANTEL, “DIRT@ SBT@ discovery of inference rules from text”, in *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pgs. 323–328, ACM, 2001a.

[Lin and Pantel, 2001b] D. LIN and P. PANTEL, “Induction of semantic classes from natural language text”, in *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD ’01, pgs. 317–322, ACM, New York, NY, USA, 2001b, URL http://doi.acm.org/10.1145/502512.502558.

[Mani et al., 2003] I. MANI, B. SCHIFFMAN and J. ZHANG, “Inferring temporal ordering of events in news”, in *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology: companion volume of the Proceedings of HLT-NAACL 2003–short papers - Volume 2*, NAACL-Short ’03, pgs. 55–57, Association for Computational Linguistics, Stroudsburg, PA, USA, 2003, URL http://dx.doi.org/10.3115/1073483.1073502.

[Mani and Wilson, 2000] I. MANI and G. WILSON, “Robust temporal processing of news”, in *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, pgs. 69–76, Association for Computational Linguistics, 2000.

[Manning, 2011] C. D. MANNING, “Part-of-speech tagging from 97% to 100%: is it time for some linguistics?”, in *Computational Linguistics and Intelligent Text Processing*, pgs. 171–189, Springer, 2011.

[Marsi et al., 2007] E. MARSI, E. KRAHMER and W. BOSMA, “Dependency-based paraphrasing for recognizing textual entailment”, in *Proceedings*

of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing, pgs. 83–88, Association for Computational Linguistics, 2007.

[Mausam et al., 2012] MAUSAM, M. SCHMITZ, R. BART, S. SODERLAND and O. ETZIONI, "Open Language Learning for Information Extraction", in *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CoNLL '12, pgs. 523–534, Association for Computational Linguistics, Stroudsburg, PA, USA, 2012, URL `http://dl.acm.org/citation.cfm?id=2390948.2391009`.

[Mayfield and Finin, 2012] J. MAYFIELD and T. FININ, "Evaluating the quality of a knowledge base populated from text", in *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction*, pgs. 68–73, Association for Computational Linguistics, 2012.

[Maynard et al., 2003] D. MAYNARD, K. BONTCHEVA and H. CUNNINGHAM, "Towards a semantic extraction of Named Entities", in *In Recent Advances in Natural Language Processing*, 2003.

[McCord et al., 2012] M. C. MCCORD, J. W. MURDOCK and B. K. BOGURAEV, "Deep parsing in Watson", *IBM Journal of Research and Development*, 56(3.4):1–3, 2012.

[McNamee and Dang, 2009] P. MCNAMEE and H. T. DANG, "Overview of the TAC 2009 Knowledge Base Population Track", in *Text Analysis Conference (TAC)*, 2009.

[Mel'cuk and Polguère, 1987] I. MEL'CUK and A. POLGUÈRE, "A Formal Lexicon in the Meaning-Text Theory (or How to Do Lexica with Words)", *Computational Linguistics*, 13(3-4):261–275, 1987.

[Mendes et al., 2012] P. N. MENDES, M. JAKOB and C. BIZER, "DBpedia for NLP: A Multilingual Cross-domain Knowledge Base", in *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey, May 2012.

[Miller, 1995] G. A. MILLER, "WordNet: a lexical database for English", *Communications of the ACM*, 38(11):39–41, 1995.

[Mitchell et al., 2009] T. M. MITCHELL, J. BETTERIDGE, A. CARLSON, E. HRUSCHKA and R. WANG, "Populating the Semantic Web by Macro-reading Internet Text", ISWC '09, pgs. 998–1002, Springer-Verlag, Berlin, Heidelberg, 2009, URL http://dx.doi.org/10.1007/978-3-642-04930-9_66.

[Ng, 2010] V. NG, "Supervised noun phrase coreference research: the first fifteen years", in *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pgs. 1396–1411, Association for Computational Linguistics, Stroudsburg, PA, USA, 2010, URL http://dl.acm.org/citation.cfm?id=1858681.1858823.

[Nivre et al., 2006] J. NIVRE, J. HALL and J. NILSSON, "Maltparser: A data-driven parser-generator for dependency parsing", in *Proceedings of LREC*, volume 6, pgs. 2216–2219, 2006.

[Palmer et al., 1986] M. S. PALMER, D. A. DAHL, R. J. SCHIFFMAN, L. HIRSCHMAN, M. LINEBARGER and J. DOWDING, "Recovering implicit information", in *Proceedings of the 24th annual meeting on Association for Computational Linguistics*, pgs. 10–19, Association for Computational Linguistics, 1986.

[Peñas and Hovy, 2010] A. PEÑAS and E. HOVY, "Filling knowledge gaps in text for machine reading", in *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, pgs. 979–987, Association for Computational Linguistics, 2010.

[Peñas and Ovchinnikova, 2012] A. PEÑAS and E. OVCHINNIKOVA, "Unsupervised Acquisition of Axioms to Paraphrase Noun Compounds and Genitives", in *(Ed.): CICLing 2012, Part I, LNCS 7181*, pgs. 388–401, Springer-Verlag, 2012.

[Pease et al., 2002] A. PEASE, I. NILES and J. LI, "The suggested upper merged ontology: A large ontology for the semantic web and its applications", in *Working notes of the AAAI-2002 workshop on ontologies and the semantic web*, volume 28, 2002.

[Pollard, 1994] C. POLLARD, *Head-driven phrase structure grammar*, University of Chicago Press, 1994.

[Pradhan et al., 1994] M. PRADHAN, G. PROVAN, B. MIDDLETON and M. HENRION, "Knowledge engineering for large belief networks", in *Proceedings of the Tenth international conference on Uncertainty in artificial intelligence*, UAI'94, pgs. 484–490, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1994.

[Punyakanok et al., 2008] V. PUNYAKANOK, D. ROTH and W.-T. YIH, "The importance of syntactic parsing and inference in semantic role labeling", *Computational Linguistics*, 34(2):257–287, 2008.

[Pustejovsky, 1991] J. PUSTEJOVSKY, "The generative lexicon", *Computational linguistics*, 17(4):409–441, 1991.

[Pustejovsky et al., 2003] J. PUSTEJOVSKY, P. HANKS, R. SAURI, A. SEE, R. GAIZAUSKAS, A. SETZER, D. RADEV, B. SUNDHEIM, D. DAY, L. FERRO and OTHERS, "The timebank corpus", in *Corpus linguistics*, volume 2003, p. 40, 2003.

[Pustejovsky et al., 2010] J. PUSTEJOVSKY, A. RUMSHISKY, A. PLOTNICK, E. JEZEK, O. BATIUKOVA and V. QUOCHI, "SemEval-2010 task 7: Argument selection and coercion", in *Proceedings of the 5th International Workshop on Semantic Evaluation*, pgs. 27–32, Association for Computational Linguistics, 2010.

[Raghunathan et al., 2010] K. RAGHUNATHAN, H. LEE, S. RANGARAJAN, N. CHAMBERS, M. SURDEANU, D. JURAFSKY and C. MANNING, "A multi-pass sieve for coreference resolution", in *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pgs. 492–501, Association for Computational Linguistics, 2010.

[Ratinov and Roth, 2009] L. RATINOV and D. ROTH, "Design challenges and misconceptions in named entity recognition", in *Proceedings of the Thirteenth Conference on Computational Natural Language Learning*, pgs. 147–155, Association for Computational Linguistics, 2009.

[Recasens, 2010] M. RECASENS, "Coreference: Theory, Annotation, Resolution and Evaluation", , 2010.

[Roberts and Harabagiu, 2011] K. ROBERTS and S. M. HARABAGIU, "Unsupervised learning of selectional restrictions and detection of argument

coercions", in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pgs. 980–990, Association for Computational Linguistics, 2011.

[Ruppenhofer et al., 2010] J. RUPPENHOFER, C. SPORLEDER, R. MORANTE, C. BAKER and M. PALMER, "Semeval-2010 task 10: Linking events and their participants in discourse", in *Proceedings of the 5th International Workshop on Semantic Evaluation*, pgs. 45–50, Association for Computational Linguistics, 2010.

[Saurí et al., 2005] R. SAURÍ, R. KNIPPEN, M. VERHAGEN and J. PUSTEJOVSKY, "Evita: a robust event recognizer for QA systems", in *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pgs. 700–707, Association for Computational Linguistics, 2005.

[Schank, 1972] R. C. SCHANK, "Conceptual dependency: A theory of natural language understanding", *Cognitive Psychology*, 3(4):552–631, October 1972.

