A constraint-based hypergraph partitioning approach to coreference resolution

Tesi Doctoral

per a optar al grau de
Doctor en Informàtica

per
Emili Sapena Masip

sota la direcció dels doctors
Lluís Padró Cirera
Jordi Turmo Borràs

Programa de Doctorat en Intel·ligència Artificial
Departament de Llenguatges i Sistemes Informàtics
Grup de Processament del Llenguatge Natural

Centre de Tecnologies i Aplicacions del Llenguatge i la Parla
Universitat Politècnica de Catalunya

Barcelona, Març 2012

Abstract

The objectives of this thesis are focused on research in machine learning for coreference resolution. Coreference resolution is a natural language processing task that consists of determining the expressions in a discourse that mention or refer to the same entity.

The main contributions of this thesis are (i) a new approach to coreference resolution based on constraint satisfaction, using a hypergraph to represent the problem and solving it by relaxation labeling; and (ii) research towards improving coreference resolution performance using world knowledge extracted from Wikipedia.

The developed approach is able to use entity-mention classification model with more expressiveness than the pair-based ones, and overcome the weaknesses of previous approaches in the state of the art such as linking contradictions, classifications without context and lack of information evaluating pairs. Furthermore, the approach allows the incorporation of new information by adding constraints, and a research has been done in order to use world knowledge to improve performances.

RelaxCor, the implementation of the approach, achieved results in the state of the art, and participated in international competitions: SemEval-2010 and CoNLL-2011. RelaxCor achieved second position in CoNLL-2011.

Contents

Chapter 1

Introduction

Managing the information contained in numerous natural language documents is gaining importance. Unlike information stored in databases, natural language documents are characterized by their unstructured nature. Sources of such unstructured information include the World Wide Web, governmental electronic repositories, news articles, blogs, and e-mails. Natural language processing is necessary for analyzing natural language resources and acquiring new knowledge.

Natural language processing (NLP) is the field of computer science and linguistics that deals with interactions between computers and humans (who use natural languages). NLP is a branch of artificial intelligence, and it is considered an AI-complete problem, given that its difficulty is equivalent to that of solving the central artificial intelligence problem, i.e., making computers as intelligent as people.

NLP tasks range from the lowest levels of linguistic analysis to the highest ones. The lowest levels of linguistic analysis include identifying words, arranging words into sentences, establishing the meaning of words, and determining how they individually combine to produce meaningful sentences. NLP tasks related to linguistic analysis include, among others, part-of-speech tagging, lemmatization, named entity recognition and classification, syntax parsing, and semantic role labeling. Typically, these tasks are based on words of isolated sentences. However, at the highest levels, many real world applications related to natural languages rely on a better comprehension of the discourse. Consider tasks such as information extraction, information retrieval, machine translation, question answering, and summarization; the higher the comprehension of the discourse, the better is their performance.

Coreference resolution is a NLP task that consists of determining the expressions in a discourse that mention or refer to the same entity. It is in the middle level of NLP, because it relies on word- and sentence-oriented analysis in order to link expressions in different sentences of a discourse that refer to

the same entities. Therefore, coreference resolution is a mandatory step in the understanding of natural language. In this sense, dealing with such a problem becomes crucial for machine translation (Peral et al., 1999), question answering (Morton, 2000), and summarization (Azzam et al., 1999), and there are examples of tasks in information extraction for which a higher comprehension of the discourse leads to better system performance.

Coreference resolution is considered a difficult and important problem, and a challenge in artificial intelligence. The knowledge required to resolve coreferences is not only lexical, morphological, and syntactic, but also semantic and pragmatic, including world knowledge and discourse coherence. Therefore, coreference resolution involves an in-depth analysis of many layers of natural language comprehension.

The objectives of this thesis are focused on research in machine learning for coreference resolution. Specifically, the research objectives are centered around the following aspects of coreference resolution:

- **Classification models.** Most common state-of-the-art classification models are based on the independent classification of pairs of mentions. More recently, models classifying several mentions at once have appeared. One of the objectives of this thesis is to incorporate the entity-mention model.
- **Problem representation.** There is no definitive representation for the problem of coreference resolution. Further research is needed in order to find more adequate coreference resolution problem representations.
- **Resolution algorithms.** Depending on the problem representation and the classification models, there can be many resolution algorithms. An objective of this thesis is to find a resolution algorithm able to handle the new classification models in the proposed problem representation.
- **Knowledge representation.** In order to manage the diverse knowledge sources employed in this problem, a symbolic and expressive representation is desirable.
- **Incorporation of world knowledge.** Some coreferences cannot be solved using only linguistic information. Often, common sense and world knowledge is essential to resolve coreferences.

The main contributions of this thesis are (i) a new approach to coreference resolution based on constraint satisfaction, using a hypergraph to represent the problem and solving it by relaxation labeling; and (ii) research towards improving coreference resolution performance using world knowledge extracted from Wikipedia. Our work contributes to each aspect of coreference resolution identified by our objectives in the following ways:

- **Classification models.** The proposed approach incorporates mention-pair and entity-mention models.

- **Problem representation.** In this thesis, we propose a representation in a hypergraph with weighted hyperedges, reducing coreference resolution to a hypergraph partitioning problem.
- **Resolution algorithms.** The proposed approach uses relaxation labeling to solve the hypergraph partitioning problem in an iterative way, which allows the incorporation of the entity-mention model. As a result of this combination of models, problem representation, and resolution algorithm, the approach overcomes the weaknesses of previous state-of-the-art approaches such as linking contradictions, classifications without context, and lack of information in evaluating pairs.
- **Knowledge representation.** This thesis motivates the use of constraints to represent knowledge.
- **Incorporation of world knowledge.** An information extraction system is developed to obtain information about entities in Wikipedia and find new coreference relations. Moreover, the entity-mention model facilitates the incorporation of constraints that take discourse coherence into account.

The structure of this dissertation is as follows. Chapter 2 introduces the concepts related to coreference resolution and presents an extended summary of the state of the art. This includes a description of the main parts of a general coreference resolution system, a brief revision of corpora and evaluation methods, details of natural language resources and knowledge sources, and a description of most of the relevant machine learning approaches to coreference resolution.

In Chapter 3, we define our proposed approach. Chapter 4 explains the details of the implementation and training methods, while our experiments and error analysis are described in Chapter 5. Chapter 6 describes our approach to incorporating world knowledge in order to improve coreference resolution performance. This chapter also includes experiments and a detailed error analysis. Finally, we present our conclusions in Chapter 7, and give a list of publications in Appendix A.

Chapter 2

Coreference resolution: state of the art

Coreference resolution is a NLP task that consists of determining which *mentions* in a discourse refer to the same entity. A **mention** is a *referring expression* that has an entity as a *referent*. By referring expression, we mean noun phrases (NPs), named entities (NEs), embedded nouns, and pronouns[1] whose meaning as a whole is a reference to an entity in the real world, and that entity is what we call the *referent*.

Coreference chains or **entities** are groups of referring expressions that have the same referent. Thus, a coreference chain is formed by all mentions in a discourse that refer to the same real entity. Given an arbitrary text as input, the goal of a coreference resolution system is to find all the coreference chains. A **partial-entity** is a set of mentions considered coreferential during resolution.

Figure 2.1 shows some mentions in a newspaper article and their corresponding coreference chains. There are five entities, and a coreference chain for the entity *Lionel Messi* is boldfaced. The difficulty of coreference resolution lies in the variety of necessary knowledge sources. First of all, in order to identify the mentions, a **morphological** and **syntactic** analysis of the document is needed. In addition, other kinds of knowledge must be used to find the complete coreference chain. For example, knowing that the mention *"star striker Lionel Messi"* refers to a person (**semantic** knowledge), we expect that this person can be mentioned using their first name or surname separately. With this knowledge, a system can add mentions of *"Messi"* to the coreference chain due to their **lexical** correspondence. Continuing with this example, **world knowledge** is essential if one wishes to add *"the young Argentine"* to the coreference chain. The acquisition and correct combination of such knowledge is what makes coreference resolution so difficult.

[1] All the pronouns with the exception of pleonastic and interrogative ones

[[FC Barcelona]$_0$ president Joan Laporta]$_1$ has warned [Chelsea]$_2$ off [**star striker Lionel Messi**]$_3$.

Aware of [[Chelsea]$_2$ owner Roman Abramovich]$_4$'s interest in [**the young Argentine**]$_3$, [Laporta]$_1$ said last night: "[I]$_1$ will answer as always, [**Messi**]$_3$ is not for sale and [we]$_0$ do not want to let [**him**]$_3$ go."

Figure 2.1: Example of coreference resolution. All mentions are annotated with a subscript indicating their coreference chain and with a different color. Boldfaced mentions refer to the entity Lionel Messi

Regarding the utility of coreference resolution, it is important to note that many real world applications related to natural language rely on it. Consider tasks such as machine translation (Peral et al., 1999), question answering (Morton, 2000), and summarization (Azzam et al., 1999). The higher their comprehension of the discourse, the better such systems will perform. Note also that the resolution of coreferences in a discourse is a mandatory step in understanding it.

Coreference, as a linguistic phenomenon, is a relation between two mentions. There are two main classes of coreference relation:

- **Direct:** identity (*Mike W. Smith* ⇔ *Smith, M.*), synonymy (*baby* ⇔ *infant*), generalization and specialization (*car* ⇔ *vehicle*).
- **Indirect:** (aka associative or bridging): part-of (*wheel* ⇔ *car*), set membership (*Ringo Starr* ⇔ *The Beatles*)

This thesis focuses on direct coreference. Readers interested in indirect coreferences can consult (Clark, 1977; Poesio et al., 1997; Asher and Lascarides, 1998; Poesio et al., 2004a).

Another consideration is the scope of coreferences. Note that coreference can occur across documents. Diverse documents, even in different languages, may mention the same real-world entity. For example, news items talking about the same politician, or Internet websites referring to the same famous artist. This kind of coreference is called cross-document coreference, while that restricted to the same document is called intra-document coreference:

- **Intra-document:** Mentions in the same document.
- **Cross-document:** Mentions in diverse documents.

Regarding the language, most of the scientific advances in this area are focused on English documents, especially because of the availability of annotated corpora. There are several published papers on resolving coreferences in

many languages, but most of the advances in algorithms, models, and knowledge sources are first developed for English texts. This is why most of the works referenced in this document focus on the resolution of coreferences or anaphora in English documents, although there are a few exceptions: Japanese (Aone and Bennett, 1995), German (Müller et al., 2002), Dutch (Hoste, 2005), Spanish and Catalan (Recasens et al., 2010a; Màrquez et al., 2012), and also Chinese and Arabic, which are included in ACE (NIST, 2003). The target of this document is not exclusively coreference resolution in English.

This thesis is a study in **direct intra-document coreference resolution**, which we call *coreference resolution* from now on for simplicity. This chapter covers the state of the art in this field, and is divided into the following sections:

1. Architecture. A description of the main parts of a general coreference resolution system.

2. Framework. A revision of corpora and evaluation methods used by most of the coreference resolution systems.

3. Knowledge sources. A description of the NLP resources and knowledge used by coreference resolution systems.

4. Approaches. An extended summary of most of the relevant machine learning approaches to coreference resolution.

5. Conclusions.

2.1 Architecture of coreference resolution systems

A coreference resolution system receives plain text as input, and returns the same text with coreference annotations as output. This section describes the architecture of a generic coreference resolution system. Each different part is introduced here, and their issues and difficulties are explained. Most existing coreference resolution systems can be considered instances of this general process, which consists of three main steps.

The first step is the detection of mentions, where text processing is needed in order to find the boundaries of the mentions in the input text. Next, in the second step, the identified mentions are characterized by gathering all the available knowledge about them and their possible compatibility. Finally, the resolution itself is performed in the third step (see Figure 2.2). The following subsections describe each of these steps.

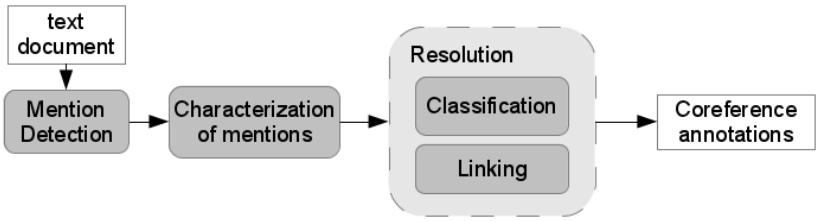

Figure 2.2: Architecture of a coreference resolution system.

The [**Technical University of Catalonia**], sometimes called [**UPC-Barcelona Tech**], is [**the largest engineering university in [Catalonia], [Spain]**]. [**The objectives of [the UPC]**] are based on internationalization, as [**it**] is [[**Spain]'s technical university with the highest number of international PhD students**] and [[**Spain]'s university with the highest number of international master's degree students**]. The [**UPC-Barcelona Tech**] is [**a university aiming at achieving the highest degree of engineering excellence**] and has [**bilateral agreements with several top-ranked European universities**].

Figure 2.3: Mentions detected in a text from Wikipedia.

2.1.1 Mention detection

The first step in resolving coreferences is to identify the mentions in the input text. As explained in the introduction, a mention is a referential NP, which also includes NEs and pronouns, with the exception of pleonastic and interrogative ones. Detecting NPs, NEs, and non-interrogative pronouns is relatively simple nowadays, given that the NLP research in these areas has left many tool kits and algorithms available for these tasks (Mitkov, 2005).

Figure 2.3 shows an example of detecting mentions in a text extracted from Wikipedia's definition of UPC. The difficulties of mention detection mostly reside in these three problems:

1 The distinction of referential NPs from non-referential ones.

2 Repetition of some nested NPs pointing to the same referent.

3 The lack of agreement in annotating mention boundaries.

The distinction of referential NPs from non-referential ones. A referential NP is one that refers to an entity, such as *Technical University of Catalonia* and *Spain's technical university with the highest number of international PhD students*. However, many NPs do not point to any entity, for example *the highest number of international PhD students*. A detailed discussion about the difficulties of this point is beyond the scope of this document, given that some

linguistic background knowledge is needed, but note that the common problem is the interpretation of the discourse above the linguistic mechanisms that form the phrases and sentences. Roughly speaking, an automatic system may detect names, named entities, pronouns, and even their syntactic function in a sentence. So, detecting NPs is *easy*. However, it is not easy to correctly identify when a noun phrase is referential.

Repetition of some nested NPs pointing to the same referent. Regarding the repetition of mentions, a set of criteria must be determined and followed during mention detection to overcome this situation. For instance, the first mention in the example in Figure 2.3 is *Technical University of Catalonia*, but annotating the whole NP including the determinant *The* is also correct. Both share the same head and are the same mention, so just one should be kept. The recommended choice is to retain the largest one in order to have more information for further steps, but either option should be considered correct. In this example, keeping the word *The* inside the mention boundaries allows the system to know that the mention is a definite NP.

The lack of agreement in annotating mention boundaries and in the interpretation of some linguistic phenomena, even between human annotators. This point is a consequence of the poor definition of mention boundaries. Mention annotations of some corpora may differ from others. This problem is related more with experiments, scoring, and contests than with the task itself, but causes the systems to need specific adaptation to the corpora to achieve regular performance. For instance, some annotators consider that two referring expressions in apposition should be included in the same mention: *[UPC, the technical university]*, whereas others annotate them as two independent mentions: *[UPC], [the technical university]*.

Given these difficulties, and the necessity of comparing the performance of different approaches in the resolution step, some authors use **true mentions**. Using true mentions means that the system is given the annotated mention boundaries as part of the input, and thus skips the mention detection step.

There is some discussion about the use of true mentions in state-of-the-art systems. The main reason for their use is clear: in order to evaluate and compare the performance of a part of a system—the resolution—the use of a common input is mandatory. Of course, analyzing the text document, detecting referential NPs, and filtering the undesirable ones according to the corpus annotation guidelines is a difficult task, and should not be trivialized. However, comparing the performance of black boxes including preprocessing, identification of mentions, and resolution does not provide enough information to determine when one resolution algorithm—or a set of features, or a training process—performs better than others. On the other hand, some authors argue that performance using true mentions leads to a rather unrealistic evaluation, given that determining whether an NP is part of an annotated coreference chain is precisely the job of a coreference resolver (Stoyanov et al., 2009; Cai and Strube, 2010b).

2.1.2 Characterization of mentions

The goal of the characterization step is to obtain as much information as possible about the mentions and the compatibility between them. Depending on the information required for further resolution, several natural language processes can be applied here. In this way, the resolution system will know, for example, the gender, number, and syntactic function of each mention. Section 2.3 describes the main knowledge sources used in state-of-the-art systems.

2.1.3 Resolution

The actual coreference resolution is performed in the resolution step. A generalization of the inner architecture of the resolution step is difficult given the diversity of approaches and algorithms used for resolution. Even so, the diverse approaches in current systems have at least two main processes in the resolution: **classification** and **linking**.

- **Classification.** This process evaluates the compatibility of elements in order to corefer. The elements can be mentions or partial entities. A typical implementation is a binary classifier that assigns class CO (coreferential) or NC (not coreferential) to a pair of mentions. It is also very typical to use confidence values or probabilities associated with the class. Classifiers can also use rankers and constraints.
- **Linking.** The linking process links mentions and partial entities in order to form the final entities. This process may range from a simple heuristic, such as single-link, to an elaborate algorithm such as clustering or graph partitioning. The input of the linking process includes the output of the classification process: classes and probabilities.

Approaches to resolution have been classified into three paradigms, depending on the use of the classification and linking processes:

- **Backward search** approaches classify mentions with previous ones, looking for the best antecedents. In this case, the linking step is typically an heuristic that links mention pairs classified as positive (single-link).
- **Two-step** approaches perform the resolution in two separate steps. The first step is to classify all of the elements, and then the second step is a linking process using algorithms such as graph partitioning or clustering to optimize the results given the classification output.
- **One-step** approaches directly run the linking process while classification is performed online.

Section 2.4.2 describes these in more detail.

2.2 Framework: corpora and evaluation

This section reviews the annotated corpora and state-of-the-art measures that are most commonly used for coreference resolution. First, we introduce the MUC, ACE, OntoNotes, and AnCora-CO corpora, and then describe the most popular metrics: MUC scorer, ACE value, B^3, CEAF, and BLANC. Finally, we give a brief summary of international competitions in the field of coreference resolution.

2.2.1 Corpora

MUC The Message Understanding Conferences (MUC) were initiated in 1987 by DARPA (Grishman and Sundheim, 1996; MUC, 1998) as competitions in information extraction. The goal was to encourage the development of new and better methods for many tasks related to information extraction. Many research teams competed against one another, and coreference resolution was included in the competition in MUC-6 (1995) and MUC-7 (1997). Annotated corpora in English for coreference are copyrighted by the Linguistic Data Consortium[2].

MUC-6 used 30 text documents with 4381 mentions for training, and another 30 documents with 4565 mentions for testing. MUC-7 consisted of 30 text documents with 5270 mentions for training, and 20 documents with 3558 mentions for testing.

ACE Automatic Content Extraction (ACE)[3] is a program that supports the automatic processing of human language in text form (NIST, 2003). Promoted by the National Institute of Standards and Technology (NIST), it was originally devoted to the three source types of newswires, broadcast news (with text derived from ASR), and newspapers (with text derived from OCR). The most recent versions of ACE may have different source types. In addition, texts are available in Chinese, Arabic, and English.

ACE annotations include information about the entities (for instance, their semantic class) and their relations that is used in other fields of information extraction. There are many ACE corpora, dating from 2002 until the present, and each one has a different size. The corpus is commonly divided into three parts according to documents of diverse nature: Broadcast News (bnews), Newspaper (npaper), and Newswire (nwire). Each of these parts is further divided into training and devtest sets. Documents in npaper are, on average, larger than the others. While an npaper document has between 200 and 300 mentions, a document in bnews or nwire has about 100 mentions.

[2] http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC2001T02
[3] http://www.nist.gov/speech/tests/ace/

MUC and ACE differences The main differences between MUC and ACE, according to (Stoyanov et al., 2009), are found in three different levels: syntactic, semantic, and task understanding, and are described as follows.

- First, at the syntactic level, the MUC annotated mentions do not include nested named entities, such as "Washington" in the named entity "University of Washington," relative pronouns, and gerunds, but do allow nested nouns. On the contrary, ACE annotations include gerunds and relative pronouns, but exclude nested nouns that are not themselves NPs, and allow some geopolitical nested named entities such as "U.S." in "U.S. officials."
- Second, ACE restricts mentions to a limited set of semantic classes: person, organization, geopolitical, location, facility, vehicle, and weapon. MUC has no limitations on entity semantic classes.
- And third, MUC does not include **singletons**. A singleton is a mention not coreferring to any other in the document. For instance, the named entity "Barcelona" in a document is annotated as a mention only if there is another mention referring to the same city, such as another occurrence of "Barcelona" or "the city."

OntoNotes The OntoNotes project has created a corpus of large-scale, accurate, and integrated annotations of multiple levels of the shallow semantic structure in text. The idea is that this rich, integrated annotation covering many linguistic layers will allow for richer, cross-layer models enabling significantly better automatic semantic analysis. In addition to coreferences, this data is also tagged with syntactic trees, high-coverage verbs, and some noun propositions, verb and noun word senses, and 18 named entity types (Pradhan et al., 2007). Moreover, OntoNotes 2.0 was used in SemEval Task 1 (Recasens et al., 2010b) and OntoNotes 4.0 (the fourth version of annotations) has been used in the CoNLL shared task on coreference resolution (Pradhan et al., 2011).

The English corpora annotated with all the layers contains about 1.3M words. It comprises 450,000 words from newswires, 150,000 from magazine articles, 200,000 from broadcast news, 200,000 from broadcast conversations, and 200,000 web data. Note that this corpus is considerably larger than MUC and ACE.

AnCora-CO AnCora-CO (Recasens and Martí, 2009) is a corpora in Catalan and Spanish that contains coreference annotations of entities composed of pronouns and full noun phrases (including named entities), plus several annotation layers of syntactic and semantic information: lemmas, parts-of-speech, morphological features, dependency parsing, named entities, predicates, and semantic roles. Most of these annotation layers are dually provided as *gold standard* and *predicted*, i.e., manually annotated versus predicted by automatic linguistic analyzers. The coreference annotation also includes singletons. AnCora-CO was

used in SemEval Shared Task 1: Coreference resolution in multiple languages (Recasens et al., 2010b). The size of AnCora-CO is about 350,000 words of Catalan and a similar quantity in Spanish.

Other corpora and languages The most widely used state-of-the-art corpora are in English. However, the availability of corpora in other languages is increasing. Figure 2.4 summarizes some of the annotated corpora for coreference resolution in other languages. Some of these were used in SemEval-2010.

Name and reference	Languages
AnCora-CO (Recasens and Martí, 2009)	Catalan, Spanish
TuBa-D/Z (Hinrichs et al., 2005)	German
PoCos (Stede, 2004)	German
KNACK-2002 (Hoste and De Pauw, 2006)	Dutch
COREA (Bouma et al., 2005)	Dutch
AnATAr (Hammami et al., 2005)	Arabic
PDT (Kucová and Hajicová, 2005)	Czech
(Sasaki et al., 2002)	Japanese, Kilivila
Ontonotes (Pradhan et al., 2007)	Arabic, Chinese
Live Memories [publication pending]	Italian

Figure 2.4: Other coreference annotated corpora.

2.2.2 Evaluation Measures

Automatic evaluation measures are crucial for coreference system development and comparison. Unfortunately, there is no agreement at present on a standard measure for coreference resolution evaluation. In this section, the most widely used metrics are explained. First, there are two metrics associated with international coreference resolution contests: the MUC scorer (Vilain et al., 1995) and the ACE value (NIST, 2003). Second, two commonly used measures, B^3 (Bagga and Baldwin, 1998a) and CEAF (Luo, 2005a), are presented. Finally, we also introduce a recently developed measure BLANC (Recasens and Hovy, 2011). B^3 and CEAF are *mention-based*, whereas MUC and BLANC are *link-based*.

The following describes in more detail what each measure quantifies as well as its strengths and weaknesses. In evaluating the output produced by a coreference resolution system, we need to compare the true set of entities (the **key** or **key partition**, i.e., the manually annotated entities) with the predicted set of entities (the **response** or **response partition**, i.e., the entities output by a system). Entities are viewed as sets of mentions. The **cardinality** of an entity is the number of mentions it contains. The MUC, B^3, and CEAF results are expressed in terms of precision (P), recall (R), and F_1, which is defined as the harmonic mean between precision and recall: $F_1 = 2*P*R/(P+R)$.

Bob$_1$ is planning to go out today. **He**$_2$ called **Charlie**$_3$ to go to **the beach**$_4$. However, **Charlie**$_5$ didn't answer **his**$_6$ call because **he**$_7$ was already at **the beach**$_8$.
Key chains: $\{1, 2, 6\}_{Bob}$, $\{3, 5, 7\}_{Charlie}$, $\{4, 8\}_{beach}$
System1 chains: $\{1, 2, 6, 7\}_{Bob}$, $\{3, 5\}_{Charlie}$, $\{4, 8\}_{beach}$
System2 chains: $\{1, 2, 3, 5, 6, 7\}_{Bob/Charlie}$, $\{4, 8\}_{beach}$

Figure 2.5: An example of coreference resolution with two system responses.

The MUC scoring algorithm This was first introduced by the MUC-6 evaluation campaign in 1995. It operates by comparing the entities defined by the links in the key and the response. First, the common links between key and response links are counted. In this context, a link corresponds to the coreferential relation between mentions. For example, a coreference chain with three mentions in the key $\{m_a,m_b,m_c\}$ has two links. More generally, a coreference chain of n mentions has $n-1$ links, which is the minimum number of links needed to create a partition. Any response chain including k common mentions of the key chain has $k-1$ common links with the key. Therefore, a response chain such as $\{m_a,m_b\}$ or $\{m_a,m_c\}$ has one link in common with the key. The link precision is the number of common links divided by the number of response links, while the recall is the number of common links divided by the number of key links, as shown in Equation 2.1.

$$Precision = \frac{Common\ links}{Response\ links}\ ;\ Recall = \frac{Common\ links}{Key\ links} \tag{2.1}$$

As has been observed (Bagga and Baldwin, 1998b; Luo, 2005b), the MUC measure is severely flawed for two main reasons. First, it is too lenient with entities containing wrong mentions: classifying one mention into a wrong entity counts as one precision and one recall error, while completely merging two entities counts as a single recall error. This can easily result in higher F-scores for worse systems. It has been pointed out (Finkel and Manning, 2008) that if all the mentions in each document of the MUC test sets were linked into one single entity, the MUC measure would give a higher score than any published system. Second, given that it only takes into account coreference links, the MUCmeasure ignores correctly clustered singletons. It is only when a singleton mention is incorrectly linked to another mention that the precision decreases. For this reason, this measure is not a good choice when working with data sets that, unlike the MUC corpora (Section 2.2.1), are annotated with singletons.

Consequently, a system that performs much better than others in terms of human understanding might be given the same or even a worse score using the MUC scorer. This is depicted in the example of Figure 2.5. From a human point of view, *System1*, which detects the three coreference chains but erroneously includes mention 7 (he) in Bob's chain, would be considered quite good, because it only fails on one pronoun and seems to "understand" that there are two people. However, *System2*, a system that joins Bob and Charlie in the same

chain, would not be considered good by a human.

Recall scores are 4/5 for *System1* and 5/5 for *System2*, because *System1* misses the links of mention 7, but *System2* finds all the links. Precision scores are 4/5 and 5/6, respectively. Consequently, the MUC scorer determines that the first system (which is good for a human) has an F-measure of 80.0%, while the second one (which is bad for a human) obtains 90.9%.

Despite its unintuitive results in some cases, the MUC scorer is the most widely used scoring algorithm in state-of-the-art coreference resolution for at least two reasons. First, the MUC corpora and MUC scorer were the first available systems, and second, the newer metrics may not be convincing enough for researchers.

The ACE value This is the scoring algorithm used to evaluate the ACE task (NIST, 2003). Each error found in the response has an associated error cost. An error can be a false alarm (a mention that is included in the response but not in the key), a miss (the opposite), or a misclassification of a coreference chain. The error cost depends on the type of entity (e.g., PERSON, LOCATION, ORGANIZATION) and on the kind of mention (e.g., NAME, NOMINAL, PRONOUN). The final error cost is the sum of the costs of all the errors made, and is normalized against the error cost that would result from a system with an unannotated output. The final score is given by subtracting the normalized cost from 1, and is often stated as a percentage.

A perfect response obtains a score of 100%, but note that a score of 0% is not the worst possible outcome. The worst response is that obtained by a system in which no coreference chains are identified. The score of a system can be negative. The interpretation of the score is that a system with a better score than another has made fewer or less important errors. However, it is important to emphasize that an ACE value of, for example, 85% does not mean that the system performs correctly for 85% of coreferences. It means that this system's error cost is 15% of the error cost of a system with an unannotated output.

The ACE value is used in several state-of-the-art works. However, as the cost is entity-type and mention-type dependent, an annotated corpus requires not only coreference chains, but also entity-types and mention-types. Of course, the ACE corpus has these annotations, but not many others do. For this reason, the use of this measure is decreasing.

B-CUBED (B^3) The B^3 measure was developed in response to the shortcomings of MUC. It shifts the attention from links to mentions by computing a precision and recall for each mention, and then taking the weighted average of these individual precision and recall scores. For a mention m_i, the individual precision represents how many mentions in the response entity of m_i corefer. The individual recall represents how many mentions in the key entity of m_i are output as coreferent.

The formula for recall for a given mention m_i is stated in (2.2), and that for precision is given in (2.3), where R_{m_i} is the response entity of mention m_i, and K_{m_i} is the key entity of mention m_i. Their cardinality is the number of mentions.

$$B^3 \text{ Recall}(m_i) = \frac{|R_{m_i} \cap K_{m_i}|}{|K_{m_i}|} \tag{2.2}$$

$$B^3 \text{ Precision}(m_i) = \frac{|R_{m_i} \cap K_{m_i}|}{|R_{m_i}|} \tag{2.3}$$

The final precision and recall are computed by averaging these scores over all the mentions.

However, this measure has also been criticized. Luo (2005b) considers that B^3 can give counter-intuitive results due to the fact that an entity can be used more than once when computing the intersection of the key and response partitions. Besides, Recasens and Hovy (2011) point out another weakness. When working with corpora where all entities are annotated and singletons appear in large numbers, scores rapidly approach 100%. More seriously, outputting all the mentions as singletons obtains a score that is close to some state-of-the-art performances.

