Universidad Carlos III de Madrid

Escuela Politecnica Superior

Tesis Doctoral
Departamento de Informática

Bootstrapping Named Entity Resources for Adaptive Question Answering Systems

Leganés, Julio de 2010

Autor: César de Pablo Sánchez

Directora: Paloma Martínez Fernández

TESIS DOCTORAL

Bootstrapping Named Entity Resources for Adaptive Question Answering Systems

Autor: CÉSAR DE PABLO SÁNCHEZ
Director: PALOMA MARTÍNEZ FERNÁNDEZ

Firma del Tribunal Calificador:

Presidente:

Vocal:

Vocal:

Vocal:

Secretario:

Calificación:

Leganés, Octubre de 2010

A mis padres, Trini *y* Resti

Agradecimientos

Esta no es mi tesis. Al menos, no es sólo mía. Cada uno de vosotros habéis puesto vuestro granito de arena para que yo pueda escribir estas líneas, las últimas de la tesis.

Mi directora, Paloma Martínez, se merece sin duda el primer lugar en agradecimientos. Sin su experiencia y dedicación el camino hubiera sido mucho más dificil. Pero, sin duda, es por su constante apoyo y paciencia que hemos llegado al final. ¡Ha sido una suerte compartir estos años!

En esta tesis se habla mucho del grupo MIRACLE, y es que, a todas luces, ha sido mi escuela en esto que llamamos investigación. Sin la colaboración de Ana González y de Antonio Moreno del Laboratorio de Lingüística de la Autonoma y de todos los profesionales de DAEDALUS esta tesis, simplemente, no hubiera sido posible. Gracias a José Carlos Gónzález, a José Luis Martínez y a Julio Villena que sembraron la semilla del Procesamiento de Lenguaje Natural y que desde entonces siempre la han alimentado con conocimientos y oportunidades.

Comidas, cafés y alguna que otra cervezita han sido la válvula de escape a tanto ordenador, tanto artículo y tanta clase. Muchas veces han sido bastante más, fuente de inspiración, de críticas, de ánimos y de sueños. He tenido la suerte de estar rodeado de amigos fantásticos tanto entre los compañeros del grupo de investigación LABDA como durante los años que pasé en Colmenarejo. Gracias especialmente a Dolores, Lourdes, Juan, Javi, David, Jessica, Elena, Mayte, Ana, Harith, Julián y Luis. Del otro bando, también con un cariño especial, a Sergio, Oscar, Fede, Rodri, Nayat, Ramón, José Daniel y Miguel Ángel. Mario y David, a vosotros os toca por partido doble. Agradecimientos aparte se merece mi inseparable compañero de aventuras y desventuras castellano-manchegas, Luis Martí.

Aunque la huella del tiempo que pasé en Yahoo! Barcelona y en DFKI Saarbrücken es ligera en este documento, personal y profesionalmente han dejado una gran marca. Gracias especialmente a Hugo Zaragoza, Jordi Atserias, Mihai Surdeanu y Massi Ciaramita por hacerme sentir como uno más. Gracias también a Thierry Declerk que no sólo nos acogió con gusto en el LT sino que nos llegó a dar cobijo, comida, bebida y buenas noches de conversación. A Alejandro Figueroa, por las dos partes, pero especialmente, por estar siempre dispuesto a hablar de preguntas y respuestas.

Por último, hay a quienes nunca podré expresarles mi agradecimiento con palabras. A mis padres, Trini y Resti, porque me habeis dado lo más valioso que tengo. A mis hermanos, Fernando y Javi, por vuestos ánimos en los momentos de flaqueza. A Isa, que se esconde en muchos de los párrafos anteriores y ha sufrido como nadie con cada uno de los que viene a continuación. Gracias por hacer mi vida tan especial. Espero que te quedes a ver todos los parrafos que quedan por escribir.

Resumen

Los Sistemas de Búsqueda de Respuestas (SBR) amplían las capacidades de un buscador de información tradicional con la capacidad de encontrar respuestas precisas a las preguntas del usuario. El objetivo principal es facilitar el acceso a la información y disminuir el tiempo y el esfuerzo que el usuario debe emplear para encontrar una información concreta en una lista de documentos relevantes. En esta investigación se han abordado dos trabajos relacionados con los SBR.

La primera parte presenta una arquitectura para SBR en castellano basada en la combinación y adaptación de diferentes técnicas de Recuperación y de Extracción de Información. Esta arquitectura está integrada por tres módulos principales que incluyen el análisis de la pregunta, la recuperación de pasajes relevantes y la extracción y selección de respuestas. En ella se ha prestado especial atención al tratamiento de las Entidades Nombradas puesto que, con frecuencia, son el tema de las preguntas o son buenas candidatas como respuestas.

La propuesta se ha encarnado en el SBR del grupo MIRACLE que ha sido evaluado de forma independiente durante varias ediciones en la tarea compartida CLEF@QA, parte del foro de evaluación competitiva Cross-Language Evaluation Forum (CLEF). Se describen aquí las participaciones y los resultados obtenidos entre 2004 y 2007. El SBR de MIRACLE ha obtenido resultados moderados en el desempeño de la tarea con tasas de respuestas correctas entre el 20% y el 30%. Entre los resultados obtenidos destacan los de la tarea principal de 2005 y la tarea piloto de Búsqueda de Respuestas en tiempo real de 2006, RealTimeQA. Esta última tarea, además de requerir respuestas correctas incluía el tiempo de respuesta como un factor adicional en la evaluación. Estos resultados respaldan la validez de la arquitectura propuesta como una alternativa viable para los SBR sobre colecciones textuales y también corrobora resultados similares para el inglés y otras lenguas.

Por otro lado, el análisis de los resultados a lo largo de las diferentes ediciones de CLEF así como la comparación con otros SBR apunta nuevos problemas y retos. Según nuestra experiencia, los sistemas de QA son más complicados de adaptar a otros dominios y lenguas que los sistemas de Recuperación de Información. Este problema viene heredado del uso de herramientas complejas de análisis de lenguaje como analizadores morfológicos, sintácticos y semánticos. Entre estos últimos se cuentan las herramientas para el Reconocimiento y Clasificación de Entidades Nombradas (NERC en inglés) así como para la Detección y Clasificación de Relaciones (RDC en inglés).

Debido a la dificultad de adaptación del SBR a distintos dominios y colecciones, en la segunda parte de esta tesis se investiga una propuesta diferente basada en la adquisición de conocimiento mediante métodos de aprendizaje ligeramente supervisado. El objetivo de esta investigación es adquirir recursos semánticos útiles para las tareas de NERC y RDC usando colecciones de textos no anotados. Además, se trata de eliminar la dependencia de herramientas de análisis lingüístico con el fin de facilitar que las técnicas sean portables a diferentes dominios e idiomas. En primer lugar, se ha realizado un estudio de diferentes algoritmos para NERC y RDC de forma semi-supervisada a partir de unos pocos ejemplos (bootstrapping). Este trabajo propone primero una arquitectura común y compara diferentes funciones que se han usado en la evaluación y selección de resultados intermedios, tanto instancias como patrones. La principal propuesta es un nuevo algoritmo que permite la adquisición simultánea e iterativa de instancias y patrones

asociados a una relación. Incluye también la posibilidad de adquirir varias relaciones de forma simultánea y mediante el uso de la hipótesis de exclusividad obtener mejores resultados. Como característica distintiva el algoritmo explora la colección de textos con una estrategia basada en indización, que permite adquirir conocimiento de grandes colecciones. La estrategia de selección de candidatos y la evaluación se basan en la construcción de un grafo de instancias y patrones, que justifica nuestro método para la selección de candidatos. Este procedimiento es semejante al frente de exploración de una araña web y permite encontrar las instancias más parecidas a las semillas con las evidencias disponibles.

Este algoritmo se ha implementado en el sistema SPINDEL y para su evaluación se ha comenzado con el caso concreto de la adquisición de recursos para las clases de Entidades Nombradas más comunes, Persona, Lugar y Organización. El objetivo es adquirir nombres asociados a cada una de las categorías así como patrones contextuales que permitan detectar menciones asociadas a una clase. Se presentan resultados para la adquisición de dos idiomas distintos, castellano e inglés, y para el castellano, en dos dominios diferentes, noticias y textos de una enciclopedia colaborativa, Wikipedia. En ambos casos el uso de herramientas de análisis lingüístico se ha limitado de acuerdo con el objetivo de avanzar hacia la independencia de idioma.

Las listas adquiridas mediante bootstrapping parten de menos de 40 semillas por clase y obtienen del orden de 30.000 instancias de calidad variable. Además se obtienen listas de patrones indicativos asociados a cada clase de entidad. La evaluación indirecta confirma la utilidad de ambos recursos en la clasificación de Entidades Nombradas usando un enfoque simple basado únicamente en diccionarios. La mejor configuración obtiene para la clasificación en castellano una medida F de 67,17 y para inglés de 55,99. Además se confirma la utilidad de los patrones adquiridos que en ambos casos ayudan a mejorar la cobertura. El módulo requiere menor esfuerzo de desarrollo que los enfoques supervisados, si incluimos la necesidad de anotación, aunque su rendimiento es inferior por el momento. En definitiva, esta investigación constituye un primer paso hacia el desarrollo de aplicaciones semánticas como los SBR que requieran menos esfuerzo de adaptación a un dominio o lenguaje nuevo.

Abstract

Question Answering (QA) systems add new capabilities to traditional search engines with the ability to find precise answers to user questions. Their objective is to enable easier information access by reducing the time and effort that the user requires to find a concrete information among a list of relevant documents. In this thesis we have carried out two works related with QA systems.

The first part introduces an architecture for QA systems for Spanish which is based on the combination and adaptation of different techniques from Information Retrieval (IR) and Information Extraction (IE). This architecture is composed by three modules that include question analysis, relevant passage retrieval and answer extraction and selection. The appropriate processing of Named Entities (NE) has received special attention because of their importance as question themes and candidate answers.

The proposed architecture has been implemented as part of the MIRACLE QA system. This system has taken part in independent evaluations like the CLEF@QA track in the Cross-Language Evaluation Forum (CLEF). Results from 2004 to 2007 campaigns as well as the details and the evolution of the system have been described in deep. The MIRACLE QA system has obtained moderate performance with a first answer accuracy ranging between 20% and 30%. Nevertheless, it is important to highlight the results obtained in the 2005 main QA task and the RealTimeQA pilot task in 2006. The last one included response time as an important additional variable of the evaluation. These results back the proposed architecture as an option for QA from textual collection and confirm similar findings obtained for English and other languages.

On the other hand, the analysis of the results along evaluation campaigns and the comparison with other QA systems point problems with current systems and new challenges. According to our experience, it is more difficult to tailor QA systems to different domains and languages than IR systems. The problem is inherited by the use of complex language analysis tools like POS taggers, parsers and other semantic analyzers, like NE Recognition and Classification (NERC) and Relation Detection and Characterization (RDC) tools.

The second part of this thesis tackles this problem and proposes a different approach to adapting QA systems for different languages and collections. The proposal focuses on acquiring knowledge for the semantic analyzers based on lightly supervised approaches. The goal is to obtain useful resources that help to perform NERC or RDC using as few annotated resources as possible. Besides, we try to avoid dependencies from other language analysis tools with the purpose that these methods apply to different languages and domains. First of all, we have study previous work on building NERC and RDC modules with few supervision, particularly bootstrapping methods. We propose a common framework for different bootstrapping systems that help to unify different evaluation functions for intermediate results. The main proposal is a new algorithm that is able to simultaneously acquire instances and patterns associated to a relation of interest. It also uses mutual exclusion among relations to reduce concept drift and achieve better results. A distinctive characteristic is that it uses a query based exploration strategy of the text collection which enables their use for larger collections. Candidate selection and evaluation are based on incrementally building a graph of instances and patterns which also justifies our evaluation function. The discovery approach is analogous to the front of exploration in a web crawler and it is able to find the most similar instances to the available seeds.

This algorithm has been implemented in the SPINDEL system. We have selected for evaluation the task of acquiring resources for the most common NE classes, Person, Location and Organization. The objective is to acquire name instances that belong to any of the classes as well as contextual patterns that help to detect mentions of NE that belong to that class. We present results for the acquisition of resources from raw text from two different languages, Spanish and English. We also performed experiments for Spanish in two different collections, news and texts from a collaborative encyclopedia, Wikipedia. Both cases are tackled with limited language analysis tools and resources.

With an initial list of 40 instance seeds, the bootstrapping process is able to acquire large name lists containing up to 30.000 instances with a variable quality. Besides, large lists of indicative patterns are obtained too. Our indirect evaluation confirms the utility of both resources to classify NE using a simple dictionary recognition approach. Best results for Spanish obtained a F-score of 67,17 and for English this value is 55,99. The module requires much less development effort than annotation for supervised algorithms although the performance is not in pair yet. This research is a first step towards the development of semantic applications like QA for a new language or domain with no annotated corpora that requires less adaptation effort.

Contents

List of Figures

List of Tables

List of Examples

List of Algorithms

Chapter 1

Introduction

1.1 Motivation

Ten years ago, in 2000, a study from Berkeley University estimated the growth rate of information in Internet in 0.1TB a day [1]. Today, the amount of information that is daily generated is several orders of magnitude larger, thanks, at least in part, to the phenomena of user-generated media, the so-called Web 2.0. Moreover, as Internet and the Web spreads, the content is created not only in English but in many other languages. Inside organizations we found similar challenges as the information that is available for employees, clients and collaborators is constantly increasing. Nevertheless, it is somehow a paradox that the increasing amount of information does not always enable us to achieve increased effectiveness in our work tasks. Knowledge workers employ today more time searching for specific information than ever. Therefore, enabling easier information access is key to the success of a society build upon knowledge. Lots of technologies may contribute to face this challenge and automatic language understanding by computers is just one of them. Question Answering (QA) systems are just an attempt to reduce the time and effort that a user should spend in order to find specific information by using technologies that *understand* human language. QA systems aims to provide a concrete and focused answer to an information need that is usually expressed as a question. This functionality contrasts with the most frequently deployed technology of Search Engines which is limited to produce a list of documents for a request expressed with search terms. In this sense, QA technology would provides more accurate results by locating concrete information contained in the documents and promises to reduce the workload on the side of the user. Because an automatic system would never be completely accurate, the system should also provide additional contextual information that helps a human to judge that answers are correct and adequate.

During the last years, the ability of answering certain questions has become part of most Internet Search Engines. Sometimes, this functionality has been marketed with great fanfare, others it has been silently integrated with previous ones. Direct Answers from Google, Yahoo! Shortcuts or Windows Live Search Answers, from what now has become Bing, are just the stakes of the three major search engines. It would not be fair not to mention Ask.com as the pioneer in using questions as requests, even if its beginning was based on a database of reference answers. Brainboost, Hakia, Powerset (now part of Bing too) and recently Wolphram Alpha have during the last years announced their advances on answering questions from a variety of sources. All of them, in one way or another, try to understand the semantics behind requests and data sources to find concrete information. Not only Internet Search Engines have attempted to improve information access by including QA capabilities, but other companies have developed personal an enterprise search engines like LCC, Teragram, Synapse or Priberam among others. A related but different approach has been taken by Community Question Answering systems like Naver, Yahoo! Answers or Aardvark (recently acquired by Google) to help individuals to share their

[1] http://www2.sims.berkeley.edu/how-much-info

knowledge, form interest communities and create large scale question and answer archives which can be reused.

In conclusion, no matter what the approach is, all of them try to revolve around these pillars:

- Understanding the needs of a user, as she is able to express them more naturally, using questions.
- Understanding language and how it is used in a context to compile knowledge that can be reused in order to answer questions.
- Integrating and organizing different sources of knowledge and information whether they are unstructured textual data, structured data or elicited from individuals knowledge.

This thesis is an attempt to introduce and explore the two first issues, question and language understanding particularly when information is expressed in written form taking into account the limited range of language understanding that is available nowadays.

1.2 Objectives

A QA system requires other services, tools and resources for the analysis of language. The fact that most commercial services and prototypes have addressed only English language is a solid evidence for the dependence of language tools and knowledge resources. QA in other languages has been tackled during the last years as part of shared evaluations like the Cross-Lingual Evaluation Forum (CLEF). Spanish was one of the language enabled in CLEF and we have participated in several editions in mono-lingual and cross-lingual tracks. This work is described in the first part of this thesis. It also gave birth to the second part, which is focused on the issue of language and domain adaptation in QA systems.

When dealing with technology that understand human languages there are problems that arise with every different language you consider. Some of them are specific to the object of study as the typology and characteristics vary a lot between languages. However, a practical problem for most languages is that linguistic tools and knowledge resources are not as developed as for English. Most techniques rely on the existence of these resources and tools, and therefore, they do not really adapt to different languages. Furthermore, new techniques may be developed if we would like to work with several lanaguages at once.

The main objective of this thesis is advancing the techniques that make easier the development and adaptation of Question Answering systems for different languages and domains. Our approach attempts to model the domain of a system using the most salient classes of entities and their relations. We assume that the entities and their relations would cover a large portion of the questions that could be asked. But this is not enough, we need to be able to identify the concrete entities and their relations in order to answer questions. In contrast with other approaches, we attempt to use semi-supervised machine learning like bootstrapping in order to reduce the need for specific language tools. In turn, the use of semi-supervised machine learning techniques is interesting as it would require less human work to build semantic analysis tools for concept (NE in our context) and relation extraction. The use of a shallow representation allows to be independent of language tools. Our aim is to exploit the availability of large textual corpora and reduce the need for expert annotation by using bootstrapping techniques.

We summarize the objectives of this thesis like:

- **Develop a Question Answering system for Spanish.** QA in English has been well studied in TREC conferences but the adaptation of QA systems for other languages, including Spanish started with CLEF. We have attempted to transfer the advances and techniques from English to Spanish in order to examine in depth the task, as well as the problems tackling a different language. We have always considered evaluation an important

issue to assess the progress. Therefore, we have attempted to participate in independent evaluations like CLEF. Besides, CLEF has also proposed new challenges to advance the state of the art in QA systems that we have attempted to tackle.

- **Study QA technology and in particular their adaptation.** This is an important step to fulfill the previous objective which required an study of the state of the art on QA. Several open problems have been pointed out in research roadmaps, but we have considered that adaptation is particularly important because the current systems require high expertise and very specific tools. This contrasts with the much lower barrier of entry than traditional IR technology have achieved. We have focused on the alternatives for Answer Extraction regarding language and domain adaptation.

- **Describe a model for the integration of extraction patterns for entities and relations in a QA system.** Describe the architecture of a QA system based on basic Named Entities and Relation Extraction obtained by means of semi-supervised learning.

- **Define a model for the semi-supervised acquisition of entities and relations from text.** In order to populate a knowledge base with information from the text a procedure we need a method to extract entities and relations from texts. There are three key requirements in order to advance towards this end, (1) it should require a small amount of supervision, (2) the supervised material should be easy to obtain and (3) it should not require complex language resources. In our approach we attempt to use examples of entities and their relations to bootstrap the modules because we assume it is more amenable for domain experts than tagging a text. The third requirement would help for example to retarget a basic architecture to a new language.

- **Implement and evaluate a platform for the acquisition of knowledge for multilingual and multi-domain extraction of entities and relations.** Implement a system based on these ideas and evaluate their ability of adapting to different languages and domains.

1.3 Organization

The rest of this work is structured as follows:

Chapter 2 studies the state of the art on Question Answering systems. We provide a simple operational definition of the QA task and present some of the problems that are found when trying to obtain accurate answers in response to a question. We introduce additional complications that advance towards systems that are more useful for final users too. We thoroughly survey the vast research from initial Natural Language Interfaces to Databases to current QA systems. The main architectures and the most relevant approaches for the different modules of a QA system are also described. Finally, we explore research carried out for Spanish in the context of CLEF evaluations.

Chapter 3 presents the MIRACLE QA system, a system that has been developed to answer questions from Spanish texts. The chapter include its architecture, its main modules and the results of its evaluation during the CLEF campaigns from 2004 to 2007. As the system has evolved along the years, the current description includes modules that have been developed to come to grips with the novelties in the CLEF@QA task. A general overview of our participation in this task and an outline of the lessons learned are also included in the next chapter, Chapter 4.

In Chapter 5 we return to an analysis of the state of the art for Information Extraction technologies which are an important part of any Answer Selection module. We focus on the problem of extracting knowledge and building IE modules without annotated data. This is a common problem when approaching a new language or a new domain of application. In

these situations the cost and effort of annotating a corpus could be too high. Advances in the application of semi-supervised learning and bootstrapping techniques may offer an alternative way to acquire and develop knowledge for Answer Extraction. We provide a common framework for different bootstrapping algorithms and focus on those which are directly required in most QA systems, recognition of important concepts (NE) and their relations.

Chapter 6 describes a proposal for a QA system based on bootstrapped IE components. By means of integrating the acquisition of entities and relations and their associated patterns we pretend to cover the most common request of information in a factual QA system.

The initial step to develop this proposal starts on Chapter 7. This chapter introduces an algorithm for the large scale acquisition of resources for NE recognition from unannotated text using only light supervision. It is based on a graph based exploration of textual sources and include several ideas from previous bootstrapping algorithms. We attempt to reduce to a minimum the amount of language processing tools in order to apply them to several languages and domains.

The algorithm is experimentally tested on the acquisition of a large list of names and patterns associated with NE classes. Experiments and evaluation for different language are presented in Chapter 8. It includes experiments Spanish and English as well as for different domains like news collection and Wikipedia.

Finally, Chapter 9 summarizes the conclusions and the future steps in order to achieve adaptive QA systems based on the large scale acquisition of knowledge resources for Answer Extraction in different domains and languages. We present our contributions as well as provide further lines of research regarding QA systems and large scale IE as well as their evaluation.

Chapter 2

State of the Art: Question Answering

2.1 Introduction

Few words are needed to give a general idea of the task of automatic Question Answering (QA). The user requires specific information and she requests it in the most common way for a human, using a question in natural language. The system should *understand* the need and look for the concrete information required, an answer, being as succinct as possible. Besides, the QA system should be able to provide the information needed to assess that the answer is correct. Moreover, if the system could not answer the question correctly, it should abstain from provide a wrong result. One step further, a system that engages in a dialogue with the user in order to be more effective could be devised. For example, the system could request a clarification of the questions. But, the user could also request a further explanation, correct their previous request or ask for more related information.

QA systems are nominated to be an advance over different existing Information Access applications because they should be able to reduce the abyss between the user and the system. It is important to keep in mind that the objective is to support the user seeking information and not as much as replicate human behavior like in traditional Artificial Intelligence (AI). When compared with Information Retrieval (IR) engines which usually return documents, QA systems should be able to provide more focused information, exact succinct answers. In contrast to languages for querying databases or IR engines, it is believed that natural language should be more comfortable for casual non-expert users.

The description gives an idea of the numerous research lines that are active and pursued in QA research. However, such QA system is a futuristic vision which is still far to be completed. An operative definition of QA is required in order to evaluate our advance towards the goal of accessing information in plain language. This has been the goal of research forums like the Text REtrieval Conference (TREC[1]) that have attempted to define meaningful scenarios or tasks where the progress could be measured. Similar objectives are pursued by other fora like CLEF[2] (Cross-Lingual Evaluation Forum) and NTCIR[3] (NII Test Collection for IR Systems).

2.2 Finding the Right Answer to a Question

The basic QA task is defined as a single round interaction including a question and an answer. The user is restricted to ask for an specific single fact, which, in principle, she does not know. For example *When did Joseph Stalin die?*. The objective of the system is to locate a source of

[1] http://trec.nist.gov
[2] http://clef-campaign.org/
[3] http://research.nii.ac.jp/ntcir/

information and extract the correct answer, *March 5, 1953*. The text should also provide the necessary explanation or the support facts that back that the answer is correct. The sentence *Joseph Stalin was the General Secretary of the Communist Party of the Soviet Union's Central Committee from 1922 until his death on 5 March 1953* supports the previous answer.

Besides, the system should assign a confidence to the overall process. In a laboratory setting, the task is also restricted to a given knowledge source, for example a specific collection of documents or a database. In this context, it is assumed that those facts that are asserted in the source are true even if they have changed later. On the other hand, the answers to questions that cannot be found in this collection remain unknown. The system should use a default answer string, *NIL*, with the meaning of *I don't know*. This interpretation contrasts with the *close world assumption* which is common in database models.

The task as it is stated was first proposed in TREC 2002 [Voorhees, 2002], following the developments in three previous campaigns which only required the retrieval of short focused passages containing the answer. The source of answers was the AQUAINT collection, a large collection of news in English. A similar setting was also adopted in CLEF for studying QA in other languages than English. The definition of the task is fairly inspired in the requirements described for the MURAX system [Kupiec, 1993] which in turn was designed to answer questions from an encyclopedia.

The basic definition of the QA task as stated above it is mostly concerned with questions about single facts, also called *factual questions*. These questions usually ask for details about different types of entities like persons, locations, objects or events among others. A great number of modifications and variations to the basic task have been introduced later on. The motivation is to advance towards a more complex and realistic scenario including among other novelties, new question types. In the following section, we analyze why even the simple scenario is complex and then proceed to look at other variations of the task.

2.2.1 Is QA Difficult? An example.

Given a factual question like *¿Cuándo murió Stalin?*, the objective of a QA system consists on finding the correct answer. The system should look for the answer in some source of information. For our interest we will consider textual sources in Spanish only. Besides, it should provide the context that justifies that the answer is correct too.

Consider first the different forms to ask the same question. For example, the same information need behind the question example could be also expressed by any of the alternative formulations in Example 2.1. We could argue that some of these questions are more informed than others. Some reflect that the user knows more details about Stalin or the way it died, but essentially the same answer should be expected for all of them.

Example 2.1 Questions reformulations for *¿Cuándo murió Stalin?*

¿En qué fecha murió Stalin?
¿Cuándo falleció Stalin?
¿Cuándo murió envenenado Stalin?
¿Cuándo murió Josef Stalin?
¿Cuándo murió Iosif Stalin?

Text fragments in Example 2.2 have been selected to illustrate some of the problems that a QA system may come across while trying to interpret a text for answering a concrete question. Please, take some moment to read them. These candidate passages have been manually retrieved from Google, Wikipedia and the EFE94-95[4] using the terms *Stalin* and *murió* and variations.

[4]The EFE94-95 collection has been used as the source collection for Spanish in CLEF from 2003 to 2006 and jointly with Wikipedia in 2007 and 2008

In fact, this is the first problem how to locate potential source of information and discriminate from the rest.

We have intentionally selected some of this text snippets to provide additional interesting examples. The first observation is that the richness and the variety of phenomena that is behind natural language and the way we are used to represent our world would produce infinite ways to express the same information. This could be associated with the figure of paraphrasis in natural language, but also includes other language phenomena like synonymy, co-reference, etc.

All candidate passages in bold in Example 2.2 contain some sort of correct answer. However the surface realization of the fact that *Stalin died on the 5th March 1953* varies in all of them. Most of them express additional facts that the QA system should ignore for the moment. Moreover, even if some passages could have a very similar surface realization, some may contain the correct answer while others not. Again, we expect the QA system to be able to filter the incorrect ones and end up locating the correct answer.

Example 2.2 A sample of candidate passages for question *¿Cuándo murió Stalin?*

1. *[...]La madre de Stalin murió en 1937, a cuyo funeral no asistió él. [...]*
2. *[...] Acaso Stalin, un hombre viejo y fatigado, como lo revela la fotografía de 1950 de arriba, murió de muerte natural?[...]*
3. *[...] ¿Quienes siguieron después de que murió Stalin? [...]*
4. *[...] A su anciana madre, que murió cuando Stalin tenía ya más de 60 años, la debe haber visto un par de veces en veinte años. [...]*
5. *[...] los autores sugieren que es posible que Stalin haya sido envenenado con warfarina, un adelgazador de sangre sin sabor ni color, también usado para matar ratas, durante una última cena con cuatro miembros de su politburó, el 1 de marzo de 1953.[...]*
6. ***[...]* no fue dado a conocer sino hasta junio de 1953, más de tres meses después de su *[ref:Stalin]* muerte el 5 de marzo de 1953.**
7. *[...] Lo que quiero decir es lo siguiente: la genuina Unión Soviética, la verdadera construcción de Stalin, murió el 1 de junio de 1941 cuando, sin declaración de [...]*
8. *[...] cuerpo de Stalin* [ref:Stalin Nicolás Rovira, guerillero anti-castrista] *estaba totalmente marcado por lo golpes y los bayonetazos, no murió Stalin con un tiro de gracia, Cuco Lara puso una inyección en sus [...]*
9. ***[...]* Con la muerte de Stalin en 1953 la diplomacia rusa adquiere una *[...]***
10. **Sin embargo, tras su *[ref:Stalin]* muerte el 5 de marzo de 1953, que abrió una nueva época en la historia de la URSS *[...]***
11. **Iósif Vissariánovich Dzhugashvili (castellanizado) (*Gori, Georgia, 18 de diciembre de 1878 - 5 de marzo de 1953), mejor conocido como Iósif Stalin fue el líder de la Unión Soviética desde mediados de los años 1920 hasta su muerte en 1953 *[...]***
12. ***[ref:Stalin]* Falleció el 5 de marzo de 1953, tras cuatro días de agonía.*[...]***

The mission of a QA system could be described as a way to overcome the semantic gap between the language used in the question and the way that the answer is represented in the source of answers. To achieve its goal, the question, the answer and the context should share

some sort of representation. This must be compatible with the representation that it is used in the source, whether this is textual, structured (like a relation of death dates for important people) or lies in between.

For example, one of the aspects to consider are synonyms and the use of different lexical terms. While the use of the term *murió* as used in the original question would help to access some of the passages, it is actually not part of any of the results which contain the answer. Including synonyms like *falleció* or related words like *muerte* could help to locate a larger number of texts containing the answer. By using them, we will have a better chance to find the answer but it could also be harder to found it if the number of texts is too large. An alternative could be to represent words in some sort of abstract or generalized form which uses their meaning and not just the word. A related problem appears if we consider different conjugations of the verb. In this case, a canonical representation of the verb using their infinitive *morir* can be used with similar trade-offs.

Proper names pose some similar problems and a few new ones too. It is possible to refer to Stalin, as a concept or entity in the real world, using several forms like *Joseph Stalin*, *Iósif Stalin*, *José Stalin* or *Iósif Vissariánovich Dzhugashvili* depending on the conventions and the language used. Moreover, in larger texts the references can be expressed with pronouns or other anaphoric expressions like in examples 6, 10 and 12. Besides, proper names or references to named entites (NE) are sometimes ambiguous and depend on the broader context. Like in the passage 8, where the NE does not refer to the communist leader but to another person, a guerrilla fighter.

An important information that is implicit in most questions is the type of the expected answer. In our case, as we know that we are looking for a date, an effective heuristic could filter candidate passages that do not contain a date. A simple reformulation of the question into a sort of representation like *Stalin murió en* DATE could be used to find potential answers. Though this new *representation* do not imply a deep understanding of the question it works surprisingly often. Nevertheless, we would find also lot of cases, like in passage 1, where this technique will fail. Such simple mechanism is not able to detect that the subject which dies is Stalin's mother and not Stalin himself. Linguistic analysis, and in particular syntactic analysis, could have been useful to spot this particular problem. World knowledge and a deeper representation that capture more semantics could be used too. For instance, they would be required to extract from passage 5 that as a consequence of the poisoning Stalin died. On the other hand, performing this kind of reasoning with computers is still brittle and pushes the limits of Natural Language Processing (NLP) and Knowledge Representation (KR).

On the other hand, it is also paramount to consider the type and distribution of facts in the collection also at the global level, beyond a single paragraph. We can find simple and effective heuristics. For example, the frequent association of words (or even concepts) like *Stalin*, *murió* and *1953* could be indicative of the correctness of the answer and should not be disregarded. In the same sense, it is different to extract one answer than to extract all correct answers.

The discussed examples show that the task of answering a question correctly by automatic means pose a number of different challenges. Moreover, among the factors there are not only theories and techniques but it is also important to take into account the sources that are available to extract information. Other practical questions like the availability of linguistic resources and the usefulness of language tools arise too. They usually constrain the path to perform automatic Question Answering. It is not clear which of the approaches is effective and until what point complete language understanding is needed. There is no easy answer to which of these techniques would be effective across a large number of questions. Our best option consists on trying to understand by careful experimentation which are the trade offs between techniques in order to be able to repeat success.

2.2.2 Elaborations of the QA task

Example 3 shows several example of factual questions. For all of them, the expected answer is a short answer referring to a real entity world like a person, a place or location, etc. Others are just requesting simple facts that can be expressed as dates, numbers or quantities. Although a great number of factual questions do ask for particular entities of the real world or generic concepts, the answer of a factual question could be other types of information like more generic concepts, events or forms to do something.

Example 2.3 Examples of factual questions

Question Type	Answer type	Question text	Answer
F	PERSON	¿Quién es el presidente de los Estados Unidos?	Barack Obama
F	ORG	¿Qué empresa produce el Escarabajo?	Volkswagen
F	NUMBER	¿Cuántos países forman la OTAN actualmente?	28
F	DATE	¿Cuándo fue la coronación oficial de Isabel II?	2 de junio de 1953
F	DATE	¿Cuándo murió Stalin?	5 de marzo de 1953
F	LOCATION	¿Dónde esta situado Ystad?	Suecia
F	COUNTRY	¿En qué país nació el Papa Juan Pablo II?	Polonia
F	MEASURE	¿Qué altura tiene el Kanchenjunga?	8586 metros
F	LOCATION	¿Cual es la principal religión de Timor Oriental?	Catolicismo

Almost every information need could be in fact stated as a question, which would enlarge the QA scope *ad-infinitum* (something that would please any young QA researcher). Nevertheless it is important to remember that the objective is to provide a concise answer and find methods to help the user deal with information. In this spirit other important areas of research have focused on questions asking for definitions, lists of facts or even procedures or opinions. The extent of this types of answers is usually longer and it can involve assembling information from different sources. Other systems go beyond simple questions and try to solve questions that include different types of restrictions, for instance, temporal restrictions.

2.2.2.1 Definitional Questions

Definitional questions require information that concisely defines a term or a short description that characterizes a concept. As an information need, it has been shown that they are particularly frequent in a Web context [Lowe, 2000], while not always stated in a question form. A key term in definitional questions is the *definiendum*, the term that is meant to be defined. It is usually an entity or a concept as in the Example 2.4. The expected answer could be a short description or the expansion of an acronym, but usually longer descriptions are preferred. Another important difference with factual questions is that several answers could be correct at the same time. These answers provide different perspectives or information on the defined subjects. In the QA parlance these are commonly called *nuggets*. The same nugget could be expressed in several different forms. The difficulty of a definitional QA system strives into providing not only correct answers but the most adequate ones. Besides, the definiendum could be ambiguous and, in this case, it is expected that the system would be able to identify and even clarify the sentences.

Example 2.4 Examples of definitional questions and example answers

Question	Answer
¿Qué es el Hubble?	un telescopio robótico que se encuentra en orbita espacial
	un asteroide del Cinturón de asteroides
	un cráter Lunar
¿Qué es la RKA?	Agencia Espacial y de Aviación Rusa
	Agencia Espacial Federal Rusa
	agencia gubernamental responsable del programa espacial y la investigación general de aeronáutica
¿Quién es Iosif Kobzon?	crooner Soviético
	cantante ruso
	miembro del parlamento ruso
	miembro del parlamento ruso que ha sido elegido con un mayor número de votos en la historia de Rusia

2.2.2.2 List Questions

When a user asks a factoid question she usually assumes that a unique answer exists. If several answers are expected, this fact is usually marked explicitly in the request, which ask for a list of all the possible answers. Finding a complete list of elements or facts is a complex task for humans because it could require to combine information from different sources and it is difficult to assess when the task is finished. Several examples of List questions are presented in Example 2.5. There are in fact several different categorizations to apply to list questions. Open List questions are those where the number of answers is not specified while Closed List questions expect a concrete number of answers. Another important difference is the number of answers, it is not the same to locate a few answers than a large number of correct answers. QA usually has dealt with the first kind of questions. Finally, the provenance of the answers also matters. One strategy to answer list questions consists on locating text where an enumeration or a list is already compiled. Compiling a list from partial lists or even individual answers is much harder.

Example 2.5 Examples of List questions

Question	Answers
¿Cuales son las tres repúblicas eslavas?	Rusia, Ucrania y Bielorrusia
Nombre luchadores de sumo	Ashashoryu, Hakuho y Harumafuji

2.2.2.3 Questions with Restrictions

Compound questions or questions with restrictions have received significant attention during the last years. In general, they are complex requests that need to integrate factual information from different sources to provide the correct answer or apply certain amount of reasoning. Time and spatial issues are important dimensions which usually are involved in most of these requests. Question with temporal restrictions have been part of CLEF evaluations since 2004 [Magnini et al., 2005]. Spatial restrictions have been studied as part of the geographical retrieval track in CLEF (GeoCLEF) in 2008 and 2009 [Santos and Cardoso, 2008]. Representative examples are presented in Table 2.6. These are questions that require to answer simpler ones in order to find just the interesting subset of answers.

Example 2.6 Examples of List questions

Question
¿Quién recibió el Premio Nobel de la Paz en 1989? ¿Qué político liberal fue ministro italiano entre 1989 y 1993? ¿Quién fue el primer ministro de Inglaterra antes de John Major?
Which Swiss cantons border Germany? Polynesian islands with more than 5,000 inhabitants. Brazilian architects who designed buildings in Europe. French bridges which were in construction between 1980 and 1990.