[Schubert, 2002] L. SCHUBERT, "Can we derive general world knowledge from texts?", in *Proceedings of the second international conference on Human Language Technology Research*, pgs. 94–97, Morgan Kaufmann Publishers Inc., 2002.

[Sekine and Nobata, 2004] S. SEKINE and C. NOBATA, "Definition, Dictionaries and Tagger for Extended Named Entity Hierarchy.", in *LREC*, 2004.

[Shen et al., 2007] L. SHEN, G. SATTA and A. JOSHI, "Guided learning for bidirectional sequence classification", in *ACL*, volume 7, pgs. 760–767, 2007.

[Shinyama and Sekine, 2006] Y. SHINYAMA and S. SEKINE, "Preemptive information extraction using unrestricted relation discovery", in *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, pgs. 304–311, Association for Computational Linguistics, 2006.

[Suchanek et al., 2007] F. M. SUCHANEK, G. KASNECI and G. WEIKUM, "Yago: a core of semantic knowledge", in *Proceedings of the 16th international conference on World Wide Web*, pgs. 697–706, ACM, 2007.

[Surdeanu et al., 2011] M. SURDEANU, S. GUPTA, J. BAUER, D. MCCLOSKY, A. X. CHANG, V. I. SPITKOVSKY and C. D. MANNING, "Stanford's Distantly-Supervised Slot-Filling System", in *Proceedings of the Fourth Text Analysis Conference (TAC 2011)*, Gaithersburg, Maryland, USA, November 2011.

[Surdeanu and Manning, 2010] M. SURDEANU and C. D. MANNING, "Ensemble models for dependency parsing: cheap and good?", in *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pgs. 649–652, Association for Computational Linguistics, 2010.

[Toutanova et al., 2003] K. TOUTANOVA, D. KLEIN, C. D. MANNING and Y. SINGER, "Feature-rich part-of-speech tagging with a cyclic dependency network", in *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pgs. 173–180, Association for Computational Linguistics, 2003.

[Toutanova and Manning, 2000] K. TOUTANOVA and C. D. MANNING, "Enriching the knowledge sources used in a maximum entropy part-of-speech tagger", EMNLP '00, pgs. 63–70, Association for Computational Linguistics, Stroudsburg, PA, USA, 2000, URL http://dx.doi.org/10.3115/1117794.1117802.

[Tratz and Hovy, 2011] S. TRATZ and E. HOVY, "A fast, accurate, non-projective, semantically-enriched parser", in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pgs. 1257–1268, Association for Computational Linguistics, 2011.

[Utiyama et al., 2000] M. UTIYAMA, M. MURATA and H. ISAHARA, "A statistical approach to the processing of metonymy", *Proceedings of the 18th conference on ...*, 2000, URL http://dl.acm.org/citation.cfm?id=992774.

[Verhagen et al., 2005] M. VERHAGEN, I. MANI, R. SAURI, R. KNIPPEN, S. B. JANG, J. LITTMAN, A. RUMSHISKY, J. PHILLIPS and J. PUSTEJOVSKY, "Automating temporal annotation with TARSQI", in *Proceedings of the*

ACL 2005 on Interactive poster and demonstration sessions, pgs. 81–84, Association for Computational Linguistics, 2005.

[Verhagen and Pustejovsky, 2008] M. VERHAGEN and J. PUSTEJOVSKY, "Temporal processing with the TARSQI toolkit", in *22nd International Conference on on Computational Linguistics: Demonstration Papers*, pgs. 189–192, Association for Computational Linguistics, 2008.

[Voutilainen, 2003] A. VOUTILAINEN, "Part-of-speech tagging", *The Oxford handbook of computational linguistics*, pgs. 219–232, 2003.

[Wu and Weld, 2007] F. WU and D. S. WELD, "Autonomously semantifying wikipedia", in *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pgs. 41–50, ACM, 2007.

[Wu and Weld, 2010] F. WU and D. S. WELD, "Open information extraction using Wikipedia", in *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pgs. 118–127, Association for Computational Linguistics, 2010.

[Yates and Etzioni, 2007] A. YATES and O. ETZIONI, "Unsupervised Resolution of Objects and Relations on the Web.", in *HLT-NAACL*, pgs. 121–130, 2007.

APPENDIX

APPENDIX A
GRAPH TRANSFORMATION RULES

Transformation rules for in the graph representation procedure. Written in JESS, a ruler interpreter for Java.

A.1 Rules for semantic class extraction

```
(defrule ruleHasClass1
        "relation(N1,N2, hasClass) :- relation(N1, N2, nn),
                        ner(N1)"

        ?r <- (Relation (word1 ?word1) (word2 ?w2)
                        (relation "nn"))

        (test (or (?word1 isNER "PERSON")
                        (?word1 isNER "LOCATION")
                        (?word1 isNER "ORGANIZATION")))
        (test (or (eq (?w2 getPOS) "NN")
                        (eq (?w2 getPOS) "NNS")))
        (test (neq ?word1 ?w2))

        =>
        (modify ?r (relation "hasClass")))
```

```
(defrule ruleFixHasClass1
        "relation(N1, N2, nn), relation(N2,N3, nn) :-
                        relation(N1, N2, nn), relation(N1, N3, nn),
                        ner(N1)"

        (declare (salience 1))
        ?r  <- (Relation (word1 ?word1) (word2 ?w2)
                        (relation "nn"))
        ?r2 <- (Relation (word1 ?word1) (word2 ?w3)
                        (relation "nn") (OBJECT ?o2))

        (test (or (?w1 isNER "PERSON")
                        (?w1 isNER "LOCATION")
                        (?w1 isNER "ORGANIZATION")))
        (test (or (eq (?w2 getPOS) "NN")
                        (eq (?w2 getPOS) "NNS")))
        (test (or (eq (?word3 getPOS) "NN")
                        (eq (?w3 getPOS) "NNS")))
        (test (neq ?w1 ?w2))
        (test (neq ?w1 ?w3))
        (test (neq ?w2 ?w3))
        (test (?w3 isBefore ?w2))

        =>
        (?o2 modify ?w1 ?w2)
        (modify ?r2 (word1 ?w2)))

(defrule ruleHasClass2
        "relation(N1,N2, hasClass) :- relation(N1, N2, appos),
                        ner(N1)"

        ?r <- (Relation (word1 ?w1) (word2 ?w2)
                        (relation "appos"))

        (test (or (?w1 isNER "PERSON")
                        (?w1 isNER "LOCATION")
                        (?w1 isNER "ORGANIZATION")))
        (test (or (eq (?w2 getPOS) "NN")
                        (eq (?w2 getPOS) "NNS")))
                (test (neq ?w1 ?w2))

        =>
        (modify ?r (relation "hasClass")))
```

```
(defrule ruleHasClass3
        "relation(N1,N2, hasClass) :- relation(N1, N2, abbrev),
                        ner(N1)"

        ?r <- (Relation (word1 ?w1) (word2 ?w2)
                        (relation "abbrev"))

        (test (or (?w1 isNER "PERSON")
                        (?w1 isNER "LOCATION")
                        (?w1 isNER "ORGANIZATION")))
        (test (or (eq (?w2 getPOS) "NN")
                        (eq (?w2 getPOS) "NNS")))
        (test (neq ?w1 ?w2))

        =>
        (modify ?r (relation "hasClass")))

(defrule ruleHasClass5
        "relation(N2,N1,has_) :- relation(N1, N2, appos),
                        ner(N2)"

        ?r <- (Relation (word1 ?w1) (word2 ?w2)
                        (relation "appos"))

        (test (or (?w2 isNER "PERSON")
                        (?w2 isNER "LOCATION")
                        (?w2 isNER "ORGANIZATION")))
        (test (or (eq (?w1 getPOS) "NN")
                        (eq (?w1 getPOS) "NNS")))
        (test (neq ?w1 ?w2))

        =>
        (modify ?r (word1 ?w2) (word2 ?w1) (relation "hasClass"))       )

(defrule ruleHasClass6
        "relation(N2,N1,has_) :- relation(N1, N2, abbrev),
                        ner(N2)"

        ?r <- (Relation (word1 ?w1) (word2 ?w2)
                        (relation "abbrev"))

        (test (or (?w2 isNER "PERSON")
                        (?w2 isNER "LOCATION")
```

```
                        (?w2 isNER "ORGANIZATION")))
        (test (or (eq (?w1 getPOS) "NN")
                        (eq (?w1 getPOS) "NNS")))
        (test (neq ?w1 ?w2))

        =>
        (modify ?r (word1 ?w2) (word2 ?w1) (relation "hasClass")))

(defrule ruleHasClass7
        "relation(N1,N2, hasClass) :- relation(N1, N2, nsubj),
                        ner(N1)"

        ?r <- (Relation (word1 ?w1) (word2 ?w2)
                        (relation "nsubj"))

        (test (or (?w1 isNER "PERSON")
                        (?w1 isNER "LOCATION")
                        (?w1 isNER "ORGANIZATION")))
        (test (or (eq (?w2 getPOS) "NN")
                        (eq (?w2 getPOS) "NNS")))
        (test (neq ?w1 ?w2))

        =>
        (modify ?r (relation "hasClass")))

(defrule ruleHasClass11
        "relation(N2,N1, hasClass) :- relation(N1, N2, prep_such_as),
                        ner(N2)"

        ?r <- (Relation (word1 ?w1) (word2 ?w2)
                        (relation "prep_such_as"))

        (test (or (?w2 isNER "PERSON")
                        (?w2 isNER "LOCATION")
                        (?w2 isNER "ORGANIZATION")))
        (test (or (eq (?w1 getPOS) "NN")
                        (eq (?w1 getPOS) "NNS")))
        (test (neq ?w1 ?w2))

        =>
        (modify ?r (word1 ?w2) (word2 ?w1) (relation "hasClass")))

(defrule ruleHasClass13
        "relation(N2,N1, hasClass) :- relation(N1, N2, prep_like),
```

```
                        ner(N2)"

        ?r <- (Relation (word1 ?w1) (word2 ?w2)
                        (relation "prep_like"))

        (test (or (?w2 isNER "PERSON")
                        (?w2 isNER "LOCATION")
                        (?w2 isNER "ORGANIZATION")))
        (test (or (eq (?w1 getPOS) "NN")
                        (eq (?w1 getPOS) "NNS")))
        (test (neq ?w1 ?w2))

        =>
        (modify ?r (word1 ?w2) (word2 ?w1) (relation "hasClass"))       )
```