Constrained Entity-Alignment F-Measure (CEAF) CEAF (Luo, 2005b) was proposed to solve the problem of reusing entities in B^3. It finds the best one-to-one mapping between the entities in the key and response, i.e., each response entity is aligned with at most one key entity. The best alignment is the one maximizing the total entity similarity (denoted as $\Phi(g^*)$), and this is found using the Kuhn–Munkres algorithm. Two similarity functions for comparing two entities are suggested, resulting in the mention-based CEAF and the entity-based CEAF that use (2.4) and (2.5), respectively.

$$\phi_3(K_i, R_i) = |K_i \cap R_i| \tag{2.4}$$

$$\phi_4(K_i, R_i) = \frac{2|K_i \cap R_i|}{|K_i| + |R_i|} \tag{2.5}$$

The mention-based CEAF is the more widely used. It corresponds to the number of common mentions between every two aligned entities divided by the total number of mentions. When the key and response have the same number of mentions, recall and precision are the same. On the basis of the best alignment, they are computed according to (2.6) and (2.7).

$$\text{CEAF Recall} = \frac{\Phi(g^*)}{\sum_i \phi(K_i, K_i)} \tag{2.6}$$

$$\text{CEAF Precision} = \frac{\Phi(g^*)}{\sum_i \phi(R_i, R_i)} \tag{2.7}$$

Again, CEAF is not free of criticism. It suffers from the singleton problem, just as B^3 does, which accounts for the fact that B^3 and CEAF usually get

higher scores than MUC on corpora where singletons are annotated, such as ACE, because a great percentage of the score is simply due to the resolution of singletons. In addition, the entity alignment of CEAF might cause a correct coreference link to be ignored if that entity finds no alignment in the key (Denis and Baldridge, 2009). Finally, all entities are weighted equally, irrespective of the number of mentions they contain (Stoyanov et al., 2009), so that creating a wrong entity composed of two small entities is penalized to the same degree as creating a wrong entity composed of a small and a large entity.

Bilateral Assessment of Noun-phrase Coreference (BLANC) The main motivation behind the BLANC measure is to account for the imbalance between singleton and coreferent mentions. To this end, it returns to the idea of links, but with a fundamental difference with respect to MUC: it considers both aspects of the problem, namely not only coreference links but also non-coreference links (i.e., those that hold between every two mentions that do not corefer). The sum of the two remains constant across the key and response. Although this is an idea that comes from the Rand index (Rand, 1971), BLANC puts equal emphasis on each type of link by computing the precision and recall separately for coreference and non-coreference links, and then averaging the two precision or recall scores to obtain the final score. This is shown in (2.8) and (2.9), where rc is the number of right coreference links, wc is the number of wrong coreference links, rn is the number of right non-coreference links, and wn is the number of wrong non-coreference links. Finally, the BLANC score averages the F-score for coreference links and the F-score for non-coreference links.

$$\text{BLANC Recall} = \frac{rc}{2(rc+wn)} + \frac{rn}{2(rn+wc)} \tag{2.8}$$

$$\text{BLANC Precision} = \frac{rc}{2(rc+wc)} + \frac{rn}{2(rn+wn)} \tag{2.9}$$

Four simple variations are defined for those cases when either the key or the response partition contains only singletons or a single entity. Unlike B^3 and CEAF, a coreference resolution system has to get precision and recall for *both* coreferences and non-coreferences correct simultaneously to score well under BLANC. Although this is a very new measure, and has not yet undergone extensive testing, its main weakness is revealed in the unlikely scenario of a document that consists of singletons except for one two-mention entity, as BLANC would penalize a system that outputs all the mentions as singletons too severely.

Evaluation based on system mentions An issue that has been discussed by various authors (Bengtson and Roth, 2008; Stoyanov et al., 2009; Rahman and Ng, 2009; Cai and Strube, 2010b) is the assumption made by B^3, CEAF, and BLANC that the mention set in the key partition is the same as the mention set in the response partition. Arguably, end-to-end systems may output some mentions that do not map onto any true mention, or vice versa, some true mentions may not map onto any system mention. These are called **twinless** mentions by Stoyanov et al. (2009). To handle twinless mentions, the above measures have been implemented with a few tweaks.

Bengtson and Roth (2008) simply discard twinless mentions, while Stoyanov et al. (2009) suggest two variants of B^3: ${B^3}_0$ and ${B^3}_{all}$. The former discards twinless system mentions and sets recall$(m_i) = 0$ if m_i is a twinless true mention; the latter retains twinless system mentions, and sets precision$(m_i) = \frac{1}{|R_{m_i}|}$ if m_i is a twinless system mention, and recall$(m_i) = \frac{1}{|K_{m_i}|}$ if m_i is a twinless true mention. Another adjustment for both B^3 and CEAF is proposed by Rahman and Ng (2009): remove only those twinless system mentions that are singletons, as they argue that in these cases the system should not be penalized for mentions that it has successfully identified as singletons. Recently, Cai and Strube (2010b) have pointed out several outputs that are not properly evaluated by any of the above approaches. To deal with system mentions more successfully, they present two variants of B^3 and CEAF with the following modifications:

1 Insert twinless true mentions into the response partition as singletons.

2 Remove twinless system mentions that are resolved as singletons.

3 Insert twinless system mentions that are resolved as coreferent into the key partition (as singletons).

At a closer look, it appears that the two variants introduced by Cai and Strube (2010b) can be regarded as adjustments of the key and response partitions, rather than variants of the evaluation measures themselves. By adjusting the two partitions, each true mention can be aligned to a system mention, so that both the key and response partitions have the same number of mentions, and systems are neither unfairly favored nor unfairly penalized. Màrquez et al. (2012) realized that the three adjustments by Cai and Strube (2010b) for B^3 and CEAF make it possible to apply any coreference evaluation measure. The adjustments of Màrquez et al. (2012) were adopted by the CoNLL-2011 shared task as the official scorer (Pradhan et al., 2011).

Breaking down by mention classes. In order to have a more detailed study about the results of coreference resolution systems, mentions can be grouped into meaningful classes according to their morphology and their relation to the mentions in the same coreference chain. The measures explained in this section can be modified in order to evaluate the results by mention class. The list of classes is described in Figure 2.6. They follow the ideas from Stoyanov et al. (2009) for English, and are extended by Màrquez et al. (2012) in order to evaluate some issues of Spanish and Catalan. Given that Catalan and Spanish pronouns are always gendered, the P_3U class is not relevant to them. Although these scores by class of Màrquez et al. (2012) are similar to the MUC-RC scores of Stoyanov et al. (2009), a variant of MUC, Màrquez et al. (2012) do not start from the assumption that all the coreferent mentions not belonging to the class under analysis are resolved correctly. Moreover, these scores by class can be used in any measure, not just in MUC scorer.

Short Name	Description
PN_E	NPs headed by a Proper Name that match Exactly (excluding case and the determiner) at least one preceding mention in the same coreference chain
PN_P	NPs headed by a Proper Name that match Partially (i.e., head match or overlap, excluding case) at least one preceding mention in the same coreference chain
PN_N	NPs headed by a Proper Name that do Not match any preceding mention in the same coreference chain
CN_E	Same definitions as in PN_E, PN_P and PN_N,
CN_P	but referring to NPs headed by a Common Noun
CN_N	
P_1∪2	First- and second-person pronouns that corefer with a preceding mention
P_3G	Gendered third-person pronouns that corefer with a preceding mention
P_3U	Ungendered third-person pronouns that corefer with a preceding mention
P_ELL	Elliptical pronominal subjects that corefer with a preceding mention
P_REL	Relative pronouns that corefer with a preceding mention

Figure 2.6: Description of mention classes for English, Catalan, and Spanish.

2.2.3 Shared Tasks

Over the last fifteen years, various competitions have been run to promote research in the field of coreference resolution. The first competition of this kind was MUC, which in its sixth edition (MUC-6, 1995) added a coreference resolution task. The experiment was repeated in the seventh and final edition (MUC-7, 1997). Later, a coreference resolution task was added to ACE from 2002 to the most current competitions. After a few years without competition in this area, nowadays there is a new wave of interest thanks to the SemEval-2010[4] (Recasens et al., 2010b; Recasens et al., 2009) and CoNLL-2011[5] (Pradhan et al., 2011) tasks. These last two tasks incorporate all known measures (except ACE-value) and have much larger corpora. In addition, the corpora and participants' output can be downloaded for future comparison.

2.3 Knowledge sources

This section summarizes the knowledge sources typically used by coreference resolution systems. The following subsections explain the knowledge required by the systems and the most common representation of this knowledge: the feature functions.

2.3.1 Generic NLP knowledge

In order to gather as much linguistic information as possible about the input text, a coreference resolution system first uses a set of language analyzers in

[4]SemEval-2010 Task 1 website: http://stel.ub.edu/semeval2010-coref
[5]CoNLL-2011 Shared Task website: http://conll.bbn.com

many linguistic layers, but mainly lexical and morpho-syntactic. This information is acquired before any resolution process is done. Therefore, this step is typically referred to as preprocessing. Preprocessing pipelines usually perform a tokenization (given a plain text, separate the words and punctuation symbols), lemmatization, sentence splitting, part-of-speech tagging (determining the morphological function of each word, such as determiner, noun, adjective, etc.), named entity recognition, and chunking (detecting nominal phrases), or a more elaborate parsing like dependency or constituent parsing (syntactic function of each phrase and its components). The use of semantic knowledge such as semantic roles and word sense disambiguation (resolving polysemy) are also carried out in many systems. We refer the reader to Mitkov (2005) for a deeper explanation of each of these computational linguistic issues.

The incorporation of more advanced knowledge than morphosyntax is one of the current lines of research for coreference resolution. The use of semantic knowledge, such as semantic classes and semantic role labeling, is increasingly important as more resources become available.

An example of the incorporation of semantic information is Ji et al. (2005). The authors added semantic relations to their system to refine the decisions taken by a mention-pair classifier. Once a first classification has been completed, a search for semantic relations between pairs of mentions classified as non-coreferential is performed. This improves recall by recovering missed links. The semantic relations are, for example, Employment/Membership: "Mr. Smith, a *senior programmer* at *Microsoft*."

A widely used resource is WordNet (Miller, 1995), a lexical database for English and many other languages. This groups words into sets of synonyms called *synsets*, and these synsets are connected by various semantic relations. Some of these relations are:

- Hypernyms: "canine" is a hypernym of "dog," also known as IS-A relation ("dog" is a "canine").
- Hyponyms: ("dog" is a hyponym of "canine").
- Coordinate terms (sharing a hypernym, "wolf" is a coordinate term of "dog," and vice versa).
- Holonym ("building" is a holonym of "window").
- Meronym ("window" is a meronym of "building").

WordNet can be used in many different ways, but what many systems do is to employ measures of similarity or distance. A measure of similarity means that in some way the system returns a value; the larger the value, the more similar are the meanings of the terms being compared. In contrast, the distance is a value that represents how different the terms are, where a distance of zero implies they have the same meaning. For instance, "table" and "furniture" have a smaller distance value than "table" and "professor."

Many WordNet similarities/distances use different relations, such as hypernymy, synonymy, and hyponymy, to evaluate the similarity between two synsets (Ponzetto and Strube, 2006; Rahman and Ng, 2011a). For example, a possible distance between two synsets is the number of synsets in the shortest path between them, according to hypernymous relations only.

2.3.2 Specialized knowledge

In addition to generic NLP knowledge, there is other information that is especially useful for the coreference resolution task, e.g., the structure of the document and the position of the mentions inside it. Two mentions are more likely to corefer if they are in apposition, i.e., if one mention is separated by a comma from the other: "*Michael Jackson, the youngest of the Jackson Five...*" Other structural information includes distances, such as the distance in words, sentences, and paragraphs.

Regarding the lexical level, there is also information that may help in finding coreferring mentions. For instance, string matching via variations (case sensitive, taking off the articles, etc.) and aliases (acronyms, abbreviations, nicknames, etc.) are often used.

Information about the structure of the discourse, such as rhetorical and discourse coherence, may also be of interest in resolving coreferences. Centering (Grosz et al., 1983), a theory about discourse coherence, is based on the idea that the intention of the speaker/writer is to keep the main entity in focus, which entails the use of referring expressions. During a discourse, if the speaker wants to change the main entity (the center), he/she usually does it *softly*, by introducing the new main entity and avoiding the use of anaphoric pronouns. Therefore, when looking for the antecedents of pronouns, it is plausible to assume that the center is the same as in previous sentences. When the center changes, it has to be easily inferred by the listener/reader in a coherent discourse. Therefore, in order to make it clear that the center is changing, a set of clues (e.g., using definite NPs referring to the new center entity) are usually provided.

Many studies have been done using centering and focusing (Sidner, 1979)—which is another theory based on the same linguistic and cognitive phenomena—for the problem of anaphora resolution. Originally, the use of centering and focusing theories for anaphora resolution provided a search scope that was limited to entities in the immediately preceding sentences. Both theories tackle focus/center changes between sentences, but ignore some intra-sentential issues. In addition, antecedents beyond the immediately previous sentence are ignored. Consequently, alternative models for centering (Brennan et al., 1987; Hahn and Strube, 1997; Walker et al., 1998; Strube, 1998; Poesio et al., 2004b) and focusing (Carter, 1986; Azzam, 1996) extend the search space to handle intra-sentential anaphora and distant antecedents. However, the application of these theories to coreference resolution has not yet been properly exploited.

Finally, it is also important to consider world knowledge in order to achieve coreference resolution. Ontologies, databases, and the use of resources like Wikipedia are attracting the interest of many researchers. Relations between mentions such as "*Michael Jackson*" - "*the King of Pop*," "*Barack Obama*" - "*President*," or "*Messi*" - "*the football player*" require world knowledge to be solved.

2.3.3 Knowledge representation: feature functions

Generally, machine learning systems introduce knowledge by means of feature functions. A feature function evaluates a set of mentions by some criterion. For example, a feature function determines the gender of a mention, or determines if a set of mentions agree in their gender. The values returned by the set of feature functions of the system forms the data set that the system uses in the resolution step (see Section 2.1.3). This subsection describes the most commonly used feature functions and the advances made in incorporating new feature functions that use semantic and world knowledge.

List of feature functions

In the following, there are some figures with the names and descriptions of the feature functions typically used in coreference resolution systems. In this case, the functions evaluate one or two mentions, but many of them can be generalized for groups of mentions. For example, `GENDER` evaluates the compatibility of the gender of two mentions. A positive result (y) is returned if (the heads of) both mentions have the same gender, a negative result (n) is returned if their gender is different, and an unknown (u) is returned when the gender of one or both of the mentions cannot be determined. This function can easily be generalized to compare partial entities (groups of mentions) with mentions. An explanation of the data models of mention pairs, rankers, and entity-mentions can be found in Section 2.4.1.

The feature functions described here are divided into groups regarding their level of discourse comprehension, including linguistic layers. Most of them can be found in Ng and Cardie (2002b); otherwise, a citation to the relevant paper is added if the description, or the fact that they are widely used, is not enough. For the rest of this section, let $\mathbf{m} = (m_1, ..., m_n)$ be the set of mentions in a document with n mentions, and (m_i,m_j) be a pair of mentions where $i < j$. Features starting with I/J mean that the feature is repeated for each mention m_i and m_j. Note that the feature functions included here are purely indicative, and each system may use its own version with different implementations and different returned values.

Figure 2.7 shows some typical feature functions for evaluating the position of mentions in the document or discourse. The distances are useful in cases where other information is affected by the distance between the mentions. For

Distance and position:
- DIST_TOK: Distance between m_i and m_j in tokens: number
- DIST_MEN: Distance between m_i and m_j in mentions: number
- DIST_SEN: Distance between m_i and m_j in sentences: number
- DIST_PAR: Distance between m_i and m_j in paragraphs: number
- APPOSITIVE: One mention is in apposition to the other: y, n
- I/J_IN_QUOTES: $m_{i/j}$ is in quotes or inside a NP or a sentence in quotes: y, n
- I/J_FIRST: $m_{i/j}$ is the first mention in the sentence: y, n

Figure 2.7: Positional feature functions.

instance, two mentions that are named entities with exactly the same name could corefer independently of the distance between them. However, the gender agreement of a pronoun with a proper name is only useful in the same or previous sentence, and not for larger distances. In this manner, distances are useful as a complement to other feature functions but not by themselves. This happens with many other feature functions.

Lexical:
- STR_MATCH: String matching of m_i and m_j: y, n
- HEAD_MATCH: String matching of NP heads: y, n
- SUBSTR_MATCH: $m_{i/j}$ is a substring of $m_{j/i}$: y, n

Figure 2.8: Lexical feature functions.

Lexical feature functions (Figure 2.8) are those focused on the strings and characters of the mentions. Despite their simplicity, string matching and its variations is one of the most effective feature functions for finding coreferential mentions (Soon et al., 2001; Bengtson and Roth, 2008). Although they are not included in the figure, many variations of string matching can be found in state-of-the-art systems. Some works have developed different string matching functions depending on the type of mention, e.g., a string matching function that returns 'y' if the strings are the same and the mentions are not pronouns (Ng and Cardie, 2002b).

Morphological:
- NUMBER: The number of both mentions match: y, n, u
- GENDER: The gender of both mentions match: y, n, u
- AGREEMENT: Gender and number of both mentions match: y, n, u
- I/J_THIRD_PERSON: $m_{i/j}$ is third person: y, n
- I/J_REFLEXIVE: $m_{i/j}$ is a reflexive pronoun: y, n
- I/J_POSSESSIVE: $m_{i/j}$ is a possessive pronoun: y, n
- I/J_TYPE: $m_{i/j}$ is a pronoun (p), named entity (e), or nominal (n)

Figure 2.9: Morphological feature functions.

Figure 2.9 shows some typical morphological feature functions. The number and gender agreement are essential for coreference resolution. Note that the unknown value (u) is present more often at this level than at previous ones. The difficulty in obtaining the information required by the feature functions increases as the text analysis goes deeper. Some *unknowns* are produced because the information required is really not there, such as the gender and the number of the pronoun "you," while some others are affected by the accuracy of the preprocessing.

Syntactic:
- I/J_DEF_NP: $m_{i/j}$ is a definite NP: y, n
- I/J_DEM_NP: $m_{i/j}$ is a demonstrative NP: y, n
- I/J_INDEF_NP: $m_{i/j}$ is an indefinite NP: y, n
- NESTED: One mention is included in the other: y, n
- MAXIMALNP: Both mentions have the same NP parent or they are nested: y, n
- I/J_MAXIMALNP: $m_{i/j}$ is not included in any other mention: y, n
- I/J_EMBEDDED: $m_{i/j}$ is a noun and is not a maximal NP: y, n
- BINDING_POS: Condition A of binding theory: y, n
- BINDING_NEG: Conditions B and C of binding theory: y, n
- I/J_SUBJECT: $m_{i/j}$ is the subject of the sentence: y, n, u
- I/J_OBJECT: $m_{i/j}$ is the object of the sentence: y, n, u

Figure 2.10: Syntactic feature functions.

Syntax knowledge is introduced by the functions of Figure 2.10 and their possible variations. Most of them are just descriptive and not decisive, like in the case of the distances. However, they provide information that might be useful in combination with other information.

The binding feature functions refer to the Binding theory. Binding theory is part of the principles and parameters theory (Chomsky, 1981), and imposes important syntactic intra-sentential constraints on how mentions may corefer. It can be used to determine impossible antecedents of pronominal anaphors, and to assign possible antecedents to reflexive pronouns. Some of the constraints defined in this theory have been used for automatic anaphora resolution (Ingria and Stallard, 1989; Brito and Carvalho, 1989).

Binding theory interprets reflexive pronouns, pronouns, and lexical NPs, and formulates an important syntactic constraint for each case. All three constraints use the structural relation of **c-command**, which must first be introduced. Given a syntactic tree of a sentence, a node A c-commands a node B if and only if (Haegeman, 1994):

1. A does not dominate[6] B.
2. B does not dominate A.
3. The first branching node that dominates A also dominates B.

The key constraints introduced in binding theory use the c-command relation, grammatical conditions, and the concept of **local domain**. Local domain refers to an immediate context, including the current sentence, in which short-distance anaphors may occur. The three key constraints of binding theory are listed below:

- A. Reflexive pronouns: a reflexive anaphora must be c-commanded by its antecedent, and they must agree in person, gender, and number.
- B. Pronouns: a pronoun cannot refer to a c-commanding NP within the same local domain.

[6] A dominates B means that A is a parent node of B in the syntactic tree

- C. NPs: a non-pronominal NP cannot corefer with an NP that c-commands it.

Constraints B and C are particularly useful in order to discard antecedents. So, in this case, the feature function `BINDING_NEG` may be useful to discard *false positives*, i.e., pairs of mentions that seem coreferential based on other information. In contrast, `BINDING_POS` helps to determine coreferences, but its application is rare, almost anecdotal.

Versley et al. (2008) used syntactic tree kernels to represent the structural information and determine binding and other syntactic conditions that may reveal coreference patterns inside a sentence. This technique is especially useful for resolving pronouns. Other authors also used tree kernels for similar purposes (Yang et al., 2006; Iida et al., 2006).

Semantic:
- `I/J_PERSON`: $m_{i/j}$ is a person (pronoun or a known proper name): y, n
- `ANIMACY`: Animacy of both mentions match (persons, objects): y, n
- `SEMCLASS`: Semantic class of both mentions match: y, n, u
- `ALIAS`: One mention is an alias of the other: y, n, u
- `I/J_SRL_ARG`: Semantic role of $m_{i/j}$: N, 0, 1, 2, 3, 4, M, L
- `SRL_SAMEVERB`: Both mentions have a semantic role for the same verb: y, n
- `WORDNET_SIM`: Returns a similarity value between m_i and m_j using WordNet: number

Figure 2.11: Semantic feature functions.

Figure 2.11 shows some semantic feature functions. The most widely used are `SEMCLASS` and `ALIAS`. Of the many different forms of `SEMCLASS`, a simple implementation is that used in Soon et al. (2001). In this case, the semantic class of each mention can be: person, female, male, organization, location, object, time, date, or money. These are compatible with the common classes returned by named entity recognizers and classifiers (NERC). In the case of nominal mentions (mentions that are just NPs, not pronouns or named entities), the head of the mention is disambiguated in order to obtain the WordNet synset. A search is performed inside WordNet, going up over the hypernyms until one of the classes is found. The first class found is assigned to the mention. `SEMCLASS` returns 'y' if both mentions have the same class or if they are compatible. For instance, the male and female classes are compatible with the person class.

Regarding the `ALIAS` feature function, an alias is a variation used to refer to an entity without using the entire or the official name. Variations in named entity expressions are due to a multitude of reasons: use of abbreviations, different naming conventions (e.g., Name Surname and Surname, N.), misspellings, or naming variations over time (e.g., Leningrad and Saint Petersburg) (Sapena et al., 2007). This feature function can be considered an *intelligent version* of the lexical string matching functions described above. Both mentions are compared in many ways to decide when one is an alias of the other. These methods include edit distance, the alignment of similar words, looking for abbreviations and acronyms, or other methods using prefixes and suffixes. The implementation of the `ALIAS` feature function can differ considerably from one system to another, and the reason for including this feature as semantic and not lexical is because semantic information can be used to increase its accuracy.

Several measures of distance/similarity use synsets of WordNet. Ponzetto and Strube (2006) developed the two features `WN_SIMILARITY_BEST` and `WN_SIMILARITY_AVG`, which respectively return the best score of all available WordNet similarities and their average. Note that they avoid disambiguation by scoring all possible synsets of m_i versus all possible synsets of m_j.

Ng (2007) studied the incorporation of shallow semantic features, proposing a set of features such as "semantic agreement," "semantic ACE class," and "semantic similarity." The semantic agreement feature is similar to those based on WordNet, but the aim is to avoid common disambiguation errors when senses are assigned to nouns or NPs. Ng looked for nouns in apposition with named entities, which had already been assigned a semantic class. In these cases, the sense of the noun is determined by the semantic class of the appositive NE. The second feature, "semantic ACE class," takes as its main classes those used in ACE. It considers two mentions to be semantically compatible if and only if both mentions have a common ACE semantic class. The final feature, "semantic similarity," is similar to the most frequently used WordNet distance. However, here it incorporates previous word sense disambiguation based on nouns found around the repetitions in the document of the noun being disambiguated.

World knowledge:
- `I/J_GLOSS_CONTAINS`: true when the first paragraph of $entry_{i/j}$ contains $m_{j/i}$: y, n
- `I/J_RELATED_CONTAINS`: true when $entry_{i/j}$ links to $entry_{j/i}$: y, n
- `I/J_CATEGORIES_CONTAINS`: true when categories of $entry_{i/j}$ contain $m_{j/i}$: y, n
- `GLOSS_OVERLAP`: an overlap score between the first paragraphs of $entry_i$ and $entry_j$: number
- `WIKI_RELATEDNESS_BEST`: given several relatedness scores (following Wikipedia categories of the articles in different ways) this feature chooses the highest one: number
- `WIKI_RELATEDNESS_AVG`: the average of all relatedness scores: number

Figure 2.12: World knowledge feature functions.

Figure 2.12 shows some feature functions using world knowledge for coreference resolution. In this case, only Wikipedia[7] is used, but any source of knowledge such as ontologies, databases, or Internet searches could also be useful. For instance, Rahman and Ng (2011a) uses YAGO (Suchanek et al., 2007) and FrameNet (Baker et al., 1998). Each non-pronominal mention (m_i) is searched in Wikipedia (by querying the head lemma or the named entity), and the response's entry ($entry_i$) is assigned to it. Sometimes, a disambiguation page is found instead of a direct article. In this case, the article that is finally assigned depends on the other mention of the mention-pair in the classification process. There are also cases in which no article can be assigned to a mention. The six Wikipedia feature functions in Figure 2.12 are from Ponzetto and Strube (2006).

Feature functions selection

There is some concern about the selection of feature functions. Soon et al. (2001) and Bengtson and Roth (2008) studied the relative contribution of different feature functions. However, most researchers/users have to manually select the

[7] Wikipedia is a multilingual Web-based free-content encyclopedia: http://wikipedia.org

combination of features that achieves the best results in their systems, as machine learning-based classifiers decrease system performance considerably when noisy features are added to the system. An automatic method to discover the effectiveness of the knowledge introduced to the system is a research line of interest. Hoste (2005) used Genetic Algorithms, firstly to automate the feature selection process by performing a cross-validation on the training data, and secondly to discover which features to discard, and Ponzetto and Strube (2006) used a hill climbing process in a similar way. Except for these works, we have not found any other automatic feature selection process, or any comparative studies on the subject.

2.4 Coreference resolution approaches

This section reviews state-of-the-art approaches to coreference resolution based on machine learning. The approaches follow the architecture explained in Section 2.1, which consists of three steps: mention detection, characterization of mentions, and resolution (see Figure 2.2). This section focuses on the differences between the approaches in the resolution step.

The resolution step can be conceptually divided into two processes: **classification** and **linking**. Classification is the process that evaluates the compatibility between elements, where elements can be mentions or partial entities (groups of mentions considered coreferent during resolution). The linking process uses the classification information to place these elements in the groups that form the final entities.

The coreference resolution engines in state-of-the-art systems can be classified into three paradigms, depending on their combinations of classification and linking processes in the resolution step:

- **Backward search** approaches classify mentions with previous ones looking backward for the best antecedents. In this case, the linking step is typically an heuristic that links mention-pairs classified as positive (single-link).
- **Two-step** approaches use classification and linking processes in a pipeline. The first step is just a classification of all the pairs of mentions—including their corresponding confidence values—and the second step is a linking process using algorithms such as graph partitioning or clustering to optimize the results given the classification output.
- **One-step** approaches directly run the linking process while classification is done online. In this manner, entity-mention models can be easily incorporated.

Figure 2.13 summarizes the classification of the systems included in this survey. The second column specifies which resolution step is employed. Next,

Approach	Resolution	Classification model	Linking process	Super-vised
(Aone and Bennett, 1995) (McCarthy and Lehnert, 1995) (Soon et al., 2001) (Ponzetto and Strube, 2006) (Yang et al., 2006) (Ng and Cardie, 2002b) (Ng, 2005) (Ng, 2007) (Ji et al., 2005) (Bengtson and Roth, 2008) (Stoyanov et al., 2009) (Ng, 2009) (Uryupina, 2009)	backward search	mention pairs	heuristic	yes
(Yang et al., 2003) (Denis and Baldridge, 2008)		rankers		
(Yang et al., 2008) (Rahman and Ng, 2011b) (Luo et al., 2004) (Luo, 2007)		entity-mention	global optimization	
(Klenner and Ailloud, 2008) (Nicolae and Nicolae, 2006)	two step	mention pairs	clustering graph partitioning	
(Denis and Baldridge, 2007) (Klenner, 2007) (Finkel and Manning, 2008) (Bean et al., 2004)			global optimization	weak
(Cardie and Wagstaff, 1999) (Ng, 2008)			clustering	
(Culotta et al., 2007) (Finley and Joachims, 2005) (Cai and Strube, 2010a)	one step	entity-mention	hypergraph partitioning	yes
(Yang et al., 2004) (McCallum and Wellner, 2005)			clustering graph partitioning	
(Haghighi and Klein, 2007) (Poon and Domingos, 2008)			global optimization	weak

Figure 2.13: A classification of coreference resolution approaches in state-of-the-art systems.

the third column shows the classification model used by the system, and the fourth column identifies the algorithm followed in the linking process. Finally, the fifth column shows whether the machine learning is supervised or not. Most of the systems are based on supervised learning, but there are some works that use unsupervised learning. However, we call these "weak supervised" instead of unsupervised, given that some training data have been used in the experiments.

This section is divided into two main parts. The first explains the models available for the classification process: mention-pairs, rankers, and entity-mention. The second part explains the different algorithms and combinations of classification and linking processes used by the approaches.

2.4.1 Classification models

This section introduces the three state-of-the-art classification models: mention-pairs, rankers, and entity-mention. Each one is explained in detail, revealing their strengths and weaknesses.

Mention-pairs

Classifiers based on the mention-pair model determine whether two mentions corefer or not. To do so, a feature vector is generated for a pair of mentions using, for instance, the features listed in Section 2.3.3. Given these features as input, the classifier returns a class: CO (coreferent), or NC (not coreferent). In many cases, the classifier also returns a confidence value about the decision taken. The class and the confidence value of each evaluated pair of mentions will be taken into account by the linking process to obtain the final result.