2.2.2.4 Opinions, Causes and Complex Questions

Almost any request can be formulated as a question, but keeping in mind that the objective of QA is to enable the user to find quickly the answer to a focused information need, there are some more types to consider. Complex questions have started to gain attention in evaluation forums because they need to combine QA with other research areas like Summarization or Opinion Analysis. The goal is to provide focused retrieval of information with retrieval units that are smaller than documents and even fuse information from several sources. The Document Understanding Conference (DUC) [Dang, 2006] have proposed a task where the goal is to provide a summary assembled from a number of documents in response to a complex question. A complex request is shown in Example 2.7 which would require to locate correct sources of information and identifying text passages including causes and opinions. Moreover, to successfully complete these task, locating definitions and facts is just another part of the task. Different tracks at TREC and CLEF have looked into some of these specific question types. Finding opinions about a subject in a large blog corpus has been the objective of TREC 2006 Blog track. On the other hand, a large number of questions in CLEF 2009 QA track deal with finding causes for different events and conditions. These tasks resemble definitional question answering because longer multiple answer and ranking play an important role. In contrast to traditional IR, topical relevance is not enough but discourse structure, opinionated language and the relation between the message, the source of the message and the topic should be taken into account.

Example 2.7 An scenario for Complex Question Answering

DUC num	D0641E
title	global warming
narr	Describe theories concerning the causes and effects of global warming and arguments against these theories.

2.2.2.5 Interactive and Dialogue based QA

The process of Information Seeking is much more complex than just retrieving information. In some cases it requires the completion of a task, which requires requesting several facts. In other cases, the need only becomes more specific as the enquiry process proceeds. In the basic scenario that we depicted in our Stalin example, the question and the answer represent a single round of interaction. Other scenarios that assume several interactions between the user and the system have been also explored. Kato et al. [2005] describes a proposal to evaluate this kind of interactions. Evaluation forums have considered context or topic related questions, a set of questions about the same topic that simulate a dialogue of questions and answers. Examples 2.8 and 2.9 are just drawn from different evaluations. The new challenge include identifying and

tracking the topic of the questions for their correct interpretation. Other important aspects are real-time processing and handling different forms of user feedback. While we have put the stress on language and dialogue comprehension, another approaches have explored the use of interfaces to improve communication with the user [Harabagiu and Strzalkowski, 2006].

Example 2.8 Example of a question series from NTCIR QAC-3

In which country was George Mallory born?
What was his famous phrase?
When did he say it?
How old was he when he started climbing mountains?
On which expedition did he go missing the top of the Everest
When did it happen?
At what altitude on Everest was he seen last?
Who found his body?

Example 2.9 Example of a question series in Spanish from CLEF 2008

¿A quién rescató Otto Skorzeny?
¿Cuántos planeadores usó en el rescate?
¿Dónde le juzgaron?
¿Quién le ayudó a fugarse?

2.3 Natural Language Processing to the Rescue

Natural Language Processing (NLP) is a discipline that explores automatic methods that understand, represent and generate human language. As we have seen in our example in section 2.2.1, NLP techniques could be used to the help in the QA process. This section outlines the levels of analysis that are commonly used when studying human language and the related tasks that an automatic system must solve in order to understand and be able to achieve a formal representation of language and meaning [Jurafsky and Martin, 2008; Manning and Schütze, 1999; Mitkov, 2003]. The tasks that we present are restricted to the interpretation and understanding of written language as this is the focus of the current work. An outline of the different analysis levels is shown in Figure 2.1 and an example of the results for those level that are easier to represent is available in Figure 2.2.

2.3.1 Morphological analysis

Morphological analysis consists on finding function and the structure of words in a text. Words are usually formed by morphemes, or the smaller meaning bearing units. Morphemes are usually divided into stems, the main form, and affixes. In most languages, there are phenomena like inflection, derivation or composition that help to produce new words from the main form with the same or related meaning. The main modules in a NLP system that operate at this level are:

- The **Tokenizer** splits the text in basic units of analysis named tokens, which roughly corresponds to words. European alphabets usually use spaces and punctuations to separate words and this is a good start for a tokenizer. Other alphabets, Chinese for example, do not use spaces and the task of the tokenizer is more complex.
- A **Stemmer** extracts the root form or the stem of a word isolating other components like prefixes, suffixes or infixes that are used for inflection or derivation from the base meaning.

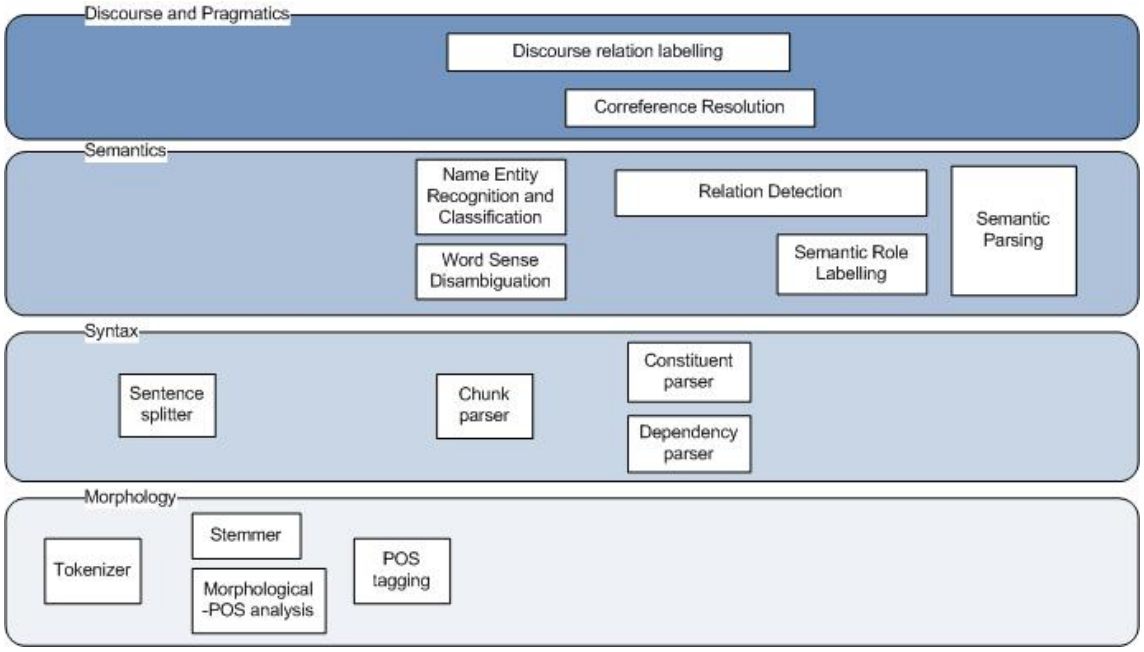

Figure 2.1: Functions of NLP processors by analysis level, left to right roughly represents typical requirements

- A **Morphological Analyzer** goes a step beyond the stemmer and it attaches a grammatical function or part of speech (POS) to each word in a text. Besides finding the stem it provides an interpretation for each of the morphemes that are part of the word and mark things like the number, the gender or the case depending on the language. English is a simple language regarding morphology but most other languages have more complex morphology, for instance Spanish or Finnish. No matter if morphology is simple or complex, to build an analyzer of this type we will need resources like dictionaries that enumerate the different POS for each word and we should compile values for each of the morphological features.

- **Part of Speech Tagger**. The morphological analyzer often provides several interpretations for each token when they are considered in isolation. In contrast, when tokens are considered in context, their possible interpretations could be reduced, typically to a single case. A POS tagger considers the context of a word and using rules or other models decides the most appropriate tag that provides the morphological analysis of each token in a sentence. Disambiguation rules could be written by linguists or they can be induced from data.

2.3.2 Syntactic analysis

The objective of syntactic analysis is to find the grammatical structure of a sentence. The process of building a syntactic analysis of a sentence is usually called parsing. There are a great number of formalisms that have been proposed to understand syntax in language, while the two most dominant families are constituent and dependency analysis.

- A **Sentence splitter** is the most basic module that works at the sentence level and its mission is to find the boundaries of a sentence.

- A **Chunk Parser** is a shallow form of parsing that marks the boundaries of the basic components in a sentence while it ignores the structure and the relationships that bind

them together. These basic components or constituents are for example nominal phrases (NP), verbal phrases (VP) or propositional phrases (PP). They are also known as chunks as they are constrained to be consecutive tokens.

- The name of **Parser** is traditionally reserved for a full constituent-based parser. It produces a syntactic tree of the sentence where each of the basic constituents is tagged with a syntactic function like subject, direct object, etc. It is important to note, that constituents are composed of phrases like NP) and VP).
- Instead, a **dependency parser** produces a directed graph for each sentence. Edges in the graph are use to express a head-modifier relationships between two lexical items (words) in the sentence. The head provides the most important meaning in the relation that the modifier complements. Most dependency parsers provide information regarding the type of dependence between pairs of terms. Examples of dependencies include the verb and the subject ($subj$), the verb and the object (obj) or the noun and a modifier ($nmod$).

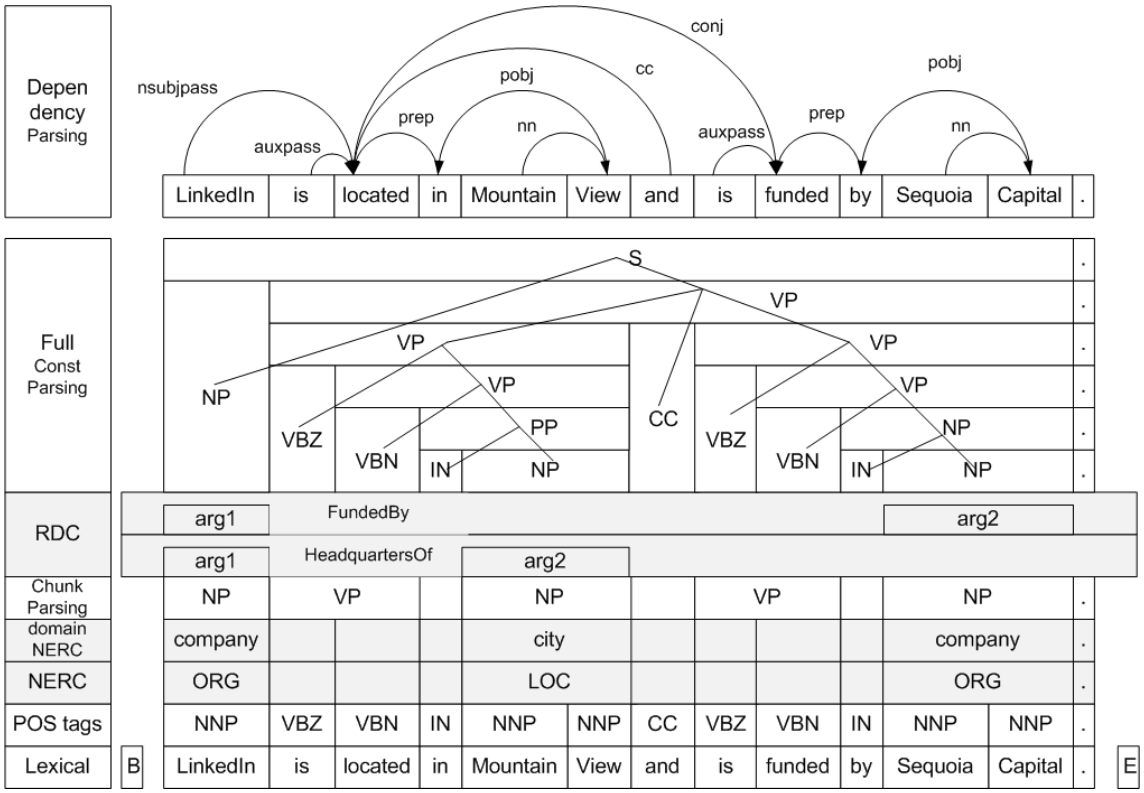

Figure 2.2: An example of the language analysis carried out by NLP processors for a sentence

2.3.3 Semantic analysis

Semantic analysis deals with the task of producing structures that represent the meaning of sentences in a broader sense. For instance, an affirmative sentence will describe a fact about the world and their semantic analysis will allow to represent and reason with this information using some sort of formalism. Semantic analysis comprises two complementary areas, the lexical semantics and the compositional semantics. Lexical semantics deals with words and their meanings, the relation with real world objects. Compositional semantics in contrast is dedicated to study how the meanings of words are composed to express knowledge about the world. There a lot of different tools that help to achieve some degree of semantic analysis, though a complete

semantic representation of language is still far. Among the tools that include semantic analysis there are:

- **Named Entity Recognition and Classification**. The objective of a NERC tool consists on marking the mentions of relevant objects from the real world in text. In addition it also provides addiotinal information like the type or the class of the entity, whether it is a person,a location or an organization. The notion of relevant objects and their types usually changes from one domain of application to another. For example, in biomedical applications genes, proteins, drugs or body parts may be considered Named Entities. Abstract concepts like temporal expressions, quantities or even measures are usually included in these tools although they do not always qualify as proper Named Entities. Section 5.5.1 provides a more detailed description of the task and the techniques used to build NERC taggers.

- **Word Sense Disambiguation (WSD)** is the process of finding the sense that is used when a word is mentioned in a given context. It is common that the same word form have different meanings or senses. For instance, in Spanish *banco* or in English *bass*. For a machine to be able to interpret the meaning of a sentence it should find which of the senses is in fact the one used in a sentence. The task of a word sense disambiguation tool consists on assigning the correct sense from those in a electronic dictionary or resource like Wordnet[5] [Fellbaum, 1998] or EuroWordnet[6].

- **Recognition of relations**. This is one of the tasks in IE that although it is only a shallow approach to semantics it is often used in QA and other NLP applications. The task involves finding and classifying the relation between pairs of concepts or entities. We can have generic relations like composition or aggregation and meronymy (part of). But other specific relations between instances or Named Entities are of interest too. Consider the one that relates persons with the place they were born, BORN_IN(PERSON,LOCATION). Although their semantics are quite specific it is informative and important for some applications. Relations of this type often relate two different entities, but there are cases that require more arguments. Section 5.5.2 introduces the details of the task and discusses the state of the art.

- **Semantic Role Labelling (SRL)** is concerned with a generic shallow semantic interpretation of sentences [Carreras and Márquez, 2005]. For each of the verbs in a sentence, a SRL system tries to find a semantic proposition and the arguments that fulfills each of the roles in the proposition. Each verb usually has one or several subcategorization frames that express a basic action and the constituents in a sentence work as arguments. For example, a verb like *shoot* has arguments like the actor, the patient, the instrument, the place and the time. The identification of the frame and its arguments is an enhanced semantic representation of the sentence. This task has some features in common to relation recognition but it attempts to be more general (achieve an interpretation for every sentence) and linguistically motivated.

2.3.4 Discourse analysis

Discourse analysis studies language beyond individual sentences. Dialogue QA systems will made use of tools usually included in discourse analysis like co-reference resolution. The co-reference module will try to identify pronouns and other referring expressions that are used to name objects, events or time points. Besides, it will try to detect when they are referring to the same discourse object. This is needed for example to maintain the context of a dialogue between a user and a system looking for answer.

[5]http://wordnet.princeton.edu/
[6]http://www.hum.uva.nl/ ewn/

2.4 Analysis of question logs

We should not pretend that a QA system answers every single question, it is simpler to build one for those questions that are in fact formulated. For example, it is reasonable to assume for factual questions that a large number of request would focus only on a defined set of entity types and relations between them. Agichtein et al. [2005] classified a large number of questions in a subset of the Encarta enciclopaedia log into definitional, explanation, navigational, binary factoids and others. Binary factoids request for relations involving two entities and amount for the third part of questions in that log. Their study shows that a set of 11 entities types and 156 relations types are enough to classify a large subset of the binary factoid questions in the log. The distribution of questions types and question instances follow a Zipf distribution in both cases. The ten most frequent relation types account for more than the 40% of types while just 5 relations cover half of the questions instances asked. Consider the relation DEATH_DATE(PERSON,DATE), each of the reformulations in Example 2.1 is counted as the same question type. Every time one of the variants appears in the log is also counted as a different question instance. Moreover they also show that these relations are relatively stable across time.

Relation	Type %	Instances %
discover(PERSON,CONCEPT)	7.7	2.9
has position(PERSON,CONCEPT)	5.6	4.6
has location(LOCATION,LOCATION)	5.2	1.5
known for(PERSON, CONCEPT)	4.7	1.7
has date(EVENT,DATE)	4.1	0.9
has discovery date (OBJECT,DATE)	3.3	1.0
creates (PERSON, OBJECT)	3.3	1.5
eats (ANIMATE,OBJECT)	2.9	1.8
has location (EVENT, LOCATION)	2.4	1.6
has alias (OBJECT, TITLE)	2.3	0.7
Total Coverage	41.5	18.2

Table 2.1: Top 10 most frequent relations by query type

Relation	Type %	Instances %
has neighbours(LOCATION,LOCATION)	0.5	21.2
has founding date (LOCATION, DATE)	1.4	11.0
has speed(ANIMATE,QUANTITY)	0.3	9.0
has color(ANIMATE,COLOR)	0.2	8.0
has length(LOCATION,QUANTITY)	5.6	6.7
Total Coverage	2.6	55.9

Table 2.2: Top 5 most frequent relations by query instances

While the results in [Agichtein et al., 2005] are in the context of a Web application which is a rather open domain, some of the conclusions could be translated to other domains. Previous studies for other search engines with question answering capabilities already have pointed that question types follow a Zipfian distribution [Lin and Katz, 2003; Lowe, 2000]. As a result, different question answering strategies can be devised for the head and the tail of the question distribution.

2.5 A Brief History of QA systems

2.5.1 Natural Language Interfaces to Databases

At the beginning of the 60's, the BASEBALL system [Green Jr et al., 1961] was developed as a natural language interface for a database of baseball results and statistics. The main objective was to enable users to query and access the information in the database without requiring them to be experts in the query language and the database schema. LIFER [Hendrix, 1977] or even SHRDLU [Winograd, 1972] succeed BASEBALL with similar objectives in different domains.

At the end of the 70's, Lehnert [1977, 1981] defines the prototypical architecture that is common nowadays for the QUALM system. It also make a seminal analysis of the kind of questions that a system should treat from the point of view of pragmatics. Later on, several systems introduced advances from NLP and reasoning with the objective of accessing more complex databases and answer more complex questions. For example, CHAT-80 [Warren and Pereira, 1982] was a system that could answer complex questions using a database of geographical facts. The Unix Consultant [Wilensky et al., 1989] in contrast was able to answer questions about commands of the Unix operating system.

All these systems were oriented to answer questions in a restricted domain by using different formalisms of semantic grammars. Quite often, they use full fledged expert systems with inference and reasoning capabilities with the objects and relations of the domain. In general, their knowledge base and rules had been carefully designed for the task at hand. Besides, inference methods were computationally expensive. Both architectural decisions became a serious limitation as the size of the knowledge base increased. Therefore, it reduced the range of practical applications that could afford this effort.

2.5.2 Question Answering on Textual Collections

MURAX [Kupiec, 1993] marked an inflection point for QA systems as it was the first succesful system that has been used as a query interface for a general knowledge encyclopedia. MURAX was the first system to combine IR techniques with shallow NLP techniques for the task. The START system [Katz, 1988, 1997] evolved from a restricted domain QA system to a Web QA system based on the semantic annotation of semi-structured resources of the web. FAQFinder [Burke et al., 1997] explored the use of database of Frequent Answered Questions (FAQ) and their answers to find the most appropriate answer for a new question. Finally, ExtrAns [Mollá et al., 1998] is one of the first system to explore restricted domain text collections and their use for domain specific QA. The usefulness of a deep semantic representation is explored and applied to technical domains like the Unix manuals or Airbus 320 manuals [Mollá et al., 2003].

We can identify these systems and those that deals with text-comprehension as the parents of modern QA systems. Although they operate on small to medium sized collection of restricted domain texts, they use more robust language analysis tools beyond ad-hoc grammar. They also approach a more realistic scenario where structure like tables or pairs of questions and answers play a role to answer users requests.

2.5.3 TREC and AQUAINT: Modern QA Systems

On 1999, the leading international evaluation forum for IR, TREC [Voorhees, 2000] organized the first QA task as a natural evolution of focused IR methods like passage retrieval. This forum has helped to define the functionality and the evaluation of current QA systems. Besides, it has been succesful in creating a research community around the theme. The first evaluation campaign tackled factual questions extracted from the FAQFinder system and those created by system designers. The expected answer were textual fragments of limited length (250 bytes

and 50 bytes). From 2002 [Voorhees, 2003] the task evolved to require the selection of an exact answer as we defined in the Section 2.2.

The TREC QA task has been carried only for English. Given the interest in QA, other evaluation fora started to offer it in other languages. That was the case of CLEF for other European languages or NTCIR for Asiatic languages. Besides particular innovations introduced in each of the evaluations, they have characterized by promoting cross-lingual QA systems where the system could answer a question with information available in another language.

2.6 The Two Sides of Question Answering

As IR systems, a QA system also has two phases of operation. The online phase is executed when the user asks a question. The offline phase groups all the tasks that the system performs as preparation to answer questions online. The system designer should decide how to distribute and carry out the different basic operations: transform the knowledge source into a suitable representation, transform the query to work with the selected representation and resolve the query. These two views have been presented in Lin and Katz [2003] as Knowledge Annotation and Knowledge Mining. However they represent two extreme solutions while there are multiple approaches to combine both.

2.6.1 Knowledge Annotation

The Knowledge Annotation (KA) approach is characterized because the extraction of knowledge and their formalization is carried out offline. This is the approach used by START [Katz et al., 2001] which is also integrated as one of the streams in ARANEA [Lin and Katz, 2003]. Knowledge is acquired from high quality sources and integrated in the knowledge base to target the process of answering frequent questions. For example, they used sources like Biography.com [7] or the CIA Factbook [8] to be able to answer questions about celebrities or geographical questions respectively. These are in fact web front ends to databases which contain a vast amount of high quality well structured data. Manual and automatic techniques may help to discover, extract and annotate this kind of knowledge. There are two main approaches to access knowledge from these structured sources:

- *Wrapping* which consists on using an adapter that transforms the question in a structured query and analyze the response. The query language is given by the access method of the remote database, typically SQL or a interface form designed for humans. Usually, a *screen scrapping* program is used to extract the answer from the generated response by using structural cues. Some modern web knowledge sources provide instead a API that can be used as a service and then there is no need for scrapping.

- *Slurping* tries to replicate the contents of the original source database in a local database. It has the advantage that a different representation from the original could be used. For example a relational database can be transformed to the triple form used in START *(object,property,value)*. Screen scraping is also used for slurping but the challenge remains on find queries to extract the hidden content. Navigational clues and previous content are used to crawl the site and reproduce the database behind.

KA was initially given for structured sources of information that were available in the Web. However, we can certainly extend the KA concept to other approaches that try to use and create structured or semi-structured knowledge bases. Moreover, this approach is becoming interesting even for open domain as more resources are published with open licenses. For example, it has

[7] http://www.biography.com

[8] https://www.cia.gov/cia/publications/factbook/

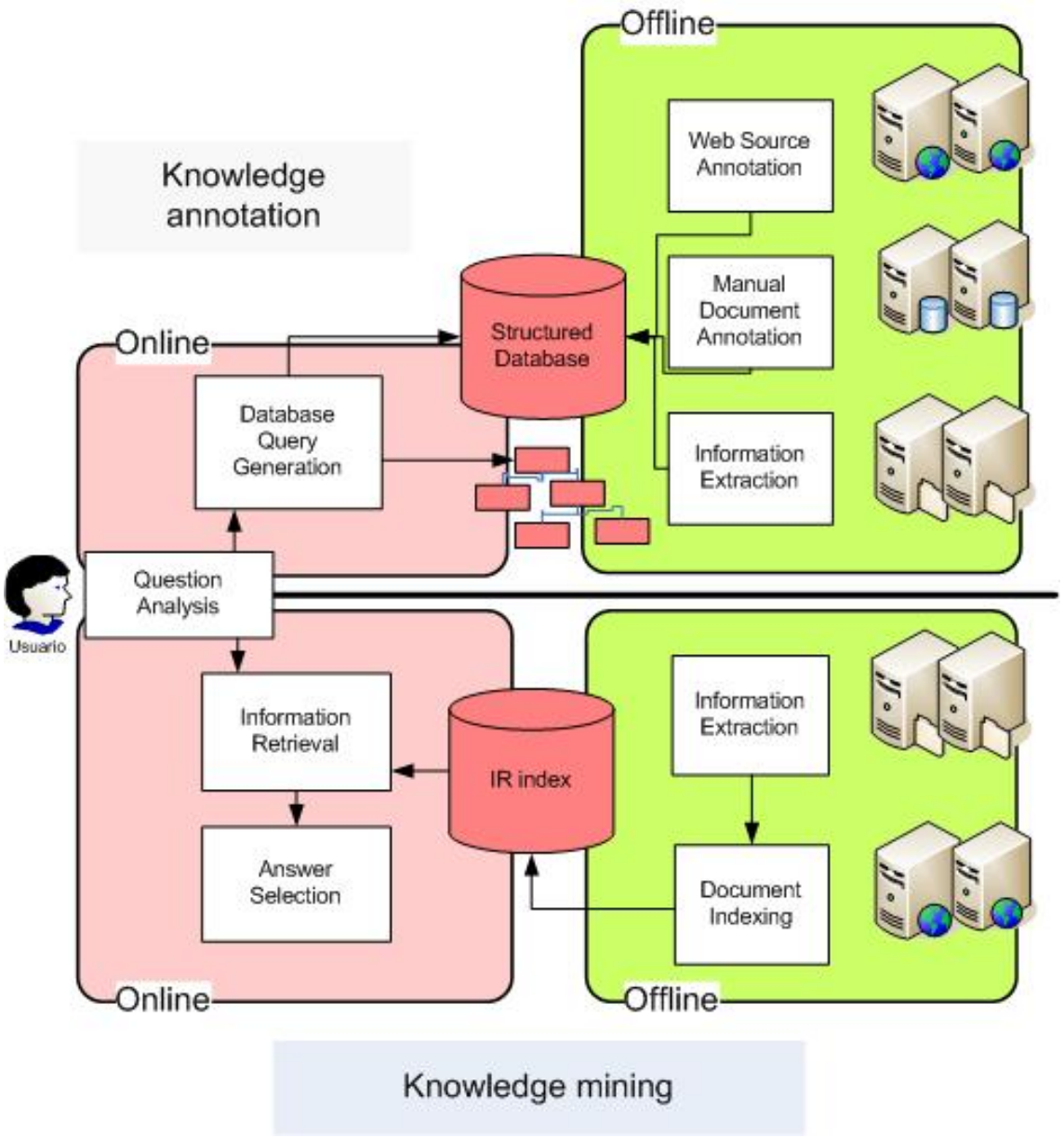

Figure 2.3: Knowledge Mining and Knowledge Annotation approach to QA

been common to use online dictionaries and other electronic resources to answer definitions or questions about acronyms (acronym expansion). Several traditional large scale resources like Wordnet has been used extensively in this manner for QA too. This approach could be seen as following the tradition of first QA systems or Natural Languages Interfaces to Databases. In contrast, they are taking profit of the advances of knowledge and data representation and integration of more than thirty years. Recent systems have endorsed the use of Semantic Web stack by using ontologies, the Resource Description Framework (RDF) language and query languages like SPARQL. Among these system we have AquaLog and PowerAqua [Lopez et al., 2006], QACID [Ferrández et al., 2009] and the architecture proposed in the QALL-ME project[9], and ORAKEL [Cimiano et al., 2008]. These systems also approach the generation of queries in new ways and make use of a wider range of lexical resources. For example, The QACID system use Textual Entailment (TE) in the questions to decide when two questions have similar

[9]http://qallme.sourceforge.net/onlinedemo.html

structures and map them to predefined queries.

2.6.2 Knowledge Mining

The approach that performs online answer extraction from unstructured text lies at the other extreme. The source of knowledge is a large collection of documents. Offline operations include the indexing of the collection in order to effective access the text but the representation remains unstructured or includes very shallow structures. Three are the online operations that a KM system performs: question analysis, retrieval of textual information and extraction of answers from the text. The architecture is depicted in Figure 2.4. Methods to find the correct answer range from those that are based on statistics like in AskMSR [Banko et al., 2002b] to the use of complex linguistic analysis and automated reasoning like in the LCC system [Harabagiu et al., 2003b]. The issue of representation and language analysis is discussed in more detail in the following sections along with the strategies used for the individual components.

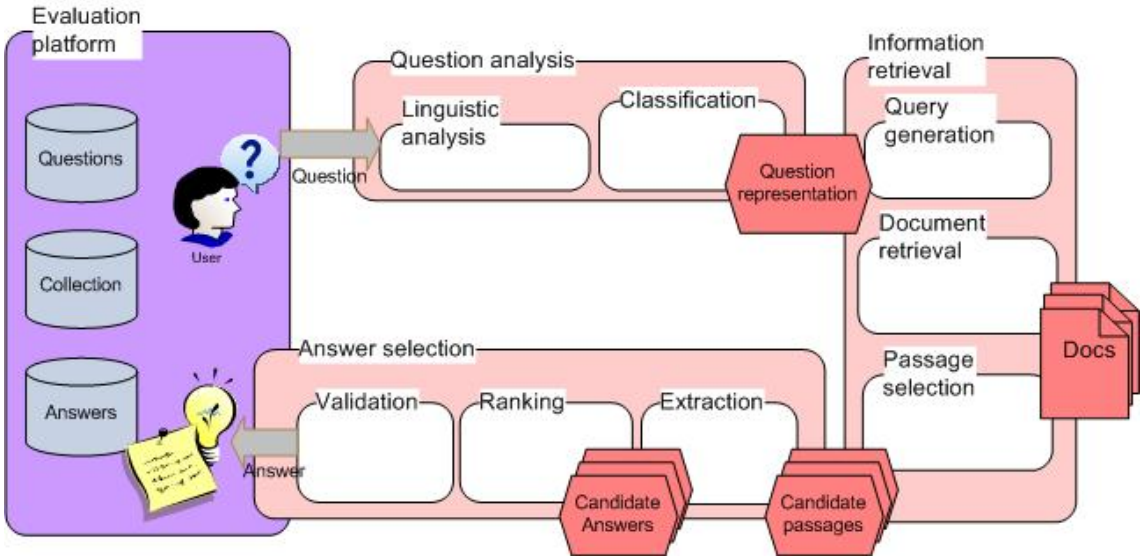

Figure 2.4: Pipeline Architecture of a QA system

The flow inside the system can be adapted depending on the type of question, whether the request is for a definition or a fact, and even what kind of fact. Although often implicitly stated, part of the design is a taxonomy of question types and answer types. There have been several proposals to organize questions types, from the simplest flat organization to a detailed taxonomy as used in [Li and Roth, 2005] or even an ontology. There have been also popular proposals for answer types like [Pasca, 2003; Sekine et al., 2002]. While the focus has been often in Named Entity types, it should be noted that other kind of answers are also possible. We outline here the three main components, their objectives as well as common submodules:

- **Question Analysis**. The analysis of the question must help to identify relevant terms that allow to locate relevant documents where the answer could be found. In addition, the analysis should provide a representation that helps to select just the correct answer among the documents. A key step is question classification which aims at clarifying which information the user is looking for.

- **Information Retrieval**. The second subsystem selects related documents based on the question representation. As documents could be too broad, more focused units of texts like paragraphs are preferred. Critical issues include how the document and the queries

should be represented, the generation of queries from questions and the ordering of the results.

- **Answer selection**. The objective of this module is to get the correct answer (or set of answers) from all the documents or paragraphs obtained in the previous step. This module would have to find a compatible interpretation between the request expressed in the question and the information expressed in the document which enables to select the correct answer. The selection of answers include their recognition in text, their validation, conflating similar answers and ranking them for final presentation. In more complex QA taks, it could include the generation of a complete and cohesive longer answer from several documents.

2.6.3 Domains and Sources of Information

The division between Knowledge Annotation and Knowledge Mining techniques that we have outlined above is too raw. At first glance, it seems that Knowledge Annotation is suited for QA applications in restricted domains where the knowledge source is well structured like manually created databases and ontologies. Mollá and Vicedo [2007] provides a good and recent overview of domain-restricted QA. On the other hand, Knowledge Mining looks suitable for applications with no large structured knowledge base where the primary information is provided in textual form. This is a typical situation in so-called open domain applications but also in other restricted domain applications too.

The distinction is artificial because there are paths that join both approaches tightly. For example, once that we understand what are the most common requests it is possible to mine knowledge offline to answer questions [Agichtein et al., 2007; Etzioni et al., 2005; Fleischman et al., 2003]. This issues are discussed in more detail in Chapter 5. On the other hand, the Web is also changing the picture for the use of open domain ontologies and other knowledge sources as it enables large scale collaborative human efforts like Wikipedia. For example, DBpedia [Bizer et al., 2007] a semi-automatic dump of the structured information contained in Wikipedia is a very interesting source of open domain knowledge. DBpedia is in fact the core to a larger number databases that are exposed as Linked Open Data [Bizer et al., 2009] using the standards created by the Semantic Web community. As the amount and range of data available grows, they become more valuable sources for answering questions. Another large database of general and encyclopedic facts in the open domain is Freebase [Bollacker et al., 2008]. Both are less curated than a traditional database and they also contain a much larger number of relations, which made them inherently less consistent.

2.6.4 Multi-strategy Architectures

In the previous section we have seen two radically different approaches to the QA task and their typical architectures. The reality is in fact not that crisp, there are systems that combine Knowledge Mining and Annotation at some point. For example, the offline extraction of facts using IE techniques could be seen as Mining Knowledge in advance guided by the domain or the need of the users. On the other hand, most Knowledge Mining systems used structured knowledge at some point, for example, to check if a name is in a list of countries or not.

Some systems, like ARANEA [Lin and Katz, 2003], decide whether to use one or other approach depending on the type of the questions and the available knowledge. QA systems usually combine a set of different strategies and diverse knowledge source in order to answer questions effectively. An strategy or a component that is very effective for answering factual questions could turn ineffective or even misleading for answering definitions. For this reason, systems use different strategies and pipelines for answering different types of questions [Cui et al., 2005b].

Several systems have used two or more strategies in parallel to answer a question. For example a system can use a compiled knowledge database and perform knowledge mining and combine candidate answers from both [Ahn et al., 2005; Chu-Carroll et al., 2003]. The main advantage of this architecture is that it may compensate errors of single strategies.

Multi-strategy architectures have an additional problem, it is necessary to combine results from several modules [Magnini et al., 2002]. This issue is discussed with more detail in Section 2.11. A common approach to set up different strategies is simply to choose different sources of knowledge. Many systems have used the Web [Banko et al., 2002a; Kwok et al., 2001], Wikipedia or Wordnet [de Rijke et al., 2005] to successfully answer questions in a general domain and complement the news source used in TREC evaluation. Multilingual systems like those evaluated at CLEF have experimented by combining information sources in different languages [Ahn et al., 2005; Tanev et al., 2005; Vicedo et al., 2005].

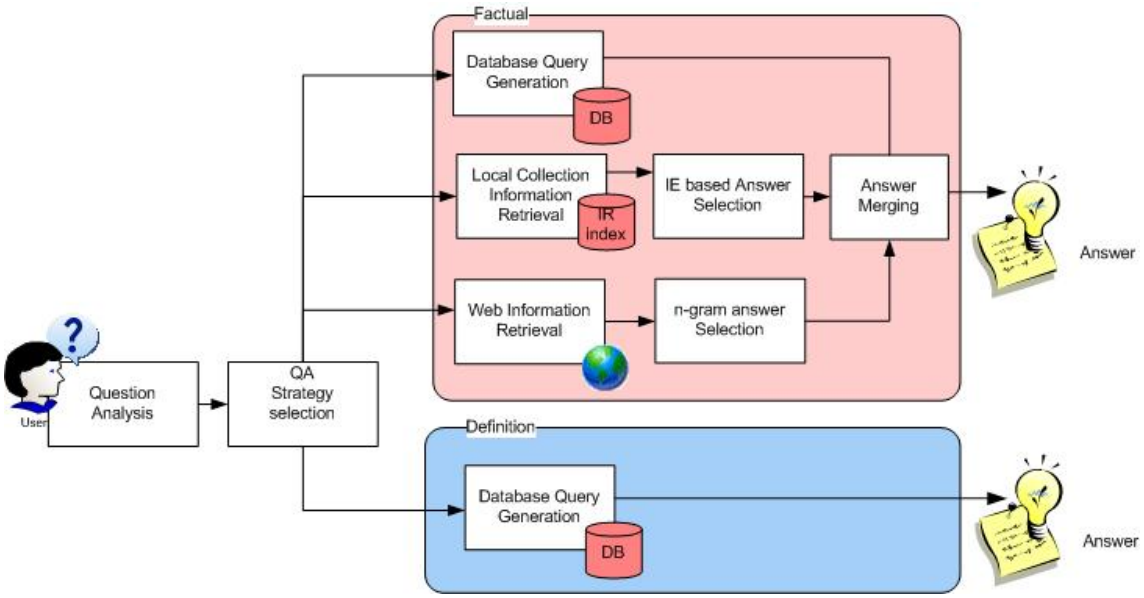

Figure 2.5: Multi-strategy QA system architecture

2.6.4.1 Strategy selection and execution

One of the issues with multiple strategies is how to select and execute the different alternatives. Most of the systems mentioned above use all the available strategies to answer a question and then combine the results. When there are lots of strategies or some of them are expensive to execute, this is clearly unpractical.

One elaboration consist on including feedback loops. For example, The LCC system (previously SMU) [Harabagiu et al., 2001; Pasca, 2003] refines the queries that sends to the IR subsystem to obtain an optimal number of documents that, according to the designers, it is the most adequate for the rest of the subsystems. The systems selects terms to add or to remove from the query, or relaxes and tightens some operators depending on the number of documents retrieved for previous queries generated from the same question.