A.2 Rules for genitive normalization

```
(defrule ruleHas1
        "relation(N1,N2,has_) :- relation(N1, N2, nn),
                        ner(N2)"

        ?r <- (Relation (word1 ?w1) (word2 ?w2) (relation "nn"))

        (test (or (?w2 isNER "PERSON")
                        (?w2 isNER "LOCATION")
                        (?w2 isNER "ORGANIZATION")))
        (test (neq ?w1 ?w2))
        (test (or (eq (?w1 getPOS) "NN")
                        (eq (?w1 getPOS) "NNS")))

        =>
        (bind ?s (str-cat "has_"(?w1 getLemma)))
        (modify ?r (word1 ?w2) (word2 ?w1) (relation ?s)))

(defrule ruleHas2
        "relation(N2,N1,has_) :- relation(N1, N2, poss),
                        ner(N2)"

        ?r <- (Relation (word1 ?w1) (word2 ?w2) (relation "poss"))

        (test (or (?w2 isNER "PERSON")
                        (?w2 isNER "LOCATION")
                        (?w2 isNER "ORGANIZATION")))
```

```
        (test (or (eq (?w1 getPOS) "NN")
                  (eq (?w1 getPOS) "NNS")))
        (test (neq ?w1 ?w2))

        =>
        (bind ?s (str-cat "has_"(?w1 getLemma)))
        (modify ?r (word1 ?w2) (word2 ?w1) (relation ?s)))

(defrule ruleHas3
        "relation(N2,N1,has) :- relation(N1, N2, poss)"

        (declare (salience -1))
        ?r <- (Relation (word1 ?w1) (word2 ?w2) (relation "poss"))

        (test (neq ?w1 ?w2))

        =>
        (modify ?r (word1 ?w2) (word2 ?w1) (relation "has"))    )

(defrule ruleHas4
        "relation(N2,N1,has_) :- relation(N1, N2, prep_of),
                        ner(N2)"

        ?r <- (Relation (word1 ?w1) (word2 ?w2) (relation "prep_of"))

        (test (or (?w2 isNER "PERSON")
                  (?w2 isNER "LOCATION")
                  (?w2 isNER "ORGANIZATION")))
        (test (or (eq (?w1 getPOS) "NN")
                  (eq (?w1 getPOS) "NNS")))
        (test (neq ?w1 ?w2))

        =>
        (bind ?s (str-cat "has_"(?w1 getLemma)))
        (modify ?r (word1 ?w2) (word2 ?w1) (relation ?s)))

(defrule ruleHas6
        "relation(N1,N2,hasClass), relation(N2,N3,hasClass) :-
                relation(N1,N2,has_N3), relation(N2,N3,hasClass), ner(N2)"
        "Si un nombre es una clase, y un segundo nombre posee
        el primero, la posesion es del tipo de la clase"

        ?r <- (Relation (word1 ?w1) (word2 ?w2) (relation "has")),
        (Relation (word1 ?w2) (word2 ?w3) (relation "hasClass"))
```

```
        (test (or (?w2 isNER "PERSON")
                        (?w2 isNER "LOCATION")
                        (?w2 isNER "ORGANIZATION")))
        (test (?w3 areConsecutive ?w2))

        =>
        (bind ?s (str-cat "has_"(?w3 getLemma)))
        (modify ?r (relation ?s)))

(defrule ruleHas7
        "relation(N2,N1,has_) :- relation(N1, N2, nsubj), ner(N2)"

        ?r <- (Relation (word1 ?w1) (word2 ?w2) (relation "nsubj"))

        (test (or (?w2 isNER "PERSON")
                        (?w2 isNER "LOCATION")
                        (?w2 isNER "ORGANIZATION")))
        (test (neq ?w1 ?w2))
        (test (or (eq (?w1 getPOS) "NN")
                        (eq (?w1 getPOS) "NNS")))

        =>
        (bind ?s (str-cat "has_"(?w1 getLemma)))
        (modify ?r (word1 ?w2) (word2 ?w1) (relation ?s)))
```

A.3 Rules for naïve semantic role labeling

```
(defrule rulePassive1
        "nsubj-> arg0"

        ?r <- (Relation (word1 ?w1) (relation "nsubj"))

        (test (neq ((?w1 getListEvent) size) 0))

        =>
        (modify ?r (relation "arg0")))

(defrule rulePassive2

        "xsubj-> arg0"

        ?r <- (Relation (word1 ?w1) (relation "xsubj"))
```

```
        (test (neq ((?w1 getListEvent) size) 0))

        =>
        (modify ?r (relation "arg0")))

(defrule rulePassive3

        "csubj-> arg0"

        ?r <- (Relation (word1 ?w1) (relation "csubj"))

        (test (neq ((?w1 getListEvent) size) 0))

        =>
        (modify ?r (relation "arg0")))

(defrule rulePassive4
        "agent-> arg0"

        ?r <- (Relation (word1 ?w1) (relation "agent"))

        (test (neq ((?w1 getListEvent) size) 0))

        =>
        (modify ?r (relation "arg0")))

(defrule rulePassive5
        "nsubjpass + arg1 -> arg2"

        (declare (salience -1))
        ?r  <- (Relation (word1 ?w1) (word2 ?w2)
                         (relation "nsubjpass"))
        ?r2 <- (Relation (word1 ?w1) (word2 ?)
                         (relation "arg1"))

        (test (neq ((?w1 getListEvent) size) 0))

        =>
        (modify ?r (relation "arg2")))

(defrule rulePassive6
        "nsubjpass-> arg1"
```

```
        ?r <- (Relation (word1 ?w1) (word2 ?w2)
                        (relation "nsubjpass"))

        (test (neq ((?w1 getListEvent) size) 0))

        =>
        (modify ?r (relation "arg1")))

(defrule rulePassive7
        "dobj-> arg1"

        ?r <- (Relation (word1 ?w1) (relation "dobj"))

        (test (neq ((?w1 getListEvent) size) 0))

        =>
        (modify ?r (relation "arg1")))

(defrule rulePassive8
        "iobj-> arg2"

        ?r <- (Relation (word1 ?w1) (relation "iobj"))

        (test (neq ((?w1 getListEvent) size) 0))

        =>
        (modify ?r (relation "arg2")))

(defrule rulePassive9
        "partmod(x1,x2)-> arg1(x2,x1)"

        ?r <- (Relation (word1 ?w1) (word2 ?w2)
                        (relation "partmod"))

        (test (neq ((?w2 getListEvent) size) 0))

        =>
        (modify ?r (word1 ?w2) (word2 ?w1) (relation "arg1"))   )

(defrule rulePassive10
 "xcomp-> arg1"

        ?r <- (Relation (word1 ?w1) (relation "xcomp"))
```

```
        (test (neq ((?w1 getListEvent) size) 0))

        =>
        (modify ?r (relation "arg1")))

(defrule rulePassive11
        "ccomp-> arg1"

        ?r <- (Relation (word1 ?w1) (relation "ccomp"))

        (test (neq ((?w1 getListEvent) size) 0))

        =>
        (modify ?r (relation "arg1")))

(defrule rulePassive12
        "xcomp + arg1 -> arg2"

        (declare (salience -1))
        ?r <- (Relation (word1 ?w1) (word2 ?w2)
                        (relation "xcomp"))
        ?r2 <- (Relation (word1 ?w1) (word2 ?)
                        (relation "arg1"))

        (test (neq ((?w1 getListEvent) size) 0))

        =>
        (modify ?r (relation "arg2")))

(defrule rulePassive13
        "ccomp + arg1 -> arg2"

        (declare (salience -1))
        ?r  <- (Relation (word1 ?w1) (word2 ?w2)
                        (relation "ccomp"))
        ?r2 <- (Relation (word1 ?w1) (word2 ?)
                        (relation "arg1"))

        (test (neq ((?w1 getListEvent) size) 0))

        =>
        (modify ?r (relation "arg2")))
```