Many systems based on the mention-pair model use decision trees for classification (Aone and Bennett, 1995; McCarthy and Lehnert, 1995; Soon et al., 2001), but many other binary classifiers can be found in state-of-the-art systems. Such classifiers include RIPPER (Ng and Cardie, 2002b), maximum entropy (Nicolae and Nicolae, 2006; Denis and Baldridge, 2007; Ji et al., 2005), TiMBL (Klenner and Ailloud, 2008), perceptrons (Bengtson and Roth, 2008), and support vector machines (Yang et al., 2006).

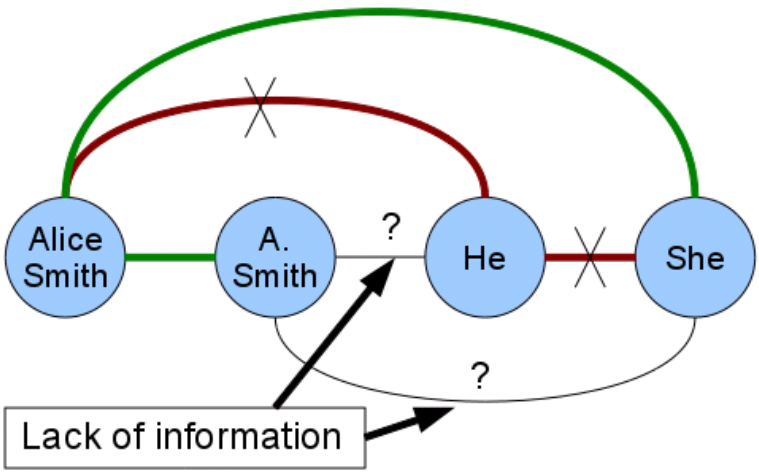

Figure 2.14: A pairwise classifier does not have enough information to classify pairs ("A. Smith," "he") and ("A. Smith," "she").

The mention-pair model has two main weaknesses: a lack of contextual information and contradictions in classifications. Figure 2.14 shows an example of lack of information. The figure is a representation of a document with four

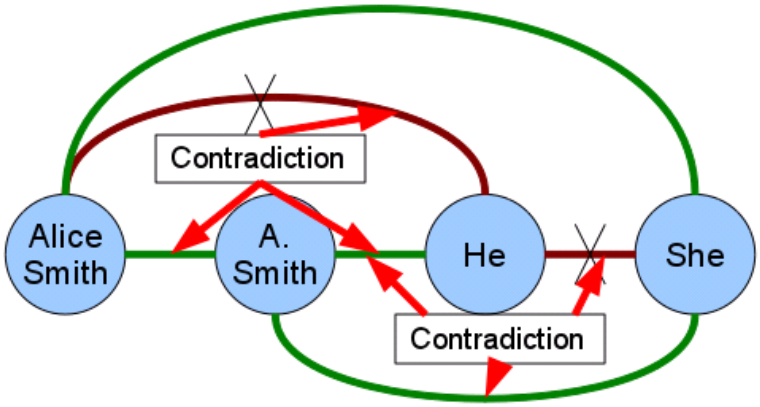

Figure 2.15: Green edges mean that both mentions corefer, and red edges mean the opposite. An independent classification of ("A. Smith," "he") and ("A. Smith," "she") produces contradictions.

mentions ("Alice Smith," "A. Smith," "he," "she"). The edges between mentions represent the classification in a mention-pair model, green means that the classifier returns the CO class, and red (also marked with an X) returns the NC class. In this case, the lack of information is due to the impossibility of determining the gender of "A. Smith." Next, Figure 2.15 shows a possible scenario with contradictions. In this scenario, the classifier has determined that the pairs ("A. Smith," "he") and ("A. Smith," "she") corefer, which causes contradictions when generating the final coreference chains given that the pairs ("Alice Smith," "he") and ("he," "she") do not corefer.

Rankers

The rankers model overcomes the lack of contextual information found using mention-pairs. Instead of directly considering whether m_i and m_j corefer, more perspective can be achieved by looking for the best candidate from a group of mentions to corefer with an active mention.

The first approach towards the rankers model was the twin-candidate model proposed by Yang et al. (2003) (motivated by Connolly et al. (1997)). The model formulates the problem as a competition between two candidates to be the antecedent of the active mention. Suppose that m_k is the active mention. The classifier must determine which of the candidates m_i and m_j would make the best antecedent. So, in this case, the output classes are not CO and NC but 1 or 2, which indicates the preferred mention between m_i and m_j to corefer with m_k. The linking process may use this information to avoid errors and contradictions.

An extension of the twin-candidate model perspective is to consider all the candidates at once, and rank them in order to find the best one (Denis and Baldridge, 2008). This method can obtain more accurate results than the twin-model due to a more appropriate context in which all the candidate mentions are considered at the same time.

Ranker models are strongly linked with backward search approaches. They select the best possible candidate to corefer with an active mention. But note, however, that a candidate is always selected, so rankers cannot determine whether an active mention forms a new chain, i.e., does not have antecedents. Consequently, using these models may require a previous process to classify mentions as anaphoric (has antecedents) or not anaphoric (the first mention of a new entity). This process is usually called an anaphoric filter, and is described in detail in Section 2.4.2.

Entity-mention

We have so far described classification models based on mentions. Even in pairwise or groupwise classifiers (i.e., mention-pair and ranker models), the active mention is always evaluated with the other mentions in the document. The main difference is the number of mentions involved in each classification. In this section, the model changes towards the concept of entity.

A partial entity is a set of mentions considered coreferent during resolution. The entity-mention model classifies a partial entity and a mention, or two partial entities, as coreferent or not. In some models, a partial entity even has its own properties or features defined in the model in order to be compared with the mentions. Due to the information that a partial entity gives to the classifier, in most cases this model overcomes the lack of information and contradiction problems of the mention-based models. For example, a partial entity may include the mentions "Alice Smith" and "A. Smith," whose genders are "female" and "unknown" respectively. In this case, the partial entity is more likely to be linked with the subsequent mention "she" than with "he" (Figures 2.14 and 2.15).

Many approaches use the entity-mention model combined with different linking processes (Yang et al., 2008; Luo et al., 2004; Lee et al., 2011; Yang et al., 2004; McCallum and Wellner, 2005). The features used for entity-mention models are almost the same as those used for mention-based models. The only difference is that the value of an entity feature is determined by considering the particular values of the mentions belonging to it.

2.4.2 Resolution approaches

Resolution approaches have been divided into three paradigms, depending on the use of the classification and linking processes. First, backward search ap-

proaches search for the best antecedent for each mention by looking backward through the document. In this case, the linking process is typically an heuristic that links the mentions classified as positive. The focus of research is on classification models and anaphoric filters. Second, some approaches use a two-step resolution process. The first step is just a static classification of all the elements, and the motivation of research in this area is to optimize the linking process. Many algorithms are used, such as graph partitioning, clustering, and integer linear programming (ILP). Finally, there are a number of one-step approaches. In this case, the linking algorithm may be similar to those used in two-step approaches, but the difference is that classification is performed online using link information already determined by the linking process. Online classification facilitates the use of the entity-mention model, given that the components of a partial entity may change during resolution.

Resolution by backward search

Many approaches to coreference resolution consider each mention of the document as an *anaphor*, and look backward until the *antecedent* is found or a stop condition is satisfied (Aone and Bennett, 1995; McCarthy and Lehnert, 1995; Soon et al., 2001; Ng and Cardie, 2002b). This behavior is inherited from anaphora resolution where, in order to resolve a pronoun or an anaphoric NP, a search is done looking backward in the document. This section describes approaches following this anaphora-styled backward search, **backward search** from now on, which includes the evolution towards ranker classification models (Yang et al., 2003; Denis and Baldridge, 2008), approaches using the entity-mention model (Yang et al., 2008; Luo et al., 2004; Rahman and Ng, 2011b), and the incorporation of anaphoric filters (Ng and Cardie, 2002a; Luo, 2007; Denis and Baldridge, 2007; Ng, 2009).

The terms *anaphor* and *antecedent* are also adopted by many authors from the anaphora resolution problem. However, using these terms in the coreference resolution context is not strictly correct. For the rest of this section, the term **active mention** is used instead of *anaphor* and **candidate mention** is used in order to refer to a *possible antecedent*. The word **antecedent** is used to refer to the candidate mention finally selected by the search algorithm.

Research into resolution based on backward searching has evolved towards the refinement of the process to select the correct antecedent. Initially, for the active mention m_j, all the mentions in reverse order of the document (i.e., from m_{j-1} until m_1) are the candidate mentions. Given the pair (m_{j-1}, m_j), the process generates a feature vector and passes it to the classifier. If the pair is classified as non-coreferential (NC), then the same process is performed for (m_{j-2}, m_j), and so on until a pair corefer (CO) or a stop condition is satisfied (Aone and Bennett, 1995; McCarthy and Lehnert, 1995; Soon et al., 2001). Generally, the stop condition is when the beginning of the document is reached, or when an arbitrary maximum distance in sentences from the active mention is exceeded.

The training set creation process of Soon et al. (2001) is representative of work based on the same idea of pairwise classifiers looking backwards. It is carried out as follows. Given a pair of mentions that is annotated as coreferential in the training corpus, the process generates several training instances. The number depends on the number of candidate mentions between the active mention and the antecedent. Specifically, if m_a and m_b are mentions annotated as coreferential ($a < b$), the pair (m_a, m_b) is a positive example, and each m_i between m_a and m_b in the document generates a pair (m_i, m_b) that is a negative example. The goal of this training data selection is to obtain a set of examples similar to those that the classifier is going to find during the resolution.

The classifier used by Soon et al. (2001), and many other studies based on the same approach, is a decision tree (DT) (typically C4.5 or C5 (Quinlan, 1993)). An example of a learned DT is shown in Figure 2.16. Each feature evaluates a pair of mentions and returns a value of *true* (t) or *false* (f). In some cases, there is also an *unknown* (u) or numeric value. The features are described in Section 2.3.3. Figure 2.16 illustrates the order followed by the DT to classify the pair. The first level of the DT depends on the value of the string matching function. If m_i and m_j match, the system classifies them as coreferential. Otherwise, the process jumps to the second level. The second level takes the decision depending on the "J_PRONOUN" feature function, which returns true when m_j is a pronoun. In this manner, the tree is followed until a final leaf is reached and the pair (m_i, m_j) is classified as CO (+) or NC (-).

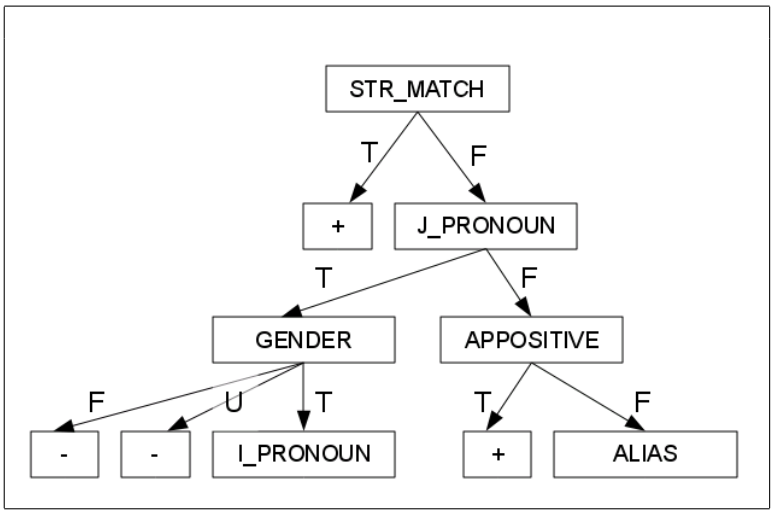

Figure 2.16: Example of a decision tree classifier.

The approach of Soon et al. (2001) achieved reasonable results on the common data sets MUC-6 and MUC-7 (62.6% and 60.4%, respectively, using the MUC scorer). This was comparable to the results of state-of-the-art non-learning systems on the same data sets[8]. Consequently, many subsequent works based on machine learning used Soon's system as a baseline.

[8] http://www-nlpir.nist.gov/related_projects/muc/proceedings/co_score_report.html

The first backward search approaches had three main weaknesses. First, the search stops suddenly when a pair is positively classified (first-link). This directly affects the precision of the system, given that the first candidate mention that agrees with the active mention is not necessarily the correct antecedent. Second, each active mention is considered an anaphor without any kind of anaphoric determination. Third, these approaches use the mention-pair model, and thus inherit the related problems of a lack of information and contradictions.

A first attempt towards solving the first-link problem is to evaluate every pair of mentions from (m_{j-1}, m_j) until (m_1, m_j), regardless of the assigned class. Once every candidate mention has been evaluated, the positively classified pair given the maximum confidence value by the classifier is the only one to be linked (Ng and Cardie, 2002b; Bengtson and Roth, 2008). In this manner, the search does not stop when a positive pair is found, and the precision of linking antecedents is improved. Approaches based on rankers and entity-mention models also overcome this problem.

The second weakness, which considers each active mention as an anaphor, is studied by anaphoric determination and is explained at the end of this section.

Regarding the lack of contextuality, a more contextual classification than pairwise classifiers is needed. Classifying each pair independently of the others does not allow the classifier to use all of the available contextual information. Rankers and entity-mention models appear to overcome this problem (see Section 2.4.1 for a description of the models).

The first approach towards a rankers model for backward search was the twin-candidate model proposed by Yang et al. (2003). This model formulates the problem as a competition between two candidates to be the antecedent of the active mention. Suppose that m_k is the active mention. The classifier decides which of the candidates m_i and m_j is the best antecedent. Each candidate mention is compared with the others in a round-robin contest. The candidate mention that wins the most comparisons is selected as the antecedent.

This methodology has two weaknesses that are mostly solved by the authors. First, if every mention before the active mention is considered as a candidate, the number of competitions to be computed is approximately the square of the number of candidates. For a large document, this computational cost might be intractable. Consequently, the model supplies a candidate filter in order to reduce the computational cost of comparing each possible pair. The filter has a window of a few sentences, and also discards the mentions that are clearly mismatched with the active mention. Second, the competition system always has a winner. Many mentions are the first mention of a new coreference chain, i.e., they are not anaphoric. The system needs some way to decide when no candidate is good enough to be considered the genuine antecedent. The solution proposed by the authors is the same filter as that used to reduce the computational cost. In cases in which the filter discards all the candidate mentions, the system decides to start a new coreference chain.

An extension of the twin-candidate model perspective is to consider all the

candidates at once, and rank them in order to find the best one (Denis and Baldridge, 2008). This method can obtain more accurate results than the twin-model, as it has a more appropriate context in which all the candidate mentions are considered at the same time. However, the scope of the rankers used in this work is also limited to a few sentences. Moreover, the system always chooses an antecedent. The solution adopted by Denis and Baldridge (2008) is a filter that discards non-anaphoric mentions. A different solution would be to find a parameter that defines the minimum cut-off value of confidence that the best candidate would need in order to be selected as the antecedent. If the first candidate in the ranking does not have a confidence value higher than the cut-off, then the active mention is considered the first of a new coreference chain.

We have so far described approaches based on mentions. In mention-pair or rankers classification models, the active mention is always evaluated with previous mentions in the document, without considering the coreferent links already determined. The main difference is the number of mentions involved in each classification. In the following, the model changes towards the concept of an entity.

The approach of Yang et al. (2008) uses the entity-mention model for backward search resolution. The main difference between this system and those described previously is that the active mention is not compared with previous mentions, but with the already-formed partial entities. Moreover, the novelty of this approach is the algorithm used for resolution. Instead of using a classifier, inductive logic programming is used in order to *reason* the entity to which the active mention should be linked. In this manner, the learning process generates a set of rules that combine first-order features with a background knowledge. An example of a rule produced by the system is:

$$\begin{array}{l} link(A,B) :- \\ \quad has\_mention(A,C), numAgree(B,C,1), \\ \quad strMatch\_Head(B,C,1), bareNP(C,1) \end{array}$$

This rule means that mention B should belong to entity A if there exists a mention C belonging to A that agrees in number with mention B, the heads of B and C satisfy string matching, and mention C is a bare NP. Each rule has an associated confidence value. During resolution, the active mention is compared with all the partial entities already formed. The applicable rule with the maximum confidence value is the one followed. In the case that no matching entity is found, the active mention starts a new entity.

A quite different approach, but one also based on the backward search paradigm, is the entity-mention model based on a Bell Tree proposed by Luo et al. (2004). A Bell Tree is a tree structure representing the process of forming entities. Each possible combination of mentions forming entities is represented by a node in the tree. The first node represents the initial state, in which the first mention m_1 forms the first partial entity (represented by [1] in Figure 2.17). At this point, the process takes into account the second mention m_2, which is

the active mention in this node (represented by 2*). There are two possibilities: (i) joining both mentions in the same entity (i.e., they corefer, represented as [12]), or (ii) creating a new entity for m_2 (represented by [1][2]). These two possibilities form the two nodes of the second layer in Figure 2.17. The process is repeated for each mention of the document, adding a new layer with all the possibilities in each step.

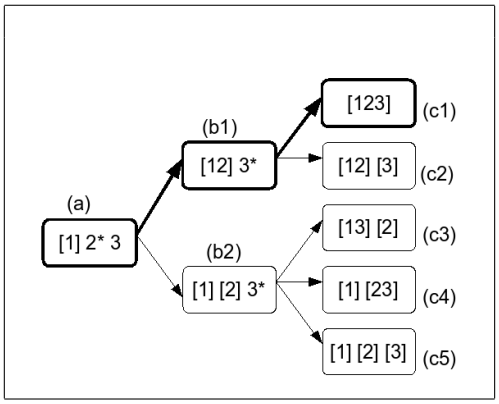

Figure 2.17: Example of a Bell Tree.

Each of the nodes created, representing a concrete formation of entities, has an associated probability, and the aim of the algorithm is to find the most probable path to the final step when the last mention is added. In order to calculate the probability of each node, the confidence values returned by a pairwise classifier are used. The original paper uses a maximum entropy classifier, which returns the probability that a partial entity and a mention corefer. The probability of a node is a combination of all the probabilities calculated along the path to that node.

In order to keep the tree size manageable, every step also performs a pruning process. Thus, the coreference chain combinations with the lowest confidence values are discarded.

Finally, Rahman and Ng (2011b) proposed what they call the *cluster-ranking* model, which is a combination of the rankers and entity-mention model in a backward search approach. In this case, the rankers order a set of partial entities as the best candidates to be antecedents of the active mention. The approach also includes an anaphoric filter.

Anaphoricity As we have seen, a common shortcoming of the backward search approaches is that every mention is initially considered an anaphor. Only in cases where no antecedent is found does the mention remain a singleton or become the first of a new coreference chain. The more candidates an active mention has, the more erroneous positive cases are possible. In order to solve this weakness, at least two solutions have been used.

The simplest solution is a limitation in the scope of the backward search. Many systems restrict the search to two or three sentences. Although most of the antecedents tend to be in the same or previous sentence as the active mention, a limitation of the search scope inevitably produces a loss of recall.

Another solution is a system for determining the anaphoricity of a mention. Thus, if a mention is considered non-anaphoric, no antecedent search will be conducted. For instance, Denis and Baldridge (2007) trained a binary classifier that determines whether single mentions are anaphoric or not. This classifier is used as a filter before the coreference classifier is applied. This kind of *anaphoric filter*, sometimes known as a *discourse new detector*, has also been applied and studied in other works (Ng and Cardie, 2002a; Uryupina, 2003; Poesio et al., 2004b; Denis and Baldridge, 2008; Ng, 2009). Its goal is to improve system precision without affecting recall. However, it does not perform as expected when used in a pipeline configuration in a backward search system. Denis and Baldridge (2007) tested why a pipeline configuration of filter+classifier caused a loss of recall. The problem was that the performance of the whole system relies heavily on the precision of the filter. On the one hand, a false positive causes a coreference chain to be joined with any other chain, forming a large one. On the other hand, a false negative causes an anaphoric mention to be considered as the beginning of a new chain, and the true coreference chain will be cut. Incidentally, an improvement in the accuracy of anaphoric filters can produce a significant improvement in the results, as has been shown in some experiments using an "oracle" system (Denis and Baldridge, 2008; Stoyanov et al., 2009; Uryupina, 2009).

Anaphoricity information can be included in the system, instead of being used as a pre-filter, to give better results (Denis and Baldridge, 2007; Finkel and Manning, 2008; Luo, 2007; Ng, 2007).

Following a similar resolution algorithm to Luo et al. (2004), Luo (2007) proposed a twin model incorporating the anaphoricity concept. The approach consists of two models. The first model evaluates the coreferentiality of the active mention with each of the partial entities that already exist. In the case that the mention clearly corefers with one of the partial entities, the mention will probably be linked to that entity. Otherwise, there is a second model that evaluates the anaphoricity of the mention. In this case, the model evaluates the probability that the mention creates a new entity. This second model compares the mention with all the entities already created at once, and uses a different set of features. The resolution process runs as follows. Given a set of N_t partial entities $E_t = (e_1, ..., e_{N_t})$ created by the t first mentions $(m_1, ..., m_t)$, the mention m_{t+1} is compared with all the N_t entities using the first model. The probability of coreferring with each one of the N_t entities is computed. The second model is then used to evaluate the probability that mention m_{t+a} creates a new entity. The Bell Tree is formed and solved, as in the previous work of Luo et al. (2004).

Ng (2009) presents another method to determine anaphoricity based on graph cuts. This system uses the information given by the coreference classifier, and at the same time assists with the coreference resolution process. Each

mention is represented as a node in a graph, and two edges connect the node with the anaphoric group and the not-anaphoric one. The weight of the edges is determined using the probabilities obtained by an anaphoricity filter and incorporates coreference probabilities. The graph is cut by taking into account the weights of the edges to find the most probable partition. The final partitioning determines which mentions are anaphoric, and this information is useful for improving the performance of the coreference resolution system.

Resolution in two steps

A different approach than the backward search is to use two separate steps for the resolution: classification and linking in a pipeline configuration, mainly focused on the second step of linking. In this case, the classification process simply evaluates the coreferentiality of each pair of mentions without taking any decision. A probability of coreference, or confidence value, attached to the classification is calculated by the classifier for pairs of mentions in the document. Finally, the linking process uses this information to find a global solution and determine the coreference chains (Klenner and Ailloud, 2008; Nicolae and Nicolae, 2006; Denis and Baldridge, 2007; Klenner, 2007; Finkel and Manning, 2008; Bean et al., 2004; Cardie and Wagstaff, 1999; Ng, 2008).

Unlike the backward seasrch approach, the resolution algorithms of two-step approaches do not depend on the order of mentions in the document. Anyway, this order is taken into account in many forms. For instance, feature functions that indicate the order of the mentions or some other constraints are frequently used. Therefore, the aim of the two-step approaches is to link the mentions into groups that optimize the global solution, given the information gathered about pairs of mentions in the classification step.

The training process of two-step approaches may change with respect to that described for backward search approaches. Depending on the order followed by the classifier and the maximum distance between mentions forming pairs, the training set includes different types of mention-pairs. Typically, two-step approaches use as many training pairs as possible.

A number of linking processes tackle coreference resolution as an optimization problem. In this manner, given the probabilities (or other types of information to evaluate coreferentiality) of the classification process, the linking process finds a global solution that maximizes them. A set of variables and an objective function are defined using the probabilities, and the resolution finds a maximization/minimization of the objective function.

The following case is an example of a global optimization process. A set of binary variables (x_{ij}) are defined to symbolize that the mentions (m_i, m_j) corefer ($x_{ij} = 1$) or not ($x_{ij} = 0$). In order to find the best solution given the probabilities of all the pairs, the following objective function is minimized:

$$\sum_{i<j} -log(P_C(i,j))x_{ij} - log(1 - P_C(i,j))(1 - x_{ij}) \tag{2.10}$$

where $P_C(i,j)$ is the probability of mentions m_i and m_j coreferring as obtained by the pairwise classifier. ILP (Klenner, 2007; Denis and Baldridge, 2007; Finkel and Manning, 2008) can be used to minimize the objective function, but any other algorithm that optimizes an objective function such as that shown above could be used.

This representation does not implicitly maintain consistency in the results. As coreference is an equivalence relation, if variables x_{ij} and x_{ik} are determined, the value of variable x_{jk} might already be determined. Specifically, when two variables have been determined to be coreferential, there is a third one that must also be coreferential. In order to keep this consistency, a set of *triangular* constraints is needed. For each three mentions m_i, m_j, m_k where $i < j < k$, the corresponding variables have to satisfy three constraints:

$$\begin{aligned} x_{ik} &\geq x_{ij} + x_{jk} - 1 \\ x_{ij} &\geq x_{ik} + x_{jk} - 1 \\ x_{jk} &\geq x_{ij} + x_{ik} - 1 \end{aligned}$$

This implies that, for a document with n mentions, the model needs $\frac{1}{2}n(n-1)$ variables and $\frac{1}{2}n(n-1)(n-2)$ constraints[9] to ensure consistency. This is an important limitation in terms of scalability.

This global optimization model can also be used to incorporate other kinds of global knowledge as well as transitivity. For example, Denis and Baldridge (2007) incorporate information about named entities and anaphoricity in order to provide more clues to the algorithm determining the start of coreference chains.

Graph partitioning is another type of linking process. Generally, it is carried out on an undirected graph in which the vertices are mentions and the edges are weighted by the confidence values (or probabilities) of the classification of the pair formed by adjacent mentions. Usually, edge weights are viewed as distances or costs, and the algorithm cuts edges whose value is above a threshold r in order to isolate the groups that represent independent entities. The cut-threshold r, or any other parameter of the graph partitioning algorithm, can be learned with training data. Figure 2.18 is an example of how the mentions "He," "A. Smith," "Alice Smith," and "She" would be represented in a graph. In this example, a resolution algorithm decides to cut the edges from mention "He," creating a coreference chain for "A. Smith," "Alice Smith," and "She." In this manner, McCallum and Wellner (2005) applied a graph partitioning algorithm

[9] $\frac{1}{6}n(n-1)(n-2)$ for each of the three triangular constraints

on a weighted, undirected graph in which vertices are mentions and edges are weighted by the pairwise score between mentions.

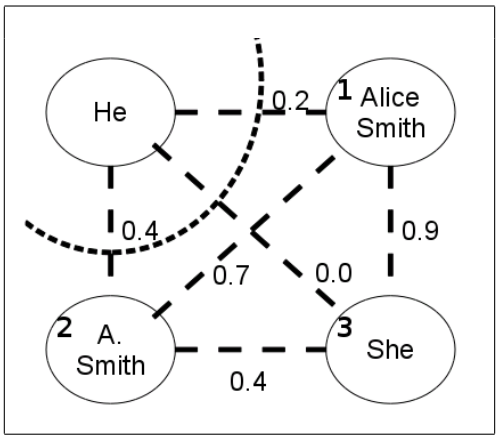

Figure 2.18: Example of graph partitioning. The weight of the edges is the coreference probability given by the classifier.

Nicolae and Nicolae (2006) presented the BestCut algorithm for graph partitioning. This represented each mention as a vertex, with edge weights assigned as the probabilities returned by a maximum entropy pairwise classifier. BestCut cuts the edges with minimum cut weight in the following manner. The weight of the cut of a graph into two subgraphs is the sum of the weights of the edges crossing the cut. This process is repeated until a stop condition, which is trained by a classifier, is satisfied. In this work, a set of graphs is used instead of a unique graph representing all the mentions in the document. In particular, the NEs and NPs are separated by their type: Person, Organization, Location, Facility, and GPE. Pronouns are not included in the process until the graph partitioning is complete.

A similar approach to graph partitioning, but based on clustering, is proposed in Klenner and Ailloud (2008). All positively (CO) classified pairs from a pairwise classifier are input to a clustering algorithm, with the confidence value of each pair classification used to calculate their cost. The clustering algorithm determines the coreference chains with the minimum cost. In this case, all mention types are included and there is no separation by semantic class, which simplifies the process.

Cardie and Wagstaff (1999) also proposed a clustering approach to coreference resolution. In this case, the approach is weakly supervised. The order of resolution follows the reverse order of the mentions found in the document, and decisions are taken greedily. For each mention, a distance is calculated with respect to all preceding mentions. This distance is defined as the weighted summation of a set of features. If the distance is lower than a cut-threshold, the mention is assigned to that cluster. Feature function weights are chosen by hand, and the cut-threshold is chosen to maximize F_1 on the development set.

Ng (2008) has proposed an expectation maximization (EM) clustering model that is unsupervised, although it is used in a weakly supervised manner in the experiments, through the use of one labeled document. A set of feature functions evaluates the compatibility of each pair of mentions, and the clustering is a squared Boolean matrix where each cell C_{ij} corresponds to the linkage of the pair of mentions (m_i, m_j). The value of C_{ij} is 1 if m_i and m_j corefer and 0 otherwise. A cluster is valid if transitivity is always satisfied. Thus, if $C_{ij} = 1$ and $C_{jk} = 1$ then C_{ik} must be 1 to satisfy transitivity. The EM process adapts the probability values of the transitive pairs in all the documents of the development data set. According to the published results, the system achieves a reasonable performance but is still not comparable to a fully supervised system.

In summary, two-step approaches can provide more coherent results than backward search systems. This is mainly because the linking process can avoid contradictions by applying transitivity, and can provide an optimized solution by comparing as many confidence values as necessary at the same time. However, the lack of contextuality in evaluating pairs separately remains. The next section describes methods that mix classification and linking processes in the same step in order to overcome this weakness.

Resolution in one step

The basic difference between one-step and two-step approaches is that classification is not a prior and independent process, but is conducted online during the linking process. The main reason for this is the use of more complex classification models, as the entity-mention model has many more possible combinations of elements than the mention-pair model. Moreover, the coreference probability associated with a set of mentions may change during resolution, given the partial results already determined. This section contains approaches based on entity-mention models (McCallum and Wellner, 2005; Culotta et al., 2007; Finley and Joachims, 2005; Yang et al., 2004; Haghighi and Klein, 2007; Poon and Domingos, 2008).

Culotta et al. (2007) uses a greedy agglomerative clustering to perform the linking process. In this case, the model is based on groups of mentions instead of pairs. Each time a new mention is considered to be included in a partial entity, the mention is compared with the whole group at the same time.