The JAVELIN system [Nyberg and Frederking, 2003] from CMU goes a step further. The system plans the best action to carry out based on the question and the results obtained by previous actions or processing steps in the strategy. Based on the available information the system can decide to introduce a loop, repeating an operation with different input, or enabling several strategies in parallel or sequential mode. Another system from Wales-Bangor University presents a modular architecture based on intelligent agents [Clifton and Teahan, 2005]. Each

agent autonomously decides whether it is appropriate for answering a given question. A statistical model to select a subset of the most fitted strategies was presented in Lita and Carbonell [2007]. Their prototype could map tens of strategies to answer a single questions but it selects only the most promising using a model of previous questions and their results.

2.7 Evaluation

There are two types of evaluations that are usually carried out within IR, QA and other information access communities, a system evaluation and a user-centered evaluation. Two are the main reasons for this division:

- There are factors concerning the design of interfaces and the interaction of the interface with the core functionality that affect the perception that the user has on the global system. In addition to functional requirements, there are factors like response time and how results are presented that have been usually mentioned as important regarding search engines. User evaluation is aimed at measuring user satisfaction with a current system.

- On the other hand, in the development process the evaluation of different system configurations should proceed without human intervention in order to be effective. System based evaluation provides an adequate compromise for the development and testing of systems in a lab setting.

The separation between system and user evaluations assumes a clear independence between their respective objectives and metrics. However, this independence is only assumed for simplicity because the relation is more intricate in the case of systems dealing with natural language. Moreover, user evaluation should provide useful advice towards system design, evaluation and research directions. For example, the actual performance of QA systems is perceived by users as insufficient. For this reason most commercial search engines offer QA functionality as a complementary help to document search. In this context, it is more important to assess the correctness of answers to avoid erroneous answers that made the user lose confident. A similar motivation is behind real time QA; users do not expect to wait long for an answer.

2.7.1 System evaluation

The classic paradigm for IR system evaluation is based on the Cranfield model [Cleverdon et al., 1966] which is also the model adopted in TREC [Voorhees, 2006]. The Cranfield model is based on the concept of test collections. Tests collections are composed of a document collection, a set of information needs that are expressed as topics and a set of relevance judgements. Relevance judgements are a list of documents that have found to be relevant for a topic. For the simplicity of the evaluation several assumptions are taken for granted. All documents are equally relevant and independent among them. In other words, once you have seen a document the rest are still relevant even if they repeat information. Other assumptions include that the topics are assumed to represent the needs of an average user and that the collection is static, no documents are added.

The evaluation of QA system is based on the same idea of test collection although some adaptation is needed. Test collections are also formed by a document collection, a set of questions and a set of answers with their supporting document. IR test collections contain binary relevance judgements, the document is either relevant or not. QA evaluation has evolved to consider not only the correctness of the answer but also their justification.

Test collections have been generated in the context of evaluation fora, like TREC, which provide a common environment including question and document collections. Results from the participant systems are manually judged and pooled. Table 2.3 presents the typical judgements

Judgement	Name	Explanation
R	Right	Correct answer which is also supported by the snippet
W	Wrong	Incorrect answer. It could happen that the snippet has the answer but the answer is not identified
X	ineXact	Answer contains additional or unnecessary information.
U	Unsupported	Correct answer but the sentence do not justify it

Table 2.3: Common judgements in QA documents

that are used in TREC and CLEF. Example 2.10 shows how they are applied in our example question. Justification requires additional text to support that the process of answering is correct. For that reason, some test collections also include the documents or the text snippets that support a concrete answer. The pooling of answers have some drawbacks because if no system finds an answer there is no key for future uses of the test collections. If a question has several answers or even different forms that the answer is found there is no guarantee that they are included. Frequently, test collections are complemented with manually generated regular expressions that improve the key answers coverage. Therefore, building a complete and reusable list of all the answers and documents that appear in a collection is not an easy and affordable task. Lin and Katz [2006] examines the difficulties of applying this methodology in post-hoc experiments, especially when the number of positive judgments is small and also provide some guidance on the work needed to compile an exhaustive test collection.

Example 2.10 An example of judgements for question *¿Cuándo murió Stalin?*

Judgement	Answer	Support
R	5 de marzo de 1953	Sin embargo, tras la muerte de Stalin el 5 de marzo de 1953, que abrió una nueva época en la historia de la URSS *[...]*
	1953	*[...]* Con la muerte de Stalin en 1953 la diplomacia rusa adquiere una *[...]*
W	1937	La madre de Stalin murió en 1937, a cuyo funeral no asistió él.
X	marzo	Sin embargo, tras la muerte de Stalin el 5 de marzo de 1953, que abrió una nueva época en la historia de la URSS *[...]*
	Stalin el 5 de marzo de 1953	Sin embargo, tras la muerte de Stalin el 5 de marzo de 1953, que abrió una nueva época en la historia de la URSS *[...]*
U	1953	[...] Acaso Stalin, un hombre viejo y fatigado, como lo revela la fotografía de 1953 de arriba, murió de muerte natural?[...]

Difficult judgement cases appear in some situations concerning the granularity of the answer, when the reference to an entity is not clear or depending on the amount of world knowledge that is required to infer that the answer is correct. Some difficult cases are presented in Example 2.11.

Besides the capacity of answering a given question, it has been pointed in shared evaluations that systems should be able to discover when not to answer. For that reason, question test collections also have so-called NIL questions, questions where there is certainty that no answer

Example 2.11 Doubtful cases for judging the question *¿Cuándo murió Stalin?*

Judgement	Answer	Support
R/X?	1953	Sin embargo, tras la muerte de Stalin el 5 de marzo de 1953, que abrió una nueva época en la historia de la URSS *[...]*
R/U/W?	1953	*[...]* los autores sugieren que es posible que Stalin haya sido envenenado con warfarina, un adelgazador de sangre sin sabor ni color, también usado para matar ratas, durante una última cena con cuatro miembros de su politburó, el 1 de marzo de 1953.
R/U?	5 de marzo de 1953	*[...]* no fue dado a conocer sino hasta junio de 1953, más de tres meses después de su muerte el 5 de marzo de 1953.

can be found in the document collection.

2.7.2 Evaluation fora

The first large scale evaluation of QA systems was celebrated in 1999 at TREC which in fact started with the retrieval of focused passages. The following TREC editions have introduced further challenges and pushing the task into a more realistic scenario using among other real questions from search engine logs. TREC 2002 was the first edition that required the extraction of exact answers. Later additions include NIL questions, list questions and dialogue scenarios.

In parallel to TREC, interest into cross-lingual and multilingual information access spun off from TREC into CLEF and NTCIR. CLEF started in 2000 with the evaluation of ad-hoc IR for European languages in monolingual, cross-lingual (topic in English and collection in Spanish for example) and multilingual settings. Since 2003, QA@CLEF started offering the evaluation in an increasing number of European languages including Bulgarian, Dutch, Finnish, French, German, Italian, Spanish and Portuguese. The definition of the QA@CLEF task started on the track of TREC, although in the last years different challenges and pilots that have looked at other aspects of QA have been enabled. They are outlined in Table 2.4. Besides, it is also important to note that a range of different collections have been used ranging from news, Wikipedia, a legal collection like JRC-Acquis and speech transcripts in European Parliament sessions.

Years	Task	Description
2003 -	CLEF QA	QA in European Languages
2004 -	Temp QA	Evaluation of Temporally Restricted Questions
2006 - 2008	Answer Validation Exercise	Textual Entailment applied to QA
2006	RealTime-QA	QA when time response is important
2006	WiQA[10]	Exploratory QA in Wikipedia, adding novel and relevant information to Wikipedia articles.
2007 -	QAST	QA on Speech Transcripts
2009 -	RespubliQA	QA on a collection of legal documents

Table 2.4: Challenges in CLEF 2003 - 2009

On the other side of the world, NTCIR [11] is also promoting similar information access and QA research on Asian languages like Chinese, Korean and Japanese. CLQA [Chen et al., 2005] has paid special attention to factual question as Named Entities in oriental languages have

[11] http://research.nii.ac.jp/ntcir/

special needs. On the other hand, QAC [Kato et al., 2005] has explored dialogue based QA and complex information needs.

2.7.2.1 QA Evaluation Measures

Given the previous framework for the evaluation of single questions, the general procedure consists on using a set of questions and average results in order to compare systems and alternative configurations. Accuracy (Acc) is the most common measure used in QA. It is defined as the fraction of correct answers divided by the number of test questions. Usually, only the first answer to a question is considered which is noted as $Acc(n = 1)$. While doing system development, it is useful to go deeper into the candidate answer list. For example, it is common to judge at least the first five candidates which is noted $Acc(n = 5)$ where n stands for the answer depth. Taking into account the answer depth requires to consider a more elaborate definition as presented in 2.1. $\delta(n)$ is just an indicator function that is 1 if a correct answer is found before the nth ranked candidate. In the example 2.12 additional functions which are used later to define additional measures are compared.

$$Acc = \frac{\sum_{i=1}^{Q} \delta(n)}{|Q|} \tag{2.1}$$

Example 2.12 Measures to evaluate the ranked list of a questions

n	judge	*correct*	*#correct*	δ	*rank*	*RR*	*TRR*
1	W	0	0	0	∞	0	0
2	R	1	1	1	2	1/2	1/2
3	W	0	1	1	∞	1/2	1/2
4	R	1	2	1	4	1/2	1/2 + 1/4
5	R	1	3	1	5	1/2	1/2+1/4+1/5
6	W	0	3	1	∞	1/2	1/2+1/4+1/5
7	W	0	3	1	∞	1/2	1/2+1/4+1/5
8	X	0	3	1	∞	1/2	1/2+1/4+1/5
9	U	0	3	1	∞	1/2	1/2+1/4+1/5
10	W	0	3	1	∞	1/2	1/2+1/4+1/5
			$\sum_{j=1}^{n} correct$	$\begin{cases} 1 & \text{if } correct(j) \\ & \text{where } j \leq n \\ 0 & \text{otherwise} \end{cases}$	$\begin{cases} n & \text{if } correct(n) \\ \infty & \text{otherwise} \end{cases}$	$max_{j<n}(\frac{1}{rank(j)})$	$\sum_{j=1}^{n} \frac{1}{rank(j)}$

Another common measure is the *Mean Reciprocal Rank*(MRR) which assigns answers a score that is proportional to the rank of the first answer in a list (*Reciprocal Rank* or *RR*). After that the results are averaged for the complete questions set. Like in accuracy the list can be limited to a maximum length n. The *Mean Total Reciprocal Rank* ($MTRR$) not only scores the first correct answer but any correct answer in the list which is useful if several answers or formulations of the same answer are possible.

$$MRR(n) = \frac{1}{|Q|} \sum_{i=1}^{Q} RR_i(n) \tag{2.2}$$

$$MTRR(n) = \frac{1}{|Q|} \sum_{i=1}^{Q} TRR_i(n) = \frac{1}{|Q|} \sum_{i=1}^{Q} \sum_{j=1}^{n} \frac{1}{rank(j)} \tag{2.3}$$

Some evaluations have considered the possibility that the QA system provide not only the answer but also the confidence that the answer is correct. The rationale behind that is, if the confidence is too low, it should be better to provide no answer. Another metric, *Confidence*

Weighted Score (CWS), has been proposed to reward systems that assess their confidence. In order to use CWS, questions are ordered by decreasing confidence on answer first. Then the formula 2.4 is used.

$$CWS = \frac{1}{|Q|} \sum_{i=1}^{Q} \frac{\#\text{ correct(1) until } i}{i} \tag{2.4}$$

Further evaluation procedures need to be defined for additional QA capabilities. NIL questions, questions that do not have an answer in the document collection are usually evaluated by using Precision and Recall in the set of all the questions.

$$Precision_{NIL} = \frac{\#\text{ of correct predictions of NIL}}{\#\text{ of predictions of NIL}} \tag{2.5}$$

$$Recall_{NIL} = \frac{\#\text{ of correct predictions of NIL}}{\#\text{ of NIL examples}} \tag{2.6}$$

Definition and List questions are also evaluated using Precision and Recall because several answers could be simultaneously valid. Precision and Recall are calculated for every question and their answer set and later average. Besides, for definitions and other complex answers the range of judgements is also extended to consider levels of relevant information (E.g. *Vital,Okay,Incorrect*).

2.7.2.2 Evaluation of QA components

The evaluation described in this section aims at the end-to-end evaluation of QA systems. In Section 2.6 we have already seen that the main approaches for QA systems are composed of complex modules that have complex interactions among them. Moreover, they are subject to the large variety of language phenomena that happens while processing questions and documents. The evaluation of components is important from the point of view of the system designer. Methodologies from QA, IR, IE and Text Classification (TC) have been adapted to evaluate individual components and their composition. On the other hand, the interplay between different modules has not been studied thoroughly and it is a shared feeling that their interdependencies are not well understood [Ferrucci et al., 2009; Greenwood, 2008].

Figure 2.6 depicts different QA strategies and outlines alternatives for the evaluation of certain modules. They correspond to the phases of Question Classification, Document and Passage Retrieval, Answer Extraction and Validation. For some of this modular evaluations specialized resources are required which in turn they should be derived from the general test collections.

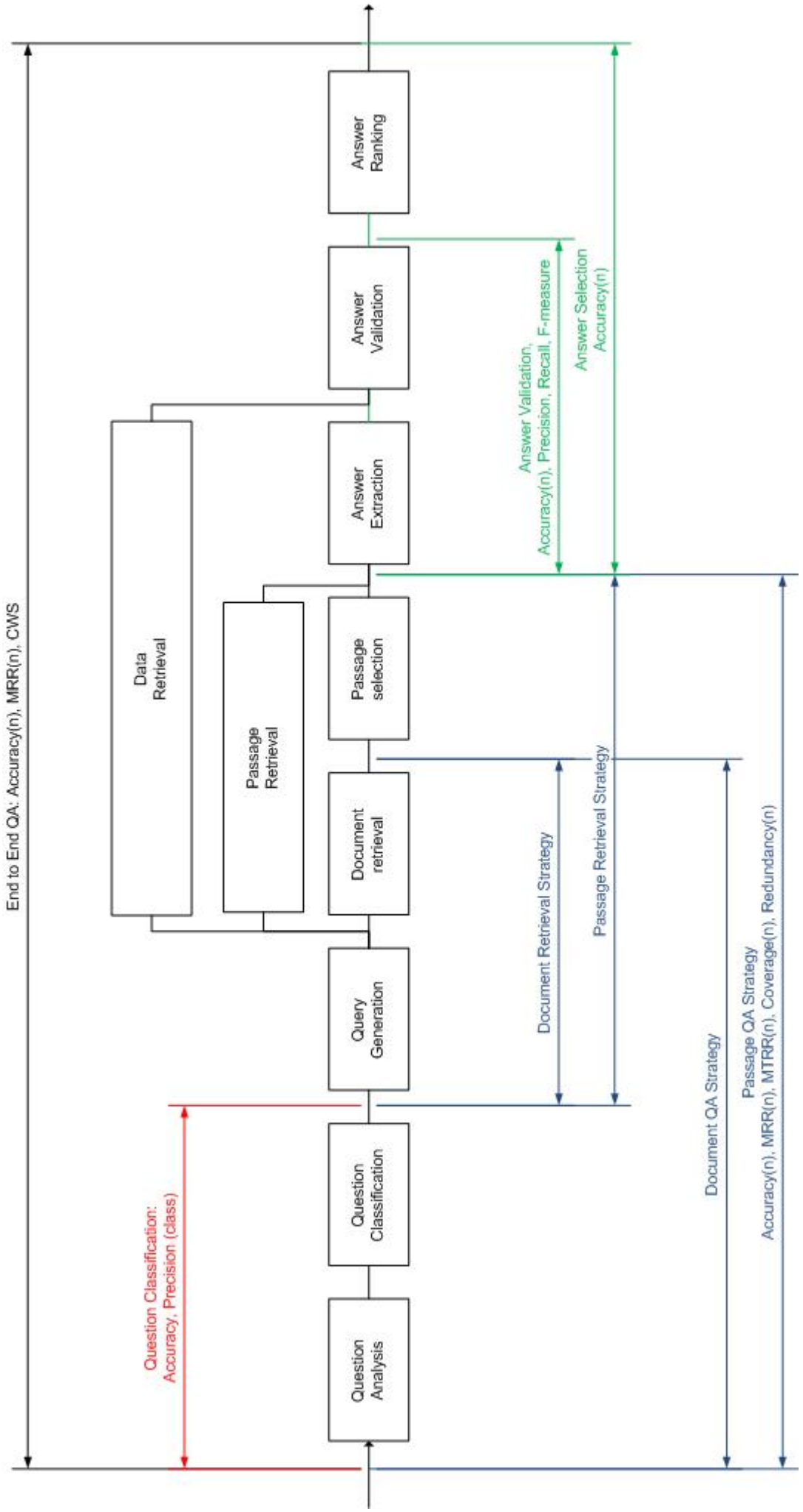

Figure 2.6: An outline of system evaluation points in a QA architecture (individual module evaluation is not represented)

System prediction	Actual value	
	positive	negative
positive	tp = true positive	fp = false positive
negative	fn = false negative	tn = true negative

Table 2.5: Contingency table for classification task

Question Classification The aim of Question Classification is detecting the type of the answer in order to trigger the adequate answering strategies. The task is often modelled as a multi-class classification problem and the most common single measure is Accuracy. The number of classes depends on the question taxonomy that is selected. As there are several classes and the number of questions per class is unevenly distributed, it is also common to report the Accuracy per class which is in fact the $Precision(c)$.

$$Accuracy = \frac{\#\text{ of correct predictions}}{\#\text{ of predictions}} = \frac{tp}{tp + fp + fn + tn} = \frac{tp}{Q} \tag{2.7}$$

$$Precision(c) = \frac{\#\text{ of correct predictions of class c}}{\#\text{ of predictions of class c}} = \frac{tp}{tp + fp} \tag{2.8}$$

$$Recall(c) = \frac{\#\text{ of correct predictions of class c}}{\#\text{ of examples of class c}} = \frac{tp}{tp + fn} \tag{2.9}$$

Document and Passage Retrieval The evaluation of Document and Passage Retrieval is aimed at quantify how feasible is to extract a correct answer from the preselected documents. QA, specially the mining approach, has often been depicted as a process that starts with maximum recall and proceeds by several filtering steps to extract precise information. It is clear that in a pipeline architecture if no document contains the answer, it will be not possible for Answer Extraction to provide a correct answer. On the other hand, the effectiveness of Answer Selection techniques depends on the amount of text that they receive and the redundancy of answers.

The automated evaluation of the retrieval step proceeds by using the answers provided in the test collection. The answers are transformed in regular expression patterns that should be present in the retrieved documents or passages. There are two forms that are commonly used to apply the patterns, strict matching and lenient matching. Consider an answer like *5 de marzo de 1953*, the strict pattern should only match exact occurrences of the string in text. A document containing just the year, *1953*, will be incorrectly evaluated with this procedure though at human would probably find this answer informative enough. The lenient alternative considers the use of individual answer tokens to assess correctness. Lenient and strict evaluations define a sort of top and upper bound for the system functionality. However, the reader should recall that finding the correct answer string in an unsupportive document do not qualify it as a correct answer. Therefore, the procedure provides only limited guidance into the characteristics of the retrieved documents.

The measures considered for retrieval for QA include MRR with large depth. Large depth is used because relevant documents can be found at low positions into the ranked list of documents and passages. Besides, Roberts and Gaizauskas [2002] propose two other related measures that has been widely adopted for the evaluation of these modules, Coverage and Redundancy.

$$Coverage(n) = \frac{1}{|Q|} \cdot |\{q \in Q | R_{D,q,n} \cap A_D\}| = \frac{1}{|Q|} \sum_{q=1}^{Q} correct(n) \tag{2.10}$$

$$Redundancy(n) = \frac{1}{|Q|} \cdot \sum_{q \in Q} |R_{D,q,n} \cap A_D| = \frac{1}{|Q|} \sum_{q=1}^{Q} \#correct(n) \tag{2.11}$$

Coverage provides the proportion of questions that have at least an answer among the retrieved documents for that question. $R_{D,q,n}$ is the set of retrieved documents for a question q in the collection D. The parameter n represents the depth considered in the ranked list. A_D is the set of documents that contain an answer as defined by any of the pattern matching methods above. Coverage is an upper bound to the maximum performance of the complete system and it has been shown with actual technology is not usually higher than 0.70. On the other hand, Redundancy accounts for the number of times that an answer appears in different documents. Higher redundancy should imply in general that answers are easier to extract, there are more chances and the combined evidence is greater. These measures are also applied to passages as well as documents, in strict and lenient modes.

Answer Selection and Validation Specific evaluation procedures have been devised for the different modules included in the Answer Selection step. Answer Extraction. Those components that are based in IE technology like NERC are evaluated using IE methodologies which are explained in more detail in Chapter 5. Nevertheless, there is more in selecting an answer than extracting a NE because type coherence with the question does not guarantee correctness. Moreover lots of different candidates could be generated from a sentence.

The framework of Answer Validation has become popular at the time of evaluating the effectiveness of the last step of the QA process. The test collection is formed by instances like those shown in Example 2.13 where the question, the answer and the sentence are provided. The Answer Validation is tested to produce a judgement on whether the answer is correct or not. In some sense, the task is similar to the evaluation that a human user does on the output of the QA system in order to check the correctness of the answer. Several resources for these kind of evaluation have created for example in the Answer Validation exercise (AVE) [Peñas et al., 2006, 2007] and also by Kaisser and Lowe [2008]. In order to create the collection they use the output of real systems or intermediate outputs that are further annotated with correct answers.

Example 2.13 Examples of tuples for Answer validation

Question	Answer	Sentence Text	Judgement
¿Cuándo murió Stalin?	5 de marzo de 1953	Sin embargo, tras la muerte de Stalin el 5 de marzo de 1953, que abrió una nueva época en la historia de la URSS *[...]*	TRUE
¿Cuándo murió Stalin?	1953	Acaso Stalin, un hombre viejo y fatigado, como lo revela la fotografía de 1953 de arriba, murió de muerte natural?	FALSE

Because it is defined as a classifications task, the proposed evaluation measures are Precision, Recall and F-measure. These are adequate as the positive class is often very unbalanced with respect to the negative class. A complementary alternative considers the accuracy averaged among the questions in the dataset. A problem with Answer Validation test collections is that they do not usually preserve the same distribution of examples than modules face in real operation.

2.7.3 User centered evaluation

User evaluation is a complicated problem as several independent variables should be controlled. The design of the experiments is complex. Besides, for the results to be meaningful a large number of users are needed to validate results with statistical significance. On the other hand, user evaluation is central for QA as a field because it helps to better understand the challenges that systems should try to solve in order to be useful.

Both TREC and CLEF have organized Interactive QA experiments in which the focus has been on the usefulness of IR systems for the task of finding concrete answers [Gonzalo et al., 2006; Gonzalo and Oard, 2005; Hersh, 2006]. Séguéla et al. [2006] compares a IR Desktop Search System and a QA system in the task of helping a user to find answers to questions. They conclude that the time and effort needed by the user is decreased by a large margin. Ogden et al. [2006] provide a different qualitative perspective on the problems that QA systems still have to solve in order to be useful for final users. Their experiments reveal that users perceive QA systems as too opaque. The users do not understand how to rephrase questions when their answers are not found or wrong. Other aspects included in their study are related to interactive communication, credibility, timely processing and interface issues.

2.8 Question Analysis

Question Analysis comprises the interpretation of the user request expressed as a question into a structured representation. This representation should capture the intention and the meaning of the original request. It also helps to organize the process to answer the request. This is a key step in the process of answering questions as errors in these steps propagate to the rest of the modules.

The character of the question representation is system specific and will depend on the level of linguistic analysis, resources and reasoning capabilities that are available. Nevertheless, there is some consensus on the type of information that is needed in a base system. An example of the basic information is presented in Example 2.14.

Example 2.14 Basic question analysis for the example question *¿Cuándo murió Stalin?*

- Question Type (QT): FACTUAL, DEATH_DATE
- Expected Answer Type (EAT): DATE
- Question Focus (QF): Cuándo
- Question Topic (QTopic): Stalin
- Keywords: Stalin, murió

There are two key processes in order to achieve such representation, Linguistic Analysis of the question and Question Classification. Linguistic Analysis is used to identify among other the Question Topic, Question Focus and at least some Keywords. The more linguistic and knowledge resources a system has, a richer and more accurate question representation is achievable. Question classification aims at detecting the Question Type and the Expected Answer Type. It also relies on the linguistic analysis of the question. The main concepts regarding question analysis are outlined below:

- **Question Type** (QT). This attribute is used to identify the nature of the information requested. The simpler question types are Factoids and Definitions. Other types of questions could be derived from these simple types by composition, like lists of facts, or by adding restrictions like temporally restricted questions. Other complex types have been

proposed like asking for opinions, procedures or causes as introduced in Section 2.2.2. The definition of these types is in fact very motivated by the applications, but other question types taxonomies have considered pragmatics or information needs as the basis of the classification [Pomerantz, 2005].

- **Expected Answer Type** (EAT). The answer type refines the type of information that is requested. It is more common in the case of Factual questions, where it is necessary to specify additional information in order to restrict the semantic category of the answer. There is some parallelism between NE taxonomies and EAT taxonomies but a one-to-one mapping is not required. The issue is specific to every system as one system may detect a concept for an EAT (MONEY) but it needs to revert to a generic abstraction (NUMBER) due to the limitations in their semantic processing. A well-known EAT taxonomy which has been used in several experiments in Question Classification is reproduced in Table 2.6 [Li and Roth, 2002].

- **Question Focus** (QF). The focus of the question is the key functional concept. There is no formal definition but it is considered as the word that provides more information to select the proper answer type. The interrogative pronoun is a good question focus for example *When*, *Who* or *Where* or their equivalent in other languages. In contrast *What* and *Which* are not useful and the focus is taken usually by the word that follows them that help to restrict the EAT.

- **Question Topic**: This the central component of the information need as the question is requesting additional information on the topic. Documents that have an answer should contain the topic, if not textually at least a reference to it. The topic is often an entity in the real world, and often represented as a Named Entity. In definition questions it is the concept to be defined or *definiendum*. It also plays an important role in order to guide the EAT and the QT identification. For example, once we know the topic is *Stalin* and that it is a person, additional domain knowledge could be used to restrict what could be asked.

The kind of NLP techniques that are of interest in Question Analysis really depends on those that are used also in the Answer Selection step. Deeper analysis could in fact be used because the amount of text is smaller. On the other hand, questions have been usually ignored in practical NLP tools and tools require adaptations. For example, the CONTEXT syntactic parser used for Webclopedia/TextMap [Hermjacob, 2001; Lin et al., 2001] was re-trained to deal with questions correctly.

The basic question representation can be enhanced using NLP tools, language, domain and common-sense knowledge. The enhanced representation can be exploited to succeed in later stages of the QA process in several forms. For example, keywords can be normalized using lemmatization. The lemmas could be enriched with synonyms or derived forms using lexical knowledge like Wordnet. Additional name variants could be added using external knowledge and Name Entities can be detected reliably. Terminology and collocations can also be detected as it has been the focus on [Grau et al., 2006]. On the other hand, syntactic, dependency and semantic representations could be employed to enable rules or heuristics that work at this level of representation.

2.8.1 Question classification

Question words or interrogative pronouns are a good start for EAT classification, and therefore Rule-based systems are common for Question Classification. On the other hand, more knowledge is needed for *What* and *Which* questions that are associated with nearly every EAT. Locating the Question Focus in a reliable manner is key to being able to discover the type of the question. A key difference in Question Classification is the type and number of types that a system uses.

Example 2.15 Enhanced Question analysis for the question *¿Cuándo murió Stalin?*

- Question type (QT): FACTUAL, DEATH_DATE
- Expected Answer Type (EAT): DATE, YEAR.
- Question Focus (QF): Cuándo
- Question Topic (QTopic): [*Stalin*: José Stalin, Stalin ...]
- Keywords:
  - *Stalin/NP/PERSON*: José Stalin, Stalin ...
  - *murió/VB*:
    - *morir/VB*, murió, muere ...
    - *muerte/NC*: ...
    - *fallecer/VB*: ...
- Constituent Representation
  - S(Cuándo VP(*murió*) NP(*Stalin*))
- Dependency Representation (dep, head)
  - (Cuándo, *murió*)
  - (*Stalin*, *murió*)
- Semantic Restriction
  - PERSON('Stalin')
  - DATE(f1)
  - DEATH_DATE(f1,'Stalin')

Rule-based systems have been used for those systems using a few types, even a dozen. Detailed taxonomies like those used in Li and Roth [2002] (reproduced in 2.6) or [Sekine et al., 2002] (used in 3.8 and 3.9) require other approaches. As the number of classes increases it becomes burdensome to create and maintain a large set of rules. Adding or deleting a new rule or condition often carry side effects that were not planned. It is also clear that the importance of wide coverage lexical resources becomes more important too.

2.8.1.1 Statistical methods

Because of the maintenance problem several works have approached it with the use of statistical and machine learning methods [Hacioglu and Ward, 2003; Zhang and Lee, 2003]. In fact, it is relatively inexpensive to create from resources like those generated in TREC. Question classification has been usually modelled as a N class classification problem and state-of-the-art discriminative classifiers like SVM [Zhang and Lee, 2003] and SNoW [Li and Roth, 2002, 2005] have been used. Similar methods have been used for other languages like Japanese [Suzuki et al., 2003], Spanish and Portuguese [Solorio et al., 2005].

In order to approach the classification into finer classes it is possible to use the structure of the taxonomy to learn a hierarchy of classifiers. The proposed architecture is a two-layer architecture. One classifier is specialized in the coarse category while other classifiers are trained for each of the descendants of that type. Robust error correction like in Hacioglu and Ward

ABBREVIATION
abbreviation
expression

DESCRIPTION
definition
description
manner
reason

ENTITY
animal
body
color
creative
currency
disease/medicine
event
food
instrument
language
letter
other
plant
product
religion
sport
substance
symbol
technique
term
vehicle
word

HUMAN
group
individual
title

LOCATION
description
city
country
mountain
other
state

NUMERIC
code
count
date
distance
money
order
other
period
percent
speed
temperature
volume/size
weight

Table 2.6: Question type hierarchy used in Li and Roth [2002] including 6 coarse and 50 fine-grained types

[2003] can also be applied in this case. Similar architectures has been also applied in Li and Roth [2005] and Suzuki et al. [2003].

Language Models techniques have also been applied to Question Classification although initially with less success. Nevertheless, due to the inherent semantic nature of the task they remain interesting. They estimate the probability that a question form is produced provided that the given information need and their EAT are known $P(Q|c)$. Using the Bayes Rule and independence assumptions is possible to produce a generative classifier like this that uses bigrams of words.

$$P(c|Q) = \arg\max_c P(Q|c)P(c) = \prod_{i=1}^{n} P(w_i|w_{i-1}, c)P(c) \tag{2.12}$$

While Zhang and Lee [2003] have shown that this kind of models performs worse than discriminative classifiers. Merkel and Klakow [2007] achieve an improvement in the performance by the use of smoothing techniques. A different technique that uses some sort of similarity is presented in Van Zaanen et al. [2005] which se a suffix-tree structure to memorize questions and their analysis. Using the question suffix tree and an approximate matching algorithm the analysis of the questions is provided by analogy with those memorized.

Although the name of Question Classification seems to imply the categorization in one single class, it could be desirable to consider that a question could be associated to several classes. This could be useful to accommodate errors or ambiguities [Croft and Metzler, 2005] in questions and minimizes the risk of making an error in this step. Several other models have avoided to do classification at all and restricting themselves to a given taxonomy. For instance, [Lin and

Pinchak, 2006] uses a statistical model to rank possible EATs while Lita and Carbonell [2004] uses clustering to select question strategies.

2.8.1.2 Linguistic representation

Lexical surface representations, like bag of words or bag of bigrams, are inherently useful to capture the main structure useful for question classification. Rule-based system usually used sequence of words or patterns but statistical system have been used these representations too. The use of syntactic information and tree kernels, similarity functions defined for syntactic trees, has improved the performance for taxonomies with few classes. On the other hand syntactic information is not by itself useful when a larger number of classes are considered [Suzuki et al., 2003; Zhang and Lee, 2003]. For fine grained classification, semantic information is far more important. Lexical resources like Wordnet have been integrated to improve the coverage of manual rules and patterns. Synonymy and hipernonymy relations could be useful in order to deduce from the Question Focus which is the EAT of a factual question. For instance, Pasca [2003] describes how Wordnet is projected to the Answer Type hierarchy of the LCC system as well as an algorithm to crawl the hierarchy to find the correct type. Li and Roth [2005] explored how to use Named Entities, Wordnet, a list of words related with each EAT and a list of distributional related words to improve the accuracy on fine-grained classification. They found Wordnet less useful when compared to the rest of resources because of sense ambiguity. Croft and Metzler [2005] support similar conclusions. In contrast, recent works have focused on locating the Question Focus [Huang et al., 2008; Krishnan et al., 2005] and integrate the Wordnet hierarchy into the learning process. Both of them have shown improvements in using WordNet.

Finally, in the standard English dataset [Li and Roth, 2002] the accuracy for the coarse-grained classes (6) is above 94% percent and for the fine-grained (50) is about 89% [Huang et al., 2008].

2.9 Information Retrieval

There are significant differences between traditional document oriented IR and IR applied to QA. Document oriented IR has been initially worried with topical relevance of documents to a query or *aboutness*, although it has broaden their scope since the 90's. Topical relevance is still important in IR for QA but it is not essential. Answers can be found in documents that are only tangentially related to their main topic. Besides, IR for QA is a focused task, where the objective is not to find documents but smaller, content-focused units that answer the question. Finally, even if topical relevance is not required, it is also not enough, because the retrieved units should contain answers.

Most of the popular IR retrieval models have been used by any of the QA systems deployed in shared evaluations. However, as the evaluation has focused in end-to-end QA performance there is not extended evidence whether one of them is more suited to the QA problem than other. The LCC system [Pasca, 2003], one of the most successful TREC systems, as well as Tellex et al. [2003], defend that the boolean model could indeed be appropriate for QA. Other systems [Ahn et al., 2005; Bouma et al., 2006; de Pablo-Sánchez et al., 2007b] have been using the Vector Model, the probabilistic or Okapi-BM25 model or language models for retrieval. Most of them have benefit from the availability of open source or research implementations like Lucene [12] (vector), Xapian [13] (Okapi), Indri(LM) [14]. These models have been widely studied in document retrieval, but as we stated before, QA systems usually benefit from smaller focused

[12] http://lucene.apache.org/
[13] http://xapian.org/
[14] http://www.lemurproject.org/

retrieval units that are usually know as passages. The granularity of these passages can vary widely from document sections to paragraphs or sentences. While it is not yet clear if there is an approach that outperforms the rest, the IR component deserves further study.

Several studies have shown that relevant passages are missing for a large part of the questions [Greenwood, 2009; Saggion et al., 2004; Tellex et al., 2003] which imposes a hard limit to the maximum performance of a QA system. Moreover, it is often required to process a large number of documents which usually affects the performance and effectiveness of later stages. Work into improving the retrieval performance for QA has been organized in three inter-related aspects: (1) the retrieval of focused units like passages, (2) query generation from questions and (3) document representation.

2.9.1 Passage Retrieval

Passage Retrieval is more suited to the needs of QA, and in particular, to Answer Selection techniques which require expensive language processing. As at least part of process is performed online the user experience is usually improved by retrieving smaller focused units. This helps to reduce the amount of online processing that is needed and overcome the robustness problems of some tools. A different alternative consists on using as much offline processing as possible. Passage retrieval systems could be considered as the direct ancestors to actual QA systems. They were popular in the first TREC evaluations. The goal initially was not to find an exact answer but a focused textual fragment that could answer the question.

Two are the main alternatives for building a passage retrieval system:

- **Passage indexing**. In this case the document is segmented into passages previous to the indexing step. Each passage is treated then as if it were a document in classic IR. In this case, the first decision consist on define what is the appropriate granularity of the passage and how the documents are going to be segmented. Alternatives for segmentation include the use of document markup or use a simple segmentation based on sentence identification. In this case several sentences could be grouped in order to form a passage. Parameters include the size in sentence and the overlap among passages. Sliding window passage segmentation considers passages that overlap maximally, being only the first and the last sentence of each passage different. More informed tiling alternatives treat to retain thematic coherence. The last include thematic segmentation like the use of TextTiling or approaches retaining correference relations. Most of them have been compared in Tiedemann and Mur [2008] which conclude that simpler N sentence approaches with no overlapping are simple and provide better results. Retrieval models need to be modified in the case of passage retrieval to account for some of the differences with document retrieval models. For example, length normalization is often disabled as passages tend to have a more uniform distribution in the length of terms than documents. In the context of models, Croft and Murdock [2004] propose an extension to the language modelling approach that is suited to sentence retrieval and therefore useful for QA. Gómez-Soriano et al. [2005] use a reranking approach based on n-grams overlap to account for the similarity between the question and the passage bearing a good candidate answer.