A.4 Rules for semantic edges addition

```
(defrule ruleBe1
        "relation(N1,N2,is) :- relation(N1, N2, nsubj),
                        relation(N1, N3, cop)"

        ?r  <- (Relation (word1 ?w1) (word2 ?w2)
                        (relation "nsubj"))
        ?r2 <- (Relation (word1 ?w1) (word2 ?w3)
                        (relation "cop") (OBJECT ?o2))

        (test (or (eq (?w1 getPOS) "NN")
                        (eq (?w1 getPOS) "NNS")))

        =>
        (?o2 delete)
        (retract ?r2)
        (modify ?r (word1 ?w2) (word2 ?w1) (relation "is")))

(defrule ruleHasAge1
        "relation(Person,Number,hasAge) :-
                        relation(Person, Number, appos)"

        ?r <- (Relation (word1 ?w1) (word2 ?w2)
                        (relation "appos"))

        (test (neq ((?w1 getListNamedEntity) size) 0))
        (test (neq ((?w2 getListNamedEntity) size) 0))
        (test (?w1 isNER "PERSON"))
        (test (?w2 isNER "NUMBER"))
        (test (neq ?w1 ?w2))

        =>
        (modify ?r (relation "hasAge")))

(defrule ruleHasAge2
        "relation(Person,Number,hasAge) :-
                        relation(Person, Number, abbrev)"

        ?r <- (Relation (word1 ?w1) (word2 ?w2) (relation "abbrev"))

        (test (neq ((?w1 getListNamedEntity) size) 0))
        (test (neq ((?w2 getListNamedEntity) size) 0))
```

```
        (test (?w1 isNER "PERSON"))
        (test (?w2 isNER "NUMBER"))
        (test (neq ?w1 ?w2))

        =>
        (modify ?r (relation "hasAge")) )

(defrule ruleGender1
        "relation(Person, Male, gender) :-
                        relation(Person, he, coreference)"

        ?r <- (Relation (word1 ?w1) (word2 ?w2)
                        (relation "coreference"))

        (test (?w1 isNER "PERSON"))
        (test (eq (?w2 getLemma) "he"))
        (test (neq ?w1 ?w2))

        =>
        (?w2 addData "GENDER" "MALE")   )

(defrule ruleGender1b
        "relation(Person, Male, gender) :-
                        relation(he, Person, coreference)"

        ?r <- (Relation (word1 ?w2) (word2 ?w1)
                        (relation "coreference"))

        (test (?w1 isNER "PERSON"))
        (test (eq (?w2 getLemma) "he"))
        (test (neq ?w1 ?w2))

        =>
        (?w2 addData "GENDER" "MALE")   )

(defrule ruleGender2
        "relation(Person, Female, gender) :-
                        relation(Person, she, coreference)"

        ?r <- (Relation (word1 ?w1) (word2 ?w2)
                        (relation "coreference"))

        (test (?w1 isNER "PERSON"))
        (test (eq (?w2 getLemma) "she"))
```

```
        (test (neq ?w1 ?w2))

        =>
        (?w2 addData "GENDER" "FEMALE") )

(defrule ruleGender2b
        "relation(Person, Female, gender) :-
                        relation(she, Person, coreference)"

        ?r <- (Relation (word1 ?w2) (word2 ?w1)
                        (relation "coreference"))

        (test (?w1 isNER "PERSON"))
        (test (eq (?w2 getLemma) "she"))
        (test (neq ?w1 ?w2))

        =>
        (?w2 addData "GENDER" "FEMALE") )
```

A.5 Rules for delete spurious relations in the enriched representation

```
(defrule ruleCleanCollapsed1
        "Delete repeated nn relations"

        ?r  <- (Relation (word1 ?w1) (word2 ?w2) (relation "nn"))
        ?r2 <- (Relation (word1 ?w1) (word2 ?w2) (relation "nn")
                        (OBJECT ?o2))

        (test (neq ?r ?r2))

        =>
        (?o2 delete)
        (retract ?r2))

(defrule ruleCleanCollapsed2
        "Delete repeated amod relations"

        ?r  <- (Relation (word1 ?w1) (word2 ?w2) (relation "nn"))
        ?r2 <- (Relation (word1 ?w1) (word2 ?w2) (relation "amod")
                        (OBJECT ?o2))

        =>
```

```
        (?o2 delete)
        (retract ?r2))

(defrule ruleCleanCollapsed3
        "Delete repeated relations"
        ?r  <- (Relation (word1 ?w1) (word2 ?w2) (relation ?s))
        ?r2 <- (Relation (word1 ?w1) (word2 ?w2) (relation ?s)
                         (OBJECT ?o2))

        (test (neq ?r ?r2))

        =>
        (?o2 delete)
        (retract ?r2))

(defrule ruleCleanAutoReferences
        "Delete auto references"

        ?r <- (Relation (word1 ?w1) (word2 ?w1) (relation ?s)
                         (OBJECT ?o))

        =>
        (?o delete)
        (retract ?r)    )

(defrule ruleFixCoreference2
        "Delete coreference between an instance and a class"

        (declare (salience 1))
        ?r  <- (Relation (word1 ?w1) (word2 ?w2)
                         (relation "hasClass"))
        ?r2 <- (Relation (word1 ?w3) (word2 ?w2)
                         (relation "coreference")
                         (OBJECT ?o2))

        =>
        (?o2 delete)
        (retract ?r2))

(defrule ruleFixCoreference2b
        "borrar la correferencia entre una entidad y su clase"

        (declare (salience 1))
        ?r  <- (Relation (word1 ?w1) (word2 ?w2)
```

```
                        (relation "hasClass"))
        ?r2 <- (Relation (word1 ?w2) (word2 ?w3)
                        (relation "coreference") (OBJECT ?o2))

        =>
        (?o2 delete)
        (retract ?r2))

(defrule ruleFixCoreference3
        "Delete coreference between an instance and a class"

        (declare (salience 1))
        ?r  <- (Relation (word1 ?w1) (word2 ?w2)
                        (relation "is"))
        ?r2 <- (Relation (word1 ?w3) (word2 ?w2)
                        (relation "coreference") (OBJECT ?o2))

        =>
        (?o2 delete)
        (retract ?r2))

(defrule ruleFixCoreference3b
        "Delete coreference between an instance and a class"

        (declare (salience 1))
        ?r  <- (Relation (word1 ?w1) (word2 ?w2)
                        (relation "is"))
        ?r2 <- (Relation (word1 ?w2) (word2 ?w3)
                        (relation "coreference") (OBJECT ?o2))

        =>
        (?o2 delete)
        (retract ?r2))

(defrule rulePointClassGovernor1
        "relation(N1,N2,hasClass), relation(N3,N1,relation) :-
        relation(N1,N2,hasClass), relation(N3,N2,relation)"

        (declare (salience -1))
        (Relation (word1 ?w1) (word2 ?w2) (relation "hasClass"))
        ?r <- (Relation (word1 ?w3) (word2 ?w2) (OBJECT ?o2))
        (test (neq ?w1 ?w2))
        (test (neq ?w1 ?w3))
        (test (neq ?w2 ?w3))
```

```
        =>
        (?o2 modify ?w2 ?w1)
        (modify ?r (word2 ?w1)))

(defrule rulePointClassGovernor2
        "Delete reciprocal hasClass relations"

        (declare (salience 0))
        (Relation (word1 ?w1) (word2 ?w2) (relation "hasClass"))
        ?r <- (Relation (word1 ?w2) (word2 ?w1)
                        (relation "hasClass") (OBJECT ?o2))
        (Relation (word1 ?w3) (word2 ?w2))

        (test (neq ?w1 ?w2))
        (test (neq ?w1 ?w3))
        (test (neq ?w2 ?w3))

        =>
        (?o2 delete)
        (retract ?r))

(defrule ruleHidePRP1
        "Avoid pronouns in the descriptors"
        (declare (salience -2))
        ?r <- (Relation (word1 ?w1) (word2 ?w2)
                        (relation "coreference"))

        (test ((?w2 getPOS) startsWith "PRP"))
        (test (?w1 getVisible))
        (test (neq ?w1 ?w2))

        =>
        (?w2 setVisible FALSE))

(defrule ruleHidePRP1b
        "Avoid pronouns in the descriptors"

        (declare (salience -2))
        ?r <- (Relation (word1 ?w1) (word2 ?w2)
                        (relation "coreference"))

        (test ((?w1 getPOS) startsWith "PRP"))
        (test (?w2 getVisible))
```

```
        (test (neq ?w1 ?w2))

        =>
        (?w1 setVisible FALSE))

(defrule ruleCollapseClasses1
        "A class is an attribute of the node"

        (declare (salience -1))
        (Relation (word1 ?w1) (word2 ?w2) (relation "hasClass"))

        (test (neq ?w1 ?w2))

        =>
        (?w1 addClass (?w2 getLemma)))

(defrule rulePositionTimeX31
        "Set temporal expressions as childs of a event node"

        ?r  <- (Relation (word1 ?w1) (word2 ?w2)
                         (relation ?relation))
        ?r2 <- (Relation (word1 ?w2) (word2 ?w3)
                         (relation ?relation2) (OBJECT ?o2))

        (test (neq ((?w1 getListEvent) size) 0))
        (test (eq ((?w2 getListEvent) size) 0))
        (test (neq ((?w3 getListTimeX3) size) 0))
        (test (neq ?w1 ?w2))
        (test (neq ?w1 ?w3))
        (test (neq ?w2 ?w3))
        (test (neq ?relation "SIMULTANEOUS"))
        (test (neq ?relation "INCLUDES"))
        (test (neq ?relation "BEGUN_BY"))
        (test (neq ?relation "ENDED"))
        (test (neq ?relation "BEFORE"))
        (test (?relation2 startsWith "prep"))

        =>
        (?o2 modify ?w2 ?w1)
        (modify ?r2 (word1 ?w1)))
```