Models based on pairs give a probability that each pair of mentions corefer ($p(x_{ij}|m_i, m_j)$). The combination of probabilities that optimizes the groups is the one that the algorithm (global optimization, clustering, graph partitioning) selects as the solution. However, the lack of contextuality in evaluating each pair is propagated. The probability of a pair of mentions being coreferent according to a maximum entropy classifier is generally like Equation 2.11:

$$p(x_{ij}|m_i, m_j) = \frac{1}{Z_{m_i,m_j}} exp \sum_k \lambda_k f_k(m_i, m_j, x_{ij}) \tag{2.11}$$

where Z is a normalization factor. In this case, each feature function f_k evaluates the pair of mentions (m_i, m_j). Thus, when the resolution process has to decide to join a group of mentions, the probability is indirectly obtained using the probabilities of all the possible pairs of mentions in the group. The work of Culotta et al. (2007) proposes a set of feature functions that evaluate the compatibility of groups of mentions and a classifier that directly returns the probability that the whole group corefer, as shown in Equation 2.12:

$$p(x_j|\mathbf{m}^j) = \frac{1}{Z_{\mathbf{m}^j}} exp \sum_k \lambda_k f_k(\mathbf{m}^j, x_j) \tag{2.12}$$

where Z is a normalization factor and $\mathbf{m}^j = \{... \, m_i \, ...\}$ is a group of mentions.

Enumerating all the possible configurations in order to find the most probable one can result in an intractable combinatorial growth (de Salvo Braz et al., 2005). Consequently, a set of reductions and practical implementations have been proposed and tested, with promising results (McCallum and Wellner, 2005; Culotta et al., 2007).

Following a different line of research, Finley and Joachims (2005) developed a support vector machine (SVM) classifier in order to learn the similarity measure used to join groups of mentions of the same coreference chain. This similarity measure is used by a *correlation clustering* algorithm. The novelty of the system is that the measure is not learned by classifying pairs of mentions. The SVM learning algorithm is modified to learn a similarity measure that directly classifies groups of mentions in a set of partitions. The *loss function* used for learning is the same function as that used in the MUC scorer (explained in Section 2.2.2), which directly associates the learning process with the final task. One of the problems with this method is the impossibility of training all the possible incorrect partitions. To solve this, two approaches are proposed in order to iteratively determine the most relevant partition examples for training. The results confirm that clustering evaluation groups of mentions performs better than a pairwise classifier, but no comparable results (e.g., a MUC test using MUC scorer) have been published.

A clustering approach similar to the ones described in Section 2.4.2 was introduced by Yang et al. (2004). In this case, the classifier used the entity-mention model and compared mentions with the already-created clusters. These clusters have their own properties, such as gender, number, semantic class, and so on. Actually, each cluster is an entity. This approach found that incorporating feature functions comparing mentions with entities improved the results.

Cai and Strube (2010a) represent the problem as a hypergraph. A set of group-oriented features form hyperedges that join all the mentions covered by the feature. For example, a feature "SameGender" joins by a hyperedge all the mentions in the document that have the same gender. A graph partitioning algorithm is then executed, taking care of the weights assigned to the hyperedges. These weights are learned in the training process. The hypergraph

representation overcomes the limitations of graphs that can only represent the mention-pair model, whose edges simply connect pairs of vertices. A hyperedge connects any number of vertices, which makes the use of the entity-mention model easier.

Finally, two unsupervised learning approaches perform a global optimization without separating the processes of classification and linking (Haghighi and Klein, 2007; Poon and Domingos, 2008).

Haghighi and Klein (2007) (H&K) developed a non-parametric Bayesian approach to coreference resolution and cross-document coreference. Their model is fully generative and produces each mention from a combination of global entity properties and local attentional states. This is a new approach that has never been used before for coreference resolution and might indicate a new research line to follow. Ng (2008) has made three modifications to the H&K model in order to remedy a set of potential weaknesses and improve the results. These weaknesses reside in the application of the model for coreference resolution. For example, for the sake of simplicity, the original H&K model only compares the heads of the mentions that could corefer. Ng's improvement uses string matching, aliases, and apposition (called strong coreference indicators) in order to determine when two mentions corefer.

Poon and Domingos (2008) presented an unsupervised approach that obtains competitive results, compared with those published by supervised, state-of-the-art systems. The system incorporates knowledge using a set of first-order, handwritten hard and soft rules. The hard rules must be satisfied in order to obtain the first partial entities. The soft rules are then applied to the rest of the mentions. Each soft rule has an associated weight that helps the resolution algorithm to determine whether the mention should be added to a partial entity. The unsupervised learning finds the weights by a process of gradient descent over the probabilities. The linking algorithm is based on Markov networks.

There is an additional system worth mentioning, although it does not use machine learning. (Lee et al., 2011) describes a multi-pass sieve system that solves coreference in one step using an agglomerative algorithm and the entity-mention model. The system consists of multiple *sieves* sorted from highest to lowest precision. Each sieve contains several rules that determine when partial entities must be linked. It is similar to a rule-based system where more precise rules are applied first. Due to the entity-mention model, the rules with, a priori, less precision have more information when they are applied, because there are already half-formed entities. The system, which does not have any training method, won first place in the CoNLL-2011 Shared Task (Pradhan et al., 2011).

2.5 Conclusion

Coreference resolution research has been very active in the last decade, since the appearance of annotated corpora and the first machine learning systems devoted

to the task. Many advances have been made, due mainly to the existence of two evaluation frameworks: MUC and ACE. More recently, other resources have appeared, such as OntoNotes, and the tasks of SemEval-2010 and CoNLL-2011 have led new evaluation frameworks that clearly show there is room for improvement.

Regarding evaluation measures, we have seen that none of those developed so far are free of defects. Furthermore, link-based metrics value different aspects than those based on mentions, so much so that Semeval-2010 could not determine a task winner, as the best score with one measure would lose with another. For CoNLL, the organizers decided to use an average of three measures to determine the final ranking of the systems. It seems clear that a final measure that correctly evaluates the task is yet to be found.

Taking into account the state of the art, the immediate future of research in coreference resolution seems to be evolving in four different ways.

- First, models for supervised learning might be improved with new procedures for a better combination of the available information: entity-mention and rankers models take care of the whole group of mentions, and seem more appropriate than mention-pair approaches.
- Second, better training or instance selection methods are needed. The mention-pair model has only 1 to 6% positive examples (depending on the corpora and the precision of the mention detection step), and this percentage is even lower when using groups of mentions for training.
- Third, the addition of semantic and pragmatic knowledge to the systems should improve their final performance. It is well-known that some ambiguities at a syntactic or lexical level need world knowledge or some kind of discourse comprehension to be solved. Thus, the use of ontologies and other resources for disambiguation is a promising line of research.
- Finally, an interesting area is the research of unsupervised and weakly supervised approaches, as the scarcity of annotated corpora for training and testing is a bottleneck for research in supervised technology.

Chapter 3

A constraint-based hypergraph partitioning approach to coreference resolution

As we have seen in the summary of the state of the art (Chapter 2), one of the possible directions to follow in coreference resolution research is the incorporation of new information such as world knowledge and discourse coherence. In some cases, this information cannot be expressed in terms of pairs of mentions, i.e., it is information that involves either several mentions at once or partial entities. Therefore, an experimental approach in this field needs the expressiveness of the entity-mention model as well as the mention-pair model in order to use the most typical mention-pair features. Furthermore, such an approach should overcome the weaknesses of previous state-of-the-art approaches, such as linking contradictions, classifications without context, and a lack of information when evaluating pairs. Also, the approach would be more flexible if it could incorporate knowledge both automatically and manually.

Given these prerequisites, we define an approach based on constraint satisfaction that represents the problem in a hypergraph and solves it by relaxation labeling, reducing coreference resolution to a hypergraph partitioning problem with a given set of constraints. The main strengths of this system are:

- Modeling the problem in terms of **hypergraph partitioning** avoids linking contradictions and errors caused by a lack of information or context.
- **Constraints** are compatible with the mention-pair and entity-mention models, which let us incorporate new information. Moreover, constraints can be both automatically learned and manually written.

- **Relaxation labeling** is an iterative algorithm that performs function optimization based on local information. It first determines the entities of the mentions in which it has more confidence, mainly solving the problem of lack of information for some pairs and the lack of context. The iterative resolution facilitates the use of the entity-mention model.

The rest of this chapter describes the details of the approach. Section 3.1 describes the problem representation in a (hyper)graph. Next, Section 3.2 explains how the knowledge is represented as a set of constraints, and Section 3.3 explains how attaching *influence rules* to the constraints means that the approach incorporates the entity-mention model. Finally, Section 3.4 describes the relaxation labeling algorithm used for resolution.

3.1 Graph and hypergraph representations

The coreference resolution problem consists of a set of mentions that have to be mapped to a minimal collection of individual entities. By representing the problem in a hypergraph, we are reducing coreference resolution to a hypergraph partitioning problem. Each partition obtained in the resolution process is finally considered an entity.

The document mentions are represented as vertices in a hypergraph. Each of these vertices is connected by hyperedges to other vertices. Hyperedges are assigned a weight that indicates the confidence that adjacent mentions corefer. The larger the hyperedge weight in absolute terms, the more reliable the hyperedge. In the case of the mention-pair model, the problem is represented as a graph where edges connect pairs of vertices.

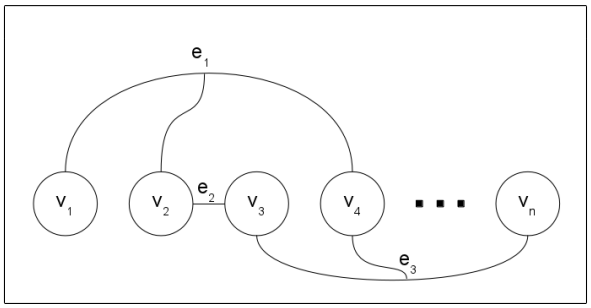

Figure 3.1: Example of a hypergraph representing the mentions of a document connected by hyperedges (entity-mention model).

Let $G = G(V, E)$ be an undirected hypergraph, where V is a set of vertices and E is a set of hyperedges. Let $\mathbf{m} = (m_1, ..., m_n)$ be the set of mentions of a document with n mentions to resolve. Each mention m_i in the document is represented as a vertex $v_i \in V$. A hyperedge $e_g \in E$ is added to the hypergraph

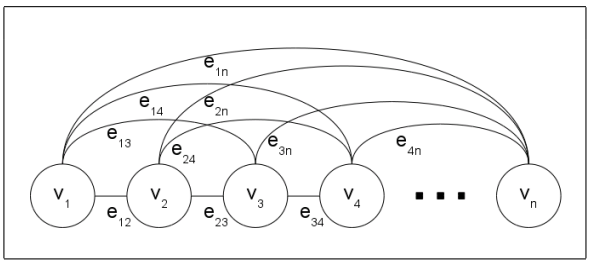

Figure 3.2: Example of a graph representing the mentions of a document connected by edges (mention-pair model).

for each group (g) of vertices $(v_0, ..., v_N)$ affected by a constraint, as shown in Figure 3.1. The subset of hyperedges that incide on v_i is $E(v_i)$.

A subset of constraints $C_g \subseteq C$ restricts the compatibility of a group of mentions. C_g is used to compute the weight value of the hyperedge e_g. Let $w(e_g) \in W$ be the weight of the hyperedge e_g:

$$w(e_g) = \sum_{k \in C_g} \lambda_k \tag{3.1}$$

where λ_k is the weight associated with constraint k.

The graph representing the mention-pair model is a subcase of the hypergraph where $|g| = 2$. Figure 3.2 illustrates a graph. For simplicity, in the case of the mention-pair model, an edge between m_i and m_j is called e_{ij}. In addition, sometimes w_{ij} is used instead of $w(e_{ij})$.

3.2 Constraints as knowledge representation

In this approach, knowledge is a set of weighted constraints where each constraint contributes a piece of information that helps to determine the coreferential relations between mentions. A constraint is a conjunction of feature-value pairs that are evaluated over all the pairs or groups of mentions in a document. When a constraint applies to a set of mentions, a corresponding hyperedge is added to the hypergraph, generating the representation of the problem explained in Section 3.1 (Figure 3.1).

Let N be the order of a constraint, i.e., the number of mentions expected by the constraint ($|g|$). A **pair constraint** has order $N = 2$, while a **group contraint** has $N > 2$. The mentions evaluated by a constraint are numbered from 0 to $N - 1$ in the order they are found in the document.

Figures 3.3 and 3.4 show examples of constraints for $N = 2$ and $N = 3$, respectively. The constraint in Figure 3.3 requires that: the distance between the mentions is just one sentence, their genders match, m_0 is not the first mention of its sentence, m_0 is a maximal NP (the next parent node in the syntactic tree is the sentence itself), m_1 also is a maximal NP, both mentions are argument 0 in semantic role labeling, and both mentions are pronouns. The constraint in Figure 3.4 applies to three mentions and requires that: the distance between consecutive mentions is one sentence, all three mentions agree in both gender and number, m_0 and m_2 are aliases, all three mentions are argument 0 in their respective sentences, and m_0 and m_2 are named entities while m_1 is a common NP[1].

DIST_SEN_1(0,1) & **GENDER_YES(0,1)** & $\overline{FIRST(0)}$ &
MAXIMALNP(0) & **MAXIMALNP(1)** &
SRL_ARG_0(0) & **SRL_ARG_0(1)** &
TYPE_P(0) & **TYPE_P(1)**

Figure 3.3: Example of a pair constraint.

DIST_SEN_1(0,1) & **DIST_SEN_1(1,2)** &
AGREEMENT_YES(0,1,2) & **ALIAS_YES(0,2)** &
SRL_ARG_0(0) & **SRL_ARG_0(1)** &
SRL_ARG_0(2) & **TYPE_E(0)** &
TYPE_S(1) & **TYPE_E(2)**

Figure 3.4: Example of a group constraint.

Each constraint has a **weight** that determines the hyperedge weight of the hypergraph (see Equation 3.1). A constraint weight is a value that, in absolute terms, reflects the confidence of the constraint. Moreover, this weight is signed, and the sign indicates whether the adjacent mentions corefer (positive) or not (negative). The use of negative information is not very extensive in state-of-the-art systems, but given the hypergraph representation of the problem, where most of the mentions are interconnected, the negative weights contribute information that cannot be obtained using only positive weights. Moreover, in our experiments, the use of negative weights accelerates the convergence of the resolution algorithm.

3.2.1 Motivation for the use of constraints

As explained in Section 3.1, the constraints and their weights finally determine the weight of the hyperedges in the hypergraph. However, there are other ways to assign weights to the edges of a graph, especially in the case of the mention-pair model where the edges connect pairs of mentions. A commonly used solution is to learn a classifier that determines whether a pair of mentions corefer or not depending on the values of the feature functions. The classifier, in addition to assigning a class, would return a confidence value for that decision. In the case of a linear SVM, for example, that value is the sum of the weights

[1] Feature functions used in our experiments are explained in detail in Section 4.2

assigned to each feature function. This confidence value could be directly used as the edge weight.

There are three main reasons to use weighted constraints instead of a classifier to determine the hyperedge weights of the hypergraph.

- Given the nature of the problem, the feature functions require a nonlinear combination. This forces the use of nonlinear classifiers, which increases the computational cost.

- Constraints are a symbolic representation of knowledge. This allows us to combine any method to generate constraints. For example, a set of constraints handwritten by linguistics experts can be added to another automatically obtained set. Moreover, symbolic knowledge is useful in situations such as error analysis.

- Constraints can be easily adapted to the entity-mention model.

Regarding the nonlinear combination of the feature functions, note that there are feature functions that are informative in themselves, such as `GENDER_YES` or `STR_MATCH`. Knowing that two mentions agree in gender (`GENDER_YES`), or that the characters of their words are exactly the same (`STR_MATCH`), provides a direct clue to the objective of determining whether these mentions corefer. The same is true with negative clues. For example, the feature function `GENDER_NO` indicates that the mentions are probably not coreferent, given that their genders do not match. A linear SVM classifier would learn a positive weight for `GENDER_YES` and `STRING_MATCH`, and a negative weight for `GENDER_NO`, so the final value would be the sum of the weights of the activated feature functions. However, there are many feature functions that provide insufficient information to determine whether it is positive or negative. For example, `I_SRL_ARG_0` indicates that, in Semantic Role Labeling, m_i serves as the argument 0. It seems that this information does not give us any clue to determine whether m_i and m_j corefer. But, combined with other information, this can be very valuable. Constraints have the expressiveness to represent such a nonlinear combination of feature values. For example, the constraint in Figure 3.3 has a precision of 95% in the OntoNotes corpus. In conclusion, a linear classifier cannot take advantage of all the knowledge offered by the feature functions. Hence, a nonlinear combination is required. Comparing constraints with nonlinear classifiers, we find that constraints offer some advantages due to their symbolic knowledge and easy adaptation to the entity-mention model, although constraints are not the only method that satisfies these requirements.

3.3 Entity-mention model using influence rules

We have explained how groups of mentions satisfying a constraint are connected by hyperedges in the hypergraph. This section explains how the entity-mention model is definitively incorporated to our constraint-based hypergraph approach. The entity-mention model takes advantage of the concept of an entity during the resolution process. This means that each mention belongs to an entity during resolution, and this information can be used to make new decisions.

In order to incorporate the entity-mention model into our approach, we define the *influence rule*, which is attached to a constraint. An **influence rule** expresses the conditions that the mentions must meet during resolution before the influence of the constraint takes effect.

An influence rule consists of two parts: condition and action.

- The **condition** of an influence rule is a conjunction of coreference relations that the mentions must satisfy before the constraint has influence. This condition is specified by joining mentions into groups, where each group represents a partial entity specified by a subscript. For instance, $(0, 1, 2)_A$ means that mentions 0, 1, and 2 are assigned to the same entity (entity A). As another example, $(0, 1)_A, (2)_B$ means that mentions 0 and 1 belong to entity A and mention 2 belongs to entity B ($A \neq B$).
- The **action** of an influence rule defines the desired coreference relation and determines which mentions are influenced. It is expressed in the same terms as the condition, specifying the mentions that are influenced and the entity to which they should belong. For instance, an action corresponding to the previous examples could be $(3)_B$. This action indicates that mention 3 is influenced in order to belong to entity B.

Constraint:
SRL_ARG_0(0) & **SRL_ARG_1(1)** & **SRL_ARG_0(2)** & **SRL_ARG_1(3)** & **DIST_SEN_0(0, 1)** & **DIST_SEN_1(1, 2)** & **DIST_SEN_0(2, 3)** & **AGREEMENT_YES(0, 2)** & **AGREEMENT_YES(1, 3)**
Influence rule: $(0, 2)_A, (1)_B \Rightarrow (3)_B$
Example:
Charlie$_0$ called **Bob**$_1$. **He**$_2$ invited **him**$_3$ to the party.

Figure 3.5: Example of an entity-mention constraint. It takes advantage of the partial entities during resolution. If mentions 0 and 2 tend to corefer, the structure indicates that mentions 1 and 3 may corefer in a different entity.

Figure 3.5 shows an example of an $N = 4$ constraint with an influence rule attached. The constraint specifies the feature functions that the involved mentions must meet, such as semantic role arguments, sentence distances, and agreements. The influence rule then determines that when mentions 0 and 2

belong to the same entity, and mention 1 belongs to a different entity, mention 3 is influenced in order to belong to the same entity as mention 1. This figure also contains some text to help understand why this kind of constraint may be useful. A mention-pair approach could easily make the mistake of classifying mentions 2 and 3 as coreferent. This is an example of introducing information about discourse coherence using an entity-mention model.

In order to retain consistency with the mention-pair model, all the constraints used in this approach are assigned a default influence rule that depends on the sign of the edge weight. In the case that the weight is positive, the last mention is influenced to belong to the same entity as the first mention, while a negative weight causes the opposite. Figure 3.6 shows the default influence rules for mention-pair constraints with both positive and negative weights.

Description	Conditions	Action
Default influence rule for a mention-pair constraint (positive weight)	$(0)_A$	$(1)_A$
Default influence rule for a mention-pair constraint (negative weight)	$(0)_A$	$(1)_B$
Example of an influence rule for an entity-mention constraint	$(0,2)_A, (1)_B$	$(3)_B$

Figure 3.6: Default influence rules for mention-pair constraints.

Note that when influence rules are used, a hyperedge is added for each subset of constraints that applies to the same group of mentions and has the same influence rule. In the case that some constraints apply to the same group of mentions but have different influence rules, a hyperedge is added to the graph for each influence rule. Therefore, in Equation 3.1, $C_g \subseteq C$ refers to the constraints that apply to the group and share the same influence rule.

3.4 Relaxation labeling

Relaxation is a generic name for a family of iterative algorithms that perform function optimization based on local information. They are closely related to neural nets and gradient steps. Relaxation labeling has been successfully used in engineering fields to solve systems of equations, in Artificial Intelligence for computer vision (Rosenfeld et al., 1976), and in many other AI problems. The algorithm has also been widely used to solve NLP problems such as part-of-speech tagging (Padró, 1998), chunking, knowledge integration, semantic parsing (Atserias, 2006), and opinion mining (Popescu and Etzioni, 2005).

Relaxation labeling (Relax) solves our weighted constraint-based hypergraph partitioning problem by dealing with (hyper)edge[2] weights as *compatibility coefficients*. In this manner, each vertex is assigned to a partition satisfying as many constraints as possible. In each step, the algorithm updates the probability of each vertex belonging to a partition. This update is performed by transferring the probabilities of adjacent vertices proportional to the edge weights.

[2]For the rest of this section, there is no distinction between edges and hyperedges

Let $V = \{v_1, v_2, \ldots, v_n\}$ be a set of variables. In our approach, each vertex (v_i) in the hypergraph is a variable in the algorithm. Let L_i be the number of different labels that are possible for v_i. The possible labels of each variable are the partitions that the vertex can be assigned. Note that the number of partitions (entities) in a document is a priori unknown, but it is at most the number of vertices (mentions) because, in an extreme case, each mention in a document could refer to a different entity. Therefore, a vertex with index i can be in the first i partitions (i.e., $L_i = i$).

The aim of the algorithm is to find a weighted labeling such that global consistency is maximized. A weighted labeling is a weight assignment for each possible label of each variable: $\mathbf{H} = (\mathbf{h}^1, \mathbf{h}^2, \ldots, \mathbf{h}^n)$, where each $\mathbf{h}^i$ is a vector containing a weight for each possible label of v_i; that is: $\mathbf{h}^i = (h^i_1, h^i_2, \ldots, h^i_{L_i})$. As relaxation is an iterative process, these weights (of between 0 and 1) vary in time. We denote the probability for label l of variable v_i at time step t as $h^i_l(t)$, or simply h^i_l when the time step is not relevant. Note that the label assigned to a variable at the end of the process is the one with the highest weight ($max(\mathbf{h}^i)$). Figure 3.7 shows an example.

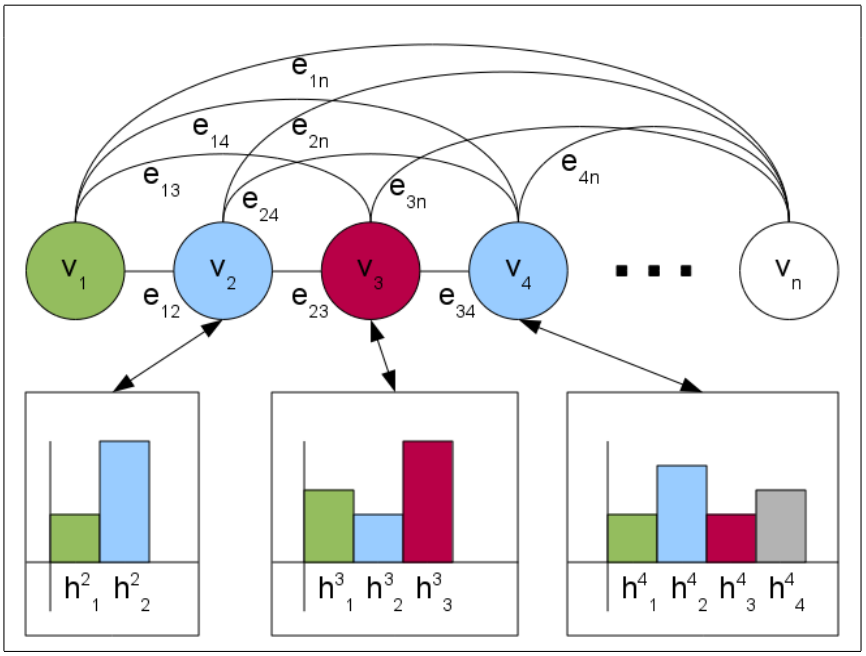

Figure 3.7: Representation of Relax solving a graph. The vertices representing mentions are connected by weighted edges e_{ij}. Each vertex has a vector h^i representing probabilities of belonging to different partitions. The figure shows h^2, h^3, and h^4. The assigned partition is the one with the highest probability. During resolution, these probability vectors are updated depending on the edge weights and the probability vectors of the adjacent vertices. The process finishes when convergence is achieved.

Maximizing global consistency is defined as maximizing the average support for each variable, which is defined as the weighted sum of the support received by each of its possible labels, that is: $\sum_{l=1}^{L_i} h^i_l \times S_{il}$, where S_{il} is the support

received by that pair from the context.

The support for a variable-label pair (S_{il}) expresses the compatibility of the assignment of label l to variable v_i compared with the labels of neighboring variables, according to the edge weights. Although several support functions may be used (Torras, 1989), we chose the following (Equation 3.2), which defines the support as the sum of the influences of the incident edges.

$$S_{il} = \sum_{e \in E(v_i)} Inf(e) \tag{3.2}$$

where $Inf(e)$ is the influence of edge e. The influence of an edge is defined by its weight and the *influence rules* attached to the constraints involved with this edge (see Section 3.3). An influence rule determines how the current probabilities for the same label of adjacent vertices (h_l^j) are combined.

Initialize:
  $H := H_0$,

Main loop:
  Repeat
    For each variable v_i
      For each possible label l for v_i
        $S_{il} = \sum_{e \in E(v_i)} Inf(e)$
      End for
      Normalize supports between -1 and 1
      For each possible label l for v_i
        $h_l^i(t+1) = \frac{h_l^i(t) \times (1+S_{il})}{\sum_{k=1}^{L_i} h_k^i(t) \times (1+S_{ik})}$
      End for
    End for
  Until no more significant changes

Figure 3.8: Relaxation labeling algorithm.

The pseudo-code for the relaxation algorithm can be found in Figure 3.8. It consists of the following steps:

1 Start with a random labeling, or with a better-informed initial state.

2 For each variable, compute the support that each label receives from the current weights of adjacent variable labels following Equation 3.2.

3 Normalize support values between -1 and 1.

4 Update the weight of each variable label according to the support obtained by each of them (i.e., increase weight for labels with support greater than zero, and decrease weight for those with support less than zero) according to the update function:

$$h_l^i(t+1) = \frac{h_l^i(t) \times (1 + S_{il})}{\sum_{l=1}^{L_i} h_l^i(t) \times (1 + S_{il})} \tag{3.3}$$

5 Iterate the process until the convergence criterion is met. The usual criterion is to wait for no more changes in an iteration, or a maximum change below some epsilon parameter. There can also be a maximum number of iterations for cases where the process does not converge.

In the following, there are some examples of the Relax implementation of the edge influences ($Inf(e)$) given the influence rules attached to the constraints.

- The simplest example is when mention m_0 has a direct influence over mention m_1. The influence rule attached to the constraint is $(0)_A \Rightarrow (1)_A$. This is determined by Equation 3.4 and is the kind of influence used in the mention-pair model.

$$Inf(e) = w(e) \times h_l^1 \tag{3.4}$$

- The next example requires that mention m_0 and mention m_1 tend to corefer during the resolution in order to influence mention m_2. The influence rule is $(0,1)_A \Rightarrow (2)_A$. In this case, the influence of the edge representing this influence rule is given by Equation 3.5. Mentions m_0 and m_1 are tending to corefer (belong to the same entity: l) when their values for label l are tending to 1 (and the other labels are tending to 0). In this case, multiplying h_l^0 and h_l^1 achieves a value close to 1, and the influence is almost the weight of the edge. In other cases when the coreference between m_0 and m_1 is not clear (or they are clearly not coreferent), at least one of the values of h_l^0 and h_l^1 is not close to 1 and the value of their product rapidly decreases, so the influence of the edge also decreases.

$$Inf(e) = w(e) \times h_l^0 \times h_l^1 \tag{3.5}$$

- Following the previous example, now suppose that in order for m_0 to influence m_2 it is required that m_1 does not belong to the same entity as m_0. In this case, h_l^1 is negated using its complementary value $(1-h_l^0)$, as is shown in Equation 3.6. The corresponding influence rule is $(0)_A, (1)_B \Rightarrow (2)_A$.

$$Inf(e) = w(e) \times h_l^0 \times (1 - h_l^1) \tag{3.6}$$

- The complexity of the influence rules can be increased arbitrarily. This last example (Equation 3.7) shows how to represent $(0,2)_A, (1)_B \Rightarrow (3)_B$, an influence rule requiring m_0 and m_2 to belong to the same entity, while m_1 belongs to a different one in order to influence m_3.

$$Inf(e) = w(e) \times h_l^1 \times (1 - h_l^0 \times h_l^2) \tag{3.7}$$

Chapter 4

RelaxCor

RelaxCor is the coreference resolution system implemented in this thesis in order to perform experiments and test the approach explained in Chapter 3. This chapter explains the implementation and training methods, before the experiments and error analysis are presented in the following chapters. RelaxCor is programmed in Perl and C++, is open source, and is available for download from our research group's website[1].

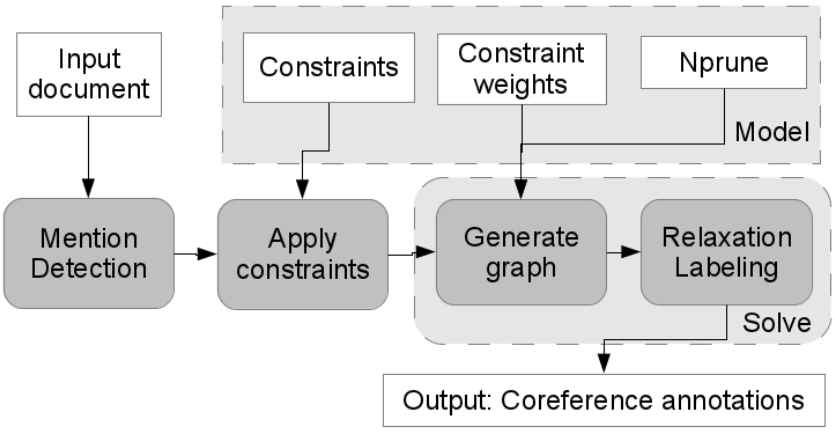

Figure 4.1: RelaxCor resolution process.