- **Passage selection**. The approach in this case is to use document indexing and to postprocess the document result in order to provide passages. Passages can be created with any of the previous techniques and are usually re-evaluated with regard to the query. An adavantage of this approach is that the length of the passage can be decided at query time. There are two additional alternatives in using the passage selection technique for QA. The resulting passages can be filtered or they can be reranked using different criteria to prioritize the process of AE. Tellex et al. [2003] is one of the most comprehensive studies of different passage retrieval and reranking strategies used in TREC. They analyzed up to eight passage retrieval strategies including some of the most succesful QA passage systems

from TREC. Passage Selection systems include heuristics like the presence and importance of question terms, their distance or density and their relative order. They conclude that systems that include term density based heuristics like MultitextQA [Clarke et al., 2006] are able to provide a larger number of relevant passages at lower rank. Monz [2004] presents another model for sentence selection adapted to the QA task that also makes use of overlap and density. On the other hand, a different alternative uses syntactic relation passage retrieval methods like Cui et al. [2005b]. Their method estimates the correlation between paths in the dependency tree of the question and the retrieved sentences and uses it to rank passages.

2.9.2 Query Generation

Another important aspect related to the retrieval process is how to transform the question into an effective query that is biased to retrieve passages containing answers. There is no obvious way to translate a question into a query and how the terms should be combined. In fact, this is an aspect that is very dependent on the query language and the features implemented in a retrieval engine. IR systems usually provide a query language that include boolean operators, phrase and proximity search. Besides, some query languages allow to modify the weight that certain terms should play into the final score. All these features are at a large extent specific to an implementation as they should combine the capabilities of the query language with the retrieval model behind. Nevertheless, query construction influences in a decisive way the process.

The baseline technique consists on adding all non-stopwords as query terms and let the retrieval systems do the rest. Nevertheless, as queries are key to locate the correct snippets to extract the answers several works have explored the ample range of options. [Bilotti et al., 2004] explores whether using stemming at indexing time or morphological expansion at query time work better. Morphological expansion provided better recall although it also requires more linguistic knowledge. However, the improvement is moderate and in fact, it depends on the metric. They use different tenses for verbs, plurals for nouns and superlatives and comparatives for adjectives which are rather regular for English. Whether these result apply to other languages and what is the trade-off is still unknown. Another conflicting issue regards the use of compound terms or multiword terms. Should NEs, acronyms, domain terminology and other compound terms be transformed into a phrase query, single terms or both?. de Pablo-Sánchez et al. [2006a] presents initial attempts to select the best query formulation for these kind of terms. Khalid et al. [2008] shows how the normalization of NE at indexing time could improve QA retrieval results. In contrast, Pizzato et al. [2006] was not able to obtain benefit when using NE in query expansion. In fact, query expansion based on different forms of relevance feedback has been controversial and difficult to perform well in QA. The reason could be that the analysis of local context for question terms does not necessarily help to locate passages containing the answer. On the contrary, Zukerman and Raskutti [2002] show a method to improve the coverage of passage retrieval by using lexical knowledge like Wordnet to produce question paraphrases. Besides, Sun et al. [2006] propose a method to discover which dependency paths between terms could be useful for term expansion. Their technique improves the results of density-based passage retrieval. The technique is extended to expand the query with new dependency path relations that are also useful for relation based passage retrieval as presented in Cui et al. [2005b].

Some systems have not used just one query for question, but several ones instead. These queries could be defined as reformulations patterns aimed at locating the answers in very specific contexts [Schlaefer et al., 2006] or they can be aimed at improving the recall of the documents retrieved by providing alternative formulations [Zukerman and Raskutti, 2002]. Web QA systems have found that there are effective forms to bias the results towards snippets that are easier to extract. Some systems [Banko et al., 2002a; Echihabi et al., 2006] have rephrased questions as affirmative sentences to locate answers on the web as almost direct hits. These technique make use of the inherent redundancy in the Web, where the same fact could be expressed closer to a

question reformulation.

The LCC system introduced the technique of query relaxation [Pasca, 2003]. This technique starts with an initial query that includes all the relevant terms from the question. The system transforms the query depending on the number of document results that aiming at a number of documents in the few tenths. If a query obtains too few results the most restrictive operators are transformed, for instance an AND operator is changed by an OR operator. Other operations could include the removal of question terms depending on their selectivity. Further related experiments on how to use different query building techniques on Boolean Retrieval systems have been carried in Saggion et al. [2004] and confirm the usefulness of the iterative refining strategies. On the other hand, the results lie behind than those systems using ranked retrieval. Radev et al. [2006] presents a method to learn building queries for Web search engines in order to retrieve relevant passages for QA. They define a set of query rewrite operations like *REMOVE*, *DUPLICATE*, etc. Using a set of question answer pairs is able to learn the sequence of transformations that should be carried in order to retrieve the best possible document set. It uses Expectation Maximization [Dempster et al., 1977] in order to estimate the probability that an operation produces better results.

It is widely accepted that all of terms in a question are not equally useful for retrieval. Relevant terms frequently co-occur with the answers, but others could be too common and introduce too much noise. For example, the term *city* in *What city is Disneyland in?* is not as useful to retrieve passages as *Disneyland*. Works like Ramakrishnan et al. [2004] have focused on tagging the role of different terms in the questions in order to take advantage of that characteristic.

2.9.3 Document Representation and Indexing

Another aspect to consider, which is also related to passage and query representation is the internal representation of documents. The standard representation of documents in IR has treated them as a weighted set of terms. It is common to apply linguistic transformations to the document words like stemming and stopword removal. More complex linguistic representations have also been proposed although multi-word recognition and complex lemmatization have been also proposed [Goñi Menoyo et al., 2006; Strzalkowski, 1999].

Prager et al. [2006a]; Radev et al. [2000] proposed the use of predictive annotation for the retrieval process of QA. With these techniques Named Entities are tagged in an offline process prior to indexing. The text and their annotations are indexed. The query language is modified to allow making explicit NE tag restrictions. This allow to retrieve only passages that already contain the desired NE types which correspond with the EAT.

Other systems have started to use customized retrieval systems that represent documents as more complex linguistic structures. Negri et al. [2003] have used dependency trees to represent sentences in the index. Tiedemann [2005] used the information of a dependency parser to build a complex index with different syntactic and NE information. The final index is formed by eleven different fields and requires the use of a Genetic Algorithm to tune the correct weights for the scoring function and the queries. Other complex graphs representations that have been proposed include those used by Mollá and Pizzato [2008] for a simplified semantic role-based representation and Hartrumpf [2005] for semantic networks. As the representation becomes more complex, the query language and the query building is also increasingly difficult.

As documents start to be represented not just as set of terms but as structured sets of linguistic information the trend to use XML and structured retrieval systems is going up [Bilotti et al., 2007; Chu-Carroll et al., 2006; de Rijke et al., 2006; Litkowski, 2005]. In this setting, linguistic annotations are casted into a tree structured data model like that of XML. Structured query languages that allow expressing restrictions on term occurring as part of Named Entities, arguments in relations or filling certain role in a frame are used. Examples include the XML Fragments language Chu-Carroll et al. [2006] and the structured annotation graph proposed by

Bilotti et al. [2008] for the Indri engine. Beyond efficiency, Bilotti et al. [2008] name some of the challenges that focused retrieval engines should face on using linguistic annotations. Most models have treated explicit annotations as hard constraints but errors and partial matches should be included too. Besides, principled methods to combine evidence from several partial matches and multiple potentially relevant structures are also needed. Nevertheless, Chu-Carroll and Prager [2007] show that the retrieval process can be improved even with state of the art annotators and provide evidence for further improvement as the quality of language annotators improve.

2.9.4 Retrieval for other Question Types

Besides the techniques presented above, some specific questions types could benefit from specialized retrieval techniques. For example, the retrieval of passages is crucial for questions with longer answers like definitions or why questions. In fact, most of the complexity for these types requires selecting the correct sentences or passages, because there is no Answer Extraction per se. The first challenge for definition questions is to find related snippets, as they usually contain few question words. External resources like Wikipedia, Wordnet [Xu et al., 2003] or the Web Chen et al. [2006] are used to build a context vector of terms related to the definiendum to improve recall. Definitional patterns could also be used to filter or re-rank sentences regarding their suitability as definitions Cui et al. [2005a]; Xu et al. [2003]. In contrast, rhetorical relations can be mined to be able to detect passages and sentences answering why questions Blair-Goldenshon [2007]; Verberne et al. [2007] and as well integrated as features in ranking functions learned from data.

2.10 Answer Selection

The Answer Selection module is best understood in terms of their inputs and their outputs. The input is usually a set of relevant text passages that the previous step considered related to the question. In addition, we have all the information that we produced in the question analysis step. The outcome should be a ranked list of answers ordered by an appropriate criterion that usually takes into account their estimated correctness. Ideally, the best ranked document is the correct one; otherwise the module should not produce an answer.

Answer Selection is a complex task that involves other components which extract candidate answers from the text and manipulate the candidate answer to complete the final results. The process includes several steps that we have organized in three block of main techniques Answer Extraction, Answer Ranking and Answer Validation. Other apparently simpler techniques include filtering and merging answers. Answer Selection is one of the most challenging modules in a QA system as it deals with large quantities of text from the passages and should apply task heuristics, language and world knowledge in order to arrive to a final answer.

2.10.1 Answer Extraction

An Answer Extractor receives a set of passages and the analyzed question and its objective is to extract text chunks that could be considered answers. One or several candidate answers or text chunks could be extracted from the same passage. It is the task of later stages like Answer Ranking or Validation to decide if the answer is in fact good. Nevertheless some of the methods can be seen as doing extraction and ranking at once because they also assign some sort of score that represent their correctness.

There are several ways to look at the techniques that have been used in Answer Extraction. The simplest exposition, which also aligns with our objective of evaluating their usefulness for different languages, makes use of the level of linguistic analysis that is used. Most of the techniques are compatible and in fact their results could be combined as explained in Section 2.11.

2.10.1.1 Knowledge-Based systems

This strategy was the one used by first QA system and in a web setting by the START [Katz et al., 2001] system which pioneered Knowledge Annotation techniques. The information is structured in Knowledge Bases that could take different forms like ontologies [Lopez et al., 2006], databases [Katz et al., 2001; Prager et al., 2006b] or expert systems [Friedland et al., 2004]. Some systems have also used lexical word networks like Wordnet to answer certain types of questions [Prager et al., 2002]. The clear advantage of Knowledge-Based system is that they perform lighter processing at the online QA usually providing direct answers. When dealing with closed-collections like in TREC and CLEF, the answers from the knowledge bases need to be projected, or it should be found again in the provided collection.

The distinctive characteristic of a Knowledge Based system is that information needs are anticipated. This is useful in order to organize the sources and the schema of the information that is going to be used. It is also possible to preselect existing Knowledge Bases to the task. In fact, compiling the knowledge is in fact the problem for both open and restricted domain QA applications. Large-scale knowledge aquisition projects like KnowItAll [Etzioni et al., 2005] aim at extracting these knowledge from the Web at large. Others like the TQA approach [Agichtein et al., 2007] use the structured part of the web like list and tables.

2.10.1.2 Data-Driven

Several system have proposed the use of Statistical Methods, Data Mining or Machine Learning directly on pairs of questions and answers as a mean to obtain a QA system. These methods rely on a simple representation of question and answer and integrate just very shallow language analysis or none.

- **Frequent N-grams**. The AskMSR system [Banko et al., 2002a] transforms the question into an affirmative sentence using some simple manual transformations. These transformations are sent as queries to a Web search engine to retrieve the most relevant snippets. Candidate answers could be retrieved as completions of the query. The example question could be transformed into *S̈talin murió en¨* which is expected to retrieve snippets containing sequences like *... Stalin murió en 1953 ....* On the other hand, irrelevant sentences would be also retrieved. The key to the method is how to select the correct string under this situation. The system relies on the rule of thumb that the words that appear more frequently on the retrieved snippets should be correct. AskMSR uses the retrieved snippets to generate all n-grams of different length up to a maximum. A scoring formula that considers length of the n-gram and its frequency is used to score candidates. The system is simple but very effective when used in a huge and redundant source like the Web, because correctness and frequency are frequently correlated for many questions. The Aranea [Lin and Katz, 2003] system includes also a stream based in this idea and refined the idea using additional Knowledge-Bases.

- **Language-model based**. This is the basis of systems like the University of Tokio [Whittaker et al., 2005a,b] which presents a complete system based on statistical language models. The typical use of a language model is to predict the probability that a sequence of words belongs to a language. The hypothesis here is that a language model of sentences that answer a given type of question could be built from a large corpora and a collection of question and answers pairs. The model is factorized into a retrieval model, an answer filter model and answer length model. The filter model estimates the probability that an answer is of the given type. The estimation of the model parameters is highly unsupervised and data driven, in contrast with more complex supervised alternatives like Ittycheriah [2006]. This system has been available in the web [15] for several languages which demonstrates the

[15] http://asked.jp

viability. Nevertheless, the performance varies widely between languages and it seems to depend on the amount and quality of the resources used for training the language models.

2.10.1.3 Shallow Linguistic Analysis

The use of NLP techniques was the natural evolutionary path from Information Retrieval systems to QA systems, especially those that have shown useful in Information Extraction. We distinguish here between those systems that perform a shallow linguistic analysis and a deep analysis. Our criterion to the division is decided by the use of full syntactic parsing. Altghough it has experimented a large increase in accuracy during last years, it is still error-prone and computationally expensive.

Most of the systems that have participated in TREC, CLEF and NTCIR combine techniques like POS-tagging, NERC or shallow parsing which fall into these categories. In contrast, full parsing of a TREC like collection is complicated and it will not scale in short to larger collections, not to mention the web. Other proper IE techniques like relation extraction are also considered here. Most of the techniques are in fact combined.

- **Entity extraction** [Abney et al., 2000]. The type or class of an entity as given by a NERC module could be used to extract answers that are compatible with the EAT of a question. Further heuristics like their proximity to the question terms could be leverage here also or at later stages like ranking. The coupling between the types of the NERC and the types of the EAT is very high. Recognizing a larger number of types gives a competitive advantage and therefore additional efforts spent into recognizing numbers, quantities and dates based on language rules usually pay off. Systems that use a NERC module have tried to extend the types using manually constructed gazetteers as well as methods to perform fine-grained categorization of entities Fleischman and Hovy [2002]; Schlobach et al. [2007].

- **Indicative patterns** The system proposed by Soubbotin [2001] was the first to use lexical indicative patterns to extract very precise answers with very good results in TREC 2001. One of the main problems with this approach is that the patterns were manually written for each question type. A large number of them may be required to obtain good recall. In this case, the question type expresses a relation between the focus and the answer like Birthdate. Later on, Ravichandran and Hovy [2002] have proposed a method that is able to acquire patterns from the web by using question and answers pairs. The method is also able to evaluate the effectiveness of the patterns. Echihabi et al. [2006]; Ravichandran et al. [2003] also show how this strategy can be implemented in a larger system and combined with other strategies in the TextMap system by ISI-USC.

- **Information Extraction**. In this category are those systems that go beyond NERC by employing IE techniques like relation extraction. The system designed by Cymfony [Srihari et al., 2006], in the present Janya, uses three layered strategies for answering questions, NE, Relations and Subject-Verb-Object structures. The question type is used to select the best strategy. If an specific strategy is not able to find an answer the system falls-back to a more generic strategy. Some relations have been predefined for the most prominent types of entities. For example, for an entity of the type Person the system is able to extract the Age, the Birthdate and Birthplace and Position among others.

2.10.1.4 Syntactic methods

Several systems have researched the utility of full syntactic analysis for QA since the inception of the TREC task. The motivation for the usefulness of syntactic analysis in Answer Extraction assumes that the analysis of a question and the analysis of a sentence bearing an answer should be similar. The role of the question particle should be fulfilled by one of the groups in the

sentence analysis. Besides, complete subtrees of the sentence parse could be ignored when the sentence contain additional spurious information.

Some of the pioneering systems on the use of deep linguistic analysis were LCC [Pasca, 2003] and IRST-itc [Negri et al., 2003; Tanev et al., 2005] which employed *dependency parsers*. One good reason for the popularity of dependency parsing was the availability of robust parsers for English like the Link Parser [Sleator and Temperley, 1993] or MINIPAR [Lin, 1998]. Other languages like Dutch have also wide coverage dependency parsers like Alpino [Malouf and van Noord, 2004]. Recently, research into data oriented dependency parsers has increased [Nivre et al., 2007] and useful parsers for different languages are becoming available. Nevertheless, syntactic parsing is still not ready for large scale text processing and analyzing a large collection is expensive. For example, Bouma et al. [2006] reports experiments on parsing CLEF corpora. They report that on 2005 they need a cluster with 128 machines during three weeks in order to parse the 1GB text with 4.1 million sentences. For that reason, syntactic analysis is in most systems only performed on the small set of selected passages. Parsing accuracy is another concern, especially for questions, which require further adaptation. There have been mainly two approaches on using syntactic information:

- **Syntactic patterns**. The system of the University of Groningen [Bouma et al., 2006] applied dependency parsing to the whole CLEF corpus. A path is defined as the set of edges that should be traversed from one term to another term in the dependency graph. They devise a retrieval system to store, index and query the paths. At query time, patterns defined on the dependency tree as composed of paths are coupled with each of the question types. Syntactic patterns built from the question were also used by the IRST-itc system [Negri et al., 2003; Tanev et al., 2005] combined with a measure for similarity.

- **Syntactic Overlap**. One of the most successful systems to employ syntactic information has been developed by the NUS team [Cui et al., 2005b]. Their strategy consisted on producing triplets of term-relation-term $P =< N_1, R, N_2 >$ which are extracted from paths in the dependency tree. The relation is in fact a sequence of syntactic tags found in the path $R = r_1, \ldots, r_n$. A corpus of questions and their answers is used to estimate the match between common relations in questions with relations in candidate sentences. This representation has been extended in Shen and Klakow [2006] with a method to measure the similarity based on the correlations among dependency relations R. Besides, it also takes into account the similarity of the arguments of the path triplet to find the best matching answers $Corr < P_i, P_j >= Corr(R_i, R_j) * Sim(N_i1, N_j1) * Sim(N_i1, N_j1)$. They show that their method outperforms NE based answer extraction and other syntactic methods. It is specially suited for questions which do not expect a NE as answer.

2.10.1.5 Semantic and Pragmatic analysis

Several English systems have used lexical resources like Wordnet. Most of them use Wordnet as a source for knowledge, in answer validation or question classification [Pasca, 2003]. One system which relied on Wordnet to build a semantic model for QA was proposed by the Universidad de Alicante [Vicedo, 2002] for TREC. Questions and candidate answers were represented like a vector of concepts or synsets and a similarity measure was used to select answers.

Semantic Role Labelling (SRL), a form of shallow semantic parsing, has been frequently cited as an enabler of QA systems [Gildea and Jurafsky, 2002] as it assigns a semantic interpretation to sentences based on verbs and their arguments. Language resources like PropBank [Palmer et al., 2005], FrameNet [Baker et al., 1998] and VerbNet [Schuler, 2005] provide the interpretation for the verb frames. Systems that perform SRL have been developed and are available like ASSERT [Pradhan et al., 2004] or SWiRL [Surdeanu and Turmo, 2005].

TREC Systems that have attempted to use SRL and related resources to perform AE include the NUS system [Sun et al., 2005], the Alyssa system [Shen and Klakow, 2006] and OpenE-

phyra [Schlaefer et al., 2007]. The main problems with deep approaches are the coverage of the resources and the accuracy and computational cost of SRL parsers. To alleviate coverage problems, most system have relied on simpler techniques as a backoff. The trivial strategy to use semantic roles has been to directly align semantic parses of questions and answers. The argument that fulfills an equivalent role to the question particle (or the EAT) is given as a candidate. Nevertheless, the correspondance between EAT and verb argument is not unique or even trivial as Palomar et al. [2008] shows for WHERE questions and ARGM-LOC arguments. Sun et al. [2005] use Wordnet verb relations to partially overcome the coverage problem. Kaisser and Webber [2007] show that the relations and hierarchy between verb frames in FrameNet also helps. Recently, Shen and Lapata [2007] have proposed a semantic structure model that represents sentences as a bipartite graph between constituents and arguments to overcome labelling errors of SRL systems. A similarity measure based in previous work with dependency relations is used to pair questions and sentences and extract answers.

Other semantic representations like full Semantic Networks have also been researched in systems like InSicht [Hartrumpf, 2005]. Due to the computational requirements of such tasks the corpus is often completely processed offline and queried online to retrieve the relevant processed sentences. It is common that not all sentences are parsed due to low-coverage of the actual rules or errors. Mollá [2006] proposes also the use of semantic representations using the formalism of Conceptual Graphs and it uses methods to learn and match patterns at the graph level.

Finally, we analyze the use of correference in QA to extract candidate answers, which has been explored in Morton [2000] and Vicedo and Ferrández [2006]. The main conclusion is that the use of correference is useful when the source is small and there is less redundant information. In larger collections it seems that the low accuracy of co-reference systems and their complexity do not pay-off yet.

2.10.1.6 Inference Based Methods

The LCC system [Harabagiu and Moldovan, 2003; Pasca, 2003] is currently the most complex system among those reported at TREC and similar evaluations. It uses dependency analysis as an intermediate step to transform the question and the candidate passages into logic formulae. Once they have accomplished the transformation, several methods are combined to extract and validate answers combining deduction and abduction. External resources like Wordnet have been combined in the inference process too.

The use of inference engines was pioneered by classical systems [Green, 1968] and also in recent ones like the HALO project [Friedland et al., 2004]. It is important to note that these systems have used manually engineered knowledge, which is clean and fairly correct. In contrast, the use of automatic generated propositions poses new challenges. Deduction is problematic in open domain and even in restricted domain where all the knowledge has not been represented. The LCC system uses abductive reasoning [Harabagiu et al., 2003a; Harabagiu and Moldovan, 2003] instead, which is useful for the problem of mining answers. They report a substantial improvement in the number of correct answers over their system based on extraction in TREC2003.

2.10.2 Answer Validation

Answer Validation has been proposed as a complementary step for newer QA systems. The objective of an Answer Validation module is to decide wether the extracted answers are correct or not. The task is often formulated as a Textual Entailment (TE) test. An hypothesis is stated and the task consists on deciding if another text support the same hypothesis or not. In Answer Validation the question and their answer form the hypothesis which should be inferred from the supporting text. General TE techniques have been applied [Bar-Haim et al., 2006; Peñas et al., 2007], for example by transforming the question answer pair into an affirmative sentence. Several forms to include TE modules in a QA system has been successfully explored

in [Harabagiu and Hickl, 2006]. In fact, TE engines incorporate a wide range of linguistic information and knowledge resources similar to those analyzed for Answer Extraction. The difference lies that the objective is not to extract an answer but to judge its correctness.

TE Approaches have included lexical and paraphrase resource, Named Entities and syntactic and semantic information. Heuristics that consider word overlap are also integrated as well as specific clues like *negation*. While the specifics of each system varyes, a common trend consists on using Machine Learning methods to estimate the contribution of each feature derived from the different analysis levels. Due to the large number of phenomena the scarcity of training pairs is an important problem. [Harabagiu and Hickl, 2006] and more recently Celikyilmaz et al. [2008] have explored methods to use unannotated data.

A different approach to Answer Validation is based on the use of external knowledge that is complementary or additional to that used in the extraction process. For example, the use of Wordnet or Wikipedia to check the correctness of an answer could be seen as an example of KB validation. In fact, logic-based approaches to answer extraction like those of LCC [Harabagiu et al., 2003a; Harabagiu and Moldovan, 2003] could be classified as validation as they used an extended version Wordnet to check for correctness. Another pioneering approach on Answer Validation used the Web as a knowledge source [Magnini et al., 2002] to validate answers by testing the hypothesis using co-ocurrence.

For some types of questions it is possible to devise specialized checking mechanism like the *QA by Dossier* approach [Prager, 2004]. Encoding world knowledge like that the person birth should have happen before their death it is possible to prune or rerank inconsistent candidate answer. In general, knowledge about the characteristic of the entities and the relations included in the question can be employed as far as they are informative. A generic alternative is *Question inversion* [Prager et al., 2006c] which uses answers to formulate complementary questions. In this case only knowledge about the relation is used to produce the inverted question. Nevertheless, the questions could be much less restrictive and even very unuseful. Several examples of different validation techniques are presented in the Example 2.16.

Example 2.16 Example of different validation techniques and their applicability

Question Answer Pair	Validation technique	
¿Cuándo murió Stalin? 5 de marzo de 1953	Textual Entailment	Stalin murió el 5 de marzo de 1953
	QA by Dossier	¿Cuándo nació Stalin? ¿Cuándo se casó Stalin?
	Question Inversion	¿Quién murió el 5 de marzo de 1953?
¿Cuál es la capital de Croacia? Zagreb	Textual Entailment	Zagreb es la capital de Croacia
	QA by Dossier	
	Question Inversion	¿De qué pais es capital Zagreb? ¿Donde está Zagreb?
¿Cómo murió Jimi Hendrix? asfixiado	Textual Entailment	Jimi Hendrix murió asfixiado
	QA by Dossier	¿Dónde murió Jimi Hendrix asfixiado? ¿Cuándo murió Jimi Hendrix asfixiado?
	Question Inversion	¿Quién murió asfixiado?

2.10.3 Answer Ranking

The objective of the Answer Ranking module is to produce an ordered set of answers. These would be the final answers as presented to the user but similar techniques could be used also at intermediate steps. The objective of the Answer Ranking module is to predict the estimated

relevance value that the answers could have for a user. Usually the estimated value is zero if the answer is incorrect and one when this is correct for the case of a factual question. Graded or rational estimated values can be useful when dealing with more complex questions.

Several heuristics have been crafted in order to assess the relevance and correctness of a particular answer. Ittycheriah [2006]; Pasca [2003] enumerate those used in their systems but almost any system uses their particular approach. Depending on the kind of information that is used they can be classified on local and global heuristics. Local heuristics only concern the relation between the question, an answer and the passage that was used for extraction. Global heuristics use information beyond the passage bearing an answer. They draw information from the whole set of candidates answer or global statistics from the collection. Most of the heuristics that can be integrated in a ranking function are fairly general. In contrast, systems with a greater level of language analysis like dependency parsing or specialized resources could leverage those also at this point.

- **Expected Answer Type relatedness**. The EAT of the question matches the class assigned to the answer. If a broader taxonomy of answer types is available, closer subtypes and supertypes could also be used in order to improve recall.

- **Questions terms**. The occurrence of questions terms in the passage supporting the answer is a good indication of the relatedness of the passage. Besides, if the relative order of the questions terms is also preserved points out to a simple paraphrase of the question. Other question analysis features like the focus of the question or the theme could be employed. For example, Ittycheriah [2006] uses Wordnet to test if a hyponym of the focus appears on the passage.

- **Named Entities** and other **complex terminology** that is mentioned in the question and also appear in the proximity of the answer are a good indicator of theme relatedness.

- **Proximity-based features**. Measure how close the answer appears to the question terms that appear in the passage.

- **Punctuations signs** and other **shallow linguistic cues** like apposition could be useful for certain questions including definitional ones.

- **Linguistic features using the parse tree or semantic roles**. If deep analysis has been performed we can give precedence to certain syntactic structures that a priori could answer a question. Depending on the verb and the type of the question the Subject, the Object or the complements are better suited to be the correct answer. Information used in answer extraction like the similarity between the trees of question and passage can also be integrated at some point.

- **Answer redundancy and similarity**. The repeated presence of the same answer in the list of candidates is a good indicator of their correctness. It could be that different texts support the same answer but also that different extraction strategies consider it correct. In any of the frequency is a good estimator of correctness and it has been exploited in large collections like the Web with very good results like in the AskMSR systemBanko et al. [2002a]. Nevertheless, it is common that the answer is expressed using different variants and the similarity between answers has also been exploited to aggregate their relevance.

- **Dictionaries and external resources** has frequently been used also as part of the validation process in the ranking step. The inclusion of an answer in Wordnet, certain gazetteers or wide resources like Wikipedia could ve used to asses their type and their relevance Ko et al. [2007a]. On the other hand IE and NLP modules often produce systematic errors and stopwords lists has been often used to clean the output. They have been also integrated in ranking frameworks.

- **Frequency information** from the collection or the Web in several forms like correlation between terms could also be added in the fashion of web validation [Magnini et al., 2003] for example.

All these heuristics need to be formalized in a mathematical expression in order to be useful. Moreover, if they are going to be combined they should be represented in a common framework, for example as binary features or as similarity functions in a range value like $[0, 1]$. The simpler method to combine them uses a function for linear combination. Selecting weights manually can yield good results but it is often a laborious task [Pasca, 2003]. It needs to be repeated for each new collection and sometimes if changes are introduced in the rest of the components. A different approach consists on learning the appropriate ranking function from data, provided that a large number of questions and answer pairs are available. The usual procedure consists on running parts of the systems and derive training data by applying the manual judgements. There are three main approaches to build and train a ranking component:

- **Pointwise ranking**. The objective is to learn a function that maps a question-answer pair to an estimated value of relevance. The order is implicit by the value proposed by the function. This is the approach proposed in Ittycheriah [2006] for a principled statistical answer selection in the IBM system. They use the Maximum Entropy Model to integrate different heuristics. In his system he combines some of the features proposed above with features derived from syntactic parses, a translation model and n-grams statistics. Their model calculates the probability that a question answer pair is correct as proposed in $P(C|A, Q) = \sum_e P(E|Q, A) * P(C|E, Q, A)$. Their model for QA is factored in a model for Question Classification and a model for Answer Selection where E is a hidden variable denoting the expected answer type of the answer. Ravichandran et al. [2003] propose a slightly different model $P(\text{Answer is } A_i|Q, A_i)$ which reranks answer candidates as the model considers each answer as an outcome. They show that this approach is more effective than using only two outcomes, which corresponds to a classifier model. Ko et al. [2007a] has extended the original classification approach to include similarity between different answer candidates. Intuitively this method is able to take into account redundancy into the ranking function. The model they propose tries to estimate $P(correct(A_i)|Q, A_1, \ldots, A_n)$ which correspond to the probability that answer A_i is correct given the question and all the answers.

- **Pairwise ranking**. The intended objective is to induce a complete order function between pairs of answers. Pairs of answers are used to train a model that provides a positive number when the first is best suited than the second, or a negative number otherwise. This function can be invoked later to order the complete list of results. Pasca [2003] trains a Linear Perceptron in order to integrate several heuristics in such function.

- **Listwise ranking**. In this approach the objective is to learn a function that maps a list of answers into another ordered list of answers. Evidence from various answers candidates can then be combined in order to produce a definitive ranking. The work by Ko et al. [2007b] extends their previous pointwise model into a Joint Graphical Model that predicts the correctness of the whole list. The model that they propose is a Boltzmann Machine [Hinton and Sejnowski, 1986] which optimizes $P(S_1, S_2, \ldots, S_n)$ where $S_i = 1$ if A_i is correct and zero otherwise. They propose an algorithm to rank the answers using the probabilities of the joint model. The main limitation of the model is its exponential complexity in the number of answers. They show that pre-selecting a small subset of the answers using their pointwise model avoids the problem and it also provides better results.

2.11 Strategy Combination

In Section 2.6 we review the two main approaches to answer questions and how lots of systems have started to combine them to increase their accuracy. In fact, almost all of the individual techniques described before can be combined into a single strategy that provides a list of answers. Experiments in the IBM system [Chu-Carroll et al., 2003; Prager et al., 2006b] and others have shown the effectiveness of system combination based on different techniques.

There are two main alternatives in order two combine strategies; combine the list of answers or proceed to combine an intermediate product like the retrieved passages or documents [Aceves-Pérez et al., 2008]. The pros and cons of each of them are not well studied. However, passage combination is only available for Knowledge Mining systems.

Blind fusion techniques [Aceves-Pérez et al., 2008] like Round Robin, Retrieval State Value (RSV), CombSum or CombMNZ can be used for both as long as the ranked list of results is also scored. Round Robin and RSV make use of the diversity of collections and assume that the scores are comparable. Passage ranking based on RSV is used for example in early versions of the PIQUANT system [Chu-Carroll et al., 2003]. CombSum and CombMNZ normalize scores and are able to use the redundancy of results just as the simpler technique of Voting. They have been informally applied in several multi-strategy systems including [Chu-Carroll et al., 2003].

On the other hand, confidence scores for the answer of QA systems where motivated by their use for fusing list of results using the self-assessment of systems. Nevertheless, as the list does not need to be normalized other combination systems have explored the use of supervised ML like TextMap [Echihabi et al., 2006] in order to combine heterogeneous systems. The main problem is that a set of question answer pairs and their provenance need to be available to adjust the combiner ranking function and it is rather system specific. Finally, Answer Validation has also been deemed as useful to combine the output of several QA systems [Peñas et al., 2007] because it does not need to rely on local scores but only on the supporting text.

2.12 QA and Language Issues

Since TREC QA began, it has focused on Question Answering in the English language. However, it is clear that language is an important variable in any Information Acesss application and forums like CLEF and NTCIR have played their role to establish QA in other languages. There are three problems in the relation between QA and languages which are monolingual QA in other languages, Cross-Lingual QA (CL-QA) and Multilingual QA (ML-QA).

The need for a QA system to be able to work in other languages than English is almost obvious in multilingual societies. It has also been remarked in several roadmaps for research in QA and related technologies [Burger et al., 2001; Vicedo, 2003]. One of the important questions is how language diversity has an effect on the overall QA research and how the techniques need to be adapted. Human languages exhibit a wide range of phenomena with different complexity among languages. For example, morphology is more complex in Spanish or Finnish than in English. Dealing with proper names is different in German but also in Asian languages. At what extent do these issues have an impact in the QA process? Beyond, the most scientific questions there are practical questions too. It is clear that there is a larger range of resources and tools to process English than many other languages. How portable or practical are the techniques when applied to other languages?

2.12.1 A Review of QA in Spanish

The activity of QA in Spanish has been carried out in the context of the QA@CLEF exercise and related tasks from 2004 to 2009 [Forner et al., 2009; Giampiccolo et al., 2007; Magnini et al., 2003, 2006, 2005; Sang et al., 2008; Vallin et al., 2006]. A total of 10 different groups have participated in at least one of the competitions which have included two main collections,

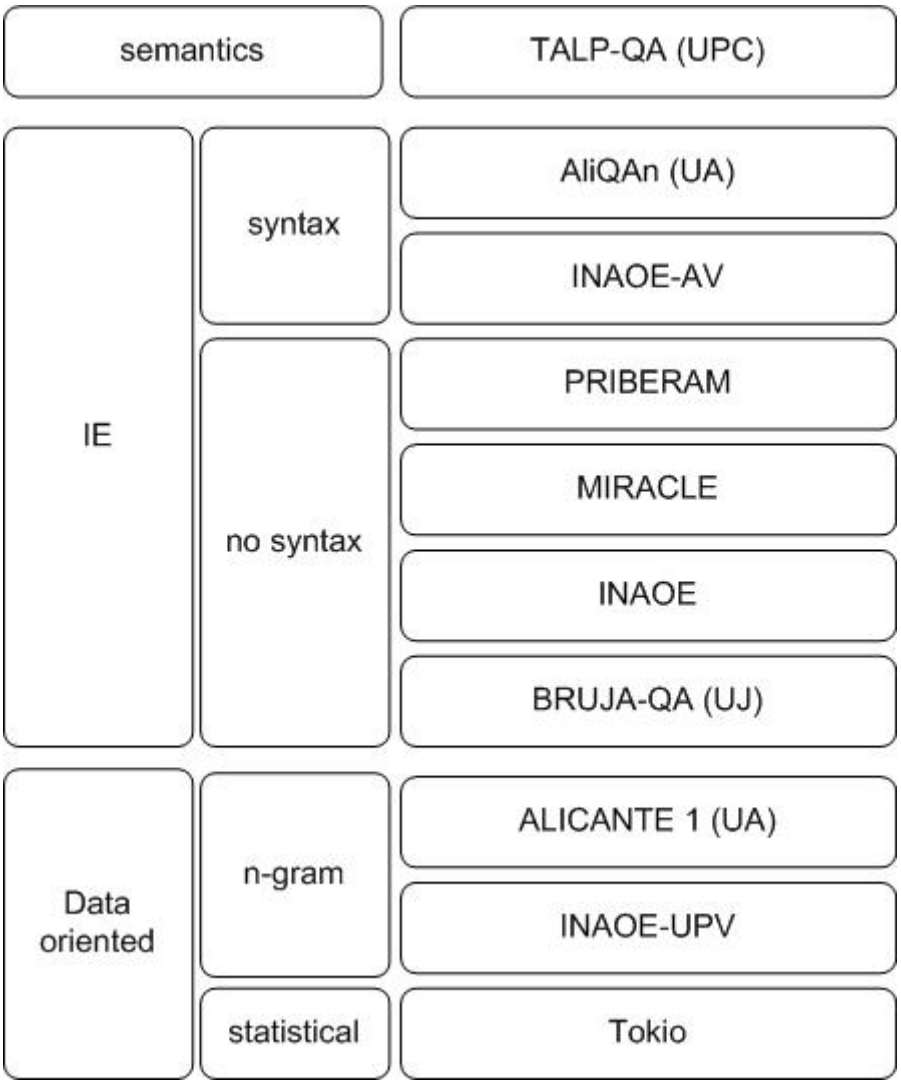

Figure 2.7: An outline of approaches for QA in Spanish

the EFE newswire and Spanish Wikipedia. Besides, the last exercise has used the JRC-Acquis collection, a collection of legal documents from the EU.

The systems have used a variety of techniques which cover a great part of the spectrum presented in previous sections. However, some innovative techniques have been developed too. Most systems have opted by the Knowledge Mining architecture based on the classic pipeline of Question Analysis, Information Retrieval and Answer Selection. That reflects the fact that no large knowledge bases that overlap with the domain are found for Spanish. In contrast, systems like INAOE have compiled large definitional knowledge bases from EFE for example in Pérez-Coutiño et al. [2005].