APPENDIX B
GRAPH REPRESENTATION

This appendix shows an example of the representation of a document as a graph as described in Chapter 3.

The document represented is:

Wu Shu-chen, the former first wife, visited her husband Chen Shui-bian in detention in the morning. She was accompanied by their son Chen Chih-chung and Lawrence Gao, a Democratic Progressive Party lawmaker.

B.1 Initial Representation

B.1.1 DOT output

```
digraph document {
"node1"[color=green]
"node1"[label="Wu[NNP,Wu]_1_1 Shu-chen[NNP,Shu-chen]_2_1,
   NER:PERSON"]
"node5"[color=green]
"node5"[label="first[JJ,first]_6_1, NER:ORDINAL"]
"node11"[color=green]
"node11"[label="Chen[NNP,Chen]_12_1 Shui-bian[NNP,Shui-bian]_13_1,
   NER:PERSON"]
"node15"[color=green]
```

```
"node15"[label="the[DT,the]_17_1 morning[NN,morning]_18_1, NER:TIME"]
"node23"[color=green]
"node23"[label="Chen[NNP,Chen]_7_2 Chih-chung[NNP,Chih-chung]_8_2,
   NER:PERSON"]
"node25"[color=green]
"node25"[label="Lawrence[NNP,Lawrence]_10_2 Gao[NNP,Gao]_11_2,
   NER:PERSON"]
"node28"[color=green]
"node28"[label="Democratic[JJ,democratic]_14_2
   Progressive[NNP,Progressive]_15_2 Party[NNP,Party]_16_2,
   NER:ORGANIZATION"]
"node2"[label=",[,,,]_3_1"]
"node3"[label="the[DT,the]_4_1"]
"node4"[label="former[JJ,former]_5_1"]
"node6"[label="wife[NN,wife]_7_1"]
"node7"[label=",[,,,]_8_1"]
"node9"[label="her[PRP$,she]_10_1"]
"node10"[label="husband[NN,husband]_11_1"]
"node12"[label="in[IN,in]_14_1"]
"node13"[label="detention[NN,detention]_15_1"]
"node14"[label="in[IN,in]_16_1"]
"node16"[label=".[.,.]_19_1"]
"node17"[label="She[PRP,she]_1_2"]
"node20"[label="by[IN,by]_4_2"]
"node21"[label="their[PRP$,they]_5_2"]
"node22"[label="son[NN,son]_6_2"]
"node24"[label="and[CC,and]_9_2"]
"node26"[label=",[,,,]_12_2"]
"node27"[label="a[DT,a]_13_2"]
"node29"[label="lawmaker[NN,lawmaker]_17_2"]
"node30"[label=".[.,.]_18_2"]
"node8"[color=blue]
"node8"[label="visited[VBD,visit]_9_1, ASPECT:NONE, TENSE:PAST,
   POLARITY:POS"]
"node18"[color=blue]
"node18"[label="was[VBD,be]_2_2 accompanied[VBN,accompany]_3_2,
   ASPECT:NONE, TENSE:PAST, POLARITY:POS"]
"node1" -> "node1" [label = "nn"]
"node8" -> "node1" [label = "nsubj"]
"node6" -> "node3" [label = "det"]
"node6" -> "node4" [label = "amod"]
"node6" -> "node5" [label = "amod"]
```

```
"node1" -> "node6" [label = "appos"]
"node11" -> "node9" [label = "poss"]
"node11" -> "node10" [label = "nn"]
"node11" -> "node11" [label = "nn"]
"node8" -> "node11" [label = "dobj"]
"node11" -> "node13" [label = "prep_in"]
"node15" -> "node15" [label = "det"]
"node8" -> "node15" [label = "prep_in"]
"node18" -> "node17" [label = "nsubjpass"]
"node18" -> "node18" [label = "auxpass"]
"node23" -> "node21" [label = "poss"]
"node23" -> "node22" [label = "nn"]
"node23" -> "node23" [label = "nn"]
"node18" -> "node23" [label = "agent"]
"node25" -> "node25" [label = "nn"]
"node18" -> "node25" [label = "agent"]
"node23" -> "node25" [label = "conj_and"]
"node29" -> "node27" [label = "det"]
"node29" -> "node28" [label = "amod"]
"node29" -> "node28" [label = "nn"]
"node29" -> "node28" [label = "nn"]
"node23" -> "node29" [label = "appos"]
"node1" -> "node6" [label=StanfordCoreference, style=dotted]
"node1" -> "node9" [label=StanfordCoreference, style=dotted]
"node1" -> "node17" [label=StanfordCoreference, style=dotted]
"node23" -> "node29" [label=StanfordCoreference, style=dotted]
"node8" -> "node18" [label=BEFORE, color=blue]
}
```