The resolution process of RelaxCor is shown in Figure 4.1. First, the mention detection system determines the mentions of the input document and their boundaries. The mention detection system is explained in Section 4.1. Alternatively, true mentions can be used when available, allowing this step

[1] http://nlp.lsi.upc.edu/web/index.php

to be skipped. Next, for each pair or group of mentions (depending on the model), the set of feature functions calculate their values, and the set of model constraints is applied. The set of feature functions used by RelaxCor and its knowledge sources are explained in Section 4.2. A (hyper)graph is then generated using the applied constraints, their weights, and the Nprune value, which is the number of edges that remain after pruning. The pruning process is justified in Section 4.3.3. Finally, relaxation labeling is executed to find the partitioning that maximizes constraint satisfaction.

Given the problem representation and the resolution algorithm, the most influential factors in the resolution are the (hyper)edge weights, which depend on the weight of the constraints. In addition, the applied constraints determine the relations between candidate mentions, and also have a big influence on the final result. Therefore, there are two main goals in the training and development process. First, we aim to find a set of constraints that reveals as many coreferent relations as possible, and second, we aim to find adequate edge weights. The training and development processes used in this work are described in Sections 4.3 and 4.4. The former explains the method for training the mention-pair model, while the latter concerns the entity-mention model.

Section 4.5 describes the set of adjustments made to the algorithm in order to improve the implementation's performance. Section 4.6 describes the adaptations required to allow the system to be executed over corpora in other languages or formats. Finally, Section 4.7 compares the system with related work in other state-of-the-art systems.

4.1 Mention detection

RelaxCor includes a mention detection system that uses part-of-speech and syntactic information. Syntactic information may be obtained from dependency parsing or constituent parsing. The system extracts one candidate mention for every:

- Noun phrase (NP).
- Pronoun.
- Named Entity (NE).
- Capitalized common noun or proper name that appears two or more times in the document. For instance, the NP "*an Internet business*" is a mention, but "*Internet*" is also added in case the word is found once again in the document.

The head of every candidate mention is then determined using part-of-speech tags and a set of rules from (Collins, 1999) when constituent parsing is used, or using dependency information otherwise. In case some NPs share the same head,

the larger NP is selected and the rest are discarded. Also, mention repetitions with exactly the same boundaries are discarded.

As a result, mention detection achieves an acceptable recall in our experiments (p. e. greater than 90% in CoNLL-2011 (Sapena et al., 2011)), but a low precision because it includes many singletons. Note that a mention detection system in a pipeline configuration acts as a filter, and the main objective at this point is to achieve as much recall as possible.

The most typical error made by the system is to include extracted NPs that are not referential (e.g., predicative and appositive phrases) and mentions with incorrect boundaries. The incorrect boundaries are mainly due to errors in the predicted syntactic column and some mention annotation discrepancies.

4.2 Knowledge sources and features

The system gathers knowledge using a set of feature functions that interpret and evaluate the input information according to some criteria. Given a set of mentions numbered from 0 to $N-1$ following the order found in the document, each feature function evaluates their compatibility in a specific aspect. RelaxCor includes features from all linguistic layers: lexical, syntactic, morphological, and semantic. Moreover, some structural features of the discourse have also been used, such as distances, quotes, and sentential positions. A feature function with only one argument indicates that it offers information about only one mention. For example, `REFLEXIVE(0)` indicates that mention 0 is a reflexive pronoun. Figures 4.2 and 4.3 show an exhaustive list of the features used and a brief description of each one.

Note that all of the feature functions are binary. The original sources that had a list of possible values have been binarized by a set of feature functions that each represent a different value. Even in numerical cases, there is a set of binary features representing the most important specific values, and the rest are placed in ranges. For instance, the distance in sentences between two mentions is represented by `DIST_SEN_0(X,Y)`, `DIST_SEN_1(X,Y)`, and `DIST_SEN_L3(X,Y)`. This means that mentions X and Y appear in the same sentence, in consecutive sentences, or at a distance of less than three sentences, respectively. We consider appearances in the same or consecutive sentences to provide valuable information, and the distance of three sentences defines the border between *near* and *far*. The use of binary features has been proven to be more efficient for the majority of machine learning algorithms. In the case of decision trees, the use of binary features favors a better performance (Rounds, 1980; Safavian and Landgrebe, 1991).

Some of the feature functions are a combination of other features, and offer redundant information. For instance, `NONPRO_STR(X,Y)` indicates that the mentions' strings match and that none of them is a pronoun. This information is also elicited by the combination of `PRONOUN(X)`=0, `PRONOUN(Y)`=0, and

STR_MATCH(X,Y)=1. However, this combined feature may help the machine learning process by simplifying the combination of features in a number of typical cases. This is a common practice in coreference resolution approaches (Ng and Cardie, 2002b; Bengtson and Roth, 2008).

There is no feature selection process. Our experiments show that a feature selection process does not significantly change system performance (see Section 5). In case a feature is not useful, it is not added to any constraint. Even where a non-useful feature is included in some constraints, those constraints will probably have a weight near to zero and will not have an impact on the resolution.

Distance and position:
Distance between X and Y in sentences:
DIST_SEN_0(X, Y): same sentence
DIST_SEN_1(X, Y): consecutive sentences
DIST_SEN_L3(X, Y): less than three sentences
Distance between X and Y in phrases:
DIST_PHR_0(X, Y), DIST_PHR_1(X, Y), DIST_PHR_L3(X, Y)
Distance between X and Y in mentions:
DIST_MEN_0(X, Y), DIST_MEN_L3(X, Y), DIST_MEN_L10(X, Y)
APPOSITIVE(X, Y): One mention is in apposition with the other
IN_QUOTES(X): X is in quotes or inside a NP
or a sentence in quotes
FIRST(X): X is the first mention in the sentence
Lexical:
STR_MATCH(X, Y): String matching of X and Y
PRO_STR(X, Y): Both are pronouns and their strings match
PN_STR(X, Y): Both are proper names and their strings match
NONPRO_STR(X, Y): String matching as in Soon et al. (2001)
and mentions are not pronouns
HEAD_MATCH(X, Y): String matching of NP heads
TERM_MATCH(X, Y): String matching of NP terms
HEAD_TERM(X): Mention head matches with the term
Morphological:
The number of both mentions match:
NUMBER_YES(X, Y,...), NUMBER_NO(X, Y), NUMBER_UN(X, Y)
The gender of both mentions match:
GENDER_YES(X, Y,...), GENDER_NO(X, Y), GENDER_UN(X, Y)
Agreement: Gender and number of all mentions match:
AGREEMENT_YES(X, Y,...), AGREEMENT_NO(X, Y), AGREEMENT_UN(X, Y)
Closest Agreement: X is the first agreement found
looking backward from Y: C_AGREEMENT_YES(X, Y),
C_AGREEMENT_NO(X, Y), C_AGREEMENT_UN(X, Y)
THIRD_PERSON(X): X is third person
PROPER_NAME(X): X is a proper name
NOUN(X): X is a common noun
ANIMACY(X, Y,...): Animacy of mentions match (person, object)
REFLEXIVE(X): X is a reflexive pronoun
POSSESSIVE(X): X is a possessive pronoun
TYPE_P/E/N(X): X is a pronoun (p), NE (e), or nominal (n)

Figure 4.2: Feature functions used by RelaxCor (1/2).

4.3 Training and development for the mention-pair model

This section describes the training and development process for the implementation of RelaxCor using the mention-pair model and the graph representation. The training process applies a machine learning algorithm over the training

Syntactic:
- DEF_NP(X): X is a definite NP
- DEM_NP(X): X is a demonstrative NP
- INDEF_NP(X): X is an indefinite NP
- NESTED(X, Y): One mention is included in the other
- SAME_MAXIMALNP(X, Y): Both mentions have the same NP parent or they are nested
- MAXIMALNP(X): X is not included in any other NP
- EMBEDDED(X): X is a noun and is not a maximal NP
- C_COMMANDS(X, Y): X c-commands Y
- BINDING_POS(X): Condition A of binding theory
- BINDING_NEG(X): Conditions B and C of binding theory
- SRL_ARG_N/0/1/2/X/M/L/Z(X): Syntactic argument of X
- SAME_SRL_ARG(X, Y,..): All mentions are the same argument
- COORDINATE(X): X is a coordinate NP

Semantic:
- Semantic class of the mentions match (the same as (Soon et al., 2001))
- SEMCLASS_YES(X, Y,...), SEMCLASS_NO(X, Y), SEMCLASS_UN(X, Y)
- One mention is an alias of the other:
- ALIAS_YES(X, Y,...), ALIAS_NO(X, Y), ALIAS_UN(X, Y)
- PERSON(X): X is a person.
- ORGANIZATION(X): X is an organization.
- LOCATION(X): X is a location.
- SRL_SAMEVERB(X, Y,...): The mentions have a semantic role for the same verb.
- SRL_SAME_ROLE(X, Y,...): The same semantic role.
- SAME_SPEAKER(X, Y,...): The same speaker.

Figure 4.3: Feature functions used by RelaxCor (2/2).

data to obtain a set of constraints. A weight is then assigned to each constraint, taking into account the precision of the constraint finding coreferent mentions.

Figure 4.4 shows the training process. A machine learning process is applied to obtain the set of constraints, but not before a data selection process unbalances the training data set. Data selection is explained in Section 4.3.1, and the constraint learning process is explained in Section 4.3.2. The learned constraints are then applied to the training data set and their precision is evaluated. The precision of each constraint determines its weight.

The development process optimizes two parameters in order to achieve maximum performance given a measure for the task. One parameter is *balance*, which is used with the constraint precision to determine the weight of the constraint. The other parameter is *Nprune*, which, as explained in Section 4.3.3, is necessary in order to achieve a constant influence across the edge weights and reduce the computational cost. The development process is explained in Section 4.3.4.

4.3.1 Data selection

Generating an example for each possible pair of mentions in the training data produces an unbalanced data set in which more than 99% of the examples are negative (not coreferent). This bias towards negative examples makes the task of the machine learning algorithms difficult. Many classifiers simply learn to classify every example as negative, which achieves an accuracy of 99% but is not at all useful. In the case of decision trees and rule induction, this imbalance

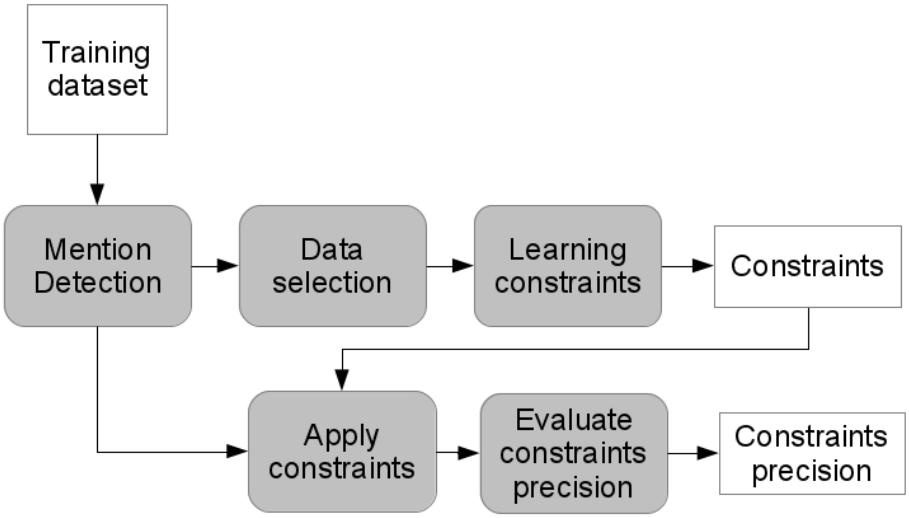

Figure 4.4: RelaxCor training process.

is also counterproductive. In addition, some corpora have more examples than the maximum affordable by the learning algorithm, given our computational resources. In this case, it is necessary to reduce the number of examples.

In order to reduce the amount of negative examples, a data selection process similar to clustering is run using the positive examples as the centroids. We define the distance between two examples as the number of features with different values. A negative example is then discarded if the distance to all the positive examples is always greater than a threshold, D. The value of D is empirically chosen depending on the corpora and the computational resources available.

4.3.2 Learning constraints

Constraints are automatically generated by learning a decision tree and then extracting rules from its leaves using C4.5 software (Quinlan, 1993). The algorithm generates a set of rules for each path from the learned tree, then checks whether the rules can be generalized by dropping conditions. Other studies have successfully used similar processes to extract rules from a decision tree that are useful in constraint satisfaction algorithms (Màrquez et al., 2000).

The weight assigned to a constraint (λ_k) is its precision over the training data (P_k), but shifted by a *balance* value:

$$\lambda_k = P_k - balance \tag{4.1}$$

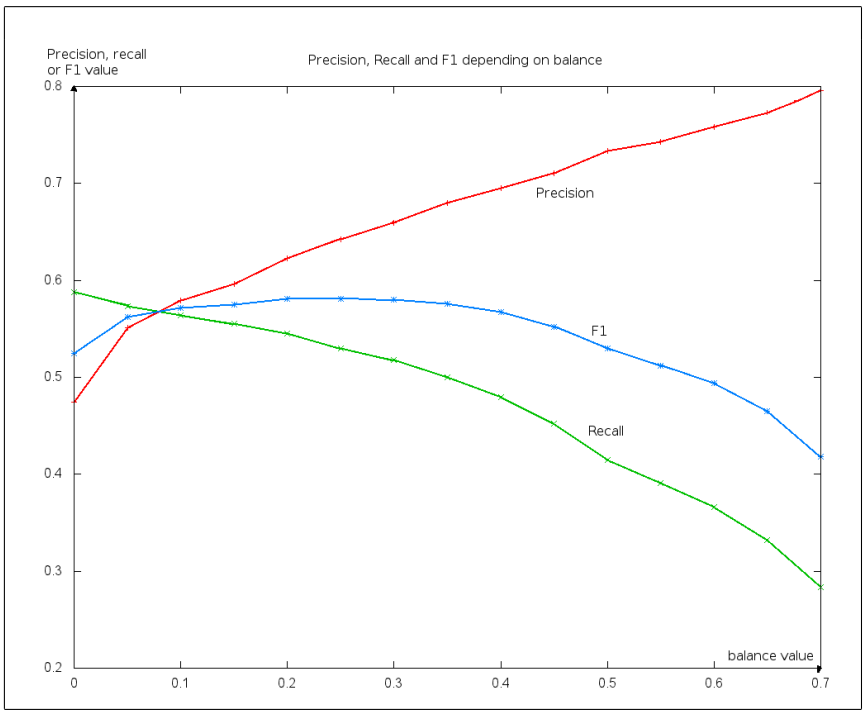

Figure 4.5: MUC's precision (red), recall (green), and F_1 (blue) for each balance value with pruning fixed to 6.

The precision here refers to the positive class, i.e., the ratio between the number of positive examples and the number of examples where the constraint applies. Note that the data selection process (Section 4.3.1) discards some negative examples to learn the constraints, but the weight of the constraints is calculated with the precision of the constraint over the whole training data.

The *balance* parameter adjusts the constraint weights to improve the balance between precision and recall. On the one hand, a high *balance* value causes most of the constraints to have a negative weight, with only the most precise having a positive weight. In this case, the system is precise but the recall is low, given that many relations are not detected. On the other hand, a low value for *balance* causes many low-precision constraints to have a positive weight, which increases recall but also decreases precision (see Figure 4.5). The correct value for *balance* is thus a compromise solution found in the development process, optimizing performance for a specific evaluation measure.

4.3.3 Pruning

As explained in Section 3.2, when a constraint applies to a set of mentions, a corresponding hyperedge is added to the hypergraph. In the case of the mention-pair model with automatically learned constraints, the most typical

case is that each pair of mentions satisfy at least one constraint, which produces an edge for each pair of mentions. There are three main issues to take into account when the problem is represented by an all-connected graph. First, the contribution of the edge weights for the resolution depends on the size of the document. Second, many weak edge weights may sum up to produce a bias in the wrong direction. And third, the computational cost of solving an all-connected graph by relaxation labeling is $O(n^3)$ (where n is the number of mentions in the document), which is a strong limitation on solving large documents. We now give an extended description of each one of these issues, and a description of the pruning process used to overcome them:

- The weight of an edge depends on the weights assigned to the constraints according to Equation 3.1. Note that the calculation of edge weights is independent of the graph adjacency. This implies that the larger the number of adjacencies, the smaller the influence of a constraint. Consequently, resolution has different results for large and small documents. Many coreference resolution systems have to deal with similar problems, especially those looking backward for antecedents. The larger the document is, the greater the set of possible antecedents to be classified by the system. This problem is usually solved by looking for antecedents in a window of a few sentences, which entails an evident recall limitation.

- Regarding the second issue, it is notable that some kinds of mention pairs are very weakly informative. For example, pairs such as (pronoun, pronoun). Many stories or discourses have a few main characters (entities) that monopolize the pronouns in the document. This produces many positive training examples for pairs of pronouns matching in gender and person, which may lead the algorithm to produce large coreferential chains joining all these mentions, even for stories where there are many different characters. For example, we have found in the results of some documents a huge coreference chain including every pronoun "he." This is because a pair of mentions ("he," "he") is usually linked with a small positive weight. Although the highest adjacent edge weight of a "he" mention may link with the correct antecedent, the sum of several edge weights linking the mention with other "he" results greater.

- Finally, the computational cost of solving an all-connected graph by relaxation labeling is $O(n^3)$. This cost is easily deduced by examining the algorithm in Figure 3.8. First, there is a loop for each variable v_i, and the number of variables is the number of mentions: n. Inside this, there is another loop for each label l of v_i, and the number of labels for v_i is $L_i = i$. The cost for these two loops is $O(\frac{n^2}{2})$. Inside the second loop, the support is calculated. The calculation of the support S_{il} for a vertex v_i and label l is an iteration over the incident edges $E(v_i)$, which is equal to n in an all-connected graph. Thus, the adjacency of the vertices depends on the size of the document. Therefore, the final computation cost of the algorithm is $O(\frac{n^3}{2})$, or $O(n^3)$ taking out the constant value.

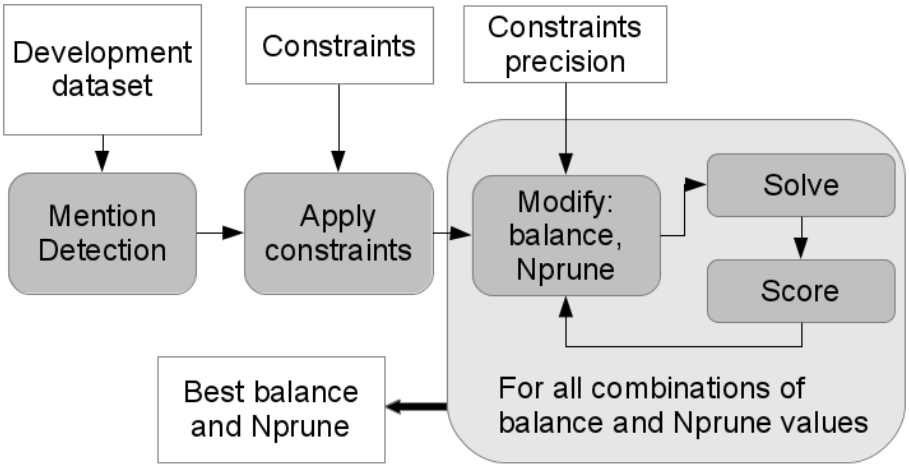

Figure 4.6: RelaxCor development process.

The pruning process turns $E(v_i)$ into a constant value N. For each vertex's incidence list $E(v_i)$, only a maximum of N edges remain and the others are pruned. In particular, the process keeps the $N/2$ edges with the largest positive weight and the $N/2$ with the largest negative weight. The value of N is chosen empirically by maximizing performance over the development data. After pruning, (i) the contribution of the edge weights does not depend on the size of the document; (ii) most edges of the less informative pairs are discarded, avoiding further confusion without limitation on distance or other restrictions that cause a loss of recall; and (iii) computational costs are reduced from $O(n^3)$ to $O(n^2)$, given that the innermost loop has a constant number of iterations (N).

4.3.4 Development

The current version of RelaxCor includes a parameter optimization process that uses the development data sets. The optimized parameters are *balance* and *Nprune*. The former adjusts the constraint weights to improve the balance between precision and recall, as explained in Section 4.3.2, and the latter limits the adjacency of the vertices, which significantly reduces the computational cost and improves overall performance, as explained in Section 4.3.3. Optimizing this parameter depends on properties like the document size and the quality of the information given by the constraints. Figure 4.6 shows the development process.

The development process calculates a grid based on the possible values of both parameters: from 0 to 1 with a stepsize of 0.05 for balance, and from 2 to 14 with a stepsize of 2 for pruning. Both parameters are empirically adjusted on the development set for an evaluation measure. For each combination of parameter values, the system is executed over the development data and evaluated with the target measure. The combination that maximizes the score is the one selected.

The Technical University of Catalonia, sometimes called UPC-Barcelona Tech, is the largest engineering university in Catalonia, Spain. The objectives of the UPC are based on internationalization, as it is [[**Spain**]$_0$**'s technical university with the highest number of international PhD students**]$_1$ and [[**Spain**]$_2$**'s university with the highest number of international master's degree students**]$_3$...
Constraint: **PN_STR(0,2)** & **HEAD_MATCH(1,3)** & **NESTED(0,1)** & **NESTED(2,3)**
Influence rule: $(0,2)_A, (1)_B \Rightarrow (3)_B$

Figure 4.7: Example of a group constraint using an influence rule to take advantage of the entity-mention model. The constraint expects four mentions, where: two of them are proper names and match in their complete strings, the other two match in their heads, mention 0 is inside mention 1, and mention 2 is inside mention 3. The influence rule says that when mentions 0 and 2 belong to the same entity (A) but mention 1 belongs to another one (B), then mention 3 should belong to entity B in order to corefer with mention 1.

4.4 Training and development for the entity-mention model

The training process for the entity-mention model is, in theory, exactly the same as for the mention-pair model, but with predefined influence rules and groups of N mentions instead of pairs. For each combination of influence rule and N, the training process has the same steps as explained in previous sections: learn constraints, apply them to the training data, calculate the weights, and perform the development process to find the optimal balance value. The positive examples are those that satisfy the final condition of the influence rule, and the rest are negative examples. However, a machine learning process to discover group constraints has a considerable cost if all the training data needs to be evaluated. The number of combinations increases exponentially as the number of implied mentions increases. Moreover, the ratio of positive to negative examples is extremely low, and a data selection process like the one used for pair constraints (Section 4.3.1) has a high computational cost.

For these reasons, the group constraints of our experiments are obtained using only the examples that the mention-pair model could not solve. Thus, after training and running RelaxCor over an annotated data set using just pair constraints, its errors are now used as examples for training the entity-mention model. The type of errors are those in which three mentions ($N = 3$) corefer $(0,1,2)_A$, but the mention-pair model has determined that just two of them corefer and discarded the third one (for example: $(0,1)_A, (2)_B$). Each time an error like this is found, the three mentions correspond to a positive example (corefer) and all other combinations of three mentions between mentions 0 and 2 are considered negative examples. The influence rules for the constraints learned this way are $(0,1)_A \Rightarrow (2)_A$, $(0,2)_A \Rightarrow (1)_A$, and $(1,2)_A \Rightarrow (0)_A$, depending on which mention was wrongly classified by the mention-pair model.

Note that when an entity-mention model has been trained this way, the resolution system is executed using both the mention-pair and entity-mention models at the same time.

Alternatively, constraints for the entity-mention model can be added manually by writing them. Figure 4.7 shows an example of an entity-mention constraint (i.e., a group constraint with an influence rule). This kind of constraint has great potential to take advantage of the structure of discourses. The example shows how the algorithm can benefit from knowing that nested mentions have some kind of relation. In the case that two coreferring mentions are related with two other mentions with the potential to corefer, the entity-mention model can use this information to find more coreference relations.

The rest of the training and development process is conducted in the same way as for the mention-pair model. The weights of group constraints are obtained by evaluating their precision over the training data, and the *balance* value is determined by a development process. However, in our experiments, the number of group constraints is typically lower than the number of pairwise ones, so there is no need for pruning.

4.5 Empirical adjustments

In addition to the training and development processes, experiments using RelaxCor have led us to discover other aspects of the theoretical model that can be adjusted during implementation to improve the performance of the system. One of these aspects is the initial state of the variables in the relaxation labeling algorithm. The initial state can be random or better-informed, and we have found that an equiprobable state with a small modification to enhance the creation of new entities is better than the random or most-informed state. Another aspect is the order of the mentions when the vertices of the graph are created. While the theoretical model says that vertices are added to the graph in the same order as the mentions are found in the document, the truth is that, given the increasing number of possible labels for each vertex, it is better to reduce the number of possible labels for the most informative mentions.

4.5.1 Initial state

The initial state of the vertices defines the a priori probability of each vertex being in each partition. There are several possible initial states, depending on the initial information known about the problem. The most frequently used are random and equiprobable ones. In the following, there is a description of the initial states we have tested:

- **Equiprobable.** The probability of vertex v_i being in a partition is the same for every possible partition:

$$h_l^i = \frac{1}{L_i}, \forall l = 0..L_i$$

- **New entities.** The probability of vertex v_i being in a new partition is double that of being in an already-created partition.

$$h_l^i = \frac{1}{L_i+1}, \forall l = 0..L_i - 1$$

$$h_{L_i}^i = \frac{2}{L_i+1}$$

- **Random.** Given the equiprobable state, a random value is added to each probability.

$$h_l^i = \frac{1}{Z}(\frac{1}{L_i} + \epsilon_l), \forall l = 0..L_i$$

  where ϵ_l is a random value ($\frac{-1}{2L_i} \leq \epsilon \leq \frac{1}{2L_i}$) and Z is a normalization parameter to force $\sum_{l \in L_i} h_l^i = 1$.

- **Informed.** The initial state is set using confidence values learned by a classifier. Initially, one of the previous initial states is used (for example, equiprobability). Then each mention (m_j) is evaluated with the classifier over the previous mentions m_i in the document. In the case that the pair (m_i, m_j) is classified as coreferent, the weight vector h^i is multiplied by the confidence value returned by the classifier and then added to the probability vector of m_j, h^j.

The experiments in Section 5 show that any initial state based on equiprobability achieves a similar performance. In contrast, a totally random (non-informed) initial state is worse than the others. The influence of a classifier modifying the initial state does not significantly improve the results. Taking into account that using a classifier increases the computational cost, this configuration has been discarded. Finally, taking into account that in a realistic scenario the majority of mentions are singletons, the best choice is *New entities*, which is based on equiprobability with enhanced formation of new coreference chains or singletons.

4.5.2 Reordering

Usually, the vertices of the graph would be placed in the same order as the mentions are found in the document (*chronological order*). In this manner, v_i corresponds to m_i. However, as suggested by Luo (2007), there is no need to generate the model following that order. In our approach, the first variables have a lower number of possible labels. Moreover, an error in the first variables has more influence on the performance than an error in later ones. It is reasonable to expect that placing named entities at the beginning is helpful for the algorithm, given that named entities are usually the most informative mentions.

Reordering only affects the number of possible labels of the variables. The chronological order of the document is taken into account by the constraints, regardless of the graph representation. Our experiments (Sapena et al., 2010a) confirm that placing named entity mentions first, then nominal mentions, and finally the pronouns, increases the precision considerably. Inside each of these groups, the order is the same as in the document.

4.6 Language and corpora adaptation

The whole methodology of RelaxCor, including the resolution algorithm and the training process, is totally independent of the language of the document. The only parts requiring adjustments are the functions that read the input data and the set of features. In most cases, these modifications are just for the different format of input data in different languages, rather than for specific language issues. However, the implementation of some features has hard-coded heuristics for English that need editing in order to use the system with other languages. In addition, some resources used by RelaxCor, such as WordNet, lists of nicknames, and countries, need to be translated or excluded.

For our participation in Semeval-2010 (Sapena et al., 2010b), the system was adapted to Catalan and Spanish by rewriting some features for these languages and avoiding the use of external resources. The same can be done for any language.

4.7 Related work

In Chapter 2, we introduced an overview of many approaches, with their classification models and resolution processes (see Figure 2.13). Our approach can be classified similarly as a one-step resolution that uses the entity-mention model for classification and conducts hypergraph partitioning for the linking process. This classification matches that of the Copa system described in (Cai and Strube, 2010a). Both approaches represent the problem in a hypergraph, where each mention is a vertex, and use hypergraph partitioning in order to find the entities. However, the differences between these two approaches are substantial. The most significant differences are as follows:

- Hypergraph generation. RelaxCor adds hyperedges to the hypergraph for each group of mentions that satisfy a **constraint**, whereas Copa adds a hyperedge for each group of mentions that satisfy a **feature**. Note that the addition of hyperedge weights representing features cannot take advantage of the nonlinear combinations offered by constraints, as explained in Section 3.2.1. Actually, in order to incorporate some nonlinearity, Copa needs combined features to introduce information such as mention type (pronoun, proper name, etc.) or distances.

- Resolution algorithm. RelaxCor uses relaxation labeling in order to satisfy as many constraints as possible. In fact, the hypergraph is just a representation of the problem. Copa uses *recursive 2-way partitioning*, a hypergraph partitioning algorithm. Copa's main contribution is not the resolution algorithm, but the hypergraph representation of the problem.

- Computational costs. RelaxCor needs to train a decision tree in order to extract a set of rules to use them as soft constraints. These constraints are then applied to the training data to calculate their weight. Copa does not use constraints, which reduces the computational cost of the training process. On the other hand, the cost of the resolution algorithm is $O(n^3)$ for Copa whereas it is $O(n^2)$ in RelaxCor thanks to the pruning process.

Chapter 5

Experiments and results

Several experiments have been performed on coreference resolution in order to test our approach. This chapter includes a short explanation and result analysis of each experiment. More details about these experiments can be found in the references. This chapter is divided into four sections describing different experiments. The first section includes experiments to tune and improve empirical aspects of the approach: the utility of reordering mentions when generating the graph, adjusting values for pruning and balance, and so on. Next, there is an explanation of a set of experiments to evaluate the performance of coreference resolution and mention detection. The scores are compared with the state of the art in diverse corpora, measures, and languages. In addition, our participation in Semeval-2010 and CoNLL-2011 shared tasks is explained in detail with performance, comparisons, and error analysis. Finally, a set of experiments using the entity-mention model are described.