Figure 2.7 shows the organization of different Spanish QA systems taking into account their Answer Selection strategy. This choice motivates at a large extent their overall architecture. The most common strategy among systems is the use of a pipeline of IE components that process the output of a retrieval system. Every team leverage their customized IE and NLP technology. There are nevertheless a broad range for alternatives, from those systems with a simple taxonomy of types (3) like INAOE [Téllez-Valero et al., 2007] and BRUJA-QA [García-Cumbreras et al., 2006] to the complexity of Priberam system [Cassan et al., 2006] which uses 86 different types. Hybrid approaches are common and integrate rule-based, dictionaries and

Figure 2.8: Outline of CLEF results for Spanish QA systems

machine learning methods.

Data oriented systems have also been applied for Spanish. The system from TU [Whittaker et al., 2007] uses the language model QA system presented in Section 2.10.1. The performance depends on the amount of training data and the training corpus, and results are worse than their equivalent English system. Using n-grams and redundancy from the local collections has been used at least in a joint INAOE-UPV system [Rosso et al., 2005] and also in one participation by UA [Tomás et al., 2006]. According to the results obtained by both system the strategy effective enough.

Syntactic analysis has been part of the AliQAn system [Roger et al., 2006] which also integrates IE components. INAOE used dependency analysis as part of their ranking systems in one of their participation [Pérez-Coutiño et al., 2006]. Small improvements were seen for certain questions though their quantitative results improved little. Finally, the TALP-QA system [Ferrés et al., 2005] is the Spanish QA system that has made use of deeper language processing. In particular they define a multilayer representation of the question which includes semantic constraints. A relaxation algorithm that fallbacks to simpler representations is also provided. Despite their use of deeper semantic representations there is no profound impact on the performance of their QA system. Finally, INAOE in 2008 [Téllez-Valero et al., 2008] experimented with using an Answer Validation step and improved the accuracy of their QA system.

Regarding Question Analysis, the Spanish QA systems show the two major approaches devised in section 2.8 the rule based and the statistical approach. The trend is for system with a simpler ontology to use data-driven methods, like BRUJA and INAOE, while rules are used in those using complex question taxonomies. With respect to the Passage Retrieval stage, most systems have relied in different strategies based on passage selection. A variety of IR subsystems have been used, but we should mention IR-n [Roger et al., 2006] and JIRS [Gómez-Soriano et al., 2005] as two remarkable approaches. IR-n uses cosine-distance for passage selection and has also been sucessfully used in TREC. JIRS use n-grams to rerank passages, which implicitly considers overlap and order heuristics. The JIRS passage retrieval module has been adopted later by two other systems which argue for their usefulness.

In general, the Spanish QA systems show a smaller degree of sophistication on the linguistic techniques than their English counterparts at TREC. This is probably due to the type of NLP tools available and their level of analysis. For example, the best performing system, Priberam, have relied so far on POS-tagging and a language for rules at this level which could perform shallow syntactic operations. On the other hand the use of knowledge and semantic resources is widespread. Both Priberam and MIRACLE use a large taxonomy of entity or answer types which seem to be populated with a large number of instances. Other forms of world knowledge with the use of the Spanish EuroWordnet [Ferrández et al., 2006] and dictionaries derived from Wikipedia [Rosso and Buscaldi, 2006; Toral and Munoz., 2006].

2.12.2 Cross-Lingual and Multilingual QA

Cross-Lingual Information Access, which CL-QA is an instance problem, is important when we would like to access information that is not accesible in our language, something that increasingly common in multilingual societies and the Web. The CL-QA language helps to jump the language barrier by allowing the user to express the question in their own language and enabling the access to information in a different language. The nomenclature of CL-QA systems usually names systems after their target and source languages. The target language is the language used in the collection of documents, while the source language is the one used in questions. For example, English-Spanish (source-target) receives questions in English that are answered from sources in Spanish. The challenge for a CL-QA system is managing the translation process while it keeps

up the same degree of accuracy and usefulness for the user than a monolingual system.

There are two main groups of techniques that have been used in CL-QA system. The first group uses a QA system for the source language as a black box. Translation components that deal with the cross-lingual issues are added outside the main QA system. The second approach breaks the QA system into subcomponents and integrates modules to manage the translation process in the subcomponent flow. From the engineering point of view, the advantages of the first approach are obvious as the approach can be applied to reuse any monolingual QA system, into a QA system while the second requires more adaptation. Nevertheless, there is no evidence that one of the approaches is definitely better than the other. In fact, the gap between cross-lingual and monolingual systems it is still very significant.

The blackbox approach can be applied in two different forms in the case of a English-Spanish system. An option would be to apply automatic translation to the target Spanish document collection and then use an English monolingual system. It has been used in [Bowden et al., 2007]. However, the most common option consists on translating questions into Spanish and using a QA system in the target language, Spanish [Al-Maskari and Sanderson, 2006; Perret, 2005]. Both options use Machine Translation (MT) components and therefore inherit their problems too. Translating the whole document collection is a task that requires intensive computation and is maybe not practical for large collections. On the other hand, translating documents works much better than translating questions, which are much shorter. A disadvantage of the question translation approach is that statistical and rule based MT have ignored questions and their performance is not yet good. Some systems solved this problem using several MT systems in parallel. Then they select the best translation or combine all the alternative translations or combine them [Aceves-Pérez et al., 2007]. Another common problem that plagues the translation of documents and questions is related to Named Entities which are crucial in factual questions. Finally, the quality of translations between different language pairs is also important.

On the other hand, the approaches that break QA systems usually assume that at least some of the components are available in the target and source language. For example, given that a Question Analysis module is available for Spanish it is possible to perform that in the source language. Once the keywords are selected they are translated into and given to a system performing Retrieval and Answer Extraction in English. Techniques from Cross Lingual IR could be employed then. The BRILI system [Ferrández et al., 2006] adopts this approach and it uses the Inter Lingual Index (ILI) of Multi-Wordnet, a system that integrates Wordnets in several languages. Besides, it also employs inter-language information in Wikipedia to translate Named Entities correctly. A final ingredient is Word Sense Disambiguation that helps to prioritize the correct sense in the context. A different approach is proposed in [Bowden et al., 2007] which translates the queries from source (English) to target (Spanish). Once the list of documents results is retrieved, they are translated again into English and the Answer Extraction module in English is applied.

2.12.3 Multilingual QA

Multilingual QA (ML-QA) has a lot in common with cross-lingual and monolingual QA. In addition, systems need to solve problems regarding how to integrate the answers which are generated from several languages. Fusion techniques from Section 2.11 have been used as in this case each unique language can be seen as an individual strategy. An additional problem is the fusion of answers from different languages.

A problem with different nature appears for systems that should work in several languages. As the number of languages grows, so does the number of heterogeneous NLP tools that need to be integrated. It is indeed challenging to deal with a variety of tools which use different formalisms, require different natural language expertise and impose different requirements. However, the expansion of the web reveals that there is a growing need of applications that process several languages effectively.

Few teams in CLEF have attempted to create systems in several languages, even monolingual ones. In fact, experienced TREC teams like ITC-irst and the UvA have produced monolingual QA systems for their native language, Italian and Dutch. The Priberam system started with a monolingual Portuguese system and adapted it to Spanish. All these systems are using several language tools including POS tagging, NERC and parsers and in general require a great deal of adaptation. On the other hand, data oriented and statistical methods have been more commonly applied to build monolingual systems in several languages. The AskEd system has been tailored for English, Chinese, Russian, Swedish and Spanish [Whittaker et al., 2007]. Other systems in CLEF like INAOE, INAOE-UPV and BRUJA have built several monolingual systems for Spanish, French and Italian for example. They use a simplified answer hierarchy of three types of answers NAMES, DATES and NUMBERS. Regular expressions and ML methods are used to recognize candidates with high recall. In both cases, though the results are impressive they are not in par with those systems using more language and knowledge resources.

Chapter 3

MIRACLE Question Answering System

3.1 Introduction

The TREC QA track has played a key role by defining the basic common QA task, providing resources for evaluation and establishing a common research methodology. More importantly, TREC has gathered a research community whose overall contribution has surpassed the expected contribution of their individuals.

CLEF and its QA task (CLEF@QA) were enacted to provide a similar research forum for other European languages beyond English. Would the ideas that have worked for English also work for other languages? It seems that specific language issues could require to adapt some proposals. Moreover, it is clear that English enjoys a privileged place in NLP and IR, it is a subject of research worldwide, while most other languages have only local communities. The range of linguistic resources and tools available would test at what degree some approaches are practical beyond English.

Beyond individual languages, another important objective of CLEF and the QA track have been to stimulate research on cross-lingual QA and multilingual QA. Besides, other research challenges, not strictly limited to language variation, have also been introduced. For instance, questions with temporal constraints were introduced in CLEF 2005 and answering questions in real-time was tested a year later. Different document collections have been used in the main track including news, encyclopedias like Wikipedia and more recently a collection of the EU parliament legislation, JRC-Acquis. Finally, CLEF has also made progress in parallel to TREC, like in the case of question groups focused on a topic.

We started the development of the MIRACLE QA system in the context of CLEF and it has been greatly influenced by the evaluation. Our participation in CLEF started in 2004, and at that time it was one of the first QA systems in Spanish. The architecture of the system was established from 2005 but since then it has been evolving to accommodate new functionality and address the new challenges proposed in CLEF@QA. As the development of the MIRACLE QA system has spanned several years we have chosen to present the main components or modules in the first sections of this chapter. The results obtained during CLEF campaigns have been presented in Section 4 including a brief description of the task, the specific configuration of the systems and the discussion of individual results. The reported results are in the range from 2004 to 2007 as the author of this thesis took the major responsibility during these campaigns. Nevertheless, other members of the MIRACLE team contributed to the system too. An overall analysis and discussion of the results and their comparison in context is also included.

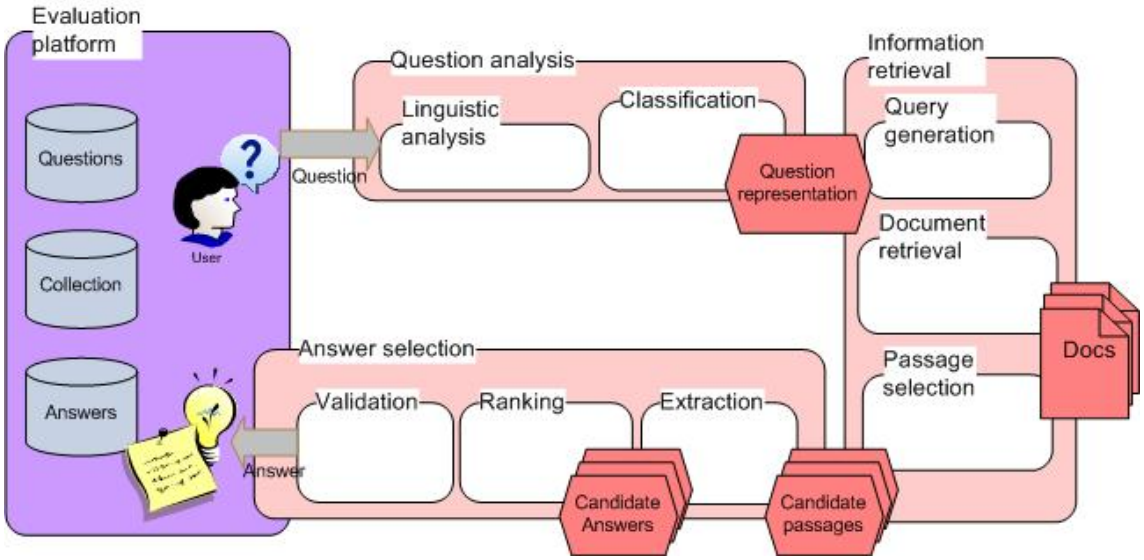

Figure 3.1: SQUASH architecture

3.2 MIRACLE Question Answering Architecture Overview

Our QA system is based on the Pipeline Architecture that we have already introduce in the Knowledge Mining approach to QA. Our efforts, as most of the works in CLEF, have centered on the acquisition and representation of knowledge from text to make it accessible for the final user. Figure 3.1 presents the main components in our core architecture which is composed of three main modules the Question Analysis, a Information Retrieval stage and a Answer Selection phase. The basic procedure starts when the user enters a question. This question is analyzed and a query is sent to the IR module. The IR module retrieves documents and selects relevant sentences. The sentences are further processed by the Answer Selection module that extracts candidates and ranks them. As a pre-processing step the collection of documents should be indexed by the IR system to perform timely access. The different modules that have been developed are described in more detailed in the sections 3.4, 3.5 and 3.6. A key part of the system, which incidentally turns out to lie outside are those artifacts used for evaluation and diagnostic. The collection of evaluation resources includes questions and answers pairs as well as collections that have been distributed for CLEF participants. A repository of those questions as well as evaluation scripts which implement some of the measures introduced in 2.7 has been continuously to measure, tune and improve the system.

The distribution of functionality in modules is one approach to describe the current architecture. A different view is provided by the depth of language understanding. Due to the actual range of language resources available for us, our QA system uses a shallow language analysis approach where Named Entities are a cornerstone. The Answer Taxonomy which models the kind of questions that the system is able to address is the main conceptual structure. The taxonomy is used to stir the behaviour and to choose the strategy of the system. The representation and processing is mainly based on the identification of terms including NE and other complex terminology. The morphological and semantic information is used throughout the three modules to make specialized processing strategies. We can conclude that our QA architecture is mainly based on terms and their types. As a practical consequence, the term oriented QA model is appropriate for current cross-lingual and multilingual needs because of their requirement of relatively shallow linguistic analysis. Nevertheless, the linguistic analysis relies on carefully crafted resources which are expensive to produce for several languages.

Finally, the core architecture has been extended to address several additional challenges

beyond the basic QA task. The pressure to develop Cross-Lingual QA and dialogue based QA has resulted in the integration of additional components that wrap the core functionality. Together with the use of different test collections and the aim to answer questions in real time has resulted in improved modularity and accentuated the need of a good design.

3.3 Linguistic Analysis with STILUS

We have used STILUS[1] language technology for the analysis of Spanish in almost all versions of the system, in particular since 2005. STILUS is a commercial tool developed by Daedalus S.A., a spin-off from Universidad Politécnica de Madrid (UPM) which is one of the founder members of MIRACLE. It was selected for the QA system because it was already a robust tool that provided thorough analysis of the language. The tool was initially aimed at spell-checking and summarization, but since then has incorporated a great deal of new capabilities. The core functionality of STILUS for Spanish includes

- Intelligent Tokenization and Sentence Splitting, including plain text and HTML formats.
- Part of Speech (POS) or morpho-syntactic analysis.
- Lemmatization.
- Morphological generation
- Spell, grammar and style checking.
- Fuzzy search of word variants
- Semantic analysis
- Semantic expansion including synonyms, antonyms and related words.
- Constituent Parsing (in development).
- Summarization

Some of this new features has been partially motivated by our mutual collaboration on IE and QA and were not available from the beginning. The core functionality, that we have used, has been improved along the years too. We have made extensive use of Tokenization, Sentence Splitting, POS tagging and Lemmatization since the beginning of the project. Semantic analysis has also been used and widely improve since 2006. Their analysis and use are described with more detail below. Morphological generation and Semantic expansion have not been tested in the context of QA yet, as well as the use of Fuzzy Word Search but they could be of utility. Parsing is currently in development and has been integrated in some versions of the MIRACLE QA system though not during the years described here.

Example 3.1 Example snippet for STILUS analysis

```
Iósif Stalin fue el máximo líder de la Unión de Repúblicas Socialistas
Soviéticas y del Partido Comunista de la Unión Soviética desde mediados de
los años 1920 hasta su muerte en 1953.
```

In order to provide a glimpse of the kind of information that is available in STILUS regarding the analysis of language we use the sentence in Example 3.1. The output of STILUS is a sequence of sentences where each token or term is analyzed. Several analysis are associated with each of the tokens. A partially manipulated output is shown in Figures 3.2 and 3.6. A human readable interpretation is taken from the web demo and shown in Figure 3.3.

[1] http://stilus.daedalus.es/soluciones-stiluscore.php

TOKEN	POS	LENGTH	ANALISIS (POS|LEMMA|WORD)	ANALISIS (cont)	NUM
Iósif_Stalin	0	12	NPMS--N-N-|Iósif_Stalin|Iósif_Stalin		1
fue	13	3	VI-S3SIL-N7|ir|fue	VI-S3SIA-N8|ser|fue	2
el	17	2	TDMSN9|el|el PragUse=@		1
máximo	20	6	ASMS--DN5|máximo|máximo	A NCMS--N-N5|máximo|m	3
líder	27	5	APMS--NN6|líder|líder	A A A N N NCFS--N-N6|líde	7
de	33	2	Y-N9|de|de PragUse=@	Y NCFS--N-N9|de|de	2
la	36	2	TDFSN9|el|la PragUse=@	PPFS3-UAN9|él|la PragU	2
Unión	39	5	NCFS--N-N5|unión|unión		1
de	45	2	Y-N9|de|de PragUse=@	Y NCFS--N-N9|de|de	3
Repúblicas	48	10	NCFP--N-N5|república|repúblicas		1
Socialistas	59	11	APMP--NN6|socialista|socialistas	A N NCFP--N-N6|socialista	4
Soviéticas	71	10	APFP--NN4|soviético|soviéticas	NCFP--N-N4|soviético|sov	2
y	82	1	CCY--N8|y|y PragUse=@		1
del	84	3	XDMSDN8|del|del PragUse=@		1
Partido_Comunista_de_la_Unión_Soviética	88	39	NPMS--N-N1|Partido_Comunista_de_la_Unión_Soviética|Partido_Comunis		2
desde	128	5	Y-N7|desde|desde PragUse=@	YFN7|desde|desde Prag	1
mediados	134	8	APMP--NN4|mediado|mediados	VPMP--BL-N6|mediar|me	2
de	143	2	Y-N9|de|de PragUse=@	Y NCFS--N-N9|de|de	3
los	146	3	TDMPN9|el|los PragUse=@	PPMP3-UAN9|él|los Prag	2
años	150	4	NCMP--N-N7|año|años	NCMP--N-N7|año|años	2
1920	155	4	NNMS----N-|1920|1920	N N N N MPFPcN|1920|192	6
hasta	160	5	Y-N6|hasta|hasta PragUse=@	YIN6|hasta|hasta PragUs	2
su	166	2	SDMS3SHN8|mío|su PragUse=@	S S S S S S S S S S S S S S S	17
muerte	169	6	NCFS--N-N6|muerte|muerte		1
en	176	2	Y-N8|en|en PragUse=@		1
1953	179	4	NNMS----N-|1953|1953	N N N N MPFPcN|1953|195	9
.	183	1	1O--|.|.		1
*					

Figure 3.2: Outline of the morphological analysis of STILUS

3.3.1 Morphological Analysis

STILUS provides tokenization and a detailed morpho-syntactic analysis for tokens. STILUS' morpho-syntactic tags are more detailed than traditional POS tags and include a large deal of information. The tags are assigned based on carefully crafted dictionary resources. Each token is characterized with several analysis which represent all the alternative interpretations for that token. Each analysis is composed of a detailed tag with several features, the lemma and the correct form of the token. The tag is composed of several features that comprise the main POS tag, its subcategorization, gender, number and the tense of verbs among several others. The detailed representation in Figure 3.3 shows detailed interpretation of the tags. The number of the different features per tag, depends on the token but they are in the range from three to ten. Besides the tag, the tool also provides a lemma for each token from the linguistic resources and if none is found, it can generate variations and corrections. STILUS provides several detailed analysis for each token. For example, in the example there are a token (*su*) with up to 17 differentiated analysis (partially hidden in the Figure 3.2, only the first analysis is shown). Most of the tokens obtain at least two or three analysis. Tag disambiguation has been included as a core feature since 2006 and prunes those analysis that are not consistent in the context. Tag disambiguation is really helpful as it reduces the complexity of further rules. On the other hand there is no attempt to reduce to a single tag, several interpretations in context are still mantained. You can compare the effect of disambiguation in the Figure 3.5.

3.3.1.1 Semantic Analysis

Another capability that is available in STILUS is the semantic analysis of the information. This feature has evolved along the years from the basic recognition of NE using resources to a more complete semantic enrichment of the analysis which includes the following information:

- **Sem_Entity**. Fine grained category of an entity. When STILUS was extended in 2005 to integrate additional semantic information, it was decided to use a similar taxonomy of types to that of ENE [Sekine et al., 2002]. An example of the kind of information can be seen in Figure 3.6. In the example, the `@` symbol is used to denote different levels in the hierarchy.

Resultado

Iósif_Stalin	**Nombre propio masculino singular** (NPMS--N-N-) ▫ *Lema* **Iósif Stalin** ▫ *Entidad semántica:* inst@nofiction@PERSON@FULL NAME
fue	**Verbo léxico intransitivo 3ª persona singular pasado simple indicativo** (VI-S3SIL-N7) ▫ *Lema* **ir** **Verbo auxiliar intransitivo 3ª persona singular pasado simple indicativo** (VI-S3SIA-N8) ▫ *Lema* **ser**
el	**Artículo masculino singular** (TDMSN9)
máximo	**Adjetivo en grado superlativo masculino singular prenominal** (ASMS--DN5) ▫ *Remisión semántica:* **grande** **Adjetivo en grado superlativo masculino singular postnominal** (ASMS--NN5) ▫ *Remisión semántica:* **grande** **Nombre común masculino singular** (NCMS--N-N5)
líder	**Adjetivo masculino singular postnominal** (APMS--NN6) **Adjetivo masculino plural postnominal** (APMP--NN6) **Adjetivo femenino singular postnominal** (APFS--NN6) **Adjetivo femenino plural postnominal** (APFP--NN6) **Nombre común masculino singular** (NCMS--N-N6) ▫ *Entidad semántica:* subc@nofiction@PERSON **Nombre común femenino singular** (NCFS--N-N6) ▫ *Entidad semántica:* subc@nofiction@PERSON
de	**Preposición** (Y-N9) **Preposición infinita** (YIN9) **Nombre común femenino singular** (NCFS--N-N9)
la	**Artículo femenino singular** (TDFSN9) ▫ *Lema* **el** **Personal femenino singular 3ª persona acusativo** (PPFS3-UAN9) ▫ *Lema* **él**
Unión	**Nombre común femenino singular** (NCFS--N-N5) ▫ *Lema* **unión**
de	**Preposición** (Y-N9) **Preposición infinita** (YIN9) **Nombre común femenino singular** (NCFS--N-N9)
Repúblicas	**Nombre común femenino plural** (NCFP--N-N5) ▫ *Lema* **república** ▫ *Capitalización canónica:* repúblicas ▫ *Entidad semántica:* subc@nofiction@LOCATION@GEO POLITICAL ENTITY@COUNTRY
Socialistas	**Adjetivo masculino plural postnominal** (APMP--NN6) ▫ *Lema* **socialista** ▫ *Capitalización canónica:* socialistas **Adjetivo femenino plural postnominal** (APFP--NN6) ▫ *Lema* **socialista** ▫ *Capitalización canónica:* socialistas **Nombre común masculino plural** (NCMP--N-N6) ▫ *Lema* **socialista** ▫ *Capitalización canónica:* socialistas **Nombre común femenino plural** (NCFP--N-N6) ▫ *Lema* **socialista** ▫ *Capitalización canónica:* socialistas
Soviéticas	**Adjetivo femenino plural postnominal** (APFP--NN4) ▫ *Lema* **soviético** ▫ *Capitalización canónica:* soviéticas **Nombre común femenino plural** (NCFP--N-N4) ▫ *Lema* **soviético** ▫ *Capitalización canónica:* soviéticas
y	**Conjunción coordinada copulativa** (CCY--N8)
del	**Contracción masculina singular** (XDMSDN8)

Figure 3.3: Output of STILUS explained

Partido_Comunista_de_la_Unión_Soviética	**Nombre propio masculino singular** (NPMS--N-N1) ▫ *Lema* **Partido Comunista de la Unión Soviética** ▫ *Entidad semántica:* inst@nofiction@ORGANIZATION@POLITICAL PARTY ▫ *Categoría semántica:* SOCIETY@POLITICS ▫ *Información geográfica:* Rusia
desde	**Preposición** (Y-N7) **Preposición finita** (YFN7)
mediados	**Adjetivo masculino plural postnominal** (APMP--NN4) ▫ *Lema* **mediado** **Verbo léxico transitivo e intransitivo plural participio masculino** (VPMP--BL-N6) ▫ *Lema* **mediar**
de	**Preposición** (Y-N9) **Preposición infinita** (YIN9) **Nombre común femenino singular** (NCFS--N-N9)
los	**Artículo masculino plural** (TDMPN9) ▫ *Lema* **el** **Personal masculino plural 3ª persona acusativo** (PPMP3-UAN9) ▫ *Lema* **él**
años	**Nombre común masculino plural** (NCMP--N-N7) ▫ *Lema* **año** ▫ *Entidad semántica:* class@nofiction@TIME TOP@PERIODX@YEAR PERIOD **Nombre común masculino plural** (NCMP--N-N7) ▫ *Lema* **año** ▫ *Entidad semántica:* subc@nofiction@TIME TOP@TIMEX@DATE
1920	**Nombre número masculino singular** (NNMS----N-) **Nombre número femenino singular** (NNFS----N-) **Numeral determinante masculino plural** (MDMPcN) **Numeral pronominal masculino plural** (MPMPcN) **Numeral determinante femenino plural** (MDFPcN) **Numeral pronominal femenino plural** (MPFPcN)
hasta	**Preposición** (Y-N6) **Preposición infinita** (YIN6)
su	**Posesivo determinante masculino singular 3ª persona** (SDMS3UUN8) ▫ *Lema* **mío** **Posesivo determinante femenino singular 3ª persona** (SDFS3UUN8) ▫ *Lema* **mío** **Posesivo determinante masculino singular 2ª persona-cortesía** (SDMS4UUN8) ▫ *Lema* **mío** **Posesivo determinante femenino singular 2ª persona-cortesía** (SDFS4UUN8) ▫ *Lema* **mío**
muerte	**Nombre común femenino singular** (NCFS--N-N6)
en	**Preposición** (Y-N8)
1953	**Nombre número masculino singular** (NNMS----N-) **Nombre número femenino singular** (NNFS----N-) **Numeral determinante masculino plural** (MDMPcN) **Numeral pronominal masculino plural** (MPMPcN) **Numeral determinante femenino plural** (MDFPcN) **Numeral pronominal femenino plural** (MPFPcN)
.	**Signo de puntuación individual** (1O--)
(final de frase)	

Figure 3.4: Output of STILUS explained (2)

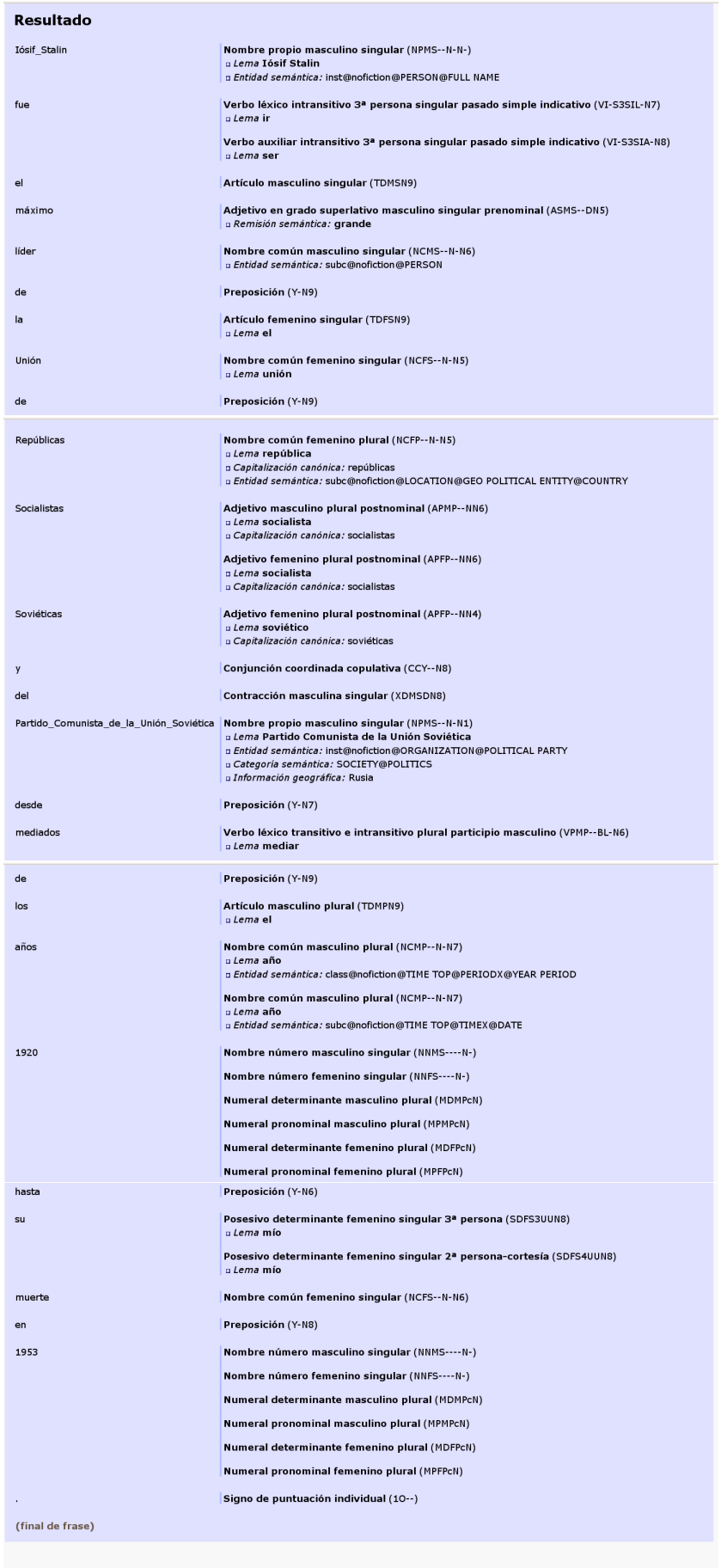

Figure 3.5: STILUS output with POS disambiguation and pruning

TOKEN	SEM_ENTITY	SEM_ENTITY (cont)	SEM_THEME	SEM_GEO	SEM_REMISSION
Iósif_Stalin	@inst@nofiction@PERSON@FULL_NAME@@@				
fue					
el					
máximo					@grande@@
líder	@inst@nofiction@OTHER_ENTITY@TITLE@POSITION_TITLE@@	@subc@nofiction@PERSON@@@@	@SOCIETY@POLITICS		
de					
la					
Unión					
de					
Repúblicas	@subc@nofiction@LOCATION@GEO_POLITICAL_ENTITY@COUNTRY@@				
Socialistas	@subc@nofiction@PERSON@@@@		@SOCIETY@POLITICS		
Soviéticas					
y					
del					
Partido_Comunista_de_la_Unión_Soviética	@inst@nofiction@ORGANIZATION@POLITICAL_PARTY@@@		@SOCIETY@POLITICS	@@@@@@ Rusia@@@	
desde					
mediados					
de					
los					
años	@class@nofiction@TIME_TOP@PERIODX@YEAR_PERIOD@@	@subc@nofiction@TIME_TOP@TIMEX@DATE@@			
1920					
hasta					
su					
muerte					
en					
1953					
.					

Figure 3.6: Outline of the semantic analysis of STILUS

- **Sem_Theme**. A broad thematic classification of a token.
- **Sem_Geo**. Hierarchical geographical association of the concept.
- **Sem_Remission**. A term that is associated with the token or from which their meaning is derived.

This information was initially asociated with Named Entities but then it was extended to other common nouns. The recognition and categorization is based on large dictionaries and gazzetteers. The result is that STILUS obtains very precise recognition like in the cases of *Iósif Stalin* and *Partido Comunista de la Unión Soviética*. In contrast, the recall is lower because those entities that do not form part of the resources are not recognized like in this examples with *Unión de Repúblicas Socialistas Soviéticas*. When such multi word terms like NE are recognized a complex token is formed by concatenating the single terms. In addition to NE, STILUS also recognizes a great deal of terminology which are also complex tokens o multiwords. STILUS does a fair amount of low level work also in recognizing abreviations, acronyms, dates and numbers.

The scope of semantic analysis has been extended to common words like *años* or *socialistas*. It has become evident that further distinctions were needed in order to distinguish between instances (*inst*) which are mainly NE and concepts which are tagged with (*subc*).

3.4 Question Analysis

The Question Analysis module is responsible for transforming the question string into a representation which should be used to select the appropriate strategy for finding answers. The analysis of the questions is used to orchestrate the behaviour of the rest of the modules in the QA system. Question Analysis in the MIRACLE system proceeds in three basic steps, Linguistic Analysis, intermediate Feature Extraction and Question Classification which are depicted in Figure 3.7. The final result of the process is an analyzed question (also known as a question analysis) which is used as input for the rest of the modules. The representation of an analyzed question is based on the Question Model, as the basic structure that describes the user information request. The Question Model has accommodated the changes that have been introduced in the QA task along the years. Besides, it also has been adapted to changes motivated by the

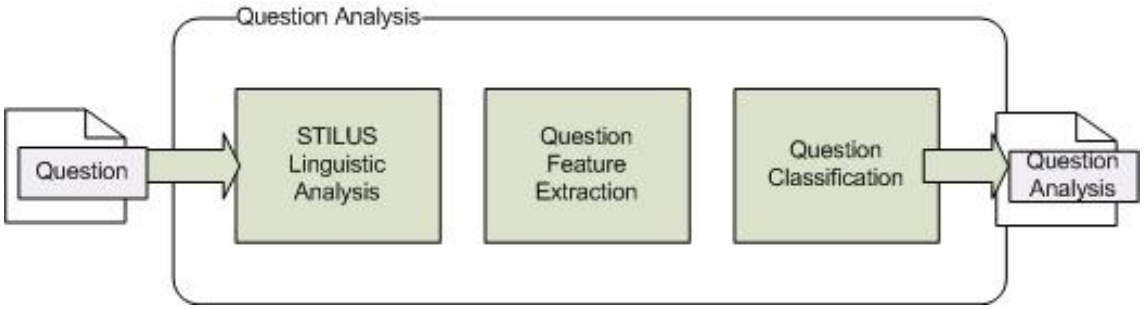

Figure 3.7: Question Analysis Pipeline

different resources and modules that have been included in the QA system. An integral part of the Question Model and in fact, the whole QA system is the Answer Taxonomy. The taxonomy organizes the kind of answers (and therefore questions) that has been considered at design time. As an important side-effect also serves to organize the strategies for answering the main types of questions.

3.4.1 Answer Taxonomy

The taxonomy that governs the behaviour of the MIRACLE QA system is determined by the answer types that can be identified with confidence. Another important constraint is that we have focus on the most common CLEF question types which are Factoid and Definition questions. Most of the Factoid questions are expecting some kind of Named Entity, therefore, we initially considered the Extended Named Entity (ENE) developed by Sekine et al. [2002]. This taxonomy, that is depicted in Figures 3.8 and 3.9, was developed to be useful and comprehensive for QA and IE task in a general domain. Starting with about 150 types, it has been recently refined and extended to 200 types [Sekine and Nobata, 2004] and later on with attributes [Sekine, 2008]. The taxonomy has been developed by manually analyzing newspapers with the aim of being able to cover the needs of open and broad domain applications like QA. On the other hand, most of the types are too specific for the NE tools and NE resources available today. Fortunately, the ENE taxonomy is hierarchically organized which makes possible to focus at different levels of the hierarchy.

The attempt to use such broad categorization using today NE tools will result yet in low accuracy. However, we decided to adopt the ENE taxonomy to represent the types of answers that the system provides because it will define our long-term goal. Besides, STILUS has adopted the taxonomy too and it is including manually tagged information regarding the entities. On the other hand, the taxonomy has been pruned and adapted to our current tools and resources in order to form our own EAT taxonomy. Additional EATs that are not strictly factoids has been added for definitional questions like Descriptions and Acronyms, for acronym expansion.