B.1.2 JSON output

```
29 32
1 {"tokens":[{"id":"1", "word":"Wu", "lemma":"Wu", "POS":"NNP",
   "NER":"PERSON", "position":"1", "numSentence":"1"}, {"id":"2",
   "word":"Shu-chen", "lemma":"Shu-chen", "POS":"NNP",
   "NER":"PERSON", "position":"2", "numSentence":"1"}],
   "DESCRIPTOR":"Wu Shu-chen", "NER":"PERSON"}
5 {"tokens":[{"id":"6", "word":"first", "lemma":"first", "POS":"JJ",
   "NER":"ORDINAL", "position":"6", "numSentence":"1"}],
   "DESCRIPTOR":"first", "NER":"ORDINAL"}
11 {"tokens":[{"id":"12", "word":"Chen", "lemma":"Chen",
   "POS":"NNP", "NER":"PERSON", "position":"12",
   "numSentence":"1"}, {"id":"13", "word":"Shui-bian",
```

```
   "lemma":"Shui-bian", "POS":"NNP", "NER":"PERSON",
   "position":"13", "numSentence":"1"}], "DESCRIPTOR":"Chen
   Shui-bian", "NER":"PERSON"}
15 {"tokens":[{"id":"17", "word":"the", "lemma":"the", "POS":"DT",
   "NER":"TIME", "position":"17", "numSentence":"1"}, {"id":"18",
   "word":"morning", "lemma":"morning", "POS":"NN", "NER":"TIME",
   "position":"18", "numSentence":"1"}], "DESCRIPTOR":"the
   morning", "NER":"TIME"}
23 {"tokens":[{"id":"26", "word":"Chen", "lemma":"Chen",
   "POS":"NNP", "NER":"PERSON", "position":"7", "numSentence":"2"},
   {"id":"27", "word":"Chih-chung", "lemma":"Chih-chung",
   "POS":"NNP", "NER":"PERSON", "position":"8",
   "numSentence":"2"}], "DESCRIPTOR":"Chen Chih-chung",
   "NER":"PERSON"}
25 {"tokens":[{"id":"29", "word":"Lawrence", "lemma":"Lawrence",
   "POS":"NNP", "NER":"PERSON", "position":"10",
   "numSentence":"2"}, {"id":"30", "word":"Gao", "lemma":"Gao",
   "POS":"NNP", "NER":"PERSON", "position":"11",
   "numSentence":"2"}], "DESCRIPTOR":"Lawrence Gao", "NER":"PERSON"}
28 {"tokens":[{"id":"33", "word":"Democratic", "lemma":"democratic",
   "POS":"JJ", "NER":"ORGANIZATION", "position":"14",
   "numSentence":"2"}, {"id":"34", "word":"Progressive",
   "lemma":"Progressive", "POS":"NNP", "NER":"ORGANIZATION",
   "position":"15", "numSentence":"2"}, {"id":"35", "word":"Party",
   "lemma":"Party", "POS":"NNP", "NER":"ORGANIZATION",
   "position":"16", "numSentence":"2"}], "DESCRIPTOR":"Democratic
   Progressive Party", "NER":"ORGANIZATION"}
2 {"tokens":[{"id":"3", "word":", ", "lemma":", ", "POS":", ",
   "NER":"O", "position":"3", "numSentence":"1"}], "DESCRIPTOR":",
   "}
3 {"tokens":[{"id":"4", "word":"the", "lemma":"the", "POS":"DT",
   "NER":"O", "position":"4", "numSentence":"1"}],
   "DESCRIPTOR":"the"}
4 {"tokens":[{"id":"5", "word":"former", "lemma":"former",
   "POS":"JJ", "NER":"O", "position":"5", "numSentence":"1"}],
   "DESCRIPTOR":"former"}
6 {"tokens":[{"id":"7", "word":"wife", "lemma":"wife", "POS":"NN",
   "NER":"O", "position":"7", "numSentence":"1"}],
   "DESCRIPTOR":"wife"}
7 {"tokens":[{"id":"8", "word":", ", "lemma":", ", "POS":", ",
   "NER":"O", "position":"8", "numSentence":"1"}], "DESCRIPTOR":",
   "}
```

```
9 {"tokens":[{"id":"10", "word":"her", "lemma":"she", "POS":"PRP$",
   "NER":"O", "position":"10", "numSentence":"1"}],
   "DESCRIPTOR":"she"}
10 {"tokens":[{"id":"11", "word":"husband", "lemma":"husband",
   "POS":"NN", "NER":"O", "position":"11", "numSentence":"1"}],
   "DESCRIPTOR":"husband"}
12 {"tokens":[{"id":"14", "word":"in", "lemma":"in", "POS":"IN",
   "NER":"O", "position":"14", "numSentence":"1"}],
   "DESCRIPTOR":"in"}
13 {"tokens":[{"id":"15", "word":"detention", "lemma":"detention",
   "POS":"NN", "NER":"O", "position":"15", "numSentence":"1"}],
   "DESCRIPTOR":"detention"}
14 {"tokens":[{"id":"16", "word":"in", "lemma":"in", "POS":"IN",
   "NER":"O", "position":"16", "numSentence":"1"}],
   "DESCRIPTOR":"in"}
16 {"tokens":[{"id":"19", "word":".", "lemma":".", "POS":".",
   "NER":"O", "position":"19", "numSentence":"1"}],
   "DESCRIPTOR":"."}
17 {"tokens":[{"id":"20", "word":"She", "lemma":"she", "POS":"PRP",
   "NER":"O", "position":"1", "numSentence":"2"}],
   "DESCRIPTOR":"she"}
20 {"tokens":[{"id":"23", "word":"by", "lemma":"by", "POS":"IN",
   "NER":"O", "position":"4", "numSentence":"2"}],
   "DESCRIPTOR":"by"}
21 {"tokens":[{"id":"24", "word":"their", "lemma":"they",
   "POS":"PRP$", "NER":"O", "position":"5", "numSentence":"2"}],
   "DESCRIPTOR":"they"}
22 {"tokens":[{"id":"25", "word":"son", "lemma":"son", "POS":"NN",
   "NER":"O", "position":"6", "numSentence":"2"}],
   "DESCRIPTOR":"son"}
24 {"tokens":[{"id":"28", "word":"and", "lemma":"and", "POS":"CC",
   "NER":"O", "position":"9", "numSentence":"2"}],
   "DESCRIPTOR":"and"}
26 {"tokens":[{"id":"31", "word":", ", "lemma":", ", "POS":", ",
   "NER":"O", "position":"12", "numSentence":"2"}], "DESCRIPTOR":",
   "}
27 {"tokens":[{"id":"32", "word":"a", "lemma":"a", "POS":"DT",
   "NER":"O", "position":"13", "numSentence":"2"}],
   "DESCRIPTOR":"a"}
29 {"tokens":[{"id":"36", "word":"lawmaker", "lemma":"lawmaker",
   "POS":"NN", "NER":"O", "position":"17", "numSentence":"2"}],
   "DESCRIPTOR":"lawmaker"}
```

```
30 {"tokens":[{"id":"37", "word":".", "lemma":".", "POS":".",
   "NER":"O", "position":"18", "numSentence":"2"}],
   "DESCRIPTOR":"."}
8 {"tokens":[{"id":"9", "word":"visited", "lemma":"visit",
   "POS":"VBD", "NER":"O", "position":"9", "numSentence":"1"}],
   "DESCRIPTOR":"visit", "ASPECT":"NONE", "TENSE":"PAST",
   "POLARITY":"POS"}
18 {"tokens":[{"id":"21", "word":"was", "lemma":"be", "POS":"VBD",
   "NER":"O", "position":"2", "numSentence":"2"}, {"id":"22",
   "word":"accompanied", "lemma":"accompany", "POS":"VBN",
   "NER":"O", "position":"3", "numSentence":"2"}],
   "DESCRIPTOR":"accompany", "ASPECT":"NONE", "TENSE":"PAST",
   "POLARITY":"POS"}
1 1 {"type":"nn"}
8 1 {"type":"nsubj"}
6 3 {"type":"det"}
6 4 {"type":"amod"}
6 5 {"type":"amod"}
1 6 {"type":"appos"}
11 9 {"type":"poss"}
11 10 {"type":"nn"}
11 11 {"type":"nn"}
8 11 {"type":"dobj"}
11 13 {"type":"prep_in"}
15 15 {"type":"det"}
8 15 {"type":"prep_in"}
18 17 {"type":"nsubjpass"}
18 18 {"type":"auxpass"}
23 21 {"type":"poss"}
23 22 {"type":"nn"}
23 23 {"type":"nn"}
18 23 {"type":"agent"}
25 25 {"type":"nn"}
18 25 {"type":"agent"}
23 25 {"type":"conj_and"}
29 27 {"type":"det"}
29 28 {"type":"amod"}
29 28 {"type":"nn"}
29 28 {"type":"nn"}
23 29 {"type":"appos"}
1 6 {"type":"coreference"}
1 9 {"type":"coreference"}
```

```
1 17 {"type":"coreference"}
23 29 {"type":"coreference"}
8 18 {"type":"BEFORE"}
```