5.1 RelaxCor tuning experiments

As described in Chapter 4, there are several RelaxCor subprocesses that must be tuned in order to optimize the performance of the system. These subprocesses are related to training and development processes, and also to the relaxation algorithm used for resolution. In the following, there is an explanation of the experiments for pruning, reordering, initial state, balance, and feature selection, including an analysis of performance and results.

Pruning. The development process is a parameter optimization for *balance* and *Nprune*. The latter limits the adjacency of the vertices, which significantly reduces the computational cost and improves overall performance (as explained in Section 4.3.3). Optimization of this parameter depends on properties such as the document size and the quality of information given by the constraints. The

	bnews	npaper	nwire	Global			
Metric:	CEAF			CEAF	B^3		
Pruning	F_1	F_1	F_1	F_1	P	R	F_1
No	67.3	64.4	69.5	67.2	**88.4**	62.7	73.3
Yes	**68.6**	**65.2**	**70.1**	**68.0**	82.3	**66.9**	**73.8**

Table 5.1: Pruning results on ACE-phase02.

	bnews	npaper	nwire	Global			
Metric:	CEAF			CEAF	B^3		
Model	F_1	F_1	F_1	F_1	P	R	F_1
Chronological order	68.6	65.2	70.1	68.0	82.3	**66.9**	73.8
Reordering	**69.5**	**67.3**	**72.1**	**69.7**	**85.3**	66.8	**74.9**

Table 5.2: Reordering results on ACE-phase02.

scores published in (Sapena et al., 2010b) are shown in Table 5.1. The table compares the algorithm scores with and without pruning the graph. ACE 2002 (phase02) has been used for this experiment with true mentions. ACE-phase02 is divided into three subsets, bnews, npaper, and nwire, that include different types of documents. The final columns tagged as "Global" refer to the average of the whole corpus. The measures included in the table are CEAF and B^3. The scores show how pruning the graph before resolution improves performance. Specifically, the score using the CEAF measure improves by 0.8 points (from 67.2% to 68.0%), and using B^3 gives an improvement of 0.5 points (from 73.3% to 73.8%). Although the improvements are not statistically significant, the pruning process reduces computational costs from $O(n^3)$ to $O(n^2)$, which is an important advantage in terms of scalability. In conclusion, the pruning process reduces costs without losing performance, which may even improve.

Reordering. This experiment compares performance when the graph is formed following the chronological order of the mentions in the document or using the reordering method explained in Section 4.5.2. Briefly, after reordering, the most informative mentions (NEs) are the ones with the fewest possible labels and probably the first to have their entity determined, which facilitates correct entity assignment for the remaining mentions. The results obtained are shown in Table 5.2. We observe significant improvements using the reordering method instead of chronological order. CEAF scores are increased by 1.7 points (from 68.0% to 69.7%), and B^3 scores improve by 1.1 points (from 73.8% to 74.9%). More details on this experiment are described in (Sapena et al., 2010a).

Initial State. As explained in Section 4.5.1, there are several possibilities for selecting the initial state in the relaxation labeling algorithm. This experiment compares the performance of different methods: random, equiprobable, almost equiprobable but enhancing new entities (New Entities), and better-informed using confidence values of a classifier (Informed). It confirms that using the initial state New Entities performs better than the random state (more than 1

point in both experiments by the CEAF measure), and slightly better than the equiprobable state, as shown in Table 5.3. The Informed state achieves similar results to New Entities (+0.04 points with CEAF), but requires more resources. Consequently, New Entities is chosen as the initial state. This experiment was undertaken for the thesis proposal (2009), and in (Sapena et al., 2010a). Note that the scores from 2010 are better than those from 2009, a result of many other improvements to the implementation.

	bnews	npaper	nwire	Global			
Metric:	CEAF			CEAF	B^3		
Initial State	F_1	F_1	F_1	F_1	P	R	F_1
Random	62,73	65,00	59,96	62,60	-	-	-
Equiprob	63,46	65,39	62,26	63,72	-	-	-
New Entities	**63,73**	**66,20**	62,34	64,11	-	-	-
Informed	63,61	66,12	**62,67**	**64,15**	-	-	-
Results in (Sapena et al., 2010a):							
Random	68.2	66.1	71.0	68.5	83.5	66.7	74.2
New Entities	**69.5**	**67.3**	**72.1**	**69.7**	85.3	66.8	**74.9**

Table 5.3: Initial State experiment. Results on ACE-phase02 using CEAF-mention (F_1) and B^3.

Balance. The development process is a parameter optimization for *balance* and *Nprune*. The former adjusts the constraint weights to improve the balance between precision and recall, as explained in Section 4.3.2. Figure 5.1 shows how precision increases and recall decreases as the value of the balance is increased. The optimal performance is found by considering F_1. Previously, RELAXCOR was using a fixed balance value of 0.5, commonly achieving scores with high precision but low recall. Experimentation in the development process has shown that the optimal F_1 is typically achieved with a balance between 0.1 and 0.4. When a balance value is determined using development data, the value is fixed in order to solve test data.

Feature selection. This experiment compares performance when the system uses different sets of the available features. The baseline in this case is an approach using decision trees in a backward search resolution, similar to (Ng and Cardie, 2002b). The features used in the baseline are the same as those used in RELAXCOR. In order to select the best set of features, a Hill Climbing process is performed using the baseline with a five-fold cross-validation over the training corpus. The Hill Climbing process starts using the whole set of features. A cross-validation is done (un)masking each feature. The (un)masked feature with most improvement is (added to) removed from the set. The process is repeated until an iteration without improvements is reached. Once the baseline has defined a selection of features, RELAXCOR is trained with the same set of features. Both systems are compared using all the features and the selection.

Figure 5.2 shows the results of the experiment. Note that the baseline has an important improvement thanks to the feature selection. In contrast, the performance of RELAXCOR using all the features or the baseline feature selection

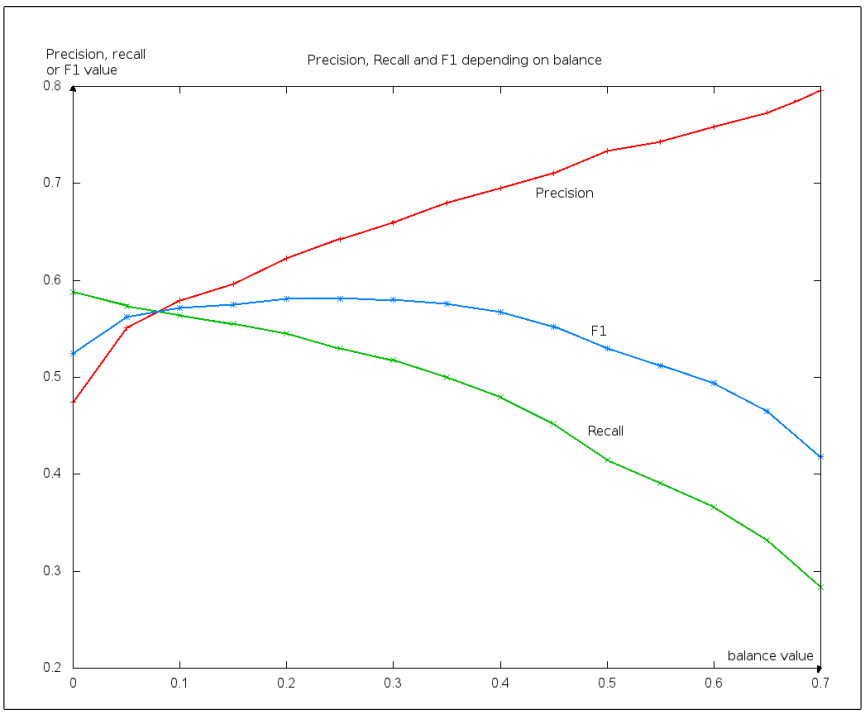

Figure 5.1: RelaxCor development process. For a fixed value of *Nprune*, this graphic shows how precision is increased and recall decreased as the value of the *balance* is increased. The optimal performance is found by taking F_1 into account. The figure shows MUC's precision (red), recall (green), and F_1 (blue) for each *balance* value with *Nprune* fixed to 6.

Approach	Features	nwire	bnews	npaper	Global ACE
Baseline	all	57,57	62,59	52,97	57,77
Baseline	selection	64,27	67,13	62,34	**64,61 (+6.84)**
RelaxCor	all	63,61	66,12	62,67	**64,15**
RelaxCor	selection	64,30	64,92	61,70	63,67 **(-0.48)**

Figure 5.2: Results on ACE02 using CEAF-mention (F_1).

is generally similar. The scores are even higher when no selection is done. Therefore, the training processes of RelaxCor does not seem to be affected by noisy features. This is an advantage in the sense that the process of feature selection is unnecessary. Any feature or constraint can be added to the system without concern.

RelaxCor results do not outperform the baseline with feature selection. Note that this experiment was undertaken for the thesis proposal (2009), and the performance of RelaxCor has been improved since then.

OntoNotes	Recall	Precision	F_1
Development	92.45	27.34	42.20
Test	92.39	28.19	43.20

Table 5.4: Mention detection results on OntoNotes (used in CoNLL-2011 Shared Task).

5.2 Coreference resolution performance

This section presents experiments and results regarding the performance of our approach for coreference resolution, including the mention detection subtask, and a comparison of results in multiple languages. The scores are compared with other approaches in state-of-the-art systems.

Mention detection. The performance of the mention detection system achieves a good recall, higher than 90%, but a low precision, as published in (Sapena et al., 2011) and reproduced in Table 5.4. The OntoNotes corpora have been used for this experiment, as they were used in CoNLL-2011. Given that the mention detection in a pipeline combination acts as a filter, recall should be kept high, as a loss of recall at the beginning would result in a loss of performance in the rest of the process. However, at this point, the precision is not a priority as long as it remains reasonable, given that the coreference resolution process is able to determine that many mentions are singletons. Moreover, the evaluation of precision on the OntoNotes corpora only takes into account mentions included in a coreference chain, not singletons. This means that the precision value is not really evaluating the precision of the mention detection system. A fair evaluation of mention detection should be performed in a corpora with annotations of every referring expression, but such a corpora is not available as far as we know.

The most typical error made by the system is to include extracted NPs that are not referential (e.g., predicative and appositive phrases) and mentions with incorrect boundaries. The incorrect boundaries are mainly due to errors in the predicted syntactic column and some mention annotation discrepancies. Furthermore, the coreference annotation of OntoNotes used in CoNLL-2011 included verbs as anaphors of some verbal nominalizations. But verbs are not detected by our mention detection system, so most of the missing mentions are verbs. The methodology of the mention detection system is explained in Section 4.1.

State of the art comparison. RelaxCor performance has been compared several times with other published results from state-of-the-art systems. In (Sapena et al., 2010a), we claimed to have the best performance for the ACE-phase02 corpora, using true mentions in the input and evaluating with the CEAF and B^3 measures. The table comparing scores with the best results found at that moment is reproduced as Table 5.5. Our approach is also compared

	bnews	npaper	nwire	Global			
Metric:	CEAF			CEAF	B^3		
Model	F_1	F_1	F_1	F_1	P	R	F_1
RelaxCor	**69.5**	**67.3**	**72.1**	**69.7**	85.3	66.8	**74.9**
MaxEnt+ILP (Denis, 2007)	-	-	-	66.2	81.4	65.6	72.7
Rankers (Denis, 2007)	65.7	65.3	68.1	67.0	79.8	66.8	72.7

Table 5.5: Comparison of results on ACE-phase02.

with the state of the art in two competitions: SemEval-2010 (Sapena et al., 2010b), and CoNLL-2011 (Sapena et al., 2011). RelaxCor achieved one of the best performances in SemEval-2010, but contradictory results across measures prevented the organization from determining a winner. In addition, RelaxCor achieved second position in the CoNLL-2011 Shared Task; Figure 5.3 reproduces the official table of results. Section 5.3 describes the shared tasks in detail. Moreover, the performance of RelaxCor is again compared with two other state-of-the-art systems in (Màrquez et al., 2012).

Languages. (Sapena et al., 2010b; Màrquez et al., 2012) show the performance of our approach for English, Catalan, and Spanish. The scores for Spanish and Catalan do not seem as good as for English, because the system was originally designed with the English language in mind. As a result, it does not include language-specific features for Spanish and Catalan, such as whether a mention is an elliptical subject or not. Despite this, RelaxCor scores for Catalan and Spanish are the best amongst the state of the art.

5.3 Shared tasks

During the development of this thesis, two international shared tasks –SemEval-2010 Task 1 and CoNLL-2011 Shared Task– have been organized in order to evaluate the state of the art in coreference resolution, compare approaches, and provide insight into many aspects of the task. We participated in both shared tasks, with our system achieving good rankings and receiving feedback and comparisons with state-of-the-art systems. In addition, part of the work realized in this thesis has been employed in the organization of the SemEval task.

5.3.1 SemEval-2010

The goal of SemEval-2010 Task 1 (Recasens et al., 2010b) was to evaluate and compare automatic coreference resolution systems for six different languages in four evaluation settings and using four different evaluation measures. This complex scenario aimed to provide an insight into several aspects of coreference resolution, including portability across languages, the relevance of linguistic

-	CEAF			MUC			B^3		
Language	R	P	F_1	R	P	F_1	R	P	F_1
ca	69.7	69.7	69.7	27.4	77.9	40.6	67.9	96.1	79.6
es	70.8	70.8	70.8	30.3	76.2	43.4	68.9	95.0	79.8
en-closed	74.8	74.8	74.8	21.4	67.8	32.6	74.1	96.0	83.7
en-open	75.0	75.0	75.0	22.0	66.6	33.0	74.2	95.9	83.7

Table 5.6: Results of RelaxCor on development data (SemEval-2010).

-	CEAF			MUC			B^3			BLANC		
Language	R	P	F_1	R	P	F_1	R	P	F_1	R	P	BLANC
Information: closed Annotation: gold												
ca	70.5	70.5	70.5	29.3	77.3	42.5	68.6	95.8	79.9	56.0	81.8	59.7
es	66.6	66.6	66.6	14.8	73.8	24.7	65.3	97.5	78.2	53.4	81.8	55.6
en	75.6	75.6	75.6	21.9	72.4	33.7	74.8	97.0	84.5	57.0	83.4	61.3
Information: open Annotation: gold												
en	75.8	75.8	75.8	22.6	70.5	34.2	75.2	96.7	84.6	58.0	83.8	62.7

Table 5.7: Results of RelaxCor on test data (SemEval-2010).

information at different levels, and the behavior of alternative scoring measures. The task attracted considerable attention from a number of researchers, but only six teams submitted results. Moreover, participants did not run their systems for all the languages and evaluation settings, thus making direct comparisons very difficult.

As discussed in the task description paper and slides, the task contributed to the coreference community with valuable resources, evaluation benchmarks, and results along several dimensions. However, some problems were also identified and discussed. These mainly related to the excessive complexity of the task, the limited number of participants, and a design decision that did not allow for a fair comparison between settings using gold-standard input information and those using automatically predicted input information.

RelaxCor participated in the SemEval task for English, Catalan, and Spanish (Sapena et al., 2010b). At the time, the system was not ready to detect mentions. Thus, participation was restricted to the gold-standard evaluation, which included the manual annotated information and also provided the mention boundaries.

The whole methodology of RelaxCor, including the resolution algorithm and the training process, is totally independent of the document language. However, in order to use the system on Spanish and Catalan documents, two parts needed some adjustments: the preprocessing and the set of feature functions. In most cases, the modifications to the feature functions were just for the different data format of different languages rather than for specific language issues. Moreover, no preprocessing was needed given that the task included a good deal of information about the mentions in the documents, such as parts-of-speech, syntactic dependency, heads, and semantic role.

RelaxCor results for development and test data sets are shown in Tables 5.6 and 5.7, respectively. The version of RelaxCor used in SemEval had a balance value fixed to 0.5, which proved to be an inadequate value. Thus, the results have high precision but a very low recall. This situation produced high scores with the CEAF and B^3 measures, due in part to the annotated singletons. However, the system was penalized by measures based on pair-linkage, particularly MUC. Although RelaxCor had the highest precision scores (even with MUC), the recall was low enough to finally obtain low scores for F_1.

Regarding the test scores of the comparison with other participants (Recasens et al., 2010b), RelaxCor obtained the best performance for Catalan (CEAF and B^3), English (closed: CEAF and B^3; open: B^3), and Spanish (B^3). Moreover, RelaxCor was the most precise system under all metrics in all languages, except for CEAF in English-open and Spanish. This confirms the robustness of the results of RelaxCor, but also highlights the necessity of searching for a balance value other than 0.5 to increase the recall of the system without losing much by way of precision. Indeed, the idea of using development (Section 4.3) to adapt the balance value occurred after these results were obtained.

The incorporation of WordNet to the English run of RelaxCor was the only difference between our implementation in the English-open and English-closed tasks. The scores were slightly higher when using WordNet, but not significantly so (75.8% vs. 75.6% for CEAF and 34.2% vs. 33.7% for MUC). Analyzing the MUC scores, note that the recall improves (from 21.9% to 22.6%), while the precision decreases a little (from 74.4% to 70.5%), which corresponds to the information and noise that WordNet typically provides.

As expected, the results for the test and development are very similar, except the Spanish (es) ones, for which the recall drops considerably from development to test. This is clearly shown in the MUC recall scores, which fall from 30.3% on development to 14.8% on test, and also indirectly affects the other scores. This issue was caused by a bug. Subsequent experiments confirmed that the test results also corresponded to the development results for Spanish. More recent results on the same corpora are published in (Màrquez et al., 2012).

As part of the organization team, we developed an open-source scorer that was published and freely available to any researcher on the competition website[1]. The same scorer, with some modifications, was also used the following year in CoNLL. In addition, many studies mainly related with the formats and preprocessing of the corpora were conducted.

5.3.2 CoNLL-2011

The CoNLL-2011 Shared Task[2] (Pradhan et al., 2011) was based on the English portion of the OntoNotes 4.0 data. The task was to automatically identify

[1] SemEval-2010 Task 1 website: http://stel.ub.edu/semeval2010-coref
[2] CoNLL-2011 Shared Task website: http://conll.bbn.com

mentions of entities and events in text, and to link the coreferring mentions together to form entity/event chains. The target coreference decisions could be made using automatically predicted information on the other structural layers including the parses, semantic roles, word senses, and named entities.

As is customary for CoNLL tasks, there was a *closed* and an *open* track. For the closed track, systems were limited to using the distributed resources, in order to allow a fair comparison of algorithm performance, while the open track allowed for almost unrestricted use of external resources in addition to the provided data.

RelaxCor participated in the closed track CoNLL task (Sapena et al., 2011). All the knowledge required by the feature functions was obtained from the annotations of the corpora, and no external resources were used with the exception of WordNet, gender and number information, and sense inventories. All of these were allowed by the task organization and are available on their website.

There were remarkable features that made this task different and more difficult than previous ones. Regarding mention annotation, it is important to emphasize that singletons were not annotated, mentions must be detected by the system, and the mapping between system and true mentions was limited to exact matching of boundaries. Moreover, some verbs were annotated as coreferring mentions.

Regarding the evaluation, the scorer used the modification of (Cai and Strube, 2010b), which was unprecedented, and the corpora had only recently been published, meaning that there were no published results to use as references.

The results obtained by RelaxCor can be found in Tables 5.8 and 5.9. Due to the lack of annotated singletons, mention-based metrics B^3 and CEAF produce lower scores—near 60% and 50% respectively—than typically achieved with different annotations and mapping policies—usually near 80% and 70%. Moreover, the requirement that systems use automatic preprocessing and do their own mention detection increases the difficulty of the task, which obviously decreases the scores in general. The official ranking score was the arithmetic mean of the F-scores of MUC, B^3, and CEAFe.

The measure that remains most stable is MUC, as it is link-based and does not take singletons into account anyway. Thus, it is the only one comparable with the state of the art at this point. The results obtained with MUC scorer show an improvement in RelaxCor's recall, a feature that needed improvement given the remarkably low SemEval-2010 results with MUC.

About 65 different groups demonstrated interest in the shared task by registering on the task web page. Of these, 23 groups submitted system outputs on the test set during the evaluation week. Eighteen groups submitted only closed track results, three groups only produced open track results, and two groups submitted both closed and open track results.

Measure	Recall	Precision	F_1
Mention detection	92.45	27.34	42.20
Mention-based CEAF	55.27	55.27	55.27
Entity-based CEAF	47.20	40.01	43.31
MUC	54.53	62.25	58.13
B^3	63.72	73.83	68.40
(CEAFe+MUC+B^3)/3	-	-	56.61

Table 5.8: RelaxCor results on the development data set (CoNLL-2011).

Measure	Recall	Precision	F_1
Mention detection	92.39	28.19	43.20
Mention-based CEAF	53.51	53.51	53.51
Entity-based CEAF	44.75	38.38	41.32
MUC	56.32	63.16	59.55
B^3	62.16	72.08	67.09
BLANC	69.50	73.07	71.10
(CEAFe+MUC+B^3)/3	-	-	**55.99**

Table 5.9: RelaxCor official test results (CoNLL-2011).

RelaxCor achieved second position in the official closed track results, as shown in Figure 5.3. The final column shows the official ranking score. The difference from the system in first place is 1.8 points, which is statistically significant, while the difference to third position is just 0.03 points and is not significant. The winning system—Stanford (Lee et al., 2011)—does not use machine learning but combines many heuristics to join mentions and partial entities, starting with the most precise ones. It is thought that the difference between RelaxCor and Stanford's system is mainly due to their use of sophisticated handwritten heuristics instead of our automatically-learned constraints.

The first column of Figure 5.3 shows the F_1 value of the mention detection evaluation. The organization evaluated mention detection with the same outputs of coreference resolution, and all participants except us excluded singletons from their output. Our score is the worst because we included singletons in our output, which heavily penalizes precision. Singletons do not affect the other measures of coreference resolution, and we did not think it necessary to clear them. Either way, given that mention detection is evaluated after coreference resolution and singletons are filtered, any detected mention incorrectly resolved as a singleton is also penalized in the mention detection evaluation. Mention detection should be evaluated before resolution, as in our experiments described in the previous section, or without clearing singletons from the output. As far as we know, the recall of RelaxCor's mention detection system was the highest of the systems that evaluated their recall before filtering singletons. (For instance, the Stanford system's recall for mention detection was 87.9% (Lee et al., 2011) and ours was 92.45%.)

System	MD	MUC	B-CUBED	CEAF$_m$	CEAF$_e$	BLANC	Official
	F	F^1	F^2	F	F^3	F	$\frac{F^1+F^2+F^3}{3}$
lee	**70.70**	59.57	68.31	**56.37**	**45.48**	73.02	**57.79**
sapena	43.20	59.55	67.09	53.51	41.32	71.10	55.99
chang	64.28	57.15	**68.79**	54.40	41.94	**73.71**	55.96
nugues	68.96	58.61	65.46	51.45	39.52	71.11	54.53
santos	65.45	56.65	65.66	49.54	37.91	69.46	53.41
song	67.26	**59.95**	63.23	46.29	35.96	61.47	53.05
stoyanov	67.78	58.43	61.44	46.08	35.28	60.28	51.92
sobha	64.23	50.48	64.00	49.48	41.23	63.28	51.90
kobdani	61.03	53.49	65.25	42.70	33.79	62.61	51.04
zhou	62.31	48.96	64.07	47.53	39.74	64.72	50.92
charton	64.30	52.45	62.10	46.22	36.54	64.20	50.36
yang	63.93	52.31	62.32	46.55	35.33	64.63	49.99
hao	64.30	54.47	61.01	45.07	32.67	65.35	49.38
xinxin	61.92	46.62	61.93	44.75	36.23	64.27	48.46
zhang	61.13	47.28	61.14	44.46	35.19	65.21	48.07
kummerfeld	62.72	42.70	60.29	45.35	38.32	59.91	47.10
zhekova	48.29	24.08	61.46	40.43	35.75	53.77	40.43
irwin	26.67	19.98	50.46	31.68	25.21	51.12	31.28

Figure 5.3: RelaxCor (sapena) achieved second position in the official closed track competition.

	Measure	Precision	Recall	F_1
Mention-pair ($N = 2$)	CEAFm	81.73	81.73	81.73
	MUC	72.92	54.17	62.17
	B^3	91.87	82.87	87.14
	CEAFe	81.95	90.47	86.00
Entity-mention ($N = 3$)	CEAFm	82.02	82.02	82.02
	MUC	73.01	54.28	62.27
	B^3	91.59	83.12	87.15
	CEAFe	82.10	90.63	86.15

Table 5.10: Comparison of RelaxCor results using just the mention-pair model ($N = 2$) with those also using the entity-mention model ($N = 3$) (corpus: SemEval-2010).

5.4 Experiments with the entity-mention model

Constraints for the entity-mention model are automatically obtained using the training data examples that the mention-pair model could not solve, with predefined influence rules and limited to $N = 3$. The training process is explained in Section 4.4. Experiments with the entity-mention model are conducted using both models at the same time. The goal of the experiments is to improve the performance of the mention-pair model itself.

Table 5.10 shows the experimental results using the SemEval-2010 English corpora. The table compares the entity-mention results (RelaxCor using $N = 3$ constraints with influence rules, including the whole set of $N = 2$ constraints) with those using mention-pairs (RelaxCor using just $N = 2$ con-

straints). The entity-mention model outperforms the mention-pair model. However, the number of really useful examples (i.e., mentions wrongly classified by the mention-pair model but correctly classified by the entity-mention model) is low. Consequently, the difference in their scores is not significant. The $N = 3$ constraints have a good precision and also an acceptable recall. However, most of the mentions affected by these constraints were already affected and correctly solved by the mention-pair model. Further research is needed in order to find more useful constraints, either by writing more elaborate group constraints or finding a better system that automatically finds them.

These results may be somewhat justified, because the entity-mention model is using the same feature functions and, consequently, the same information as the mention-pair model. In fact, the only new information is included in the conditions of the influence rules, which take into account the entities assigned to each mention during resolution. In addition, group constraints can also include, in an implicit way, information about the structure of the discourse. However, it seems clear that this new information is either too little or not relevant enough.

Even though the obtained performance does not significantly outperform the mention-pair model, we can draw some positive conclusions from these experiments. First of all, the approach is ready to use either model (mention-pair or entity-mention) in a constructive way. As soon as new feature functions specific to entity-mention models appear, the results will reflect this. One research line to follow in this field is the incorporation of feature functions following discourse theories, such as focusing and centering. Another research line is the introduction of world knowledge using these models, as explained in the next chapter.

Chapter 6

Adding world knowledge to coreference resolution

Some coreferences cannot be solved using only linguistic (lexical, morphological, and syntactic) information. Often, common sense and world knowledge is essential to resolve coreferences. For example, we can find coreferential mentions in any newspaper, such as {"Obama," "USA President"}, {"Messi," "Barcelona striker"}, or {"Beirut," "the Lebanese capital"}. In some cases, the information is introduced with appositions and the resolution algorithms can solve them, but many other cases cannot be solved without world knowledge. For example, Figure 6.1 shows a passage from a newspaper about the football player "*Lionel Messi*." The set of coreferent mentions about *Messi* are boldfaced. Note that the information that *Messi* is a striker is easily obtained, given that it is attached in the noun phrase *star striker Lionel Messi*. However, the noun phrase *the young Argentine* can only be solved by the addition of world knowledge. Otherwise, a coreference resolution system may incorrectly link this NP with other entities in the document, such as *Laporta* or *Abramovich*, or classify it as a singleton.

FC Barcelona president Joan Laporta has warned Chelsea off **star striker Lionel Messi**.

Aware of Chelsea owner Roman Abramovich's interest in **the young Argentine**, Laporta said last night: "I will answer as always, **Messi** is not for sale and we do not want to let **him** go."

Figure 6.1: Example of a coreference chain requiring world knowledge to be solved. Boldfaced mentions refer to the same entity.

Measure	Pre	Rec	F_1	Quantity
PN_E	99.7	99.4	99.6	356 (18%)
PN_P	94.5	77.9	85.4	222 (12%)
PN_N	**5.3**	**1.3**	2.1	**75 (4%)**
CN_E	97.3	71.8	82.6	149 (8%)
CN_P	87.3	**36.0**	51.0	**172 (9%)**
CN_N	**22.6**	**2.5**	4.5	**278 (14%)**
P_1∪2	74.5	61.2	67.2	134 (7%)
P_3G	88.8	85.0	86.9	187 (10%)
P_3U	78.1	59.3	67.4	356 (18%)
MUC	74.4	59.9	66.4	
CEAFm	83.0	83.0	83.0	
B^3	91.8	84.6	88.1	

Table 6.1: Results of RelaxCor on English OntoNotes from SemEval-2010 without world knowledge.

Name	Description
PN_E	NPs headed by a Proper Name that Exactly match (excluding case and the determiner) at least one preceding mention in the same coreference chain
PN_P	NPs headed by a Proper Name that Partially match (i.e., head match or overlap, excluding case) at least one preceding mention in the same coreference chain
PN_N	NPs headed by a Proper Name that do Not match any preceding mention in the same coreference chain
CN_E	Same definitions as in PN_E, PN_P, and PN_N,
CN_P	but referring to NPs headed by a Common Noun
CN_N	
P_1∪2	First- and second-person pronouns that corefer with a preceding mention
P_3G	Gendered third-person pronouns that corefer with a preceding mention
P_3U	Ungendered third-person pronouns that corefer with a preceding mention

Table 6.2: Description of the mention classes for English documents.

In order to know the importance of the coreference links that are missing due to a lack of world knowledge, more specific evaluation measures are needed. For this reason, the MUC measure used in the following tables is broken down depending on the mention classes. These classes are explained in Section 2.2.2, but we reproduce the descriptions in Table 6.2 for convenience. PN_N and CN_N are NPs that do not lexically match any preceding mention of their coreference chain. Some examples of PN_N are "*the President*" and "*Iraq's Deputy Prime Minister*"; examples of CN_N are "*a northern city*," "*the company*," and "*the young Argentine*." Table 6.1 shows the partial and total scores of RelaxCor on the test data set of OntoNotes 2.0, the same data set used for the English task in SemEval-2010. Analyzing the table, we observe that PN_N, CN_P, and CN_N are the classes with the lowest recall, especially PN_N and CNn. In addition, PN_N and CN_N have the lowest precision. The final column shows the number of mentions corresponding to the class of that row and the percentage representing the total number of coreferent mentions. Note that these three classes together represent 27% of coreferent mentions.