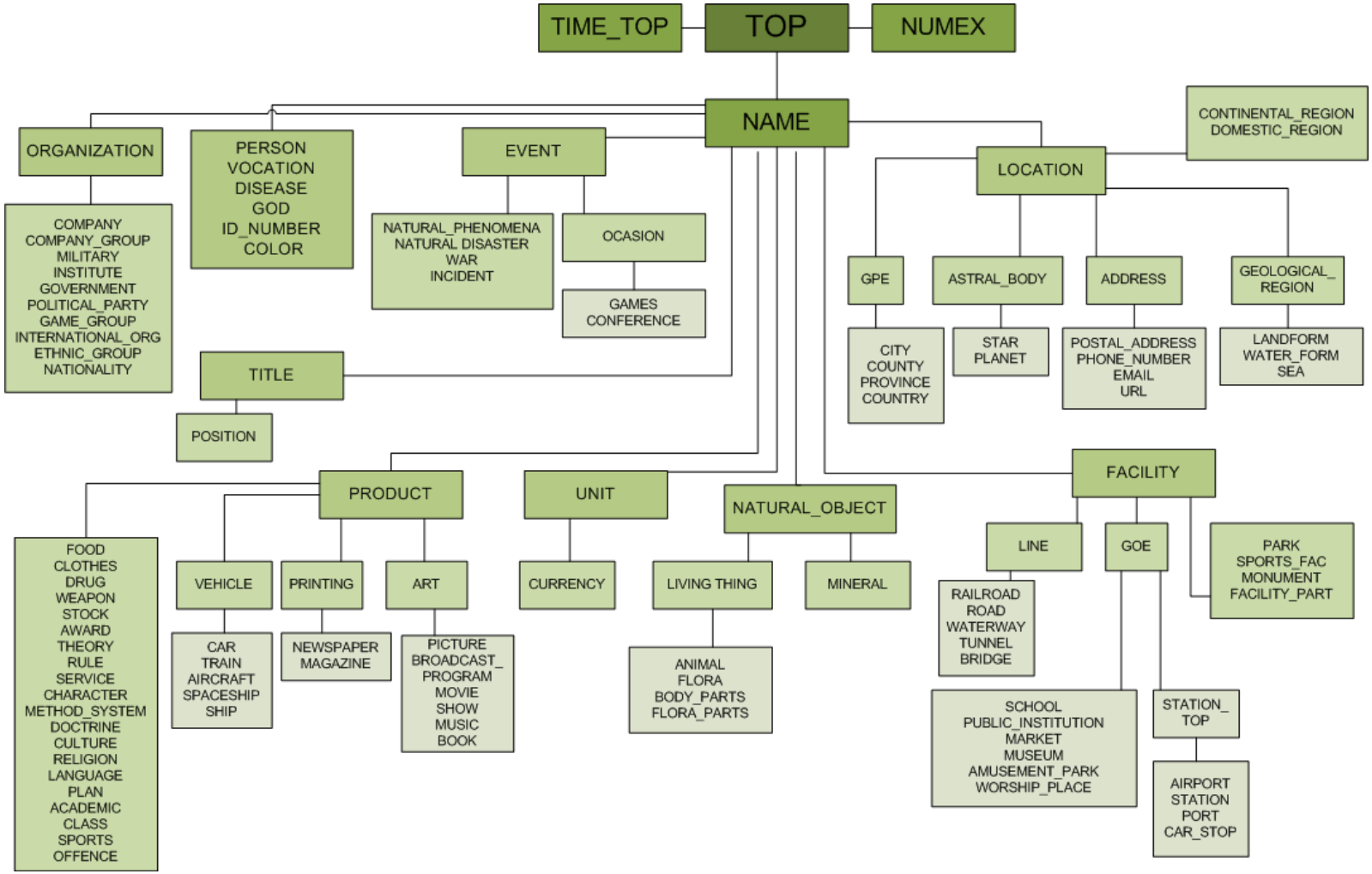

Figure 3.8: Extended Named Entity Hierarchy from [Sekine et al., 2002]

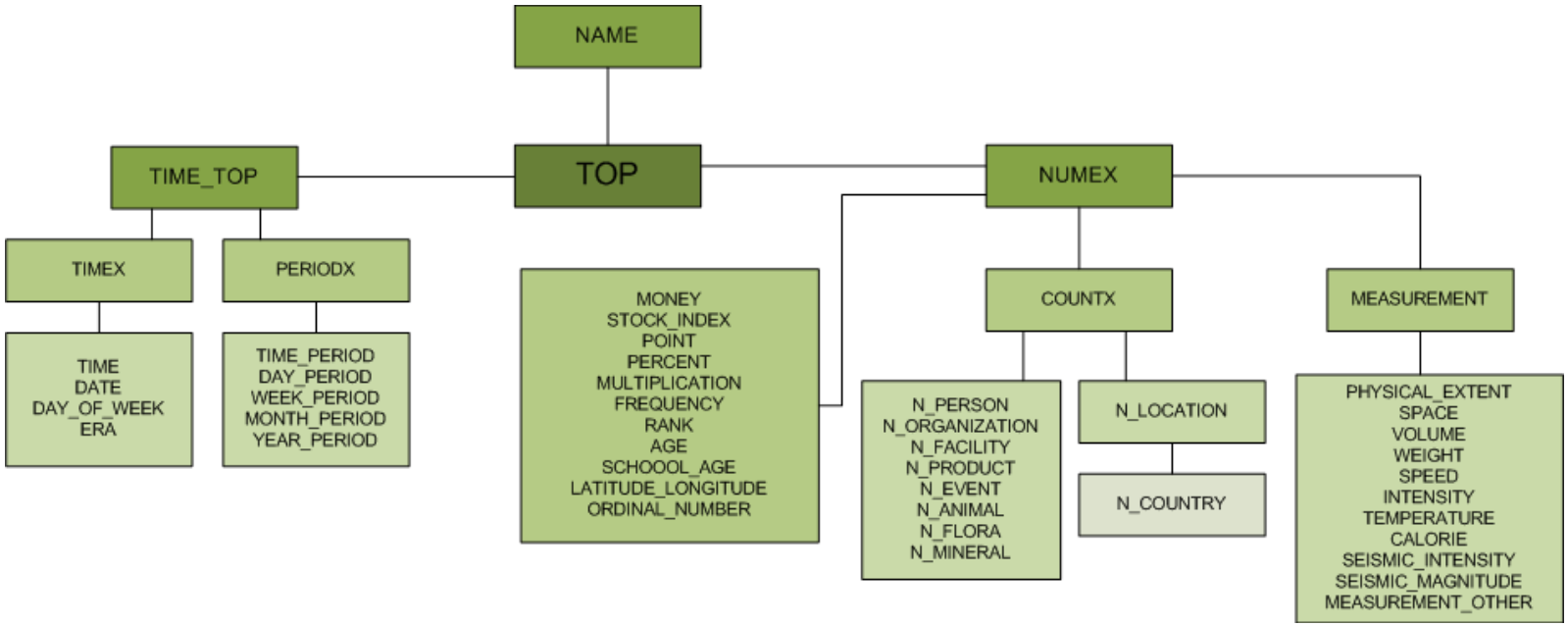

Figure 3.9: Extended Named Entity Hierarchy from [Sekine et al., 2002]

3.4.2 Question Model

Our Question Model is, in essence, similar to the ideas already presented in the state of the art. We decided to implement a simple question model because we lack complicated resources and the interpretation of questions is still error-prone. Besides, we believe that simpler models are more adequate for Cross-Lingual QA as additional errors are introduced in translating and deep linguistic representations are not available in lots of language.

The analysis of a question contains the following information which is our Question Model:

- **Question Type** (QT) is aimed at defining the type of the request. Originally, two different types of questions, FACTUAL and DEFINITION were considered. Additional types for temporally restricted questions and lists were added. Later on, their inclusion was reconsidered as they are not proper types but only add additional features to these types. Temporal restrictions can be added to almost any question type. A list states that several simultaneously correct answers should be considered.

- **Expected Answer Type** (EAT). This is the type that the user expects for a correct answer. In our system the type is assigned from reduced ENE taxonomy. Only a single type is assigned to a question so far. Due to the hierarchical nature of the taxonomy some degree of generalization is allowed, for instance the answer may be part of one the parent types or categories. When trying to define the EAT in more detail, some problems arise for certain questions. There are questions where, if the user does not know the answer neither their specific type, it could be difficult to guess an EAT. For instance, consider a question like *¿Quién creo la Prodnalog?*, it must be even difficult for a human imagine which is the unique type of the response without knowing more about *Prodnalog* itself. It is a country, an organization or a person? In these cases, additional world knowledge would be required, but this is part of the answering process itself. In order to circumvent the problems, it is practical to define the EAT as that type which can be derived from the question without requiring the answer to be known, otherwise default choices should be taken. A final concern is that several types could be valid a priori, that is in fact another way to look at the same problem. So far this has been ignored in our current system which has assigned only one EAT.

- **Question Focus** (QF) is the word that provide the most important clue about the EAT. Most of the interrogatives pronouns help to determine the type particularly in a concrete domain. The question pronoun *Dónde* is associated with places or locations in most domains while in a medical context it signals a body part (*¿Dónde se encuentran las amigdalas?*). On the other hand, certain interrogative determiners like *Qué* y *Cuál* in Spanish could be used to introduce any type. In these cases, additional lexical knowledge is required to detect the EAT. For English language, Wordnet have usually been used as a resource. In our system we have compiled lists of common words which signal certain EAT. The question focus is located by looking for the first common noun after the interrogative pronoun. Additional rules address common ambiguities between nouns and verbs and affirmative requests of information.

- **Question Topic** (QTopic). This is the concept that introduces the theme of the question. The topic is often mentioned or referred in any sentence that bears an answer, though not necesarily in the same form than the question. The topic is usually a concept and it is frequently a NE.

- **Query terms** are those terms that are considered useful to retrieve candidate documents which contain the answer, and therefore are used in queries. Query Terms from the question usually exclude the interrogative pronoun and other stopwords. The Question Focus is also excluded because it introduces additional noise in the returned documents, they are

Question text	$t_1, \ldots, t_n$
Question type	FACTUAL,DEFINITION
Expected answer type	NAME,TIME_TOP,NUMEX ...
Question topic	(i, j)
Question focus	(i, j)
Query terms	$(i_1, j_1) \ldots (i_k, j_k)$
Relevant terms	$(i_1, j_1) \ldots (i_k, j_k)$
Temporal Flag	TRUE,FALSE
Temporal restrictions	(i, j)
List Flag	TRUE,FALSE
Instance/Subtype Flag	INSTANCE/SUBTYPE

Table 3.1: Question analysis model

informative of good documents but not selective. This is specially true for Question Focus terms that are linked to NE types like *año* or *actor*. While they could certainly appear in documents bearing an answer they are too common in documents that are not necessarily related to the question itself.

- **Relevant Terms** are those which are expected to co-occur often in the passage bearing an answer. Query Terms and in particular the Question Topic are very informative about the topicality of a sentence. On the other hand, the Question Focus it is also mentioned close to an answer, though with lower frequency.

- **Temporal Flag**. This flags signals whether the answer is subject to be interpreted with a temporal restriction of any kind.

- **Temporal restrictions** contains the terms in the span of text that express a temporal restriction. Later on, when temporal expressions have been normalized they also contain their associated normalization. Several alternatives to apply temporal restrictions have been considered recently in [Martínez-González et al., 2009].

- **List Flag**. This flag indicates if several simultaneously correct answers should be expected for the question. A couple of rules help to discover whether the answer should be a list. The first heuristic uses the number feature of the terms in the Question Focus. The use of plural often indicates a request for a list like in the case of *¿Quiénes fueron los asesinos de Stalin?* or *¿Qué personas asesinaron a Stalin?*. A second heuristic consider if an explicit number of answers is requested like in *Nombre los tres asesinos de Stalin?*.

- **Instance/Subtype Flag**. Most factual questions require a NE as an answer and therefore a concrete instance of a concept. Nevertheless, a fraction of questions do require to answer with a concept that is an hyperonym of the type and not just an instance. This flag is used with these purpose which was also needed to differentiate between the taxonomic relation in the ENE hierarchy and the member relation. While *presidente* could be seen as a subtype of PERSON and *Stalin* as a particular instance. Subtypes where included in STILUS and could be used in place of our handcrafted lists.

Table 3.1 outlines all the elements in the Question Model. The Question is represented as a sequence of terms or tokens that contain their linguistic analysis which can be indexed by their position. The Question Topic and the Question Focus use these indexes to refer to the start and end of their specific values. Query and Relevant terms are formed by a set of index pairs for each term. Some concrete analysis that apply the model are presented in Example 3.2.

Example 3.2 Examples of question classification

Question	QT	EAT	Focus	Topic	QueryTerms	RelTerms	LF	TF	TempRest	IT
¿Qué es el Mossad?	D	ORG	Mossad	Mossad	Mossad	Mossad	T	F	–	–
¿Quién es Yves Saint Laurent?	D	PERSON	Yves Saint Laurent	Yves Saint Laurent	Yves Saint Laurent	Yves Saint Laurent	T	F	–	–
¿Qué año le fue concedido el premio Nobel a Thomas Mann?	F	TIMEX.YEAR	año	Thomas Mann	concedido, premio, Nobel, Thomas Mann	año, concedido, premio, Nobel, Thomas Mann	F	F	–	I
¿Cuánto aumenta la población mundial cada año?	F	NUM	población	población	aumenta, población, mundial, año	aumenta, población, mundial, año	F	F	–	I
¿Cuál es el nombre de pila del juez Borsellino?	F	PERSON.FIRST	nombre de pila	Borsellino	juez, Borsellino	juez, Borsellino	F	F	–	I
¿Qué cargo detenta Ariel Sharon?	F		cargo	Ariel Sharon	detenta, Ariel Sharon	cargo, detenta, Ariel Sharon	F	F	–	I
¿Qué empresa de automóviles produce el "Escarabajo"?	F	ORG	empresa de automóviles	Escarabajo	automóviles, produce, Escarabajo	empresa, automóviles, produce, Escarabajo	F	F	–	I
¿Dónde tiene lugar el Motorshow?	F	LOCATION	Dónde	Motorshow	Motorshow	tiene, lugar, Motorshow	F	F	–	I
¿Cómo murió Jimi Hendrix?	F		Cómo	Jimi Hendrix	murió, Jimi Hendrix	murió, Jimi Hendrix	F	F	–	I
Nombre un edificio envuelto por Christo.	F	LOCATION.FACILITY	edificio	Christo	envuelto, Christo	edificio, envuelto, Christo	F	F	–	I
¿Quién recibió el Premio Nobel de la Paz en 1989?	F	PERSON	Quién	Premio Nobel de la Paz	recibió, Premio Nobel de la Paz	recibió, Premio Nobel de la Paz	F	T	en 1989	I
¿Qué político liberal fue ministro de Sanidad italiano entre 1989 y 1993?	F	PERSON	político	ministro de Sanidad italiano	liberal, ministro, Sanidad, italiano	político, liberal, ministro, Sanidad, italiano	F	T	entre 1989 y 1993	I
¿Qué países forman parte del Tratado de Libre Comercio de América del Norte?	F	LOCATION	paises	Tratado de Libre Comercio de América del Norte	forman, parte, Tratado de Libre Comercio de América del Norte	países, forman, parte, Tratado de Libre Comercio de América del Norte	T	F	–	I

3.4.3 Linguistic Analysis and Feature Extraction

When the question is introduced by the user, the first operation is the linguistic analysis using STILUS. The second step considers the identification of certain intermediate features that serve to classify the question and, in general, to fill the Question Model that has been presented which is used throughout the QA process. The following intermediate features are used:

- IsInterrogative. A boolean attribute that signals if the information request is in interrogative form or it is an imperative sentence like *Nombre tres luchadores de sumo.*
- InterrogativeLemma. A feature to store which is the lemmatised form of the interrogative particle of a question. The system relies on the morphological analysis of STILUS to locate it and avoid prepositions like in question of the type *De [qué] está recubierto el continente antártico* o *¿A [cuánto] asciende el premio para la ganadora de Wimbledon?.*
- IsNounBeforeVerb. Also a boolean flag that detects if a noun appears before a verb, which is a good indicator of questions where a noun is the Question Focus, and therefore the key to find the EAT. Consider questions like *¿Qué [cargo] detenta Ariel Sharon ?* or *¿Cuántos [países] forman la ONU actualmente?*
- FirstNounLemma. For most questions where the interrogative particle is not enough to detect the type, the first noun is usually the Question Focus as in the example *¿Qué [cargo] detenta Ariel Sharon?.* On the other hand exceptions to this general rule exists like *¿Cuántos [países] forman la ONU actualmente?.* While in this case the EAT is a NUMEX.COUNTX and the interrogative provides the clue, the noun *países* would help to detect a further specialization which is the count.
- MainContentVerbLemma. First verb which is not copulative or auxiliar and could establish a relation. Again, rules implemented in STILUS information help to locate it correctly.
- FirstEntityType. The NE type or types of the first entity following the ENE taxonomy. This entity is often the Question Topic.

Features of the Question Model are filled in directly or using additional rules from the previous information. The most complex process is Question Classification and due to their significance some of the rules developed are explained in more detail.

3.4.4 Question classification

Question Classification was initially aimed at detecting the EAT of the question. As the task of QA has been extended with new question types and further complications the module has also been extended. It produces additional question attributes of the Question Model, like the Question type, Temporal, List and Instance/Subtype Flags.

Question Classification is performed with a set of linguistic rules defined on the extracted features defined above. The module was initially developed using a generalization study of CLEF 2004 Spanish data, but has been revised every year to include new rules and features that improve coverage. The rules were initially expressed as a decision tree and hard-coded in Java. Since 2008, the rule language described in [Martínez-González et al., 2009] has been used to implement similar functionality in a more reusable language. Questions are first classified regarding their QT and later different rules are applied to the EAT for Factual and Definition questions. The List Flag and Instance/Subtype Flag are filled in taking into account morphological and semantic information of the analysis of the Question Focus. Finally, the Temporal Flag is enabled if any temporal expression has been detected in the question which is carried by combining STILUS basic date analysis with specialized rules.

Definitional questions are detected based on the following basic patterns which can be translated into tests which use the previous features and the linguistic analysis.

- ¿Qué SER NOUN/NE?
- ¿Que SIGNIFICAR *?
- DEFINIR/DESCRIBIR *

The rest of the questions are considered factual and the EAT is also assigned using a similar list of rules. A set of rules links each of the pronouns to their default type in the general domain. Another resource, a list of words that are common Question Focus are used to assign the EAT. In 2008 STILUS integrated subtypes of the EAT taxonomy as part of the analysis tools and made the coverage of this lists wider. Some examples of the original lists are presented in Table 3.2. Finally, a third set of rules are needed to avoid some common questions formulations which are complex, periphrastic or posed in an indirect way like in the question *¿Cómo se llama el Síndrome de Down?*. Other problems include dealing with ambiguity and other analysis errors in common words which potentially affect the process.

3.4.5 Topic Identification and Co-Reference in Question Series

Since 2007 a significant number of questions were introduced in series or groups. The purpose of the task was to simulate a more complex information seeking dialogue between the user and the system. In a question group the topic is presented in the first question and the following questions are related to the same topic. Guidelines restricted the topic to any kind of entity or event. The topic could be mentioned in the first question of the group or it could be the answer to the first question. In contrast, an analysis of the groups revealed that sometimes the topic can shift within a group of questions.

We started by including a topic identification system which was later complemented with a co-reference system adapted to question and answer dialogues. The main idea behind topic identification is that not all kinds of information, in particular different NE types, have the same probability a priori to be a topic for a group of questions. Some of the NE types, like persons, are subject to motivate a QA dialogue. Numbers or measures on the other hand are less common. Nevertheless, this preference is often modified by including a referential expression in the following question that helps to locate the common topic. In the common case, the referential expression in a QA dialogue would be expressed as ellipsis because the topic is shared between the two parts, user and system. On the other hand a topic shift is naturally introduced by the use of a referring expression that recalls a different entity or event. This is a very simplified view of the theory of centering (Grosz et al, 1995). The example from group 2011 in CLEF 2007 topic set in Table 3.3 illustrates both cases.

EAT	Focus Term
NAME.LOCATION	lugar, montaña, río, zona, región, pueblo, continente, museo, cárcel, edificio, estadio
NAME.LOCATION.GPE.CITY	ciudad, capital
NAME.LOCATION.GPE.COUNTRY	país, estado, nación
NAME.ORGANIZATION	organización, compañía, grupo, equipo, asociación, empresa, firma, ejército, institución, partido, comisión, iglesia, banda, banco, multinacional, organismo
NAME.PERSON	persona, hijo, marido, mujer, seudónimo, pseudónimo,amante, padre, madre, presidente, diputado, ministro, secretario, dictador, directivo, responsable, capitán, general, actor, director, escritor, cantante, guitarrista, bajista, batería, entrenador, jugador, juez, magistrado, abogado, político, ex, líder, dirigente, personalidad
NAME.PERSON.LAST_NAME	apellido
NAME.VOCATION	profesión, cargo
NAME.PRODUCT	título, premio, torneo, película, disco, periódico
NAME.UNIT.CURRENCY	moneda
NAME.NAME_OTHER.NATIONALITY	nacionalidad
NUMEX.MEASURE	altura, profundidad, tamaño, número, superficie, anchura, extensión, media, esperanza, edad, distancia, porcentaje, kilómetro
NUMEX.MONEY	precio, gasto, euro, dólar, peseta, cambio
TIME_TOP.TIMEX	tiempo, hora, minuto, segundo, semana, lustro, siglo
TIME_TOP.TIMEX.DATE	fecha, día, aniversario, cumpleaños
TIME_TOP.TIMEX.MONTH	mes
TIME_TOP.TIMEX.YEAR	año
TIME_TOP.PERIODX	edad, tiempo
ACRONYM	sigla, acrónimo

Table 3.2: Table with word lists for EAT identification

Example 3.3 Tracking a topic in a question series

QUESTION	ANSWER	REFERENT LIST	REFERRING EXPRESSIONS	TOPIC
¿Quién fue Hermann Emil Fischer?	químico alemán			Hermann Emil Fischer
¿Qué premio recibió en 1902?	Premio Nobel de Quimica	R1 = (Hermann Emil Fischer, químico alemán)	E1(ellipsis)	E1=R1= (Hermann Emil Fischer, químico alemán)
¿Quién recibió el Premio Nobel de Literatura ese año?	Theodor Mommsen	R1 = (Herman Emil Fischer, químico alemán) R2 = Premio Nobel de Química R3 =1902	E2= ese año E3= Premio Nobel de Literatura	E2 = R3 = 1902

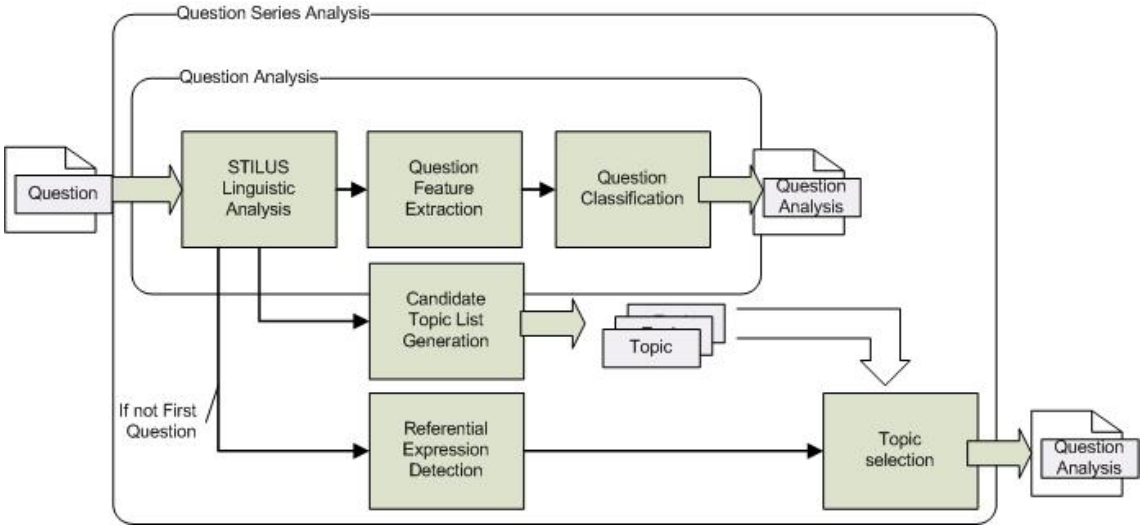

Figure 3.10: Topic identification and tracking for question series

A subsystem based on these ideas was initially implemented in 2007 and enhanced in 2008. It was implemented as a wrapper around the basic Question Analysis functionality which helps to track the topic as new related questions from the user are received. It works in two main steps, as the topic is expressed in the first question and it should be tracked for the second question. In addition to the basic Question Analysis, the first step generates a list of candidate topics from the question and its answer. When the second question is introduced, the user would refer to some of the candidate topics in the first question. To decide which of them is the topic we should locate referential expressions and find the best candidate. We explain the modules in more detail in the next sections.

3.4.5.1 A Priori Candidate Topic Ordering

The first simplification to the problem is that we only consider as candidate topics nominal groups and preference is given to NE. The topic for a QA dialogue could be the initial Question Topic, the Answer which should be linked to the Question Focus or maybe other NE mentioned in the question. Our system applies a second simplification which produces an ordered list of candidates considering the QA functions and their NE type:

- Entities of subtypes `NUMEX` (numbers and quantities) are ignored as topics for questions series. We believe it is improbable for a number, if not representing any other type of entity, to be a natural topic. We use the subject, the topic of this question, as the topic of the question series.

- Entities of subtypes `TIMEX` (dates, years, etc...) are considered after the rest of the candidates topics. It is not usual to dialogue about certain periods if they are not explicitly named. If this is the case, they will usually be mentioned explicitly like in the case of *ese año* in Example 3.3 which is used to retrieve the year as a topic.

- If the question asks for a definition like *Quién es George Bush?*, the topic (Named Entity) and their answer *(presidente de los Estados Unidos)* will be added together to the candidate lists because they are esentially the same topic. A similar case occurs with questions with reverse questions like *Quién es el presidente de los Estados Unidos?*.

- The question follows the pattern `Qué NP * ?` " like *¿Qué organización se fundó en 1995?*. In these cases the noun group that follows the interrogative article should be the focus of

the question. Both the answer and the focus would be added as the same topic at the top of the list. We should remark that this case is different from a question beginning with a preposition like *En qué lugar... ?*.

- For the rest of the classes the answer is the highest ranked topic and the rest of the candidates are introduced in the same order than they appear in the question.

Referential Expression Detection. Most referring relations in Spanish questions are realized by ellipsis and in those cases the a priori selection works well. In other words, ellipsis is used when there is no doubt that the referenced entity is common ground for the two parts in a dialogue. In contrast when an explicit referring expression is introduced, it usually signals a topic shift. In that case, it is suggested that the order of the candidates probably changed. The first step is to detect the most common referential expressions in questions and answers dialogues: definite noun phrases (*esa persona*, pronouns (*él*) and Named Entities (*Fischer*) are identified. Rare cases like epithets or verb nominalizations have been so far ignored as well as other complicated referential phenomena that require further linguistic analysis.

Topic Selection. Once the referring expressions have been selected the next step consists on selecting their co-referent from the list of candidate topics. For each candidate pair of referent and referring expressions we calculate if they satisfies some agreement constraints. We have implemented five different constraints based on the linguistic information that is available after analysis: number, genre, lemma, semantic type and acronym expansion. A candidate pair that satisfy more constraints than the a priori best rated candidate for co-reference could be promoted if the score is higher. So far the weights have been adjusted manually using previous examples and counterexamples.

The previous schema is based on the structure of the information seeking dialogue and how we introduce new topics or modify the topic in a conversation. Once the question group topic is identified, it is included as the Question Topic and as an additional relevant term to the analysis of the second question. The topic would help to locate related documents and filter relevant sentences. Obviously, the whole process is vulnerable to the accuracy of the whole QA process. The most common problem appears when the answer is incorrect but it is identified as the topic of the group. Fortunately, a more complex dialogue would allow for correction and clarification. On the other hand, additional co-reference phenomena in QA dialogues like the use of partitives, meronymy or collective co-reference are subject to future work [Jurafsky and Martin, 2008; Vicedo and Ferrández, 2006].

3.5 Information Retrieval

The goal of the Information Retrieval module consists on selecting a fraction of the documents in the collection where the answer has to be found. Although our Answer Selection is able to process a large number of documents and it is relatively lightweight, the final result depends on the quality of the input documents and the amount of noise that arrives. The desired output of the IR module should include:

- broad coverage along a range of questions, or, in other words, relevant documents are returned for most questions.

- high redundancy, if an answer is stated in several documents is better to find all of them.

- low signal to noise ratio, as irrelevant sentences and documents would produce spurious answers it is better to process only related sentences.

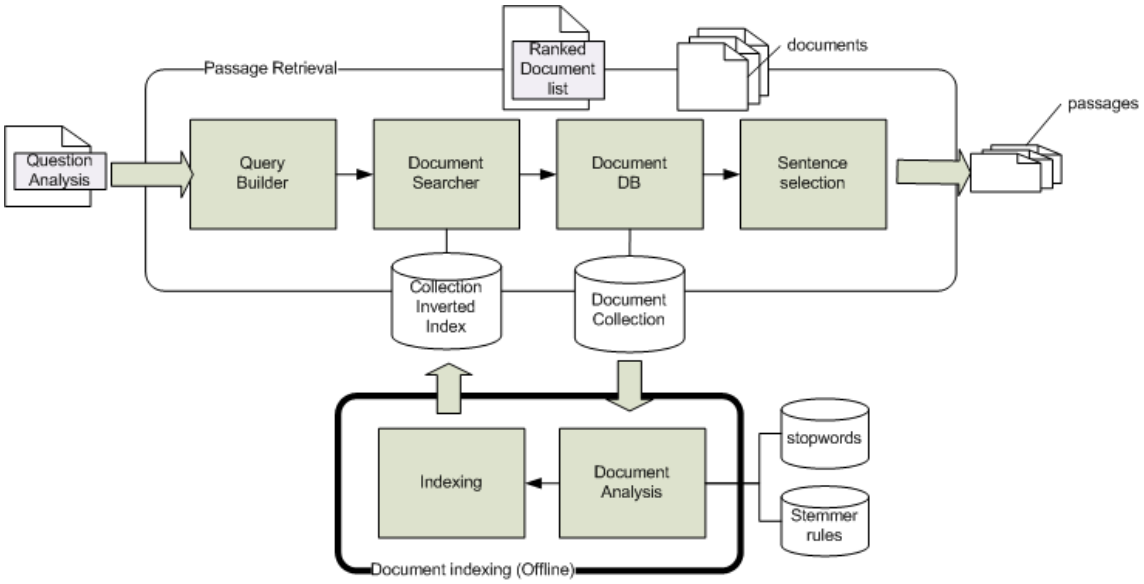

Figure 3.11: Passage Retrieval Pipeline

To accomplish these goals we opted from the beginning for a passage filtering approach, which aims at retrieving documents with high recall. Later on, promising sentences are selected aiming at improving the precision. Our final architecture is depicted in Figure 3.11 and has evolved from the initial requirements. The input to the IR module is the question analysis which guide the rest of the process. Indexing has been performed at the document level in most experiments though is now flexible enough to accommodate smaller units. The passage filtering step produces single sentences that are the most appropriate units for our extraction methods that do not use co-reference at all. While the document retrieval stage is expected to produce high recall, the objective of the filtering approach is to focus on the selection of very promising sentences. As the platform evolved, it become evident that different strategies for producing queries from a question had very different results. To support the use of different Query Generation strategies a new module was introduced. The need became more compelling as different retrieval engines were introduced and their query languages differ. The introduction of different engines also required the adaptation of the Collection Indexing mechanisms. Finally, in our experience, it seems that strategies to find appropriate documents should also be adapted to different collections.

The different components are described with more detail below. Two different document retrieval engines have been used. Xapian was used from 2004 to 2006 while Lucene was introduced in 2007 and has become the main engine now. Besides, Web search engines like Google has been used as a service in some demos and it has been also integrated in the framework. While the appropriateness of different engines and different retrieval models is still discussed, the change from Lucene to Xapian was in fact motivated by system designs requirements. Our platform is developed in Java while Xapian is developed in C++ and it is used through JNI bindings. We found time consuming to maintain two development environments and problems arouse when migrating between OS environments (Windows and different flavors of Linux). On the other hand, we found many advantages with the migration to Lucene. In general, it is better documented and has a more stable and flexible API. It is portable and easy to deploy, not only the source, but the indexes can be easily transported between machines. Besides, development tools are better and the community around it has created useful and reusable resources. Along the years, several researchers have adapted other retrieval models to Lucene and it would be easier to integrate them now.

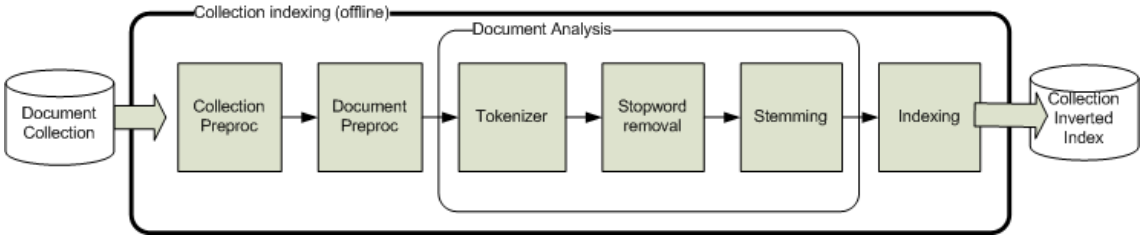

Figure 3.12: Collection Indexing Pipeline

3.5.1 Document Analysis and Collection Indexing

Typical IR engines represent a document as a some kind of bag of terms regardless of the retrieval model that they use. This is a simplified but effective document representation which practically ignores most relations between terms, except for their co-occurrence in the same document. So far, it is widely accepted that this representation has consistently outperformed other alternatives in the automatic representation of document content, at least for topical requests.

The bag of terms is inherently a boolean representation, the terms could be present or not. In fact, the most common logical models of documents associate weights to terms. Common weights make use of the term frequency (TF) and the inverse document frequency (IDF). TF is the frequency of a term in a document. Document frequency is the number of documents in which a term appears, and their normalized inverse is IDF. The weighted bag of terms representation and the equivalent weighted vector representation, where absent documents obtain a zero weight, are common models for the logical representation of documents.

$$d_j =< w_1, \ldots, w_{N_{d_j}} >= [w_1, \ldots, w_i, \ldots w_N] \tag{3.1}$$

On the other hand, the physical view is based on the inverted indexes that store list of postings. Postings relate the term with the ID of the document in which they appear. Additional information regarding the document frequency and the collection frequency of a term should also be stored in order to support document relevance calculations. Finally, the logical view of a document can be extended to a weighted bag of terms where engines and models differ on the procedure to estimate their weights as well as document relevance. Some additions to the basic bag of term representation, consider distance between terms. Most IR engines implement useful query operators like phrase search that require additional physical information too, like the document offset of the term in order to be efficiently executed.

Document analysis techniques determine the final logical view of a document as they define the terms that are used to represent the document. The system has used three main operations: tokenization, stopword removal and stemming. The pipeline of components is depicted in Figure 3.12.

3.5.1.1 Tokenizer

Its goal is to split the documents into terms or tokens. The most basic tokenizer splits by using non-alphanumeric characters as term separators, like spaces and punctuations marks. Most elaborated rules are required in some cases. Since the adoption of Lucene we have used their framework for tokenizers and document analysis. Before that, both a simple tokenizer and STILUS were used for this step.

3.5.1.2 Stopword list

The usual stopword list for Spanish contains the most common words in the language using inflected forms as it is usually applied to the raw text. Adaptations for the collection and for the task, like including common question pronouns have demonstrated better intermediate results while the system was developed. A comprehensive list of stopwords can be accessed at the University of Neuchatel place [2].

3.5.1.3 Stemming

We have used the Snowball Spanish stemmer which is widely accepted for the Spanish language. Snowball is a metalanguage that allows to define rules for stemming for different languages. The meta-algorithm for stemming is explained in detail in [Porter, 2001] and in the maintained site [3]. It applies a series of transformations to reduce words into a single form which remove the main morphological information to produce a representative form. In Spanish, most of the word types are modified with morphological information but verbs, nouns and adjectives are the most important for QA. The goal of stemming is to reduce the token or word to a normalize form which groups tokens with the same meaning despite their different form. In contrast with lemmatisation, the normalize form does not have to be linguistically motivated. Snowball for Spanish removes clitic pronouns, common derivative suffixes, morphological suffixes and performs also acute character normalization. One of the common problems with stemming algorithms in Spanish are irregular forms which usually result in completely different wordforms, for example with the verb *ir*. Snowball takes into account some of the irregular verbs and remove suffixes with special rules. However, as the modifications affect the root of the terms no stemmer is appropiate. Another problem with stemming that is common to different languages is how to deal with Named Entities for IR and whether their stemming affects the overall performance of IR and QA.

Before the documents could be analyzed, some additional pre-processing is often required. The pre-processing is specific to each collection and their distribution format. For instance, if the collection is distributed in a file, single directory or multiples directory and how it bundles individual documents. It also depends on the document format which includes additional metadata beyond the indexable text. The metadata could be removed or indexed in a different field. For example, the EFE collection included one SGML file whith all the news from one day. To index the documents, it was required to parse each file into several documents. Besides, each EFE document like the one presented in Example 3.4 has to be preprocessed to index only certain fiels, initially only the TEXT and TITLE were indexed. On the other hand, Wikipedia has very different structure. For example, in the HTML dump the files are organized in a three level directory hierarchy that is alphabetically organized. Each file contains one Wikipedia entry or document which is marked with HTML and should be cleaned before indexing. In fact, every collection presents their own characteristics regarding the organization of documents in files and directories, markup format and character encoding. In our system, we have divided the task in two submodules for collection and document preprocessing that are adapted for each collection.

3.5.2 Document Retrieval with Xapian

The first engine that was used in our system was Xapian, a C++ open source engine based on the Probabilistic Retrieval Model [Robertson and Spärck Jones, 1976] that includes the Okapi BM25 scoring function [Robertson et al., 1994]. The BM25 model is often cited because it has obtained very good results in ad-hoc document retrieval task in TREC competitions. Xapian was one of the most popular open source packages at that time but was limited to provide barely

[2] http://members.unine.ch/jacques.savoy/clef/index.html
[3] http://snowball.tartarus.org/

Example 3.4 A document example from the the EFE collection

```
<DOC>
<DOCNO>EFE19940101-00001</DOCNO>
<DOCID>EFE19940101-00001</DOCID>
<DATE>19940101</DATE>
<TIME>00.28</TIME>
<SCATE>POX</SCATE>
<FICHEROS>94F.JPG</FICHEROS>
<DESTINO>ICX EXG</DESTINO>
<CATEGORY>POLITICA</CATEGORY>
<CLAVE>DP2403</CLAVE>
<NUM>736</NUM>
<PRIORIDAD>U</PRIORIDAD>
<TITLE>   GUINEA-OBIANG
           PRESIDENTE SUGIERE RECHAZARA AYUDA EXTERIOR CONDICIONADA
                                                                                ...
</TITLE>
<TEXT>    Malabo, 31 dic (EFE).- El presidente de Guinea Ecuatorial, Teodoro
 Obiang Nguema, sugirió hoy, viernes, que su Gobierno podría rechazar
 la ayuda internacional que recibe si ésta se condiciona a que en el
 país haya "convulsiones políticas".
    En su discurso de fin de año, Obiang afirmó que "la democracia
 pluralista no es sinónimo de desorden ni de convulsiones", y el

[...]

   Por último, basó el desarrollo global de Guinea Ecuatorial en la
 combinación de los conceptos de libertad, seguridad ciudadana y
 desarrollo económico y social. EFE
    DN/FMR
    01/01/00-28/94
</TEXT>
</DOC>
```

the indexing functionality. Fortunately, Xapian has continued evolving and it is far more easier to adapt to new uses with less effort. We briefly present some of the characteristics of the system along with how it has been used and adapted in our QA system.