B.2 Enriched Representation

B.2.1 DOT output

```
digraph document {
"discourseReferent31" [label="Wu[NNP,Wu]_1_1
   Shu-chen[NNP,Shu-chen]_2_1  NER: PERSON  DESCRIPTOR: Wu Shu-chen
    POS: NP GENDER:[FEMALE] CLASS:WIFE ", color=green]
"discourseReferent32" [label="first[JJ,first]_6_1  NER: ORDINAL
   DESCRIPTOR: first  POS: NP ", color=green]
"discourseReferent33" [label="Chen[NNP,Chen]_12_1
   Shui-bian[NNP,Shui-bian]_13_1  NER: PERSON  DESCRIPTOR: Chen
   Shui-bian  POS: NP CLASS:HUSBAND ", color=green]
"discourseReferent34" [label="the[DT,the]_17_1
   morning[NN,morning]_18_1  NER: TIME  DESCRIPTOR: the morning
   POS: NP ", color=green]
"discourseReferent35" [label="Chen[NNP,Chen]_7_2
   Chih-chung[NNP,Chih-chung]_8_2  NER: PERSON  DESCRIPTOR: Chen
   Chih-chung  POS: NP CLASS:SON CLASS:LAWMAKER ", color=green]
"discourseReferent36" [label="Lawrence[NNP,Lawrence]_10_2
   Gao[NNP,Gao]_11_2  NER: PERSON  DESCRIPTOR: Lawrence Gao  POS:
   NP ", color=green]
"discourseReferent37" [label="Democratic[JJ,democratic]_14_2
   Progressive[NNP,Progressive]_15_2 Party[NNP,Party]_16_2  NER:
   ORGANIZATION  DESCRIPTOR: Democratic Progressive Party  POS: NP
   ", color=green]
"discourseReferent38" [label=",[,,,]_3_1  DESCRIPTOR: ,  POS: , "]
"discourseReferent39" [label="the[DT,the]_4_1  DESCRIPTOR: the  POS:
   DT "]
"discourseReferent40" [label="former[JJ,former]_5_1  DESCRIPTOR:
   former  POS: JJ "]
"discourseReferent41" [label="wife[NN,wife]_7_1  DESCRIPTOR: wife
   POS: N "]
"discourseReferent42" [label=",[,,,]_8_1  DESCRIPTOR: ,  POS: , "]
"discourseReferent43" [label="visited[VBD,visit]_9_1  DESCRIPTOR:
   visit  POS: V  TENSE: PAST  ASPECT: NONE  POLARITY: POS
   ",color=blue]
"discourseReferent45" [label="husband[NN,husband]_11_1  DESCRIPTOR:
   husband  POS: N "]
```

```
"discourseReferent46" [label="in[IN,in]_14_1  DESCRIPTOR: in  POS:
   IN "]
"discourseReferent47" [label="detention[NN,detention]_15_1
   DESCRIPTOR: detention  POS: N "]
"discourseReferent48" [label="in[IN,in]_16_1  DESCRIPTOR: in  POS:
   IN "]
"discourseReferent49" [label=".[.,.]_19_1  DESCRIPTOR: .  POS: . "]
"discourseReferent51" [label="was[VBD,be]_2_2
   accompanied[VBN,accompany]_3_2  DESCRIPTOR: accompany  POS: V
   TENSE: PAST  ASPECT: NONE  POLARITY: POS ",color=blue]
"discourseReferent53" [label="by[IN,by]_4_2  DESCRIPTOR: by  POS: IN
   "]
"discourseReferent54" [label="their[PRP$,they]_5_2  DESCRIPTOR: they
    POS: PRP$ "]
"discourseReferent55" [label="son[NN,son]_6_2  DESCRIPTOR: son  POS:
   N "]
"discourseReferent56" [label="and[CC,and]_9_2  DESCRIPTOR: and  POS:
   CC "]
"discourseReferent57" [label=",[,,,]_12_2  DESCRIPTOR: ,  POS: , "]
"discourseReferent58" [label="a[DT,a]_13_2  DESCRIPTOR: a  POS: DT "]
"discourseReferent59" [label="lawmaker[NN,lawmaker]_17_2
   DESCRIPTOR: lawmaker  POS: N "]
"discourseReferent60" [label=".[.,.]_18_2  DESCRIPTOR: .  POS: . "]
"discourseReferent33" -> "discourseReferent45" [label = "hasClass"]
"discourseReferent37" -> "discourseReferent35" [label =
   "has_lawmaker"]
"discourseReferent33" -> "discourseReferent47" [label = "prep_in"]
"discourseReferent41" -> "discourseReferent32" [label = "amod"]
"discourseReferent43" -> "discourseReferent51" [label = "BEFORE",
   color=blue]
"discourseReferent35" -> "discourseReferent59" [label = "hasClass"]
"discourseReferent31" -> "discourseReferent41" [label = "hasClass"]
"discourseReferent54" -> "discourseReferent35" [label = "has_son"]
"discourseReferent41" -> "discourseReferent40" [label = "amod"]
"discourseReferent41" -> "discourseReferent39" [label = "det"]
"discourseReferent51" -> "discourseReferent36" [label = "arg0"]
"discourseReferent43" -> "discourseReferent33" [label = "arg1"]
"discourseReferent31" -> "discourseReferent33" [label =
   "has_husband"]
"discourseReferent43" -> "discourseReferent34" [label = "prep_in"]
"discourseReferent59" -> "discourseReferent58" [label = "det"]
"discourseReferent51" -> "discourseReferent31" [label = "arg1"]
```

```
"discourseReferent43" -> "discourseReferent31" [label = "arg0"]
"discourseReferent35" -> "discourseReferent55" [label = "hasClass"]
"discourseReferent35" -> "discourseReferent36" [label = "conj_and"]
"discourseReferent51" -> "discourseReferent35" [label = "arg0"]
}
```

B.2.2 JSON output

```
27 22
31 {"1":{"tokens":[{"id":"1", "word":"Wu", "lemma":"Wu",
   "POS":"NNP", "NER":"PERSON", "position":"1", "numSentence":"1"},
   {"id":"2", "word":"Shu-chen", "lemma":"Shu-chen", "POS":"NNP",
   "NER":"PERSON", "position":"2", "numSentence":"1"}],
   "DESCRIPTOR":"Wu Shu-chen", "CLASS":["WIFE"], "NER":"PERSON"},
   "9":{"tokens":[{"id":"10", "word":"her", "lemma":"she",
   "POS":"PRP$", "NER":"O", "position":"10", "numSentence":"1"}],
   "DESCRIPTOR":"she", "GENDER":"FEMALE"},
   "17":{"tokens":[{"id":"20", "word":"She", "lemma":"she",
   "POS":"PRP", "NER":"O", "position":"1", "numSentence":"2"}],
   "DESCRIPTOR":"she", "GENDER":"FEMALE"}, "DESCRIPTOR": "Wu
   Shu-chen",  "NER": "PERSON",  "POS": "NE", "GENDER":["FEMALE"],
   "CLASS":["WIFE"]}
32 {"5":{"tokens":[{"id":"6", "word":"first", "lemma":"first",
   "POS":"JJ", "NER":"ORDINAL", "position":"6",
   "numSentence":"1"}], "DESCRIPTOR":"first", "NER":"ORDINAL"},
   "DESCRIPTOR": "first",  "NER": "ORDINAL",  "POS": "NE"}
33 {"11":{"tokens":[{"id":"12", "word":"Chen", "lemma":"Chen",
   "POS":"NNP", "NER":"PERSON", "position":"12",
   "numSentence":"1"}, {"id":"13", "word":"Shui-bian",
   "lemma":"Shui-bian", "POS":"NNP", "NER":"PERSON",
   "position":"13", "numSentence":"1"}], "DESCRIPTOR":"Chen
   Shui-bian", "CLASS":["HUSBAND"], "NER":"PERSON"}, "DESCRIPTOR":
   "Chen Shui-bian",  "NER": "PERSON",  "POS": "NE",
   "CLASS":["HUSBAND"]}
34 {"15":{"tokens":[{"id":"17", "word":"the", "lemma":"the",
   "POS":"DT", "NER":"TIME", "position":"17", "numSentence":"1"},
   {"id":"18", "word":"morning", "lemma":"morning", "POS":"NN",
   "NER":"TIME", "position":"18", "numSentence":"1"}],
   "DESCRIPTOR":"the morning", "NER":"TIME"}, "DESCRIPTOR": "the
   morning",  "NER": "TIME",  "POS": "NE"}
35 {"23":{"tokens":[{"id":"26", "word":"Chen", "lemma":"Chen",
   "POS":"NNP", "NER":"PERSON", "position":"7", "numSentence":"2"},
   {"id":"27", "word":"Chih-chung", "lemma":"Chih-chung",
```