According to the study published in (Màrquez et al., 2012), the relation of partial scores for RelaxCor in Table 6.1 can be roughly generalized to any other system using similar information. In other words, most systems exhibit their worst performance for the PN_N, CN_P, and CN_N classes.Moreover, the percentage of mentions of these classes is usually around 20–30% in English corpora (Stoyanov et al., 2009) and even in other languages such as Spanish and Catalan (Màrquez et al., 2012). Therefore, these classes require attention in order to improve global performance. Note that PN_N and CN_N are NPs that do not match any preceding mention in the same coreference chain, which means that lexical information is not at all useful. Moreover, it seems clear that the other linguistic layers (morphological, syntactic, semantic) are not helping either. This is the main reason for encouraging research on adding world knowledge to coreference resolution systems.

Our experiments along this line have been focused on extracting information about the named entities of the document using Wikipedia. Once information has been extracted, the document is searched for mentions that match this information. For instance, when a document mentions "*Samsung*," the system searches "*Samsung*" in Wikipedia and extracts, among other things, that Samsung is a *company*. In the case that there are mentions in the document with the head "*company*," those mentions are marked as candidates to corefer with mention "*Samsung*." Experiments have been carried out regarding methods to extract such information from Wikipedia and incorporate it to improve coreference resolution.

In state-of-the-art systems, we can find some other attempts to incorporate world knowledge to coreference resolution, using either Wikipedia (Ponzetto and Strube, 2006; Uryupina et al., 2011) or other resources such as YAGO and Freenet (Rahman and Ng, 2011a). In our work, we developed an elaborate method to extract the information that uses some techniques not used before, such as a selection of the most informative mentions and an entity disambiguation process. Moreover, the methodology presented in this chapter also explores the incorporation of knowledge using the entity-mention model. We also include an extended error analysis that could be useful for further research in this field.

This chapter presents our approach to incorporating world knowledge to coreference resolution using Wikipedia, represented in Figure 6.2. First, a methodology to discover the real-world entities mentioned in a document is needed. This methodology selects a particular set of mentions of the document and disambiguates them in order to find the corresponding Wikipedia entry. Section 6.1 describes the process to select the most informative mentions, and Section 6.2 shows the disambiguation process. An information extraction process is then applied to Wikipedia in order to obtain some alternative names (aliases) and properties of the entities, populating our knowledge base. Section 6.3 explains the information extraction process. Next, this knowledge is incorporated to the system using two different models: feature functions and constraints. Both models are detailed in Section 6.4. Sections 6.5 and 6.6 describe our experiments and analyze their errors, respectively. Finally, the conclusion in Section 6.7 identifies some directions to follow in this research.

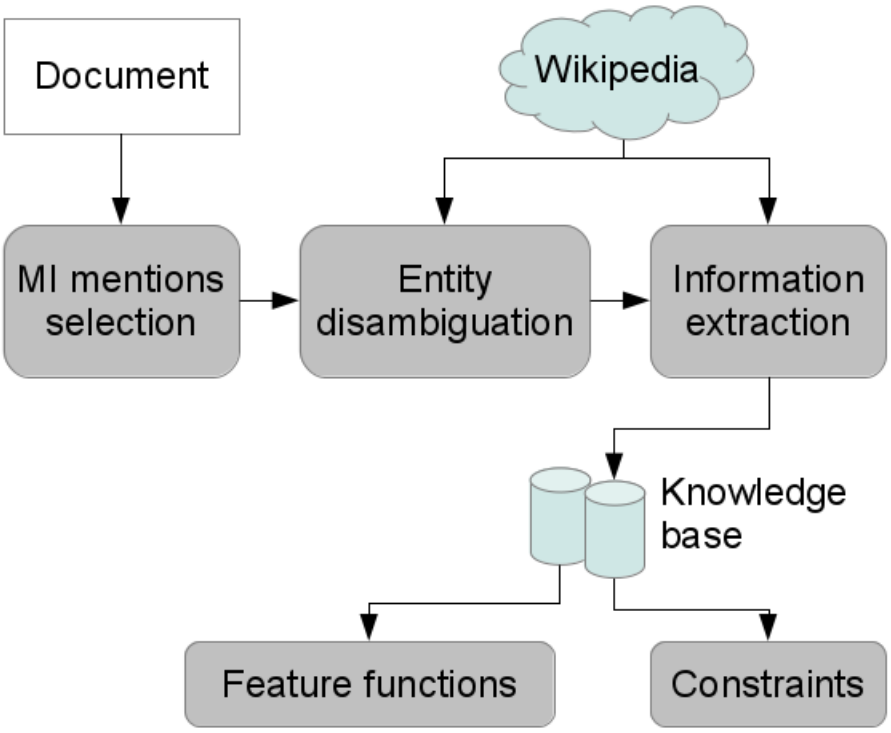

Figure 6.2: Process to add information from Wikipedia to coreference resolution.

6.1 Selecting the most informative mentions

In order to extract information about the **real-world entities** mentioned in a document, they should first be detected by taking into account the mentions in the document. In our coreference resolution approach, mentions in the document are classified as one of three types: NE mentions, pronouns, or nominal mentions. **NE mentions** are those whose heads have been annotated as a named entity by a NERC system in the preprocess. Therefore, a NE mention is the whole NP that has a NE as a head. For example, "*star striker Lionel Messi*" is a NE mention given that *Lionel Messi* is annotated as a NE. Pronoun mentions are those with pronoun heads, and nominal mentions are NPs whose head is a common noun, or even a proper name, but not annotated as a NE.

One approach to finding real-world entities mentioned in a document is to use the NE mentions of that document. However, on the one hand, not all the NE mentions are informative enough, and on the other hand, many of them may be pointing to the same real-world entity. Using every NE mention in a document may add noise to the process. For instance, consider a document with "*Bill Clinton*" and, some sentences later, "*Clinton.*" If we try to get information about "*Clinton*" from Wikipedia, we obtain a page about the English family name Clinton with a lot of non-relevant information that may lead to erroneous

results. However, given that "*Bill Clinton*" appears in the same document, it seems more convenient to select the most informative NE mention and discard the less informative ones, like "*Clinton*," which are probably pointing to the same real-world entity. Therefore, an approach to selecting the most relevant and unambiguous NE mentions has been developed as follows:

- For each class (Person, Organization, Location):
  - Let L be the list of all NE mentions of the class.
  - Apply the `ALIAS` feature function for each pair of NE mentions in L.
  - Let a *clique* be a group of NE mentions in L where `ALIAS` is true for all the pairs.
  - For each clique, select the longest NE mention.

After this selection process, a list of the most informative NE mentions, **MI mentions** from now on, is given to the next module that searches them in Wikipedia. Note that an entity disambiguation process is needed in order to find the Wikipedia entry that best fits each MI mention.

6.2 Entity disambiguation

Entity disambiguation is a NLP task that consists of determining which entity in the real world—in our case, which Wikipedia entry—is referred to by a mention. The entity disambiguation process is mandatory in order to finally extract information from Wikipedia about an entity. Given the list of MI mentions, this process has to assign a Wikipedia entry to each of them. Note that in many cases, the real entity pointed to by a NE is not clear. Therefore, in such cases, the entity disambiguation process will choose the most popular entity. For instance, the named entity "*Michael Jackson*" in a document may refer to several people in the world, from musicians to basketball players. However, the entity disambiguation process has to choose the most probable one given the named entity and its context. In addition, a named entity may refer to an entity that is not included in Wikipedia, and the entity disambiguation process should be able to detect this.

The approach to entity disambiguation in this work is very simple, given that the task is beyond the scope of this thesis. The approach uses Google as an information retrieval system to find the most relevant pages in Wikipedia given a named entity and its closest context. The process consists of the following steps:

1. A *query* is extracted from the MI mention. The **query** is defined as the head and all the nouns, proper names, and adjectives that appear immediately before the head. For example, given the mention "*the 42nd*

President of the United States Bill Clinton," the query is "*President United States Bill Clinton.*"

2. A search is conducted in Google using the template: `"<query> site:en.Wikipedia.org"`. This asks Google for the most relevant pages in English Wikipedia according to the query. If another language is used, the site can be changed to that corresponding to the target language (i.e., es.Wikipedia.org for Spanish). The first result, a Wikipedia URL, is kept and the rest are discarded.

3. The Wikipedia entry is filtered in order to avoid special pages and other types of pages that do not correspond to a Wikipedia entry. The Wikipedia URL selected in the previous step passes this filter if:

   - The URL fits this pattern: `en.Wikipedia.org/wiki/<page>`.
   - The *page* extracted in the first pattern does not fit with: `List_of*` and `Category*`.
   - The *page* name does not include the character "*:*" (used in many special pages).
   - The *page* is not a disambiguation page.
   - The Wikipedia page includes the head of the MI mention—or a string that matches as an alias—in the title or in the first sentence of the first paragraph.

If the URL selected in step 2 does not pass the filtering of step 3, the system considers that the MI mention does not exist in Wikipedia.

6.3 Information extraction

For each Wikipedia entry obtained in the entity disambiguation step, the following information extraction process is performed. The process extracts information from the entry (description, infoboxes, and categories), and also from entries linking to that entry, as found in the "What links here" section. For each entry, there are two lists to fill with as many values as possible. These lists are: **Names**, where all official names, nicknames, and aliases are included; and **Properties**, indicating the most descriptive aspects or qualities of the entity. The information extracted is added into one of these two lists. A description of the process followed to extract information at each part of the process is given below.

Description. The first paragraph of a Wikipedia entry is considered the description of the entry, excluding eventual elements such as tables of contents and infoboxes that may be placed before the description. The description typically starts with the complete name of the entity, some aliases, and the most descriptive properties of the entity. Figures 6.3 and 6.4 show the first paragraphs of the entries for UPC and Bill Clinton.

The **Technical University of Catalonia**, sometimes called **UPC-Barcelona Tech**, is the largest engineering university in Catalonia, Spain. The objectives of the UPC are based on internationalization, as it is Spain's technical university with the highest number of international PhD students and Spain's university with the highest number of international master's degree students. The UPC-Barcelona Tech is a university aiming at achieving the highest degree of engineering excellence and has bilateral agreements with several top-ranked European universities.

Figure 6.3: Description of UPC in Wikipedia.

William Jefferson "Bill" Clinton (born William Jefferson Blythe III; August 19, 1946) is an American politician who served as the 42nd President of the United States from 1993 to 2001. Inaugurated at age 46, he was the third-youngest president. He took office at the end of the Cold War, and was the first president of the baby boomer generation. Clinton has been described as a New Democrat. Many of his policies have been attributed to a centrist Third Way philosophy of governance, while on other issues his stance was center-left.

Figure 6.4: Description of Bill Clinton in Wikipedia.

The description is preprocessed to obtain tokenization, parts-of-speech, NEs, and dependency parsing. The first named entity is then extracted as the official name. Moreover, the official name is usually boldfaced. Next, a set of patterns combining strings and parts-of-speech extract the aliases that are typically found just after the official name. For example, in Figure 6.3 the pattern is "sometimes called `<alias>`." Official names and aliases are added to the *Names* list. After names and aliases, a set of patterns extract the most descriptive qualities or aspects of the entities. The patterns are basically the verbs "be" and "become," followed by a NP. That NP is extracted as a descriptive NP. In addition, the head and the term of each descriptive NP are also extracted. All three (NP, term, and head) are added to the *Properties*. Figures 6.5 and 6.6 show the properties extracted for UPC and Bill Clinton, respectively.

Infoboxes and categories are the most structured part of Wikipedia's content, and therefore the easiest from which to extract information. From infoboxes, all the contents of the following fields are extracted: `fullname`, `name`, `office`, `title`, `profession`, `company_name`, `playername`, `occupation`, `nickname`,

Descriptive NP	Term	Head
the largest engineering university in Catalonia	largest engineering university	university
Spain's technical university with the highest number of international PhD students	technical university	university
a university aiming at achieving the highest degree of engineering excellence	university	university

Figure 6.5: Properties extracted from the description of UPC.

Descriptive NP	Term	Head
an American politician who served as the 42nd President of the United States from 1993 to 2001	American politician	politician
the third-youngest president	third-youngest president	president
the first president of the baby boomer generation	first president	president
center-left	center-left	center-left

Figure 6.6: Properties extracted from the description of Bill Clinton.

`official_name`, `native_name`, `settlement_type`, `type`. The values of the fields related to names and aliases are added to the *Names* list, while the others are added to the *Properties* list. All categories are also added into *Properties*.

What links here is a special page of Wikipedia that lists the entries that link to the current entry. Note that the information gathered from the entry is the official information, but it is not always the description that people most commonly use to refer to that entity. For instance, extracting information from the description, infoboxes, and categories of the entry "*Samsung*," we find that Samsung is *"a South Korean [multinational conglomerate [corporation]]"* from the description and a *company* taking into account the categories. However, looking into the entries that link to Samsung, new properties can be found such as *manufacturer*, *competitor*, and *electronics company*.

The methodology to extract information from the entries linking to the current entry is as follows. First, sentences including a link to the current entry are selected and the rest of the document is discarded. For each sentence, a set of patterns are matched in order to extract new information. The patterns are as follows:

- **Anchor text.** The text used to link to the entry, which is typically the name or an alias. All the anchor texts used to link the entry are added to the *Names* of the entity. The pattern takes advantage of the wiki format, where links are annotated inside brackets. Pattern: `[[entry|<anchor text>]]`.
- **Left term.** The set of nouns and adjectives to the left of the anchor text are added to the *Properties* list. Pattern: `(NNP?|JJ)* [[entry(|*)?]]`.
- **Such as.** In some cases, the entry is linked in the middle of a comma-separated list of other similar or related entries. In many cases, these lists are introduced by a sentence including some information about the following listed entries, and an expression such as `include`, `such as`, or `like`. The pattern is then defined as follows: "`<property> such as entry1, entry2,..., entryN`" where one of the listed entries is the current entry.
- **Appositions**. Similar to coreference resolution, a document linking to an entry that has a NP in apposition is probably describing some property of

Nouns and adjectives at the left of the anchor text:
...the logo of **electronics company** *Samsung*, and the logo of the engineering consultancy Atkins...

`include` and `such as` patterns:
Cash register manufacturers `include` CHD, ELCOM, SAM4S, Casio, NCR, IBM, Panasonic, *Samsung*, Wincor-Nixdorf, Uniwell, RCH S.p.A., United Bank Card, Sharp, ...
Major competitors today `include`, in the main business, Alcatel-Lucent, Huawei, Nokia Siemens Networks and ZTE, with Cisco, IBM, EDS, Accenture, Nokia, Motorola, *Samsung*, LG Electronics, NEC, Sharp and most recently Apple Inc., competing with aspects of the business.
Korean companies `such as` LG, Hyundai and *Samsung* have established...

Figure 6.7: Properties of Samsung extracted from Wikipedia entries linking to Samsung.

the linked entry. So, NPs in apposition to the link are also added to the *Properties* list. Pattern: `[[entry(|*)?]], <noun phrase>`.

Figure 6.7 shows some examples of the extraction patterns for the entry Samsung. These sentences have been taken from entries in Wikipedia linking to the entry Samsung.

All the NPs, terms, and heads extracted are added to the *Names* or *Properties* list with an associated counter. In the case that an expression was already in the list, the counter is increased. This value is analogous to a confidence value associated with each expression—the most repeated expressions are the most reliable. In order to avoid incorrect information as much as possible, we define a threshold below which all the *Names* and *Properties* are discarded.

6.4 Models to incorporate knowledge

Once information about the entities mentioned in a document has been extracted from Wikipedia, the next step is to find a way to incorporate such information to the coreference resolution system. Two approaches for the incorporation of this knowledge have been studied. The first is to add some feature functions for the mention-pair model that evaluate whether a pair of mentions may corefer according to Wikipedia's information, similar to other state-of-the-art studies (Ponzetto and Strube, 2006; Rahman and Ng, 2011a). The second approach adds a set of constraints to the hypergraph connecting groups of mentions, using the entity-mention model.

6.4.1 Features

In order to incorporate the knowledge extracted from Wikipedia, a set of feature functions are defined. In this approach, these new feature functions are used to evaluate pairs of mentions, and some learned constraints may use them as any other feature function. As explained in previous sections, a MI mention may be assigned to a Wikipedia entry, and then an information extraction process obtains *Names* and *Properties*. These feature functions are only applied to pairs including a MI mention and any other mention but pronouns, and use the information in *Names* and *Properties* to determine its value.

- `WIKI_ALIAS(MI mention,` X`)`: This function compares the names in the *Names* list with the head of X. If the strings match, a true value is returned. A true value is also returned when X is a MI mention assigned to the same entry. Otherwise, the function returns false.
- `WIKI_DESC(MI mention,` X`)`: This function compares the properties in the *Properties* list with the X term. When all the words of the X term are included in a property, the returned value is true. Otherwise, the function returns false.

6.4.2 Constraints

World knowledge can also be incorporated by adding constraints relating the mentions that may corefer given the extracted information about the entities. In this case, the features of the previous model are now replaced by constraints. In addition, other constraints can be added to take advantage of the entity-mention model. The following is a list of constraints used in our experiments.

- Constraint `cAlias` is added for each pair of mentions that satisfy the same conditions as `WIKI_ALIAS`.
- Constraint `cDesc` is added for each pair of mentions that satisfy the same conditions as `WIKI_DESC`.
- Constraint `cWiki3`, a $N = 3$ constraint, is added for each combination of three mentions (0, 1, 2) where 0 is a MI mention, 1 is a NE mention alias of 0, and 2 is a nominal mention or a NE mention that satisfies `WIKI_ALIAS` or `WIKI_DESC` with 0. This constraint tries to link the nominal mention with the closest NE mention that may corefer. The influence rule is $(0, 1)_A \Rightarrow (2)_A$, i.e., 2 is influenced when 0 and 1 corefer.
- Constraint `cStructWiki3`, an $N = 3$ constraint, is added for each combination of three mentions (0, 1, 2) where 0 is a MI mention, 1 is a NP that satisfies `WIKI_ALIAS` or `WIKI_DESC` with 0, and 2 is a NE mention alias of 0. In addition, the three mentions have the same syntactic function and are found in consecutive sentences. The influence rule associated with this constraint is $(0, 2)_A \Rightarrow (1)_A$, i.e., 1 is influenced when 0 and 2 corefer.

`cWiki3`
Output from *the Organization of Petroleum Exporting Countries* is already... As a result, the effort by some oil ministers to get *OPEC* to approve... *The organization* is scheduled to meet in Vienna...

`cStructWiki3`
Google Inc. is offering new applications... *The company* is going to... Predictably, *Google* has highlighted user profiles...

Figure 6.8: Examples of the application of $N = 3$ constraints `cWiki3` and `cStructWiki3`.

Figure 6.8 shows examples of the constraints `cWiki3` and `cStructWiki3`. The idea behind `cWiki3` is to link the nominal mention (2, *The organization*) with a closer mention in the document than the MI mention (0, *the Organization of Petroleum Exporting Countries*). Linking nearest mentions may take advantage of information given by other constraints, such as syntactic patterns. When *the Organization of Petroleum Exporting Countries* is tending to corefer with *OPEC*, mention *The organization* is influenced by both mentions. The second case, `cStructWiki3`, takes advantage of a typical discourse structure where the same entity is the subject of some consecutive sentences, in this case three. First mention 0, *Google Inc.*, is the MI mention, whereas 2 (*Google*) is just an alias. In the middle of these we find a nominal mention (*The company*), which is the one we wish to find using world knowledge. Both $N = 3$ constraints are expected to have high precision but low recall.

Note that `cAlias` and `cWiki` are equivalent to the feature functions of the previous model. The difference is that, in the case of constraints, they are always applied when `WIKI_ALIAS` and `WIKI_DESC` are true, and so their weight is added to the edge weight of that pair in the hypergraph. However, in the case of the model using feature functions, the application of constraints to that pair depends on the constraints learned by the model.

6.5 Experiments and results

The experiments consist of the execution of RelaxCor using each one of the models to incorporate information. RelaxCor + features incorporates the new features to the original model and repeats the training process from the beginning. Constraints are learned using these new feature functions mixed with all the others (a detailed list of features is in Section 4.2). RelaxCor + constraints incorporates the new constraints. In this case, the learning process uses the constraints already learned for RelaxCor and adds the new constraints to the model. The training process then follows its common behavior by calculating the weight of the constraints using their precision in the training files.

	Baseline			Features			Constraints		
Measure	Pre	Rec	F_1	Pre	Rec	F_1	Pre	Rec	F_1
PN_E	99.7	99.4	99.6	**100**	*98.0*	*99.0*	**100**	*99.2*	99.6
PN_P	94.5	77.9	85.4	*92.9*	*76.6*	*84.0*	*93.6*	**78.8**	**85.6**
PN_N	5.3	1.3	2.1	**15.0**	**4.0**	**6.3**	**14.8**	**5.3**	**7.8**
CN_E	97.3	71.8	82.6	97.3	**72.5**	**83.1**	97.3	**72.5**	**83.1**
CN_P	87.3	36.0	51.0	**90.2**	**43.0**	**58.3**	**89.2**	**43.0**	**58.0**
CN_N	22.6	2.5	4.5	**32.1**	**3.2**	**5.9**	**31.0**	**3.2**	**5.9**
P_1∪2	74.5	61.2	67.2	**76.9**	*59.7*	67.2	**77.1**	*60.4*	**67.8**
P_3G	88.8	85.0	86.9	*87.6*	**86.6**	**87.1**	*87.6*	**86.6**	**87.1**
P_3U	78.1	59.3	67.4	*76.2*	*54.8*	*63.7*	*76.1*	*55.3*	*64.1*
MUC	74.4	59.9	66.4	**75.9**	*59.6*	**66.8**	**75.4**	**60.3**	**67.0**
CEAFm	83.0	83.0	83.0	**83.4**	**83.4**	**83.4**	**83.5**	**83.5**	**83.5**
B^3	91.8	84.6	88.1	**92.6**	*84.5*	**88.4**	**92.3**	**84.7**	**88.4**

Table 6.3: Results of RelaxCor on English OntoNotes 2.0 from SemEval-2010 with world knowledge. Baseline means RelaxCor using mention-pair model, Features is RelaxCor using features `WIKI_ALIAS` and `WIKI_DESC`, and Constraints means RelaxCor with the set of constraints of Section 6.4.2. Scores higher than the baseline are boldfaced and scores lower than the baseline are italicized.

Table 6.3 shows the results obtained when adding world knowledge compared to the results of RelaxCor without world knowledge. The first three columns list the results of RelaxCor using the mention-pair model, as explained in Chapter 4, the next three columns are the results of RelaxCor adding the features of Section 6.4.1, and the final three columns are the scores for RelaxCor with the constraints of Section 6.4.2. Improvements are boldfaced while a decreased score is shown in italics. Note that the main improvements are focused around PN_N, CN_P, and CN_N as expected. Moreover, the global scores also improve, but the global improvements are not statistically significant.

While there are improvements in our target classes (PN_N, CN_P, and CN_N), there are some collateral effects that decrease the performance for other classes such as PN_P and P_3U (ungendered pronouns: *"it"*). The latter is a strong decrease and, given that the class P_3U represents 18% of the total coreferent mentions, this affects the global results. This decrease in pronoun classification performance is related to the balance value learned in the development process.

Another phenomenon to take into account in the case of RelaxCor + features is that the improvement in global scores is in precision but not in recall. This is because the development process is optimizing scores for the CEAF measure, which encourages precision more than recall compared with the MUC scorer.

Regrettably, the improvements achieved seem too little given the necessary effort to extract the knowledge.

Microsoft has bought *Illusion Novelty*. The owners of *the company in New York* will receive about 10 million dollars...

Figure 6.9: Example of an erroneous coreference link due to a lack of information about Illusion Novelty. Given that the system has extracted that Microsoft is a company but there is no information about Illusion Novelty, a link is added between *Microsoft* and *the company* causing an erroneous coreference link.

6.6 Error analysis

Errors in output have been analyzed for both experiments (using features and constraints). In general terms, we have found the following main problems:

- The number of new coreferential relations found is too few.
  - Most of the cases affected by the new information are coreference relations already found by other clues, such as the alias function, apposition, or some syntactic patterns.
  - New coreference relations are found, but the weights of the constraints that link the mentions is not high enough to change the final results. Increasing these weights causes many new errors in other cases with unreliable information.
  - The information is not found in Wikipedia, or the incorporation (`WIKI_ALIAS`, `WIKI_DESC`, or constraints) is not taking this information into account and is finally ignored.
  - Many MI mentions are not found in Wikipedia.
- The addition of noise (incorrect coreference links).
  - The newly added information, even when it is correct, is not always synonymous with coreference. Coreference relations depend on the context. For example, the system may extract that *Pau Gasol* is a *basketball player*, but this does not mean that each mention of a *basketball player* in the document corefers with *Pau Gasol*.
  - Given that not every MI mention is found in Wikipedia, or the information of some MI mentions may be incomplete, the most popular MI mention with most information in the knowledge base is the one having more links to other mentions. This situation causes an imbalance towards the most populated entities in the knowledge base. Figure 6.9 shows an example. *Microsoft* is found in Wikipedia, and its *Names* and *Properties* lists are highly populated. However, *Illusion Novelty* is not found in Wikipedia. This causes the system to prefer to link the NP headed by *company* with *Microsoft* instead of the correct one, which is *Illusion Novelty* in this case.

In addition, each step of the process contributes by losing some information that might be useful for finding some coreferent relations, and also by adding errors, noise, and incorrect information that causes misclassifications at the end:

- The named entity selection may be missing some named entities that seem like aliases of others but do not refer to the same entity. For instance, in a sports newspaper we may find *Inter Milan* and *Milan* referring to two different football clubs, but they may erroneously be considered aliases of the same entity. Consequently, *Milan* is not considered a MI mention and no coreference relations are added.

- The entity disambiguation process may be missing some entities or incorrectly linking them to Wikipedia entries corresponding to other entities.

- The information extraction process is introducing incorrect information. This problem is mainly solved by filtering the *Names* and *Properties* with a low confidence value, but this filtering also discards some useful information.

In summary, although performance is slightly improved on average, few new coreference relations are found, taking into account the potential for improvement identified in the introduction. Moreover, some of these new relations do not change the final output and, even worse, many of them are incorrect. In addition, some coreferences that were correctly solved before this process are now incorrectly classified. In particular, the recall of ungendered pronouns has decreased considerably.

6.7 Conclusions and further work

Although it is clearly necessary to incorporate world knowledge to move forward in the field of coreference resolution, the process required to introduce such information in a constructive way has not yet been found. In this thesis, we tested a methodology that identified the real-world entities referred to in a document, extracted information about them from Wikipedia, and then incorporated this information in two different ways in the model. However, it seems that neither of the two forms work very well, and that the results and errors are in the same direction: the slight improvement of the few new relationships is offset by the added noise. Other state-of-the-art systems have better improvements than ours (Ponzetto and Strube, 2006; Uryupina et al., 2011; Rahman and Ng, 2011a), but these also seem too modest given the large amount of information used and the room for improvement outlined in the introduction.

The problem seems to lie with the extracted information rather than the model to incorporate it. The extracted information is biased in favor of the more famous and popular entities –those in Wikipedia, and having larger entries– that causes the system to find more information about these entities over the rest, including false positives and causing an imbalance against entities with little or no information in Wikipedia. On the other hand, it is not possible to use negative information in the absence of complete information. For instance, given the example of Figure 6.9 about *Microsoft*, *Illusion Novelty*, and *New York*, although there is no information about *Illusion Novelty*, *Microsoft* and *the*

company in New York could be negatively linked given that the knowledge base knows that *Microsoft* is in *Redmond*. Thus, one might expect *Illusion Novelty* and *company in New York* to be linked by elimination. But the extracted information would have to be far more reliable and complete than currently. For example, once information has been extracted about *Samsung* and it is known to be a *company* and an *electronic manufacturer*, should it be negatively linked with mentions like *the supplier*? Clearly not. One possible solution would be to use ontologies to add a logical reasoning process to incorporation of the information. In this way, would know that *companies* and *suppliers* are compatible, while *Redmond* and *New York* are not. But this reasoning is still beyond our reach, and may be more complicated than the task of coreference resolution itself.

Therefore, we believe that research in this field should focus on the extraction of more reliable and concise information, so that the information added, no matter how little, should always be constructive and avoid false positives. On the other hand, we would need to find some process of reasoning to expand the scope of the information obtained using logic and common sense. Only then could the full potential of the knowledge base be exploited.

Chapter 7

Conclusions

As we have seen in the summary of the state of the art, two possible directions that research in coreference resolution should follow are the use of more expressive models than mention-pairs to manage the problem, such as entity-mention, and the incorporation of new information, such as world knowledge and discourse coherence. In some cases, this information cannot be expressed in terms of pairs of mentions. That is, it is information that involves either several mentions at once or partial entities. Therefore, an experimental approach in this field requires the expressiveness of the entity-mention model combined with the most typical features of the mention-pair model.

In this thesis, we defined an approach based on constraint satisfaction that represented the problem in a hypergraph and solved it by relaxation labeling, reducing coreference resolution to a hypergraph partitioning problem under a set of constraints. Our approach managed mention-pair and entity-mention models at the same time, and was able to introduce new information by adding as many constraints as necessary. Furthermore, our approach overcame the weaknesses of previous approaches in state-of-the-art systems, such as linking contradictions, classifications without context, and a lack of information in evaluating pairs.

The system developed, RelaxCor, has achieved state-of-the-art results using only the mention-pair model without new knowledge. Moreover, experiments with the entity-mention model showed how the system is able to introduce knowledge in a constructive way.

In addition, as explained in Section 4.3.1, we have proposed a method based on the clustering of examples in which all positive examples are included, while the negative examples most similar to the positive ones are kept and the rest are discarded. This method reduces the number of negative examples without losing any positive information.

Regarding the feature function selection, many works just manually select the most informative feature functions and discard the noisy ones. Few re-

searchers have incorporated an automatic feature function selection process. We have made a small contribution in this area by selecting feature functions through a Hill Climbing process (Sapena et al., 2010a).

The other contributions of this thesis include techniques for performance optimization such as balance, pruning, and reordering. The balance parameter was used to find the optimal point between precision and recall, while the pruning process reduced the computational cost and avoids the system performance being dependent on the size of the documents. Both techniques were included in the development process that facilitated the optimization of the system for a target measure. The reordering process improved performance by reducing the number of possible labels assigned to the most informative mentions, which caused the most reliable coreferential relations to be resolved first.