3.5.2.1 Indexing

Regarding indexing, Xapian uses the simple logical view that a document is a bag of terms. Nevertheless additional information could be added to the documents but only as document metadata. As our use of Xapian dates back to 2004, versions 0.8 and 0.9 were used. These versions used the Quartz format engine that used a sort of B-tree for the vocabulary. Recent versions have included the ability to define fields for documents that allow to handle structure as well as other modifications as a result to the new database format Flint.

3.5.2.2 Ranking model

Document ranking is based on the Probabilistic Retrieval Model which in a few words is based on the estimation of the probability that a document is relevant for a given query, $P(Rel|d_j, q)$.

Rel is just a binary variable that indicates whether it is relevant or not. Documents are ranked using the ratio between the probability of being relevant and their negation. Several assumptions like independence between terms and rank preserving transformations are used to arrive for a final single operational ranking formula. The Okapi BM25 formula used in Xapian is adapted to include additional factors that have been motivated by IR practice and is outlined in Equation 3.2. The Retrieval Score Value (RSV) is the sum of the values for each query term as it is derived from the term independence assumption.

$$RSV(d_j, q) = \sum_{q_i \in q} \frac{(k_3 + 1) \cdot q_i}{k_3 + q_i} \cdot \frac{(k_1 + 1) t f_i}{K + t f_i} \cdot W_i \tag{3.2}$$

Each query term is the product of three different factors. The first factor modules the importance of query terms and is controlled by parameter k_3. The second factor is related with the frequency of the query term in the document. It defines a saturation function which modulates the relative weight of terms that have high frequency. The intuition is that it is more informative if a term appears or not than if the same term occurs 10 or 100 times in the same document. These factor has to be adjusted for each collection and their influence can be tuned with k_1. The function K is expanded in 3.3 and it can also be tuned to control the influence of document length in results. Consider that a long document would probably have higher term counts or higher term frequencies. The parameter b which is set in the range $[0 - 1]$ controls the final influence of the term weight while L is defined in Equation 3.4 as the ratio between the lenth of the document and the average length. The third factor is the one provided by the theoretical model and represents the relevance weight assigned to each term according to the theoretical model.

$$K = k_1(b \cdot L + (1 - b)) \tag{3.3}$$

$$L = dl/avdl \tag{3.4}$$

The theoretical Probabilistic Retrieval Model assumes term independence and also simplifies the model for the binary ocurrence of terms, so it only accounts if the term exists or not in the document. This is known as the Binary Independence Model (BIM) and the final equation is shown in Equation 3.5.

$$W_i^{BIM} = log\left(\frac{P(t_i|Rel)(1 - P(t_i|\overline{Rel}))}{P(t_i|\overline{Rel})(1 - P(t_i|Rel))}\right) \tag{3.5}$$

Probability estimates were proposed by Robertson and Spärck Jones [1976] using the contingency table in 3.3 where bolded values are supposed to be easy to obtain from the collection. The term weighting formula in 3.6 is derived by applying the table and simple smoothing. Nevertheless, this model assumes that R (the amount of relevant documents) and r_i (relevant documents indexed by the term t_i) are known which is often not true when using ad-hoc retrieval without any relevance judgment. This is also the case of how IR systems are used for QA. We know some of the terms, those in the question, but ignore which are in relevant documents. The solution then goes to consider that all documents indexed by the term t_1 are relevant and then the weights would be given by formula 3.7 which is basically a term modelling the inverse document frequency (IDF) of the term in the collecion.

$$W_i^{RSJ} = log\left(\frac{(r_i + 0.5) \cdot (N - R - n_i + r_i + 0.5)}{(n_i - r_i + 0.5) \cdot (R - ri + 0.5)}\right) \tag{3.6}$$

$$W_i^{IDF} = log\left(\frac{N - n_i + 0.5}{n_i + 0.5}\right) \tag{3.7}$$

Documents	Relevant	Non-relevant	Total
indexed by term t_i	$\mathbf{r_i}$	$n - r_i$	$\mathbf{n}$
not indexed by term t_i	$R - r_i$	$N - R - n + s$	$N - n$
Total	$\mathbf{R}$	$N - R$	$\mathbf{N}$

Table 3.3: Contingency table for the estimation of $P(Rel|d_j, t_i)$

Operator	Description	Score
Term query	searches for a single term	allow defining a query term weight
AND	return documents by both sub-queries	the scores are summed
OR	return documents by either sub-queries	sum the scores for documents in both branches
AND_NOT	return documents with the first term and not the second	the score is provided by left
FILTER	return documents by both sub-queries	the scores are from the left
AND_MAYBE	return documents by the left sub-query	the scores from the right are summed if they exist
XOR	like OR, but documents in both branches are not given as results	sum scores
NEAR	return documents where the terms appear near as defined by a window w	sums the scores
PHRASE	return documents where the terms are in window w and in the specified order	sums the scores
ELITE_SET	select and elite set of terms from first documenst and performs a new OR query (blind relevance feedback)	sums the scores

Table 3.4: Xapian Query operators

In our use of Xapian for QA we have not performed special tuning of the parameters and we have reverted to the default values that are recommended by the system designers.

3.5.2.3 Query Language and Query Generation

As almost any other search engine, Xapian provides a boolean query language in order to express queries. We have directly used the query language and defined the way that questions should be mapped to a Xapian query. The Xapian Query language is outlined in Table 3.4. The alternative would have been to directly produce query vectors for the retrieval with the appropriate weights. This alternative, in principle, seemed much more difficult. In fact, this is the role of the query language, as the probabilistic model implicitly defines the use of OR operator for queries. The approach taken by Xapian is to perform the probabilistic search and then apply a boolean filter to the results. An alternative method would have been to apply the boolean constraints and then perform ranking on the resulting set. Note that both approaches do not guarantee to provide the same results than the theoretical model.

There are a large number of alternatives to transform the questions into a Xapian query using the operators in the query language. An example is shown for a simple question in Table

Question	*¿Cuando murió José Stalin?*
Single Terms	*murió José Stalin*

operator	query
OR	murió OR (José OR Stalin)
AND	murió AND (José AND Stalin)
AND_MAYBE	murió AND_MAYBE (José AND_MAYBE Stalin)
AND_MAYBE	murió AND_MAYBE (Stalin AND_MAYBE José)
AND_MAYBE	José AND_MAYBE (murió AND_MAYBE Stalin)
AND_MAYBE	José AND_MAYBE (Stalin AND_MAYBE murió)
AND_MAYBE	Stalin AND_MAYBE (murió AND_MAYBE Stalin)
AND_MAYBE	Stalin AND_MAYBE (José AND_MAYBE murió)
NEAR	NEAR({murió, José, Stalin }, 20)
PHRASE	PHRASE({murió, José, Stalin }, 20)
PHRASE	PHRASE({murió, Stalin, José }, 20)
PHRASE	PHRASE({José, Stalin, murió }, 20)
PHRASE	PHRASE({José, murió, Stalin }, 20)
PHRASE	PHRASE({Stalin, murió, José }, 20)
PHRASE	PHRASE({Stalin, José, murió }, 20)

Table 3.5: Alternative queries withouth structure in the question

3.5 where no structure is known for the terms that form the query. All queries are at some point meaningful while the ordering of the results definitely could change with every formulation. For simplicity the same operator is used to join the terms and the the terms are represented with no stemming. The OR query alternative provides maximum recall if we consider the complete list of results. In contrast, if only a subset of the results is processed we could not guarantee that the most relevant results for successful Answer Extraction would appear in that subset. The actual ranking would depend on the relative frequencies of the different terms. Another important observation is that for some of the operators the order of the terms is in fact important.

The structure of complex terms like NEs can be useful in order to find queries that map with more precision to the desired documents. Table 3.6 considers the formulation of a query for just the name *Jose Stalin* but if we consider as a compound term in contrast to single terms. Our intuition would say that it is more important to find sentences that match *Stalin* than those that match *José*. Querying with an OR operator is again the strategy which will provide the maximum expected recall though it probably would include lot of noise in the form of unrelated documents. AND, NEAR and PHRASE are reasonable queries if precision is needed while PHRASE_1 could in general be preferred to PHRASE_2. AND_MAYBE and in particular, AND_MAYBE_2 are a good trade-off between obtaining good recall and using the intuition than the term *Stalin* should be more important. So far, this intuition is based on just an example and it could be argued that the range of cases is too breadth. Consider among others cases like *John Smith*, *José García* or *Organización Mundial de Comercio*.

Given the large range of options, our strategy has been to provide a generic infrastructure that allows to define queries from terms and sentences that could make used of specific heuristics. General strategies can be overriden depending on the type of the question or the type of the term. For example, we have worked with a hierarchy of objects that considers terms, composed terms, NE and their subtypes. A query generation strategy can be adapted to each of these types of terms. As our system has been concerned with real time QA and simplicity, we have opted to generate just one query for question. A range of alternatives has been tested along different years while the use of OR for maximum recall and the AND_MAYBE operator with order inversion has proved the most useful one. This choice has shown suitable for our specific

Named entity	*José Stalin*
operator	query
OR	José OR Stalin
AND	José AND Stalin
AND_MAYBE_1	José AND_MAYBE Stalin
AND_MAYBE_2	Stalin AND_MAYBE José
NEAR	NEAR({José, Stalin }, 5)
PHRASE 1	PHRASE({José, Stalin }, 5)
PHRASE 2	PHRASE({Stalin, José }, 5)

Table 3.6: Queries using a NE

Answer Extraction and Answer Merging strategies.

3.5.3 Document Retrieval with Lucene

Lucene is a framework for the development of IR engines or search engines. It is also an open-source development, but in this is case it is written in Java. Lucene is based on the Vector Space Model (VSM) [Salton et al., 1975] in order to represent and score documents. The basic model is extended to allow additional structure in the documents.

3.5.3.1 Indexing

Documents in Lucene are composed of fields and each of them is in turn a bag of terms or a vector of terms. The Fielded Document design is interesting for QA because it allows to exploit additional structure of the document. For example, the same term can be given different weights if it appears in the title and in the text. Fielded documents also allow to have different analysis of the same document text, for example a field could store plain tokens and the other NE. Equation 3.8 shows their mathematical representation. A single index stores the different fields which can be combined in a single query. While a similar functionality can be accomplished with several indexes in other systems, Lucene features greatly simplify their management and development. Another important point of Lucene is that the document analysis framework, including analyzers and stemmers, is well designed and extensible.

$$d_j =< f_1, \ldots, f_k, \ldots, f_L > \text{ where } f_i \text{ is a field} \tag{3.8}$$

$$d_j =< f_1 :< w_{1,1}, \ldots, w_{1,N_1} >, \ldots, f_k :< w_{1,1}, \ldots, w_{1,N_k} > \ldots, f_L :< w_{1,1}, \ldots, w_{1,N_L} >>$$
where $w_{i,j}$ is a the weight for term t_j in field f_i

$$d_j = [w_{1,1} \ldots w_{1,N_k}, w_{2,1} \ldots w_{k-1,N_{k-1}}, w_{k,1} \ldots w_{k,N_k}, w_{k+1,1} \ldots w_{L-1,N_{L-1}}, w_{L,1} \ldots w_{L,N_k}]^T \tag{3.9}$$

While most of the results presented make use of a single field representation of the document, we have also make a profit from the multi-field functionality for several uses like priming titles in Wikipedia articles or time indexing. On the other hand it must be said that using several fields for representing a document requires additional work on tuning the weights or boost factors of different fields.

3.5.3.2 Ranking

Lucene uses the VSM in order to represent documents but their ranking function it is extended to the structured case. In the basic VSM, each document is represented as a vector of weighted terms. The similarity between documents is measured in terms of the cosine distance between vectors as in 3.10. To produce a IR ranking formula, the query is just represented as a document and the cosine similarity is used to produce the rank. Again, to produce a formula that is easier to compute at query time, several rank preserving transformations are carried out (3.11 and 3.12). The final scoring formula is in essence a product of the weights of the matching terms in the query normalized by a document length factor.

$$sim(d_i, d_j) = cos(d_i, d_j) = \frac{d_i \cdot d_j}{|d_i| \cdot |d_j|} \tag{3.10}$$

$$RSV(d_i, q) = rank(cos(d_i, q)) = rank\left(\frac{d_i \cdot q}{|d_i| \cdot |q|}\right) = rank\left(\frac{d_i \cdot q}{|d_i|}\right) \tag{3.11}$$

$$RSV(d_i, q) = rank\left(\sum_{t_j \in d_i \cap q} w_{i,j} \cdot \frac{1}{|d_i|}\right) = rank\left(\sum_{t_j \in q} w_{i,j} \cdot \frac{1}{|d_i|}\right) = rank\left(\sum_{t_j \in q} w_{i,j} \cdot len_i\right) \tag{3.12}$$

The final formula is adapted in Lucene to include a common type of weights which is a variation of $TF - IDF$. $TF - IDF$ stands for the product of the term frequency in a document (tf_j) and the inverse of the frequency of this term across documents (df_j). For practical reasons, it is common to apply monotonic transformations to this factors to soften their influence. Lucene default choice which is shown in Equation 3.13 uses a logarithmic transformation. Besides, it also modifies slightly the length factor and uses a quadratic function of the length which further penalizes (or compensates) very long documents.

$$RSV(d_i, q) = \sum_{t_j \in q} w_{i,j} \cdot \ln_i = \sum_{t_j \in q} tf_{i,j} \cdot log\left(\frac{n}{df_j + 1}\right) \cdot \frac{1}{count(d_i)^2} \tag{3.13}$$

When several fields are used for indexing the scoring formula is extended like in Equation 3.14. The individual field RSV scoring is combined by means of boost factors. Beyond the basic scoring formula, the Lucene framework allows to rewrite user defined similarity measures and also to include ad-hoc weights to queries which can be used to modify the scoring value.

$$RSV(d_i, q) = \sum_{t_{k,j} \in q} boost_k \cdot RSV_k(d_i, q) \tag{3.14}$$

3.5.3.3 Query Language and Query Generation

The set of query operators that are implemented in Lucene are presented in Table 3.7. The main combiner operations include Boolean and Phrase operators but Lucene includes also a wider range of operators to match single terms beyond exact matching. However, in the context of QA most of the single term operators like Range or Prefix are of no use. Fuzzy queries could have been used for detecting variant NE but we have not explored them yet.

Although Boolean and Phrase operators look similar to those in Xapian, a closer inspection yields significant differences. The first difference consists on the way operators and the scoring function relate to. In Lucene the boolean operators are applied before producing the scoring. For instance, an AND operation does not return results if any of the terms are not matched. Phrase queries are also interpreted in slightly different form because it reorders documents depending on a distance between the original query and the span in the document. With the query language

Operator	Description	Score
Term Query	searches for a single term	
Range Query	searches for all terms in the range defined by terms *begin-end*	
Prefix Query	all terms that have the common prefix	
Wildcard Query	terms could include a prefix	
Fuzzy Queries	the term matches also close variants as measured by Levensthein string distance, a factor to measure similarity is used	the weight of the terms is modified by the distance
Boolean Queries	search performs a boolean operation between two subqueries by setting two flags for each operator: required, prohibited	sum scores
	AND	$required \& \neg prohibited$
	OR	$\neg required \& \neg prohibited$
	NOT	$\neg required \& prohibited$
Phrase Queries	requires all the terms to appear in order, a slop factor controls rotation or skips to match terms in order	sums scores, score is modified by the number of edits, less edits mean higher score

Table 3.7: Query operators in Lucene

offered by Lucene, the best basic model consists on OR all the query terms. Precision on the ranked document lists can be improved by including additional subqueries with complex terms using conjunction like *Jose OR Stalin OR (Jose AND Stalin)*. The effect of these kind of queries with a varying number of query terms is however not clear. As in the case of Xapian, this is motivated in practical observations more than on a solid theory.

The main conclusion to draw from the analysis of the query languages is that a flexible framework that allow to experiment different queries is needed. We reused our schema to specify specialized strategies for terms, complex terms and different NE types with Lucene and make use of the recognized structure. In the long term, future IR models that are able to capture the dependency and structure between queries could provide better alternatives to transform questions into meaningful retrieval queries.

3.5.4 Sentence filtering

The last step of the retrieval module is in charge of selecting only relevant and promising sentences to feed them to Answer Selection module. It reduces significantly the search space by providing sentences instead of documents. The module proceeds by retrieving and splitting in sentences the first N_{doc} documents using STILUS tools. Sentences are filtered using a simple and inexpensive procedure. Sentences or snippets that contain a number of relevant terms from the question are considered for further processing. The absolute threshold have been set to different values depending on the number of relevant terms that the question contains.

The parametrization used in most submissions have considered two different kind of questions, those with just few relevant terms and the rest. Definitions and other simple question contain usually just a relevant term as the rest of the words are usually question words or stopwords. Consider questions like *¿Qué es BMW?* or *¿Qué es el Altlantis?*. The focus of the definition is the only relevant term and that should be matched. For longer questions, at least two relevant terms are required. Besides, a simple heuristic method that maps partial matches of NEs (*Stalin*) to complex NEs (*Josef_Stalin*) as stated in the question is also used. It only

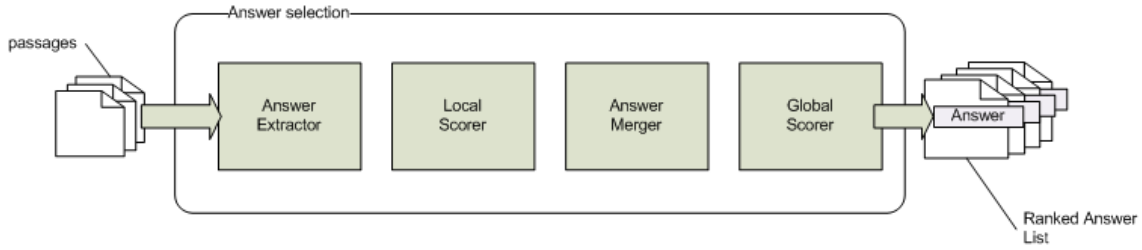

Figure 3.13: Answer Selection Pipeline

takes into account the overlap between simple and complex terms ignoring stopwords. Though this method is not really precise it has been generally useful.

In general, the Sentence Filter is oriented to obtain good precision, by retaining sentences related to the question. It is designed in conjuction with the later Answer Selection in mind which also uses term overlap heuristics though they can be defined on language generalizations like lemmas. A different technique that uses Minimal Weight Span selection as proposed in Monz [2004] has been also implemented and tested for filtering. The implementation details differ slightly from the initial proposal, because we used it to filter and score sentences, and not larger passages of variable length. Nevertheless, we found no overall improvement when we tested it in the context of our system.

3.6 Answer Selection

The Answer Selection step has been designed as a pipeline of modules (see Figure 3.6): Answer Extraction, Local Scoring, Answer Merging and Global Scoring. Different strategies have been implemented in the system along the years. Especially, two different strategies have been used in 2004 and from 2005 until 2009. The first strategy relied on soft matching based on Hidden Markov Models which were trained from the Web. The second strategy for Answer Extraction could be framed into a more classical view of IE, which is often used in systems that view QA as IR followed by IE. This strategy is based on NE recognition and it has used extensively STILUS language analysis tools. In fact, some of the developments in STILUS has been influenced to make it flexible and adequate for QA.

Finally, a third approach motivates the work of the second part of this thesis. It is also based on a IE pipeline, which will perform NE recognition and Relation Extraction. The main difference however would be in the way that these functions are created. While STILUS and their rules has been manually crafted, the purpose of the second part is to study the acquisition of resources that would help in NE and RE. The long-term goal consists on advance towards language and domain independent techniques at a reasonable level of performance.

3.6.1 Answer Extraction

The core Answer Extractor receives the analysis from the processed question and a passage of text, in particular a sentence, and produces zero, one or several candidate answers. Each answer is associated with its original sentence, also known as snippet. In the Named Entity selection approach the key information is the Expected Answer Type of the question. The taxonomy of EAT from the Question Analysis has to be mapped to concepts from the NE taxonomy. The design that we have used relied on a bank of filters. Each filter is specialized and is activated only when the question is classified with the given EAT. The system have used STILUS as a base for Spanish linguistic analysis and the filters rely on its annotations. General post-processing filters are applied in the final stage as well. For instance, they can filter common errors, stopwords and

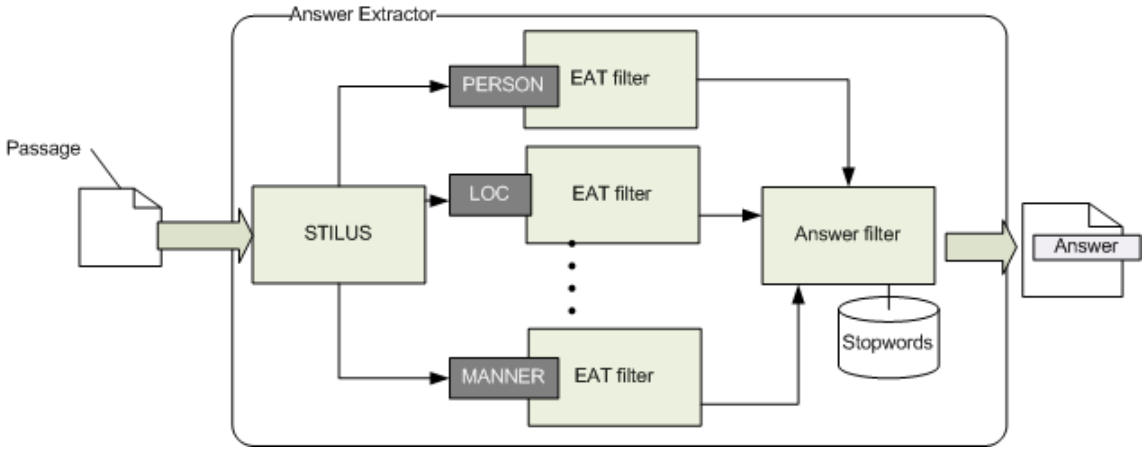

Figure 3.14: Answer Extraction Pipeline

question terms from the list of candidate answers. The architecture of the Answer Extractor is depicted in Figure 3.14.

3.6.2 EAT Filters

Since the adoption of STILUS, it was clear that a mechanism to adapt the results of the linguistic analysis to the taxonomy of answers was required. The main reason was to bridge the gap between the annotations provided by STILUS and those required as useful in the QA context. Even after STILUS adopted the ENE taxonomy to categorize entities, there are differences in purpose. High precision was the priority in STILUS while a QA system could benefit by improving the recall at recognizing entities. We enumerate some of the advantages:

- Help to introduce new types of questions like definitions, previously uncategorized entities or domain specific entities.
- Additional and complementary rules for NE types that improve recall are easier to implement.
- Help to use different operational definitions for certain types or include additional processing like normalization of dates.
- Combine replicated functionality like additional NERC modules, list of addional Name Resources, etc.

The solution proposed to improve recall and reduce the semantic mismatch between the analysis tools and the QA system was to rely on a mechanism to implement EAT specific filters. The aim of the filter is to take as input a processed sentence and produce answer candidates for a given Answer Type. The most basic operation consists on extracting tokens based on the annotations provided by STILUS. The test can target any of the annotation levels provided like the text, the lemma, the morphological tags or the semantic information and should take into account that multiple annotations for each token do exist. Besides, a mechanism to join and split tokens is also needed. It should help to describe how the annotations are created or inherited for new tokens. Finally, it should be possible to cascade several automata in order to reuse definitions and compose complex EAT filters.

The filters are based on a regular automata with basic transducing capabilities which processed the sequence of tokens generated by STILUS. The fact that the automata works on tokens and needed to test conditions at several information levels is important, because it made impossible to reuse other tools. A filter automata is defined by:

- A set of Predicates. Boolean Predicates could be defined to target conditions at different information levels of a token analsyis. Particularly, regular expressions on tokens and tags were useful. Examples of predicates that were defined are presented in Table 3.8. In fact their actual form is different as they have just been provided as Java classes. In order to be able to compose complex predicates and use boolean operators we relied on the utilities provided by the common-utils library from the Apache project[4]. The basic binary boolean operators were supported like AND, OR, NOR and XOR. Besides, useful efficient implementations of operators with several predicates like ALL_TRUE and ANY_TRUE were provided.

- A set of Actions. One or several actions could be performed in the sequence of tokens that are matched by the automaton. Operations consisted on adding the matched tokens to the result by composing the sequence of tokens into a complex token, replacing the token in place and adding filter specific semantic information.

- The structure of the automaton was defined by a set of initial states, a set of final states and a set of transitions. Transitions are triplets s, d, P defined by a source state s , a destination state d and the predicate P which should be matched in order to complete the transition. All states with no incoming edges are supposed to be initial states and those with no outcoming edges in turn are final states. Because predicates can be complex and they are composed, for example $P_2 = P_1 \bigvee P_0$, interpretations problem can arise if no order is defined. Consider a subgraph with transitions $s_1 \xrightarrow{P_1} s_2$, and $s_1 \xrightarrow{P_2} s_3$. If a token fulfills P_1 and P_2 withouth an interpretation order the path would be underspecified. We decided to use the convention of top-down order in transition declaration to avoid this problem.

The interpretation of the automaton proceeds by scanning a sentence $S = w_1, w_2 \ldots, w_n$ from left to right trying to match a sequence of states from start to end in the automaton using token w_1 and consuming a token in each transition. When no end state is reached or a predicate is not fulfilled, the algorithm backtracks to the last choice and proceeds with the next transition to test. When a final state is reached the actions defined are executed. If no match is found it advances to next the token, in this case w_2. The automaton is executed until the sequence of tokens is exhausted.

An specialized answer filter was written for every EAT using this kind of rule automata to produce candidate answers. A simplified example is shown in Figure 3.15. The simplest filters could just add STILUS tokens to the results. More complex filters could be composed of a dozen of predicates and elaborated programs. For some EAT types like MANNER and OTHER , it is difficult to establish a precise model of the answer based on the information available. For example, MANNER (usually manner of death) could be tagged as an adjective or a past participle (*asesinado*) or a preprositional phrase (*de una puñalada*) or even both (*asesinado de una puñalada*). Some of the filters that could be defined with the morphological information are too loose and would have low precision.

3.6.3 Answer Extraction with Paraphrase Patterns Acquired from the Web (CLEF 2004)

For our first participation in CLEF, we attempted a very different answer extraction strategy. Following from our analysis of the CLEF 2003 question dataset we realized that a large number of the questions were in fact asking for relations between entities. For example, the question *¿Cual es la capital de Croacia?* could be seen as asking to complete the predicate `capital(Croacia,X)` which in turn is an specific instance of CAPITAL(COUNTRY,CITY). Questions were grouped

[4]http://commons.apache.org/

Predicate	Example	Explanation
text = value	text = `de`	the text of a token is the same than a string
text in set(value)	text in `de,las,los`	the text of a token is any of a set of strings
text like RegExp	text like `isPunct()`	the text of a token matches a regular expresion, like punctuations
lemma = value	lemma = `de`	the lemma of a token is the same than a string
lemma in set(value)	lemma in `de,las,los`	the lemma of a token is any of a set of strings
lemma like RegExp	lemma like `.*s`	the lemma of a token matches a regular expresion
tag = value	tag = `NPMS--N-N-`	a token is tagged with a certain tag (Proper Noun, masculine, Singular, etc.)
tag is empty	tag is `empty`	the token does not have a tag, unknown word
tag like RegExp	tag like `/bN.*`	the tag matches a certain regular expresion, useful to check certain features or feature sets. For intance, the tag is a noun
tag like RegExp	tag like `/bN.FS.*`	the token is tagged as a noun, femenine and singular
SemEntity = value	SemEntity = `@inst@nofiction @PERSON@FULL_NAME`	the token is semantically tagged with a value, as a full name for person
SemEntity is empty	SemEntity is `empty`	the token have no value in SemEntity
SemEntity like RegExp	SemEntity like `.*FIRST_NAME.*`	the token is tagged as a FIRST_NAME

Table 3.8: Example of frequently used predicates

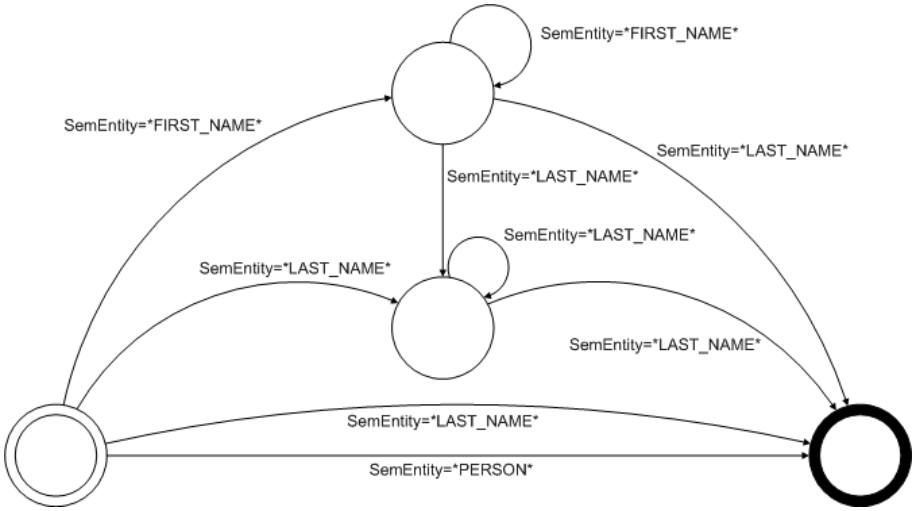

Figure 3.15: An sketch of an EAT filter automata for PERSON names (simplified from the used one)

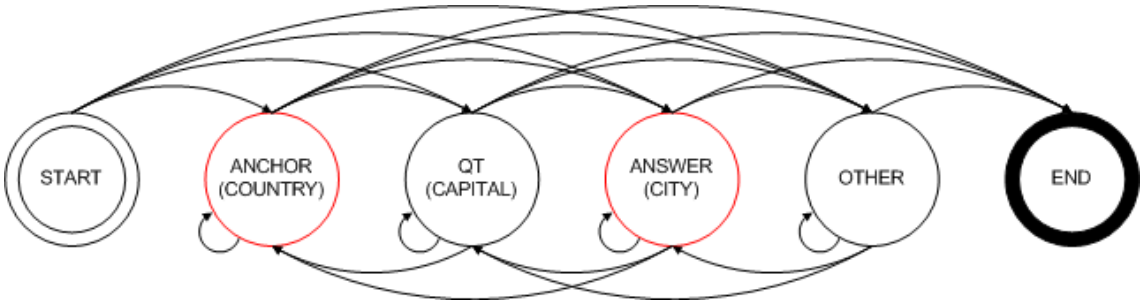

Figure 3.16: The HMM structure for learning patterns, states are question specific annotations and the transitions are triggered by words

into common predicates and we proceed first by manually annotating the questions with three different types of terms: the known term in the relation (Question Topic, (COUNTRY)), the EAT (CITY) and the terms which denote the relation, *capital* in this case. 17 different groups were identified though they were not as concrete as relations but more like related clusters. We tried to keep a balance between preserving their meaning and their size. When there was no specific types we used general placeholders like (QT) and (ANSWER). We trained a model to extract answers for each of the relations based on a Hidden Markov Model (HMM) Rabiner [1989] at the syntactic-morphological level. The model tried to capture the syntactic structure of the sentences which convey the relation meaning.

Once the questions were manually analyzed we implemented a system that was able to reproduce our manual annotation and used it for question analysis. Our system used the ms-tools [Turmo et al., 1998], a package for language processing developed by TALP-UPC and UAB NLP groups which mixed statistical and rule based methods for the analysis of Spanish. ms-tools integrated a POS tagger (MACO) and a partial parser (TACAT) as well as a NERC module. We modified TACAT rules to prevent prepositional attachment as it was more appropriate to our interests and added some rules to parse questions correctly. Once the questions were tagged and parsed, a set of manually developed rules were used to assign a template to questions and map to our relation specific tags.

We used question and answer pairs to build the models to each of the manually identified clusters. Question terms and answers strings were combined in a query and sent to Google using the Google API [5] with the restriction that only Spanish documents should be retrieved. The top 100 snippets from the results were split into sentences using heuristics, analyzed and annotated with question and answer terms. Only snippets which contained at least one question and one answer term were selected to train the model. We trained a HMM in which hidden states were syntactic-semantic tags assigned to the chunks and symbols were POS tags. To estimate the transition and emission probabilities of the automata, we used Maximum Likelihood Estimation and counted the frequencies of bigrams for CHUNK-CHUNK and CHUNK-POS. Besides, simple add-one smoothing was used to partially avoid the data sparsity problem [Manning and Schütze, 1999].

In order to extract answers the models were paired with their own question type. Sentences provided by the retrieval step were analyzed with the same pipeline than training snippets. Syntactic chunks that contained any of the terms in the question were retagged with their semantic tags and were used as anchors. The system selected pieces in a window of 10 words around anchor terms that went to the phase of HMM decoding. For decoding a variant of N-best recognition strategy was used to identify the most probable sequence of states (syntactic and semantic tags) that could have originated the POS sequence. The special semantic tag that identifies the answer (ANSWER) represents the state for words that should be part of

[5] http://code.google.com/apis/soapsearch/reference.html

the answer. The decoding algorithm was guided to visit states marked as anchors in order to find a path that passes through the answer state. The algorithm also assigned a score to every computed path and candidate answer based on the log probabilities of the HMM.

3.6.4 Local Scorer

The role of the Local Scorer is to judge which extracted candidate answers are more suitable to be the correct answer. Several heuristics have been combined along the years. The local scores made use of the text of the candidate answers but also the context n in which it appears, the snippet or sentence. In CLEF, they have been combined in order to provide an unified local score. The heuristics has been manually combined in a single local scoring function ($LS(A_i, S_j, Q)$) which scores every answer(A) in the context of their snippet (S_j) for a given question (Q).

- **Retrieval score**. The score provided by the document or passage retrieval stage. The overall relevance of the documents is usually a good starting point to estimate the relevance of a sentence.
- **Term overlap between question and sentence**. The ratio of relevant terms from the question contained in the sentence. We have experimented with different normalizations including the document, question and both which is similar to the Jaccard measure. Besides, the overlap can be measured at the token or at the lemma level. We have usually removed stopwords and performed at the lemma level.

$$LS_{overlapQ}(S, Q) = \frac{|Q \bigcap S|}{|Q|} \tag{3.15}$$

$$LS_{overlapS}(S, Q) = \frac{|Q \bigcap S|}{|S|} \tag{3.16}$$

$$LS_{Jaccard}(S, Q) = \frac{|Q \bigcap S|}{|Q \bigcup S|} \tag{3.17}$$

- **TF-IDF scores on the snippets**. IR engines score documents and passages using the term frequency (TF) and inverse document frequency (IDF) of terms from the whole document collection. When several terms are used to form a query the distribution of terms in the results is usually very different from the collection. The distribution of terms in the collection could produce results that are highly skewed by the low document frequency of one of the question terms. In contrast, other questions terms that may help to locate the answer or to disambiguate the topic, may be underrepresented in the results. On 2005 we experimented different forms to weight relevant question terms (q_i) in order to take these effects into consideration. Term frequencies and inverse term frequencies were calculated using the set of selected passages $S =< S_1, \ldots, S_j >$.

$$LS_{tfidf}(S, Q) = \sum_i w(q_i, S) \tag{3.18}$$

$$w(q_i, S) = \frac{tf(q_i)}{df(q_i, S)} \tag{3.19}$$

$$df(q_i, S) = \sum_j \delta(q_i, S_j) \tag{3.20}$$

$$\delta(q_i, S_j) = \begin{cases} 1 & \text{if } q_i \in Sj \\ 0 & \text{if } q_i \notin Sj \end{cases} \tag{3.21}$$

- **Distance from keywords to answers**. The average distance of keywords to answers has been also used to score and differentiate between several candidate answers provided in the same sentence. In general, candidate answers that appear closer to relevant terms are probably correct.

$$LS_{distance}(A, S, Q) = \sum_i w(q_i, S) * distance(A, S, q_i) \tag{3.22}$$

$$distance(A, S, q_i) = \frac{d_{max} - d(A, q_i)}{d_{max}} \tag{3.23}$$

$$d(A, q_i) = \begin{cases} A_s - q_i & \text{if } i < s \cup |s - i| < d_{max} \\ 0 & \text{if } s \leq i \leq e \\ q_i - A_e & \text{if } i > e \cup |e - i| < d_{max} \\ 0 & \text{otherwise} \end{cases} \tag{3.24}$$

3.6.5 Answer Merging

Redundancy is one of the main differences between Question Answering and traditional Information Extraction. Information from several documents or sentences can be aggregated in order to find the correct answer or a better estimate of the confidence. The same answer could be expressed in several different surface forms. Besides, extraction errors will produce incomplete answers as well as answers that are longer than the correct ones. Merging all these possible candidates could be complicated. We conflate answers using two different tests. The obvious one consists on merging those answer string that are exactly the same. The second merges those that have a large overlap when answer are tokenized and cleaned from stopwords and question terms. Specialized rule-based heuristics have been used for certain types of answers like dates (date normalization started to be applied in 2006) and abbreviations (check the correspondance of initials and expansion). Lately, additional world knowledge like that provided by SemRemission tags in STILUS has also been incorporated.