"POS":"NNP", "NER":"PERSON", "position":"8",
"numSentence":"2"}], "DESCRIPTOR":"Chen Chih-chung",
"CLASS":["SON", "LAWMAKER"], "NER":"PERSON"}, "DESCRIPTOR":
"Chen Chih-chung", "NER": "PERSON", "POS": "NE",
"CLASS":["SON", "LAWMAKER"]}
36 {"25":{"tokens":[{"id":"29", "word":"Lawrence",
"lemma":"Lawrence", "POS":"NNP", "NER":"PERSON",
"position":"10", "numSentence":"2"}, {"id":"30", "word":"Gao",
"lemma":"Gao", "POS":"NNP", "NER":"PERSON", "position":"11",
"numSentence":"2"}], "DESCRIPTOR":"Lawrence Gao",
"NER":"PERSON"}, "DESCRIPTOR": "Lawrence Gao", "NER": "PERSON",
"POS": "NE"}
37 {"28":{"tokens":[{"id":"33", "word":"Democratic",
"lemma":"democratic", "POS":"JJ", "NER":"ORGANIZATION",
"position":"14", "numSentence":"2"}, {"id":"34",
"word":"Progressive", "lemma":"Progressive", "POS":"NNP",
"NER":"ORGANIZATION", "position":"15", "numSentence":"2"},
{"id":"35", "word":"Party", "lemma":"Party", "POS":"NNP",
"NER":"ORGANIZATION", "position":"16", "numSentence":"2"}],
"DESCRIPTOR":"Democratic Progressive Party",
"NER":"ORGANIZATION"}, "DESCRIPTOR": "Democratic Progressive
Party", "NER": "ORGANIZATION", "POS": "NE"}
38 {"2":{"tokens":[{"id":"3", "word":", ", "lemma":", ", "POS":", ",
"NER":"O", "position":"3", "numSentence":"1"}], "DESCRIPTOR":",
"}, "DESCRIPTOR": ", ", "POS": ", "}
39 {"3":{"tokens":[{"id":"4", "word":"the", "lemma":"the",
"POS":"DT", "NER":"O", "position":"4", "numSentence":"1"}],
"DESCRIPTOR":"the"}, "DESCRIPTOR": "the", "POS": "DT"}
40 {"4":{"tokens":[{"id":"5", "word":"former", "lemma":"former",
"POS":"JJ", "NER":"O", "position":"5", "numSentence":"1"}],
"DESCRIPTOR":"former"}, "DESCRIPTOR": "former", "POS": "JJ"}
41 {"6":{"tokens":[{"id":"7", "word":"wife", "lemma":"wife",
"POS":"NN", "NER":"O", "position":"7", "numSentence":"1"}],
"DESCRIPTOR":"wife"}, "DESCRIPTOR": "wife", "POS": "N"}
42 {"7":{"tokens":[{"id":"8", "word":", ", "lemma":", ", "POS":", ",
"NER":"O", "position":"8", "numSentence":"1"}], "DESCRIPTOR":",
"}, "DESCRIPTOR": ", ", "POS": ", "}
43 {"8":{"tokens":[{"id":"9", "word":"visited", "lemma":"visit",
"POS":"VBD", "NER":"O", "position":"9", "numSentence":"1"}],
"DESCRIPTOR":"visit", "ASPECT":"NONE", "TENSE":"PAST",
"POLARITY":"POS"}, "DESCRIPTOR": "visit", "POS": "V",
"ASPECT": "NONE", "TENSE": "PAST", "POLARITY": "POS"}

```
45 {"10":{"tokens":[{"id":"11", "word":"husband", "lemma":"husband",
   "POS":"NN", "NER":"O", "position":"11", "numSentence":"1"}],
   "DESCRIPTOR":"husband"}, "DESCRIPTOR": "husband",  "POS": "N"}
46 {"12":{"tokens":[{"id":"14", "word":"in", "lemma":"in",
   "POS":"IN", "NER":"O", "position":"14", "numSentence":"1"}],
   "DESCRIPTOR":"in"}, "DESCRIPTOR": "in",  "POS": "IN"}
47 {"13":{"tokens":[{"id":"15", "word":"detention",
   "lemma":"detention", "POS":"NN", "NER":"O", "position":"15",
   "numSentence":"1"}], "DESCRIPTOR":"detention"}, "DESCRIPTOR":
   "detention",  "POS": "N"}
48 {"14":{"tokens":[{"id":"16", "word":"in", "lemma":"in",
   "POS":"IN", "NER":"O", "position":"16", "numSentence":"1"}],
   "DESCRIPTOR":"in"}, "DESCRIPTOR": "in",  "POS": "IN"}
49 {"16":{"tokens":[{"id":"19", "word":".", "lemma":".", "POS":".",
   "NER":"O", "position":"19", "numSentence":"1"}],
   "DESCRIPTOR":"."}, "DESCRIPTOR": ".",  "POS": "."}
51 {"18":{"tokens":[{"id":"21", "word":"was", "lemma":"be",
   "POS":"VBD", "NER":"O", "position":"2", "numSentence":"2"},
   {"id":"22", "word":"accompanied", "lemma":"accompany",
   "POS":"VBN", "NER":"O", "position":"3", "numSentence":"2"}],
   "DESCRIPTOR":"accompany", "ASPECT":"NONE", "TENSE":"PAST",
   "POLARITY":"POS"}, "DESCRIPTOR": "accompany",  "POS": "V",
   "ASPECT": "NONE",  "TENSE": "PAST",  "POLARITY": "POS"}
53 {"20":{"tokens":[{"id":"23", "word":"by", "lemma":"by",
   "POS":"IN", "NER":"O", "position":"4", "numSentence":"2"}],
   "DESCRIPTOR":"by"}, "DESCRIPTOR": "by",  "POS": "IN"}
54 {"21":{"tokens":[{"id":"24", "word":"their", "lemma":"they",
   "POS":"PRP$", "NER":"O", "position":"5", "numSentence":"2"}],
   "DESCRIPTOR":"they"}, "DESCRIPTOR": "they",  "POS": "PRP$"}
55 {"22":{"tokens":[{"id":"25", "word":"son", "lemma":"son",
   "POS":"NN", "NER":"O", "position":"6", "numSentence":"2"}],
   "DESCRIPTOR":"son"}, "DESCRIPTOR": "son",  "POS": "N"}
56 {"24":{"tokens":[{"id":"28", "word":"and", "lemma":"and",
   "POS":"CC", "NER":"O", "position":"9", "numSentence":"2"}],
   "DESCRIPTOR":"and"}, "DESCRIPTOR": "and",  "POS": "CC"}
57 {"26":{"tokens":[{"id":"31", "word":", ", "lemma":", ", "POS":",
   ", "NER":"O", "position":"12", "numSentence":"2"}],
   "DESCRIPTOR":", "}, "DESCRIPTOR": ", ",  "POS": ", "}
58 {"27":{"tokens":[{"id":"32", "word":"a", "lemma":"a", "POS":"DT",
   "NER":"O", "position":"13", "numSentence":"2"}],
   "DESCRIPTOR":"a"}, "DESCRIPTOR": "a",  "POS": "DT"}
59 {"29":{"tokens":[{"id":"36", "word":"lawmaker",
```

```
   "lemma":"lawmaker", "POS":"NN", "NER":"O", "position":"17",
   "numSentence":"2"}], "DESCRIPTOR":"lawmaker"}, "DESCRIPTOR":
   "lawmaker",  "POS": "N"}
60 {"30":{"tokens":[{"id":"37", "word":".", "lemma":".", "POS":".",
   "NER":"O", "position":"18", "numSentence":"2"}],
   "DESCRIPTOR":"."}, "DESCRIPTOR": ".",  "POS": "."}
;Sentence:1
31 31 {"type":"coreference"}
41 40 {"type":"amod"}
41 39 {"type":"det"}
43 33 {"type":"arg1"}
33 45 {"type":"hasClass"}
31 33 {"type":"has_husband"}
43 34 {"type":"prep_in"}
43 31 {"type":"arg0"}
41 32 {"type":"amod"}
31 41 {"type":"hasClass"}
33 47 {"type":"prep_in"}
;Sentence:2
35 55 {"type":"hasClass"}
35 36 {"type":"conj_and"}
51 35 {"type":"arg0"}
37 35 {"type":"has_lawmaker"}
54 35 {"type":"has_son"}
59 58 {"type":"det"}
35 59 {"type":"hasClass"}
51 36 {"type":"arg0"}
51 31 {"type":"arg1"}
;DifferentSentences:
31 31 {"type":"coreference"}
43 51 {"type":"BEFORE"}
```