Experiments to add world knowledge were performed in order to improve the coreference resolution performance. Although these last experiments did not achieve a significant improvement, the reason seems to be more related to the type and source of information and its extraction than the approach used to incorporate it.

The list of publications in Appendix A shows the contribution of our research to the field of coreference resolution and other related fields such as alias assignment and entity disambiguation. We also participated in the organization of SemEval Task 1: "Coreference resolution in multiple languages," providing an insight to the state of the art. In addition, this thesis contributed to the research community by releasing the code for RelaxCor and a scorer, which was used in the SemEval-2010 and CoNLL-2011 shared tasks on coreference resolution.

Regarding research in coreference resolution, the immediate future seems to be evolving towards the following areas:

- The addition of semantic and pragmatic knowledge. As seen in Chapter 6, there is room for improvement in the cases where more knowledge is needed. About 30% of coreference relations need some semantic information to be solved. A main line of research in coreference resolution should be the incorporation of semantic and pragmatic knowledge.
- New representations of the problem. The hypergraph with weighted hyperedges proposed in this thesis has been proven to be a good solution for representing the problem. However, the corpora used to train and test it is a set of documents with a size limitation, such as newspaper news and blogs. Other types of input to this model, and many of the state-of-the-art models so far, can be unfeasible. For instance, is hypergraph partitioning an appropriate representation to resolve the coreferences of a whole book? A high number of vertices may increase the computational cost to unaffordable limits. As another example, suppose that a machine translation system for an international meeting requires coreference resolution to give a good translation, but translations are conducted in real time. Can coreferences be resolved in real time?

- Better training methods and instance selection methods. Even though this field is more related to general machine learning, some training and instance selection methods can be studied in order to discriminate the useful information from the rest.

- Unsupervised and weakly supervised approaches. An interesting area is the research of unsupervised and weakly supervised approaches, as the scarcity of annotated corpora for training and testing is a bottleneck for research in supervised technology.

- Multilingual systems. Although most coreference resolution systems are not language dependent, research into coreferences in multiple languages is expected to be undertaken in the immediate future. Indeed, the SemEval-2010 task was for coreferences in multiple languages, and in 2012 there is going to be another CoNLL shared task dedicated to coreference resolution in multiple languages.

Agraïments

M'agradaria dedicar les últimes paraules d'aquesta tesi a fer un petit homenatge a les persones que m'han influït positivament durant tot aquest temps.

En primer lloc, un sincer agraïment als meus directors de tesi, Jordi Turmo i Lluís Padró. Per la seva ajuda, paciència i comprensió. Sense la seva guia i el seu coneixement aquesta tesi no hauria estat possible.

També és de ben nascut ser agraït amb qui posa els diners. En el cas d'aquesta tesi, son un complexe conjunt de projectes de recerca finançats en part pel ja desaparegut Ministerio de Ciencia e Innovación i per fons europeus[1]. Així que, gràcies diners públics!

Vull donar les gràcies també a tots els companys de despatx –i de despatxos propers– que he tingut durant aquest temps. Edgar, Pere, Maria, Meritxell, Ignasi, Montse, Roberto, Jordi, Jesús, Xavi, Muntsa, Cristina, Sergi, Leo, Gemma, Marina, Solmaz i Stefan. Gràcies pel vostre bon humor, pels cafès de cada dia, per les freak excursions, per les chancheces (incloent les chancheces finales!), per les partides de tants jocs i per tot el que hem fet junts durant aquests anys però, sobretot, per haver-me estat acompanyant en aquest viatge.

Per últim, però no per això menys important, una forta abraçada a la meva família i els meus amics que han sabut recolzar-me en tot moment. Emili (papa), M. Antonieta (mama), Eli, Victor, Alex i Mireia. A vosaltres, a la resta de la família i a tots el meus amics, gràcies per suportar-me i creure en mi!

I com no, el meu agraïment més especial és per la Celia. Gràcies Celia per estar al meu costat i estimar-me tal com soc.

A tots, moltes gràcies!

[1]Research supported by the Spanish Science and Innovation Ministry, via KNOW project (TIN2006-15049-C03-01) and KNOW2 project (TIN2009-14715-C04-04), and from the European Community's Seventh Framework Programme (FP7/2007-2013) under Grant Agreement number 247762 (FAUST)

Appendix A

List of publications

- Emili Sapena, Lluís Padró and Jordi Turmo. **Alias Assignment in Information Extraction** *Procesamiento del Lenguaje Natural, pg. 105–112.* September, 2007. (Sapena et al., 2007)

  This paper is a first approach towards coreference resolution but focuses on alias assignment. Alias assignment is an Information Extraction task related to Coreference Resolution and Entity Matching. Coreference resolution determines whether some mentions in the document, such as Michael Jacksonand Jackson, refer to the same real-world entity, while Entity Matching tackles the same problem with data extracted from different documents or databases. Alias assignment decides whether a named entity can be an alias of an entity or a set of entities in a database. Variations in named entity expressions are due to multiple reasons: use of abbreviations, different naming conventions (for example, Name Surname and Surname, N.), aliases, misspellings, or naming variations over time (for example, Leningrad and Saint Petersburg). This paper describes a machine learning method for alias assignment.

- Emili Sapena, Lluís Padró and Jordi Turmo. **A Graph Partitioning Approach to Entity Disambiguation Using Uncertain Information** *Advances in Natural Language Processing, pg. 428–439.* 6th International Conference, GoTAL 2008. August, 2008. (Sapena et al., 2008)

  This paper uses a graph partitioning approach solved by relaxation labeling to solve an Entity Disambiguation problem. Entity Disambiguation and Coreference Resolution are similar problems from the point of view of problem representation and resolution. While Coreference Resolution decides which mentions in a document refer to the same entity, Entity Disambiguation decides which entries in a database should be joined because they refer to the same entity. This paper was our first experiment representing the problem in a graph and solving it by relaxation labeling.

- Marta Recasens, Toni Martí, Mariona Taulé, Lluís Màrquez and Emili Sapena. **SemEval-2010 Task 1: Coreference Resolution in Multiple Languages**. *SEW-2009 Semantic Evaluations: Recent Achievements and Future Directions*, 2009. (Recasens et al., 2009)

  This paper presents the task "Coreference Resolution in Multiple Languages" to be run at SemEval-2010 (5th International Workshop on Semantic Evaluations). This proposal is in collaboration with linguists from Universitat de Barcelona (Marta Recasens, Toni Martí and Mariona Taulé). This task aims to evaluate and compare automatic coreference resolution systems for three different languages (Catalan, English, and Spanish) by means of two alternative evaluation metrics, thus providing an insight into (i) the portability of coreference resolution systems across languages, and (ii) the effect of different scoring metrics on ranking the output of the participant systems. This paper talks about three languages, but we finally run the task with six languages, adding Italian, Dutch, and German to the list.

- Emili Sapena, Lluís Padró and Jordi Turmo. **A Global Relaxation Labeling Approach to Coreference Resolution**. *Proceedings of the 23rd International Conference on Computational Linguistics, COLING*, Beijing, China. August, 2010. (Sapena et al., 2010a)

  This paper presents our approach to coreference resolution (explained in Chapter 3) using just the mention-pair model. This is the first version of RelaxCor (the implementation of the approach, Chapter 4). The approach achieves performances in the state of the art.

- Emili Sapena, Lluís Padró and Jordi Turmo. **RelaxCor: A Global Relaxation Labeling Approach to Coreference Resolution** *Proceedings of the ACL Workshop on Semantic Evaluations (SemEval-2010)*, Uppsala, Sweden. July, 2010. (Sapena et al., 2010b)

  Our participation in SemEval-2010 Task 1: Coreference Resolution in Multiple Languages. RelaxCor achieves a good performance in the three languages we participate in: English, Spanish, and Catalan. Given the disparity of run scenarios and scores with different measures, it was not possible to determine a winner. However, RelaxCor achieves the best scores with the CEAF and B^3 measures in some scenarios, and demonstrates robustness across languages.

- Marta Recasens, Lluís Màrquez, Emili Sapena, Toni Martí and Mariona Taulé. **SemEval-2010 Task 1: Coreference Resolution in Multiple Languages**. *Proceedings of the ACL Workshop on Semantic Evaluations (SemEval-2010)*, Uppsala, Sweden. July, 2010. (Recasens et al., 2010c)

This paper summarizes the task of SemEval-2010: Coreference Resolution in Multiple Languages. The goal was to evaluate and compare automatic coreference resolution systems for six different languages (Catalan, Dutch, English, German, Italian, and Spanish) in four evaluation settings and using four different metrics. Such a rich scenario had the potential to provide insight into key issues concerning coreference resolution: (i) the portability of systems across languages, (ii) the relevance of different levels of linguistic information, and (iii) the behavior of scoring metrics.

- Emili Sapena, Lluís Padró and Jordi Turmo. **RelaxCor Participation in CoNLL Shared Task on Coreference Resolution**. *Proceedings of the Fifteenth Conference on Computational Natural Language Learning (CoNLL): Shared Task, pg. 35–39.* Association for Computational Linguistics. Portland, Oregon, USA. June, 2011. (Sapena et al., 2011)

  Our participation in the CoNLL-2011 Shared Task: Modeling Unrestricted Coreference in OntoNotes (Pradhan et al., 2011). This task attracted several researchers in the area and was very competitive. The organization was focused on a simplified scenario with one language (English) and one measure (an average of three common measures). This simplification allows the organization to easily compare outputs and determine a final ranking. The scenario was as realistic as possible, forcing the use of mention detection systems and using only automatic preprocessing information. RelaxCor achieved second position in the official ranking. This paper describes our participation and the changes in the approach from the version of the previous year.

- Lluís Màrquez, Marta Recasens and Emili Sapena. **Coreference Resolution: An Empirical Study Based on SemEval-2010 Task 1**. *LRE special issue* (Accepted). (Màrquez et al., 2012)

  This paper presents an empirical evaluation of coreference resolution that covers several interrelated dimensions. The main goal is to complete the comparative analysis from the SemEval-2010 task on *Coreference Resolution in Multiple Languages*. To do so, the study restricts the number of languages and systems involved, but extends and deepens the analysis of the system outputs, including a more qualitative discussion. The paper compares three automatic coreference resolution systems for three languages (English, Catalan, and Spanish) in four evaluation settings, and using four evaluation measures. Although the different dimensions are strongly interdependent, making it very difficult to extract general principles, the study reveals a series of interesting issues in relation to coreference resolution: the portability of systems across languages, the influence of the type and quality of input annotations, and the behavior of scoring measures.

Bibliography

References

C. Aone and S.W. Bennett. 1995. Evaluating automated and manual acquisition of anaphora resolution strategies. In *Proceedings of the 33rd annual meeting on ACL*, pages 122–129.

N. Asher and A. Lascarides. 1998. Bridging. *Journal of Semantics*, 15(1):83–113.

J. Atserias. 2006. *Towards Robustness in Natural Language Understanding.* Ph.D. Thesis, Dept. Lenguajes y Sistemas Informáticos. Euskal Herriko Unibertsitatea. Donosti. Spain.

S. Azzam, K. Humphreys, and R. Gaizauskas. 1999. Using coreference chains for text summarization. In *Proceedings of the Workshop on Coreference and its Applications*, pages 77–84. Association for Computational Linguistics.

S. Azzam. 1996. Resolving anaphors in embedded sentences. In *Proceedings of the 34th annual meeting on Association for Computational Linguistics*, pages 263–268, Morristown, NJ, USA. Association for Computational Linguistics.

A. Bagga and B. Baldwin. 1998a. Algorithms for scoring coreference chains. *Proceedings of the Linguistic Coreference Workshop at LREC*, pages 563–566.

A. Bagga and B. Baldwin. 1998b. Algorithms for scoring coreference chains. In *Proceedings of the Linguistic Coreference Workshop at LREC 98*, pages 563–566, Granada, Spain.

C.F. Baker, C.J. Fillmore, and J.B. Lowe. 1998. The berkeley framenet project. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics-Volume 1*, pages 86–90. Association for Computational Linguistics.

D. Bean, E. Riloff, S. Dumais, D. Marcu, and S. Roukos. 2004. Unsupervised Learning of Contextual Role Knowledge for Coreference Resolution. *HLT-NAACL 2004: Main Proceedings*, pages 297–304.

E. Bengtson and D. Roth. 2008. Understanding the value of features for coreference resolution. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 294–303. Association for Computational Linguistics.

G. Bouma, R. Groningen, and W. Daelemans. 2005. Coreference resolution for extracting answers corea.

S. Brennan, M. Friedman, and C. Pollard. 1987. A centering approach to pronouns. In *Proceedings of the 25th annual meeting on Association for Computational Linguistics*, pages 155–162, Morristown, NJ, USA. Association for Computational Linguistics.

A. Brito and R. Carvalho. 1989. *Logic grammars and pronominal anaphora.* Ph.D. thesis, Berkshire, UK, UK.

J. Cai and M. Strube. 2010a. End-to-end coreference resolution via hypergraph partitioning. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 143–151. Association for Computational Linguistics.

J. Cai and M. Strube. 2010b. Evaluation metrics for end-to-end coreference resolution systems. In *Proceedings of SIGDIAL*, pages 28–36, University of Tokyo, Japan.

C. Cardie and K. Wagstaff. 1999. Noun phrase coreference as clustering. *Proceedings of the 1999 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, pages 82–89.

D. Carter. 1986. A shallow processing approach to anaphor resolution. Technical Report UCAM-CL-TR-88, University of Cambridge, Computer Laboratory, 15 JJ Thomson Avenue, Cambridge CB3 0FD, United Kingdom, phone +44 1223 763500, May.

N. Chomsky. 1981. *Lectures on Government and Binding.* Foris, Dordrecht.

H.H. Clark. 1977. Bridging. *Thinking*, pages 411–420.

M. Collins. 1999. *Head-driven statistical models for natural language parsing.* Ph.D. thesis, University of Pennsylvania.

D. Connolly, J.D. Burger, and D.S. Day. 1997. A machine learning approach to anaphoric reference. In *New Methods in Language Processing*, pages 133–144.

A. Culotta, M. Wick, and A. McCallum. 2007. First-Order Probabilistic Models for Coreference Resolution. *Proceedings of NAACL HLT*, pages 81–88.

R. de Salvo Braz, E. Amir, and D. Roth. 2005. Lifted First-Order Probabilistic Inference. *IJCAI.*

P. Denis and J. Baldridge. 2007. Joint Determination of Anaphoricity and Coreference Resolution using Integer Programming. *Proceedings of NAACL HLT*, pages 236–243.

P. Denis and J. Baldridge. 2008. Specialized models and ranking for coreference resolution. *Proceedings of the EMNLP, Hawaii, USA.*

P. Denis and J. Baldridge. 2009. Global joint models for coreference resolution and named entity classification. *Procesamiento del Lenguaje Natural*, 42:87–96.

P. Denis. 2007. *New Learning Models for Robust Reference Resolution.* Ph.D. dissertation, University of Texas at Austin.

J.R. Finkel and C.D. Manning. 2008. Enforcing transitivity in coreference resolution. In *Proceedings of the 46th Annual Meeting of the ACL HLT: Short Papers*, pages 45–48. Association for Computational Linguistics.

T. Finley and T. Joachims. 2005. Supervised clustering with support vector machines. *ACM International Conference Proceeding Series*, 119:217–224.

R. Grishman and B. Sundheim. 1996. Message Understanding Conference-6: a brief history. *Proceedings of the 16th conference on Computational linguistics-Volume 1*, pages 466–471.

B. Grosz, A. Joshi, and S. Weinstein. 1983. Providing a unified account of definite noun phrases in discourse. In *Proceedings of the 21st annual meeting on Association for Computational Linguistics*, pages 44–50, Morristown, NJ, USA. Association for Computational Linguistics.

L. Haegeman. 1994. *Introduction to Government and Binding Theory.* Blackwell Publishers.

A. Haghighi and D. Klein. 2007. Unsupervised coreference resolution in a nonparametric bayesian model. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 848–855, Prague, Czech Republic, June. Association for Computational Linguistics.

U. Hahn and M. Strube. 1997. Centering in-the-large: computing referential discourse segments. In *Proceedings of the eighth conference on European chapter of the Association for Computational Linguistics*, pages 104–111, Morristown, NJ, USA. Association for Computational Linguistics.

S. Hammami, L. Belguith, and A.B. Hamadou. 2005. Arabic Anaphora Resolution: Corpora Annotation with Coreferential Links. *The International Arab Journal of Information Technology*, 6(5):481–489.

E. Hinrichs, S. Kübler, R. Kübler, and K. Naumann. 2005. A unified representation for morphological, syntactic, semantic, and referential annotations. In *In ACL Workshop on Frontiers in Corpus Annotation II: Pie in the Sky, Ann Arbor*.

V. Hoste and G. De Pauw. 2006. Knack-2002: a richly annotated corpus of dutch written text. In *Proceedings of The fifth international conference on Language Resources and Evaluation*, pages 1432–1437. ELRA.

V. Hoste. 2005. *Optimization Issues in Machine Learning of Coreference Resolution.* PhD thesis.

R. Iida, K. Inui, and Y. Matsumoto. 2006. Exploiting syntactic patterns as clues in zero-anaphora resolution. In *Proceedings Coling/ACL*, pages 625–632. Association for Computational Linguistics.

R. Ingria and D. Stallard. 1989. A computational mechanism for pronominal reference. In *Meeting of the Association for Computational Linguistics*, pages 262–271.

H. Ji, D. Westbrook, and R. Grishman. 2005. Using semantic relations to refine coreference decisions. *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 17–24.

M. Klenner and É. Ailloud. 2008. Enhancing Coreference Clustering. In *Proceedings of the Second Workshop on Anaphora Resolution.* WAR II.

M. Klenner. 2007. Enforcing consistency on coreference sets. In *Recent Advances in Natural Language Processing (RANLP)*, pages 323–328.

L. Kucová and E. Hajicová. 2005. Coreferential Relations in the Prague Dependency Treebank. In *Proceedings of the 5th International Conference on Discourse Anaphora and Anaphor Resolution, San Miguel, Azores*, pages 97–102.

H. Lee, Y. Peirsman, A. Chang, N. Chambers, M. Surdeanu, and D. Jurafsky. 2011. Stanford's multi-pass sieve coreference resolution system at the conll-2011 shared task. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning: Shared Task*, pages 28–34, Portland, Oregon, USA, June. Association for Computational Linguistics.

X. Luo, A. Ittycheriah, H. Jing, N. Kambhatla, and S. Roukos. 2004. A mention-synchronous coreference resolution algorithm based on the bell tree. In *Proceedings of 42nd ACL*, page 135.

X. Luo. 2005a. On coreference resolution performance metrics. *Proc. of HLT-EMNLP*, pages 25–32.

X. Luo. 2005b. On coreference resolution performance metrics. In *Proceedings of the Joint Conference on Human Language Technology and Empirical Methods in Natural Language Processing (HLT-EMNLP 2005*, pages 25–32, Vancouver, B.C., Canada.

X. Luo. 2007. Coreference or not: A twin model for coreference resolution. In *Proceedings of NAACL HLT*, pages 73–80.

L. Màrquez, L. Padró, and H. Rodríguez. 2000. A machine learning approach for pos tagging. *Machine Learning Journal*, 39(1):59–91.

L. Màrquez, M. Recasens, and E. Sapena. 2012. Coreference resolution: An empirical study based on semeval-2010 shared task 1. *LRE Special Issue, Accepted*.

A. McCallum and B. Wellner. 2005. Conditional models of identity uncertainty with application to noun coreference. *Advances in Neural Information Processing Systems*, 17:905–912.

J.F. McCarthy and W.G. Lehnert. 1995. Using decision trees for coreference resolution. *Proceedings of the Fourteenth International Conference on Artificial Intelligence*, pages 1050–1055.

G.A. Miller. 1995. WordNet: a lexical database for English.

R. Mitkov. 2005. *The Oxford handbook of computational linguistics*. Oxford University Press, USA.

T.S. Morton. 2000. Using coreference in question answering. *NIST SPECIAL PUBLICATION SP*, pages 685–688.

1998. *MUC-7 — Proceedings of the 7th Message Understanding Conference*. http://www.muc.saic.com/.

C. Müller, S. Rapp, and M. Strube. 2002. Applying Co-Training to reference resolution. *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 352–359.

V. Ng and C. Cardie. 2002a. Identifying anaphoric and non-anaphoric noun phrases to improve coreference resolution. *Proceedings of the 19th international conference on Computational linguistics-Volume 1*, pages 1–7.

V. Ng and C. Cardie. 2002b. Improving machine learning approaches to coreference resolution. *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 104–111.

V. Ng. 2005. Machine learning for coreference resolution: From local classification to global ranking. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, page 164. Association for Computational Linguistics.

V. Ng. 2007. Shallow semantics for coreference resolution. *IJCAI 2007*, pages 1689–1694.

V. Ng. 2008. Unsupervised models for coreference resolution. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*. EMNLP-08.

V. Ng. 2009. Graph-cut-based anaphoricity determination for coreference resolution. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics on ZZZ*, pages 575–583. Association for Computational Linguistics.

C. Nicolae and G. Nicolae. 2006. Best Cut: A Graph Algorithm for Coreference Resolution. *Proceedings of the 2006 Conference on EMNLP*, pages 275–283.

US NIST. 2003. The ACE 2003 Evaluation Plan. *US National Institute for Standards and Technology (NIST), Gaithersburg, MD.[online*, pages 2003–08.

L. Padró. 1998. *A Hybrid Environment for Syntax–Semantic Tagging*. Ph.D. thesis, Dep. Llenguatges i Sistemes Informaics. Universitat Politecnica de Catalunya, February. http://www.lsi.upc.es/˜padro.

J. Peral, M. Palomar, and A. Ferrández. 1999. Coreference-oriented interlingual slot structure & machine translation. In *Proceedings of the Workshop on Coreference and its Applications*, pages 69–76. Association for Computational Linguistics.

M. Poesio, R. Vieira, and S. Teufel. 1997. Resolving bridging references in unrestricted text. In *Proc. of the ACL Workshop on Operational Factors in Robust Anaphora Resolution*, pages 1–6.

M. Poesio, R. Mehta, A. Maroudas, and J. Hitzeman. 2004a. Learning to resolve bridging references. In *Proceedings of the 42nd annual meeting of the Association for Computational Linguistics*, pages 143–150.

M. Poesio, R. Stevenson, B.D. Eugenio, and J. Hitzeman. 2004b. Centering: A parametric theory and its instantiations. *Computational Linguistics*, 30(3):309–363.

S.P. Ponzetto and M. Strube. 2006. Exploiting semantic role labeling, WordNet and Wikipedia for coreference resolution. *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, pages 192–199.

H. Poon and P. Domingos. 2008. Joint unsupervised coreference resolution with Markov Logic. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 650–659. Association for Computational Linguistics.

A.M. Popescu and O. Etzioni. 2005. Extracting product features and opinions from reviews. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 339–346. Association for Computational Linguistics.

S.S. Pradhan, E. Hovy, M. Marcus, M. Palmer, L. Ramshaw, and R. Weischedel. 2007. OntoNotes: A unified relational semantic representation. *INTERNATIONAL JOURNAL OF SEMANTIC COMPUTING*, 1(4):405.

S. Pradhan, L. Ramshaw, M. Marcus, M. Palmer, R. Weischedel, and N. Xue. 2011. Conll-2011 shared task: Modeling unrestricted coreference in ontonotes. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning (CoNLL 2011)*, Portland, Oregon, June.

J.R. Quinlan. 1993. *C4.5: Programs for Machine Learning*. Morgan Kaufmann.

A. Rahman and V. Ng. 2009. Supervised models for coreference resolution. In *Proceedings of EMNLP 2009*, pages 968–977, Suntec, Singapore.

A. Rahman and V. Ng. 2011a. Coreference resolution with world knowledge. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 814–824. Association for Computational Linguistics.

A. Rahman and V. Ng. 2011b. Narrowing the modeling gap: A cluster-ranking approach to coreference resolution. *Journal of Artificial Intelligence Research*, 40(1):469–521.

W. Rand. 1971. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66(336):846–850.

M. Recasens and E. Hovy. 2011. BLANC: Implementing the Rand Index for coreference evaluation. *Natural Language Engineering*, doi:10.1017/S135132491000029X.

M. Recasens and M.A. Martí. 2009. AnCora-CO: Coreferentially annotated corpora for Spanish and Catalan. *Language Resources and Evaluation*.

M. Recasens, A. Martí, M. Taulé, Ll. Màrquez, and E. Sapena. 2009. SemEval-2010 Task 1: Coreference Resolution in Multiple Languages. *SEW-2009 Semantic Evaluations: Recent Achievements and Future Directions*.

M. Recasens, L. Màrquez, E. Sapena, M.A. Martí, M. Taulé, V. Hoste, M. Poesio, and Y. Versley. 2010a. SemEval-2010 Task 1: Coreference Resolution in Multiple Languages.

M. Recasens, L. Màrquez, E. Sapena, M.A. Martí, M. Taulé, V. Hoste, M. Poesio, and Y. Versley. 2010b. SemEval-2010 Task 1: Coreference resolution in multiple languages. In *Proceedings of the 5th International Workshop on Semantic Evaluations (SemEval-2010)*, Uppsala, Sweden.

M. Recasens, L. Màrquez, E. Sapena, M.A. Martí, M. Taulé, V. Hoste, M. Poesio, and Y. Versley. 2010c. Semeval-2010 task 1: Coreference resolution in multiple languages. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 1–8, Uppsala, Sweden, July. Association for Computational Linguistics.

R. Rosenfeld, R. A. Hummel, and S. W. Zucker. 1976. Scene labelling by relaxation operations. *IEEE Transactions on Systems, Man and Cybernetics*, 6(6):420–433.

EM Rounds. 1980. A combined nonparametric approach to feature selection and binary decision tree design. *Pattern Recognition*, 12(5):313–317.

S.R. Safavian and D. Landgrebe. 1991. A survey of decision tree classifier methodology. *Systems, Man and Cybernetics, IEEE Transactions on*, 21(3):660–674.

E. Sapena, L. Padró, and J. Turmo. 2007. Alias assigment in information extraction. In *Proceedings of SEPLN-2007*. Sevilla, Spain.

E. Sapena, L. Padró, and J. Turmo. 2008. A Graph Partitioning Approach to Entity Disambiguation Using Uncertain Information. In *Proceedings of the 6th international conference on Advances in Natural Language Processing*, pages 428–439. Springer-Verlag Berlin, Heidelberg.

E. Sapena, L. Padró, and J. Turmo. 2010a. A Global Relaxation Labeling Approach to Coreference Resolution. In *Proceedings of 23rd International Conference on Computational Linguistics, COLING*, Beijing, China, August.

E. Sapena, L. Padró, and J. Turmo. 2010b. RelaxCor: A Global Relaxation Labeling Approach to Coreference Resolution. In *Proceedings of the ACL Workshop on Semantic Evaluations (SemEval-2010)*, Uppsala, Sweden, July.

E. Sapena, L. Padró, and J. Turmo. 2011. Relaxcor participation in conll shared task on coreference resolution. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning: Shared Task*, pages 35–39, Portland, Oregon, USA, June. Association for Computational Linguistics.

F. Sasaki, C. Wegener, A. Witt, D. Metzing, J. Ponninghaus, et al. 2002. Co-reference annotation and resources: A multilingual corpus of typologically diverse languages. In *Proceedings of the 3nd International Conference on Language Resources and Evaluation (LREC-2002)*, pages 1225–1230. Citeseer.

C. Sidner. 1979. Towards a computational theory of definite anaphora comprehension in english discourse. Technical report, Cambridge, MA, USA.

W.M. Soon, H.T. Ng, and D.C.Y. Lim. 2001. A Machine Learning Approach to Coreference Resolution of Noun Phrases. *Computational Linguistics*, 27(4):521–544.

M. Stede. 2004. The Potsdam commentary corpus. In *Proceedings of the 2004 ACL Workshop on Discourse Annotation*, pages 96–102. Association for Computational Linguistics.

V. Stoyanov, N. Gilbert, C. Cardie, and E. Riloff. 2009. Conundrums in noun phrase coreference resolution: Making sense of the state-of-the-art. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2*, pages 656–664. Association for Computational Linguistics.

M. Strube. 1998. Never look back: an alternative to centering. In *Proceedings of the 17th international conference on Computational linguistics*, pages 1251–1257, Morristown, NJ, USA. Association for Computational Linguistics.

F.M. Suchanek, G. Kasneci, and G. Weikum. 2007. Yago: a core of semantic knowledge. In *Proceedings of the 16th international conference on World Wide Web*, pages 697–706. ACM.

C. Torras. 1989. Relaxation and neural learning: Points of convergence and divergence. *Journal of Parallel and Distributed Computing*, 6:217–244.

O. Uryupina, M. Poesio, C. Giuliano, and K. Tymoshenko. 2011. Disambiguation and filtering methods in using web knowledge for coreference resolution. In *Proceedings of the 24th International Florida Artificial Intelligence Research Society Conference*.

O. Uryupina. 2003. High-precision identification of discourse new and unique noun phrases. In *Proceedings of the ACL 2003 Student Workshop*, pages 80–86.

O. Uryupina. 2009. Detecting anaphoricity and antecedenthood for coreference resolution.

Y. Versley, A. Moschitti, M. Poesio, and X. Yang. 2008. Coreference systems based on kernels methods. In *Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1*, pages 961–968. Association for Computational Linguistics.

M. Vilain, J. Burger, J. Aberdeen, D. Connolly, and L. Hirschman. 1995. A model-theoretic coreference scoring scheme. *Proceedings of the 6th conference on Message understanding*, pages 45–52.

M. Walker, A. Joshi, and E. Prince. 1998. *Centering Theory in Discourse*. Clarendon Press, Oxford.

X. Yang, G. Zhou, J. Su, and C.L. Tan. 2003. Coreference resolution using competition learning approach. In *ACL '03: Proceedings of the 41st Annual Meeting on Association for Computational Linguistics*, pages 176–183.

X. Yang, J. Su, G. Zhou, and C.L. Tan. 2004. An NP-cluster based approach to coreference resolution. In *Proceedings of the 20th international conference on Computational Linguistics*, page 226. Association for Computational Linguistics.

X. Yang, J. Su, and C.L. Tan. 2006. Kernel-based pronoun resolution with structured syntactic knowledge. *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the ACL*, pages 41–48.

X. Yang, J. Su, J. Lang, C.L. Tan, T. Liu, and S. Li. 2008. An entity-mention model for coreference resolution with inductive logic programming. *Proceedings of ACL-08: HLT*, pages 843–851.