When several answer are merged, a representative candidate for the answer cluster is required. We found that the simple heuristic that chooses the longest candidate is good enough. For experiments that should report a representative snippet we have decided to use the highest scored snippet. Although not frequent, we have found cases where the correct answers selected a bad snippet with this strategy. This effect has been more frequent in instances of the system working on the Web. An issue that deserves further investigation is the problem of selecting a good supporting snippet. It seems that is very different to perform this task in CLEF or TREC collections where redundancy is very low than in Web collections. Selecting a good representative snippet of the answer from the Web is harder.

3.6.6 Global Scorer

Once answers are clustered the final score can be modified to take into account the issue of answer redundancy. We have performed this step by considering a linear combination between the local answer score and the redundancy score. The weights of the two factors have been tuned manually. N_{max} is a parameter for the maximun number of different answers that we consider significant.

$$GS(A_i, Q) = \alpha * max_j(LS(A_i, S, Q)) + (1 - \alpha) * \frac{N_{A_i}}{N_{max}} \tag{3.25}$$

Our experience shows that small weight values for the redundancy already provide advantages in CLEF collections. Nevertheless, more principled ways to tune the correct values are needed as they depend on the range of the scores returned by local scoring functions.

Chapter 4

MIRACLE Participation in QA@CLEF

The QA task at CLEF was initiated as a pilot task in 2003 [Magnini et al., 2003] to foster research on Cross-Lingual QA and Monolingual QA in other languages than English which had been the only focus of TREC. On 2004, the pilot task was confirmed as a regular task. CLEF 2004 [Magnini et al., 2005] offered evaluation on nine source European languages (Bulgarian, Dutch, English, Finnish, French, German, Italian, Portuguese and Spanish) and seven target language (Dutch, English, French, German, Italian, Portuguese and Spanish). Six monolingual tasks and 50 cross-lingual task were setup for participants and our group debuted, with their participation in the Spanish monolingual task. The last evaluation has been carried in 2009 and has always included new challenges, new languages and new collection types.

MIRACLE has taken part since 2004 in the Spanish track until now. During these years in which the described system has been developed and enhanced. We report main results and conclusions for our participation since 2004 to 2007 as this is the period where the author of these thesis was responsible for the architecture, development of main QA modules, submissions of runs and discussion of the results. Nevertheless, a QA system is a complex software architecture and all the members from the MIRACLE team contributed important ideas, codes, analysis and their precious time to achieve the goal of submit our best results every year. This work would not have been possible withouth their help.

4.1 CLEF 2004

MIRACLE took part in 2004 in the monolingual Spanish track at CLEF@QA. The objective of the evaluation was to compare approaches for open domain QA. The target corpora was formed for all languages by collections of newspapers and news agencies articles. In the case of Spanish the corpus was formed by news from the EFE newswire service dating from years 1994 and 1995. The main statistics of the corpora are summarized in Table 4.1.

The collection is formatted using SGML and encoded in Latin-1 (ISO-8859-1). An example document is shown in Example 3.4. Documents were identified in the collection by the DOCID field which was also required as a reference to judge answers. They contain unstructured plain text (no bold or italics markup) on the TITLE and TEXT fields which were intended as the

Collection	Size (MB)	Documents
EFE 1994	509	215,738
EFE 1995	577	238,307

Table 4.1: Target collection for ES-ES task at 1994

source of information. Titles are generally capitalized and the TITLE field includes references to news locations as is shown in the example. The style of the text follows journalistic conventions. The TEXT field includes also additional information like place, date, source agency, author signature and automatic timestamping with more or less fixed patterns. Nevertheless, this data does not follow a unique convention across time. Proper document metadata includes fields for DATE, TIME and CATEGORY. DATE and TIME refer to the editorial date and time of the news article. A small proportion of the articles contain duplicates or nearly duplicates, as they contain small typographic and style corrections.

The evaluation question set was formed by 200 questions which contained factual and definition questions. Definitional questions were aimed to be an innovation with respect to 2003 edition and they represented a 10% of the total questions. Another 10% of the questions were decided to be NIL questions with no correct answer in the target collection. The questions were generated by the groups carrying the evaluation and covered different types like PERSON, LOCATION, ORGANIZATION, OTHER, MEASURE, TIME, OBJECT and MANNER. The full set of questions for all languages was later available as the MultiEight corpus [Magnini et al., 2005]. The questions contained information about the question types, factual information (F) or definition (D). Some examples are presented in Example 4.1.

The evaluation of the systems considered the extraction of correct answers and evaluated only the first correct answers, due to the large number of submissions. The evaluation measure selected was $Acc(n = 1)$ while for NIL questions, Precision and Recall were used. The use of confidence values was optional and for those systems that provide it the Confidence Weighted Score measure was used too.

Example 4.1 Some examples of CLEF 2004 questions

F	ES	ES	0001	¿Qué año le fue concedido el premio Nobel a Thomas Mann?
F	ES	ES	0002	¿Cuánto aumenta la población mundial cada año?
F	ES	ES	0003	¿Cuál es el nombre de pila del juez Borsellino?
-	- -	- -	- -	- - -
F	ES	ES	0005	¿Qué cargo detenta Ariel Sharon?
-	- -	- -	- -	- - -
F	ES	ES	0007	¿Qué empresa de automóviles produce el "Escarabajo"?
F	ES	ES	0008	¿Dónde tiene lugar el Motorshow?
F	ES	ES	0009	¿Cómo murió Jimi Hendrix?
D	ES	ES	0010	¿Qué es el Mossad?
-	- -	- -	- -	- - -
F	ES	ES	0012	¿De qué está recubierto el continente antártico?
-	- -	- -	- -	- - -
D	ES	ES	0029	¿Quién es Yves Saint Laurent?
-	- -	- -	- -	- - -

4.1.1 Results and descriptions of the runs

The system that MIRACLE group developed for QA@CLEF 2004, miraQA, was our first attempt to face the Question Answering task. The system was developed for the monolingual Spanish subtask and followed the typical three stage architecture that is used in the Knowledge Mining approach. It contained modules for Question Analysis, Document Retrieval and Answer Selection. The question typology was simpler and grouped questions into 17 clusters according to the examples we obtained from the previous CLEF 2003 evaluations. The Question Analysis was based on manual patterns that mapped to each group and probably had low coverage. Document retrieval was based on Xapian and Answer Extraction used the patterns obtained by

Question type	Right	Wrong	IneXact	Unsupported	*Acc*@1
Factoid	18	154	4	1	10
Definition	0	17	3	0	0
Total	18	174	7	1	9

Table 4.2: MIRACLE results on CLEF 2004 (miraQA)

the method described in Section 3.6.3. Besides the online QA phase, our approach required a system to train the models. The system used pairs of questions and answers from the CLEF 2003 dataset to query Google and select relevant snippets.

We submitted one run for the monolingual Spanish task (mira041eses). Our system was unable to compute the confidence measure. The same approach was used for factoid and definition questions, though different extraction models were used. Definitional questions in CLEF 2004 (and the few examples in CLEF 2003) were mainly short descriptive phrases including factual information like nationalities, professions, etc. We believed that the approach could be equally useful for both types of questions. NIL answers were returned whenever no answer candidates were found. That could be due to the Document Retrieval step, which did not find related documents, or to the Answer Extraction, which filtered all candidate sentences without extracting answers.

The results for the submitted run are outlined in Table 4.2 and leave a very low accuracy of 10% for factual questions. For definition questions the results are even more negative and no questions were correctly answered. 8 more questions were judged as inexact or unsupported mainly because the extraction method was unable to identify the complete answer. The same number of questions were answered correctly across the answer types considered by the organization, although our analysis reflect that questions of the MEASURE type were correctly answered more frequently.

During development and at later stages, it became evident that robustness of the language tools was an important issue. The accuracy of tools and their appropriateness for QA was not as expected, but definitely not the main source of failure. Training data (question-answer pairs from 2003) were probably too few and our method did not find and estimate correct models. Besides, methodological issues regarding overfitting and the selection of so much classes made the data sparseness problem even worse.

4.2 CLEF 2005

The CLEF 2005 task [Vallin et al., 2006] was a slight evolution from the task already defined in 2004. Most groups requested the task to be stable in order to achieve better performance and polish methods and systems. Document target collections remained the same and a new evaluation set of 200 questions was produced for each language. Experience from previous evaluations show that MANNER and OBJECT questions were problematic for evaluation. Agreement between human judges on what was a correct answer for this types was not high. Their relative importance in number was reduced, though nothing was noted on the official guidelines. The proportion of Factoid and Definitional questions changed and the last amount for 25% of the test set. The most important change was the inclusion of 30 temporally restricted questions. Table 4.2 includes several examples for most of the types that were representative.

4.2.1 Run Descriptions and System Configuration

For our second participation in the CLEF-QA task, we submitted six runs with Spanish as a target language, but different source languages, Spanish, English and Italian. The system was

Example 4.2 Some examples of CLEF 2005 questions

D	0001	ES	ES	¿Qué es BMW?
-	- -	- -	- -	- - -
D	0003	ES	ES	¿Quién es Nelson Mandela?
-	- -	- -	- -	- - -
F	0006	ES	ES	Nombre un edificio envuelto por Christo.
F	0007	ES	ES	¿A cuánto asciende el premio para la ganadora de Wimbledon?
F	0008	ES	ES	¿Con qué grupo ha cantado Robbie Williams?
F	0009	ES	ES	Nombre una película en la que se hayan usado animaciones por ordenador.
T	0010	ES	ES	¿Quién recibió el Premio Nobel de la Paz en 1989?
T	0011	ES	ES	¿Quién hizo el personaje de Superman antes de quedar paralizado?
-	- -	- -	- -	- - -
F	0013	ES	ES	¿Cuándo nació Christopher Reeve?
-	- -	- -	- -	- - -
F	0025	ES	ES	¿Qué conferencia de la UE adoptó la Agenda 2000 en Berlín?
-	- -	- -	- -	- - -
T	0029	ES	ES	¿Qué político liberal fue ministro de Sanidad italiano entre 1989 y 1993?
-	- -	- -	- -	- - -
F	0033	ES	ES	¿Dónde nació Supachai Panitchpakdi?
F	0034	ES	ES	¿Qué deporte practica Adrian Mutu?
F	0035	ES	ES	¿Quiénes eran los dos firmantes del tratado de paz entre Jordania e Israel?
F	0036	ES	ES	¿Qué alfabeto tiene sólo cuatro letras "A, C, G, y T"?
-	- -	- -	- -	- - -
F	0096	ES	ES	¿Qué organismo español se encarga de informar sobre los movimientos sísmicos?
-	- -	- -	- -	- - -
F	0108	ES	ES	¿Cuántos ejemplares de ballena "Minke" quedan en el mundo?
-	- -	- -	- -	- - -
T	0137	ES	ES	¿Qué se celebró en Nápoles del 8 al 10 de julio de 1994?
-	- -	- -	- -	- - -
F	0181	ES	ES	¿Cómo murió Olof Palme?
-	- -	- -	- -	- - -

almost a new development based on the experience acquired from our first contribution on 2004. It used different resources from MIRACLE's toolbox as well as open source components and already had the architecture described before in this chapter.

The system was already composed of a Rule-based Question Analysis module. The Question Classification steered up the strategy to answer a question. Document Retrieval was based on Xapian and used a high number of document resuls ($N_{doc} = 100$) to achieve the maximum recall at this step. Two runs, mira051 and mira052 were submitted for each language pair where the main difference between configurations was on:

- Answer Extraction of MANNER and OTHER questions. Runs mira051 extracted mainly noun phrases using regular expressions defined in the morphological analysis of STILUS. Rules were expressed in the formalism explained in Subsection 3.6.2. We found difficult to establish a good model of the answer, for that reason in our second group of runs (mira052) we experimented with a simpler extractor, candidates in this extractor are selected as chunks between content words. Due to the removal of most of these questions (conflated

Run	R	X	U	*Acc*@1(F)	*Acc*@1(D)	*Acc*@1(T)	*Acc*@1
mira051ESES	**51**	11	0	**26,27**	34	9,38	**25,5**
mira052ESES	46	14	0	22,03	34	9,38	23
mira051ENES	39	7	1	16,95	28	15,62	19,5
mira052ENES	39	8	2	16,95	28	15,62	19,5
mira051ITES	36	10	0	16,95	26	9,38	18
mira052ITES	35	11	0	16,95	24	9,38	17,5

Table 4.3: MIRACLE results on CLEF 2005

Run	*Acc*@1	*CWS*	*K*1	*Corr*
mira051ESES	**25,5**	**0,12372**	**-0,3021**	0,3157
mira052ESES	23,0	0,10382	-0,3432	**0,3160**
mira051ENES	19,5	0,09376	-0,3922	0,2300
mira052ENES	19,5	0,08809	-0,3943	0,2278
mira051ITES	18,0	0,06829	-0,4379	0,2244
mira052ITES	17,5	0,07186	-0,4471	0,2192

Table 4.4: MIRACLE results on CLEF 2005 (Global measures)

in OTHER) in the evaluation their influence was limited.

- Answer Ranking. Runs mira051 scored sentences using only the inverse frequency of relevant terms from the query that appear in the retrieved documents. Roughly, it corresponds to usig plain IDF on the retrieved documents to rank sentences. In this case, answer inherit the same score than their snippet or sentence. Runs mira052 used a weighted combination of $TF - IDF$ in the relevant documents (as described in Subsection 3.6.4) and average distance from keywords to answers.

For cross-lingual runs we used external translation services to translate questions. In particular for the submitted experiments Systran[1] was used. Two pair of languages were submitted English-Spanish (ENES) and Italian-Spanish (ITES) without any additional treatment of questions. Temporal questions were not addressed in any special form in these evaluations.

4.2.2 Monolingual experiments

Results for the six submitted runs for evaluation are outlined in Table 4.3. Table 4.5 presents the same results split by the coarse types used by the evaluation organizers. Best results were achieved in the mira051ESES run but the difference does not seem significant for $Acc(n = 1)$. On the other hand, the two runs contained a moderate number of different answers at this position. Unfortunately, with our results it seems that the answer on whether distance from keyword to answer helps to improve answer selection remains open.

4.2.2.1 Results analysis by type

The system obtains better performance for definition questions which are classified regarding the type of the *definiendum*, ORGANIZATION or PERSON. Both types could be answered with high frequency with nominal and descriptive phrases defined as automaton filters on the morphological information provided by STILUS. Other common type that was already addressed

[1] http://www.systran.co.uk/

	Correct Answers														
	Definition		Factoid						Temporally restricted					Total	
run	Or [25]	Pe [25]	Lo [21]	Me [17]	Or [22]	Ot [19]	Pe [20]	Ti [19]	Lo [6]	Me [7]	Or [6]	Ot [6]	Pe [7]	#	Acc(1)
mira051ESES	8	9	7	3	6	4	10	1	-	2	-	-	1	51	25.50
mira052ESES	8	9	6	3	4	2	10	1	-	2	-	-	1	46	23.00
mira051ENES	6	8	6	5	2	2	5	-	-	2	1	1	1	39	19.50
mira052ENES	6	8	6	5	2	2	5	-	-	2	1	1	1	39	19.50
mira051ITES	6	7	2	1	4	1	8	4	1	-	-	-	2	36	18.00
mira052ITES	5	7	2	1	4	1	8	4	1	-	-	-	2	35	17.50

Table 4.5: MIRACLE results by type on CLEF 2005

were definition questions that could be answered by expanding the acronym. In general, definition questions had simpler structure and contain a NE which was used to formulate the query. As far as the query retrieves relevant documents the question could be easily solved. On the other hand, errors in definitions are due to three causes: (1) ambiguity in NE, (2) using a different name variation for the question than that used in text, (3) excessive number of relevant documents and extraction of spurious phrases.

The focus our work was in Factual questions where the introduction of STILUS tools and resources provided a significant improvement over our CLEF 2004 results. Our performance on factual questions was very similar overall to the rest of the systems. STILUS tools leveraged a large dictionary of NE which was already quite complete for 2005. Moreover, the additional rules helped to improve the recall of the Answer Extraction phase significantly. Nevertheless, there was significant room for improvement as the number of answers judged as inexact was very large. They amount to a fifth of correct answers. In particular, our results were specially good for question with a PERSON EAT, but also moderate for LOCATIONS and ORGANIZATIONS. The list of person names was large in STILUS and also the automaton carefully engineered.

Finally, temporally restricted questions (T) were answered with exactly the same strategy than other factual questions. The performance of the system was in contrast much lower. Our post experiment analysis and the experiments carried for 2006 system tuning support the idea that much of the problem was due to including temporal restrictions in queries. Identifying temporal restrictions and treating them with a different strategy regarding query has been important to improve results for these type of question.

4.2.2.2 Error analysis

In our experience, errors in a QA system are difficult to be tracked down to a single cause, although this simplification is needed in order to focus research and development errors. For example, low precision in the Answer Extraction module will make the task of the Answer Selection module most difficult and therefore more prone to errors. Table 4.6 shows an estimation of the errors that could be tracked down to certain modules. Manual analysis of the question classification reveals that 80,50% were correctly classified regarding their QT and EAT. Most of these errors are due to the Question Focus of question with interrogatives 'Qué' and 'Cuál' not being part of the list resources. None of these errors would be easy to solve without a more comprehensive resource like Wordnet that maps could map names to EAT.

Table 4.7 presents the manual judgement of correct answers for run mira052ESES at different cuts from the ranked list of documents. The results for the analysis of the monolingual run indicate that only 115 of the 180 questions with an answer in the collection (20 questions have NIL answers) do find relevant documents that contains an answer. Then, even with a perfect Answer Selection step, only 63,89% of the questions would obtain a correct answer. Table 4.8 shows a similar analysis for the list of ranked answers for run mira052ESES which reveals a

Module	Error
hline Question analysis	25,98%
Document retrieval recall	20,81%
Answer extraction recall	11,83%
Answer ranking	40,84%

Table 4.6: Error analysis for the different stages for run mira052ESES

measure	ESES	ENES	ITES
Acc@20	52.22	44.44	43.89
Acc@40	63.89	55.56	56.11

Table 4.7: Questions with a correct answer at document retrieval rank, at this step both runs used the same modules

further leak of answers due to the selection, extraction and ranking of answers.

4.2.3 Cross-lingual experiments

Cross-lingual experiments were produced translating questions and submitting them to the Spanish monolingual QA system. As could be expected, accuracy was lower for cross-lingual runs with a loss between 5% and 7%. Additional errors were due to the translation process. The cross-lingual and monolingual question datasets were aligned that means that additional conclusions can be obtained by comparing the result for the original question. On the other hand, cross-lingual questions were manually translated from Spanish by the evaluation group. This generation process would probably lead to question formulations that are closer to Spanish than what a native English or Italian speaker would use. The most common error causes are:

- the order of the words in the question is altered, which cascades question analysis and question classification errors in the monolingual QA process.
- incorrect translation of Named Entities and Acronyms. Their incorrect translation degrades queries and therefore the performance of the document retrieval. Detecting NEs in the source language would avoid most of this errors for cognate NEs. For the rest of NEs, specific transliteration techniques would help to produce useful variants for translation.
- incorrect translation of other terms also affects the performance. In general, most of the lower performance in Document Retrieval as shown in Table 4.7 may be tracked back to translation errors.
- 'Qué' and 'Cuál' questions in Italian were also incorrectly translated.

On the other hand, we have detected that for some questions, the answer is found in one of the cross-lingual runs while the monolingual run fail to provide a correct answer, at least as a first choice. In these cases, we believe that the translation step chose a different lexical alternative which happened to be a good paraphrase or it had a positive interplay with the

Acc(n=X)	1	2	5	10	20	40
	23	31	40.5	44.5	47	49.5

Table 4.8: Questions with a correct answer at answer retrieval rank for run mira052ESES

#	ES	IT
F 0008	¿Con qué grupo ha cantado Robbie Williams?	¿Cantó Williams Granzas en cuál grupo?
F 0013	¿Cuándo nació Christopher Reeve?	¿Cuándo Christopher Reeve nació?
F 0014	¿En qué año se casó el Príncipe Carlos con Diana?	¿Se casó el príncipe Carlo Diana en qué año?
D 0026	¿Qué es la FIFA?	¿Qué es el MIEDO?
F 0125	¿Cuál es la misión principal de la sonda Ulises?	¿Cuál es Ulises la misión principal de la sonda?
#	ES	EN
F 0001	¿Qué es BMW?	¿Cuál es BMW?
F 0007	¿A cuánto asciende el premio para la ganadora de Wimbledon?	¿Cuánto está el premio para el ganador hembra en Wimbledon?
T 0011	¿Quién hizo el personaje de Superman antes de quedar paralizado?	¿Quién jugó el papel de Superhombre antes de que paralizarase?

Table 4.9: Some common errors in the translation of queries

rest of the steps. For example, for the question #98 *¿Qué grupo encabeza Franck Goddio?*, the Italian translation *¿Cuál grupo conduce Franck Goddio?* is useful to find the answer thanks to the use of a different verb.

4.3 CLEF 2006

The QA@CLEF task [Magnini et al., 2006] reached a widespread popularity in 2006 with an increasing number of groups taking part in the evaluation. New pilot tasks were also enacted which explored important QA questions, AVE (Answer Validation Exercise), WiQA (Exploratory QA in Wikipedia) and Real-Time QA. Our team participated in the main task and the two last pilot tasks.

The main task was slightly modified from previous years as it has been the norm to introduce new challenges for senior participants. Questions no longer were provided with their question type and the systems should infer it from the question text. Besides, new question types were introduced, a small number of list questions (10) as well as some new definition question subtypes, asking for definitions for objects and other NE. A selection of some of the questions is presented in Example 4.3. The expected output of the system was also modified. Systems were allowed to produce up to 10 answers per question. For each answer a supporting snippet was required, while previously a document ID was used.

The Real Time QA pilot task took place in Alicante in September 20 during the CLEF workshop. Evaluation guidelines for TREC and CLEF provided participants with a week to process questions. There are several motivations for this setting as the systems are just prototypes and could make use of complicated analysis techniques which will become affordable in the future. On the other hand, interactivity with QA systems is an important characteristic in succesful applications and a very important factor is time response. The evaluation procedure was modified in this task to account not only for the accuracy but also for the time used to answer questions [Llopis et al., 2009, 2007]. Our system had been already designed towards real-time and we worked on making robust an additional optimization technique.

On the other hand, the WiQA task was interested into a different kind of requests for QA and innovating also in the nature of the corpus. Wikipedia was used as a target collection for the first time in this task. Moreover, the task focused not only on questions but in providing novel and relevant information that should be included in a Wikipedia article. Our team participated in the

Example 4.3 Some examples of CLEF 2006 questions

- -	- -	- -	- - -
ES	ES	0001	¿Qué es el Atlantis?
- -	- -	- -	- - -
ES	ES	0005	¿Quién es Iosif Kobzon?
- -	- -	- -	- - -
ES	ES	0006	¿A qué país invadió Irak en 1990?
ES	ES	0007	¿Cuántos países forman la OTAN actualmente?
ES	ES	0008	¿Qué organización dirige Yaser Arafat?
- -	- -	- -	- - -
ES	ES	0010	Nombre una película en la que haya participado Kirk Douglas en el periodo de 1946 a 1960.
ES	ES	0011	Dé el nombre de alguien que haya ganado el Premio Nobel de Literatura entre 1945 y 1990.
- -	- -	- -	- - -
ES	ES	0012	¿Cuándo fue la coronación oficial de Isabel II?
ES	ES	0014	¿Cuándo murió Stalin?
- -	- -	- -	- - -
ES	ES	0016	¿Quién escribió la novela fantástica titulada "El señor de los anillos"?
- -	- -	- -	- - -
ES	ES	0024	¿Qué altura tiene el Kanchenjunga?
ES	ES	0025	¿Cómo se le llama también al Síndrome de Down?
- -	- -	- -	- - -
ES	ES	0033	¿Qué es la quinua?
ES	ES	0034	¿Qué empresa se hizo cargo de Barings después de su quiebra en Febrero de 1995?
ES	ES	0040	¿Cuál es la palabra alemana más larga?
- -	- -	- -	- - -
ES	ES	0053	¿Qué es Lufthansa?
- -	- -	- -	- - -
ES	ES	0063	¿Quiénes fueron los cosacos?
- -	- -	- -	- - -
ES	ES	0130	¿Qué organismo presidía Primo Nebiolo durante los Campeonatos del Mundo de atletismo de Gotemburgo?
- -	- -	- -	- - -
ES	ES	0142	¿Qué países forman parte del Tratado de Libre Comercio de América del Norte?
ES	ES	0143	Nombre los tres Beatles que siguen vivos.
- -	- -	- -	- - -
ES	ES	0153	¿Cuántas veces ha ganado Zinedine Zidane el US Open?
- -	- -	- -	- - -

organization for Spanish and we proposed a system that reused a great number of components of the main QA system. The system and the results are not described here for clarity but they can be found in [de Pablo-Sánchez et al., 2006c].

4.3.1 Runs description

In our third participation in the CLEF@QA task, the MIRACLE group submitted two runs for the Spanish monolingual subtask. Runs differ in the way they considered Named Entities and

Run	R	X	U	*Acc*@1(F)	*Acc*@1(T)	*Acc*@1(D)	*Acc*@1(L)	*Acc*@1
mira061ESES	37	3	6	**21.30**	15.00	16.67	10.00	18.50
mira062ESES	**41**	4	7	**21.30**	**17.50**	**23.81**	10.00	**20.50**

Table 4.10: Results for CLEF 2006 (by question type)

other complex expressions in passage selection and ranking processes. The system was adapted at a certain extent to new requirements of 2006 guidelines for the QA task:

- Question type is inferred from the question.
- List questions are treated as factual questions and several answers are returned.
- A method to select representative support snippets (sentences) and not just documents based on the scoring of sentences.

Other significant changes on the system are:

- The query generation strategies were refined and the method to use complex NE queries described in Subsection 3.5.2.3 was implemented. These changes were aimed at improving the recall of the system.
- Time Normalization was improved and allowed for better treatment of temporal questions and questions with temporal restrictions.
- Answer Merging was improved using heuristics to remove stopwords and conflate similar answer candidates. Time normalization also allowed to merge this particular EAT.
- The structure for Answer Ranking, which clearly separates local factor from global factors, was introduced also this year.

Two runs were submitted to the Spanish monolingual task. They differed in the way that use multiwords or complex terms. Complex terms include multiword units such as most NE, TE and numerical expressions. As we indexed the EFE collection using simple terms it was not clear which strategy could work best for selecting and scoring relevant documents and sentences. Both runs decompose complex proper names in a query into simple terms joined with the operator AND_MAYBE to retrieve documents. The difference between runs lies on the method used for sentence selection and ranking. Run mira061ESES selected and scored sentences taking into account the whole multiword. This strategy selects those sentences in which the expression is the same that appears in the question and it is clearly oriented to favour high precision. In contrast, run mira062ESES used individual terms that compose the term. This run would favour recall because it could match different common forms of referring the same entity for example, but it is also problematic as it could select noisy sentences that refer to other entities.

4.3.2 Results

Table 4.10 presents the official results for the two monolingual Spanish runs that were submitted. General performance for Factual and Definition questions were quite similar for both runs while it seems that mira062ESES obtained better accuracy score and CWS (Table 4.11). It seems that the precision oriented run (mira061ESES) ranked lower some useful sentences and answers. This is also supported by the higher number of NIL responses in this run.

The analysis by question type reveals that our efforts to improve the treatment of temporal expression were succesful, as the performance for these questions is closer to factual than the previous years. This is also supported by the fact that the system obtains good scores for

Run	*Acc*@1	NIL-F	r
mira061ESES	18.50	0.34	0.136
mira062ESES	**20.50**	**0.35**	**0.145**

Table 4.11: Global Results for CLEF 2006

			Time(s)	
Run	N_d	MRR	on-line	off-line
official run 1	100	0.41	549(496)	156
official run 2	40	0.33	198(182)	73
unofficial run 3	10	0.30	48	14

Table 4.12: Evaluation results for MIRACLE Real Time QA runs

the particular class of F-TIME questions. On the other hand, much work remained to do on List questions. Although they have been identified in the Question Analysis phase, no special processing has been implemented and they were treated as factoids. One of the problems that remains to be solved is to find appropiate scoring functions. It seems that correct answers and clearly incorrect answers do obtain very similar scores. Distance and term weight factors need to be reconsidered despite our fruitless experience in the previous year.

Comparing our results with last year system performance, the system obtained lower scores along all question types. In our opinion, this effect is a consequence of the increasing difficulty of questions.

4.3.2.1 CLEF 2006 Real-time QA

It is widely acknowledged that some of the interesting scenarios for a QA system impose time constraints that limit the processing and resources a system could use to answer a question. The Real-Time QA at CLEF 2006 explored the trade off between answer accuracy and processing time. Evaluation proceeds on-site and systems were asked to answer a subset of 20 questions in monolingual Spanish. The overall time taken to answer the questions was also recorded in wall clock time.

We submitted two runs with the configuration used for mira062ESES, using on-line document processing. The full set of retrieved documents are processed with STILUS and the EAT filters for Answer Selection when the answer is issued. Therefore, it was natural to test the effect of the number of documents retrieved N_d in the overall accuracy. The first run used $N_d = 100$ documents while the second run used only $N_d = 40$ documents. After the workshop we have performed additional experiments using $N_d = 10$ documents. We also measured the time more accurately in the same question set. Finally, the system was fixed to use a preprocessed STILUS collection instead of the original textual document. NE detection and filtering are still performed on-line. The system was formed by a mixture of components of Java and C++ (Xapian, STILUS) and run in a Linux laptop with a 1,7GHz and 1GB of memory. Overall results are outlined in Table 4.12.

Our system performed very well at this task, and it was ranked first with in MRR and other metrics used to compare systems [Llopis et al., 2009]. It was surprising that our results were above those usually obtained at the main task. The small sample of questions used in these experiments does not allow to make any claim, but our impression was that the testset contained longer questions than usual that our system was able to answer better.

Nevertheless, the most important result is that Real-Time QA could be achieved with a simple architecture like the one presented. Although not a realistic architecture for moderate size and large collections, in the case of corpus pre-processing the system could answer in less

Error type	%
Question classification	7,79
Question analysis	18,18
Document retrieval	38,96
Sentence selection	3,89
Answer extraction	11,03
Answer ranking	20,12

Table 4.13: Error analysis for run mira061

Error type	2005	2006
Questions classification & analysis	25.98	25.97
Document retrieval	20.81	38.96
Sentence & answer extraction	11.83	14.92
Answer ranking	40.84	20.12

Table 4.14: Error comparison between 2005 and 2006 runs

than a second per question if using only the first 10 documents. The time for processing a question is reduced in a factor of three or four. Obvious improvements to the architecture require processing not documents but to work in a sentence by sentence basis. Research on specific approaches of IR for QA is needed to maintain accuracy scores in RT-QA not only aiming at obtaining high recall but also high precision. That would allow to reduce the time processing requirements that almost all NLP module require.

4.3.3 Error Analysis

With respect to our 2005 error classification, we added two new types this year to better understand the treatment of the questions and answers in our system. The estimation of errors for run MIRA061ESES are shown in the Table 4.14. In the first column, we show the percentage of error types with respect to the total of errors considering Wrong, ineXact and Unsupported types (in the case of MIRA061ESES, 154 questions).

Comparing the error classification in both years (see Table 4.14), we can observe significant differences in the approaches. While question analysis and answer extraction (although with addings and modifications) have obtained basically the same results in both years; in document retrieval and answer ranking the results have been reversed. A possible explanation of bad results in document retrieval is the increasing difficulty of the questions. We should advise that the methods for the analysis of errors between the two year differ. In 2005, internal logs were analyzed while in 2006 only the final output, including the first 10 questions were used.

4.4 CLEF 2007

The 2007 edition of CLEF@QA consolidated the evaluation model followed in past editions but also introduced new challenges in several aspects [Giampiccolo et al., 2007]. One of the changes was aimed at doing the task more interactive and considered the problem of answering several questions about a related topic. Some examples of a topic-related question group are presented in Example 4.4. The topic of the information seeking dialogue is introduced in the first question and the system should identify it which is an additional complication compared to a similar task evaluated at TREC. The topic could be directly stated in the question or it can be also the first answer, therefore only indirectly referred in the question. An example of the first case is the

Figure 4.1: A document example from Wikipedia

group 2011, where the question states the topic *Hermann Emil Fischer*. The second scenario is represented in group 2000 and, in this case, the topic is the correct answer, *Colegio Hogwarts de Magia y Hechicería*. Although the guidelines state that the groups of questions will have only one topic, a small number of groups exhibit topic shift (Eg. group 2011).

Example 4.4 An example of topic related questions in CLEF 2007

Source	Target	Question ID	Group ID	Question text
ES	ES	0001	2000	¿En qué colegio estudia Harry Potter?
ES	ES	0002	2000	¿Cuál es el lema del colegio?
ES	ES	0003	2000	¿En qué casas está dividido?
ES	ES	0004	2000	¿Quién es el director del colegio?
–	–	- - -	- - -	–
ES	ES	0020	2011	¿Quién fue Hermann Emil Fischer?
ES	ES	0021	2011	¿Qué premio recibió en 1902?
ES	ES	0022	2011	¿Quién recibió el Premio Nobel de Literatura ese año?
–	–	- - -	- - -	–

The second modification of the task has been the introduction of new target collections. For each of the target languages a new collection based on a dump of Wikipedia dated November 2006 was used. The collections were offered in two formats, in XML and HTML, and only actual page entries were considered as a source of answers. Additional page data including templates, discussion, category pages and revisions were discarded. Examples of Wikipedia pages and their formatting, as used in the Wikipedia XML corpus used in the evaluation, are presented too.

Example 4.5 The XML encoding used in the Wikipedia XML corpus

```
<?xml version="1.0" encoding="UTF-8"?>
<html xmlns="http://www.w3.org/1999/xhtml"
xmlns:wx="http://ilps.science.uva.nl/WikiXML/wx" xml:lang="es" lang="es">
<head>
<title>Teodoro Obiang</title>
<meta name="wx_namespace" content="0"/>
<meta name="wx_pagename" content="Teodoro_Obiang"/>
<meta name="wx_page_id" content="54308"/>
</head>
<body>
<div id="wx_article">
<wx:section level="1" title="Teodoro Obiang" id="wxsec1">

<h1 class="pagetitle" id="wx1">Teodoro Obiang</h1>

<div class="wx_image" wx:align="right" wx:thumb="thumb" id="wx2">

<a href="/wiki/Imagen:T%C3%A9odoro_Obiang_Nguema_Mbasogo.jpg"
title="Teodoro Obiang Nguema Mbasogo" wx:linktype="image"
wx:pagename="Imagen:Téodoro_Obiang_Nguema_Mbasogo.jpg" id="wx3">

<img src="/wiki/Imagen:T%C3%A9odoro_Obiang_Nguema_Mbasogo.jpg"
alt="Teodoro Obiang Nguema Mbasogo" width="300" id="wx4"/></a>

<div class="thumbcaption" id="wx5">Teodoro Obiang Nguema Mbasogo</div>
</div>

<p id="wx6"><b id="wx7">Teodoro Obiang Nguema Mbasogo</b>
(nacido el <a href="/wiki/5_de_junio" title="5 de junio"
wx:linktype="known" wx:pagename="5_de_junio"
wx:page_id="7240" id="wx8">5 de junio</a> de <a href="/wiki/1942"
title="1942" wx:linktype="known" wx:pagename="1942" wx:page_id="9974"
id="wx9">1942</a>) ha sido el dictador de [...]

<p id="wx13">Nacido en el seno del clan <a href="/wiki/Mongomo"
title="Mongomo" wx:linktype="known" wx:pagename="Mongomo"
wx:page_id="315369" id="wx14">Mongomo</a> en el pueblo de
<a href="/wiki/Acoacan" class="new" title="Acoacan" wx:linktype="unknown"
wx:pagename="Acoacan" id="wx15">Acoacan</a> an la actual frontera con Gabón,
Obiang comenzó su carrera en una academia militar [...]

</wx:section></div>
<div id="wx_categorylinks">
<a href="/wes/index.php?title=Especial:Categories&amp;article=Teodoro_Obiang"
title="Especial:Categories" wx:linktype="known"
wx:pagename="Especial:Categories" id="wx44">Categorías</a>:
[...]
</div>
<div id="wx_languagelinks">
[...]
</div>
</body>
</html>
```

4.4.1 System description

The system architecture used in 2007 is depicted in Figure 4.2. It is composed of two streams. The first stream uses the EFE newswire collection as a source of answers while the second uses Wikipedia. Each stream produces a ranked list of answers that are merged and combined by the Answer Source Mixer component described below. The two QA streams share a similar basic pipeline architecture and work as an independent QA system, with different configuration parameters, collections, etc. Question Analysis is shared by the two streams as the processing is common. The module for managing context and anaphora resolution in topic-related question series was developed for this year and is also shared by the two streams.

Several major changes were introduced in 2007 in the base QA system. For example, the Document Retrieval module which was based on Xapian was replaced by Lucene. The change was motivated because Lucene was easier to deploy across architectures and simplified overall system management as it was developed in Java. The linguistic analysis provided by STILUS was also improved by including a feature that pruned POS tags based on context and reduced tag ambiguity. Besides, the Collection Processing module that was developed for Real-Time QA experiments was finally consolidated.

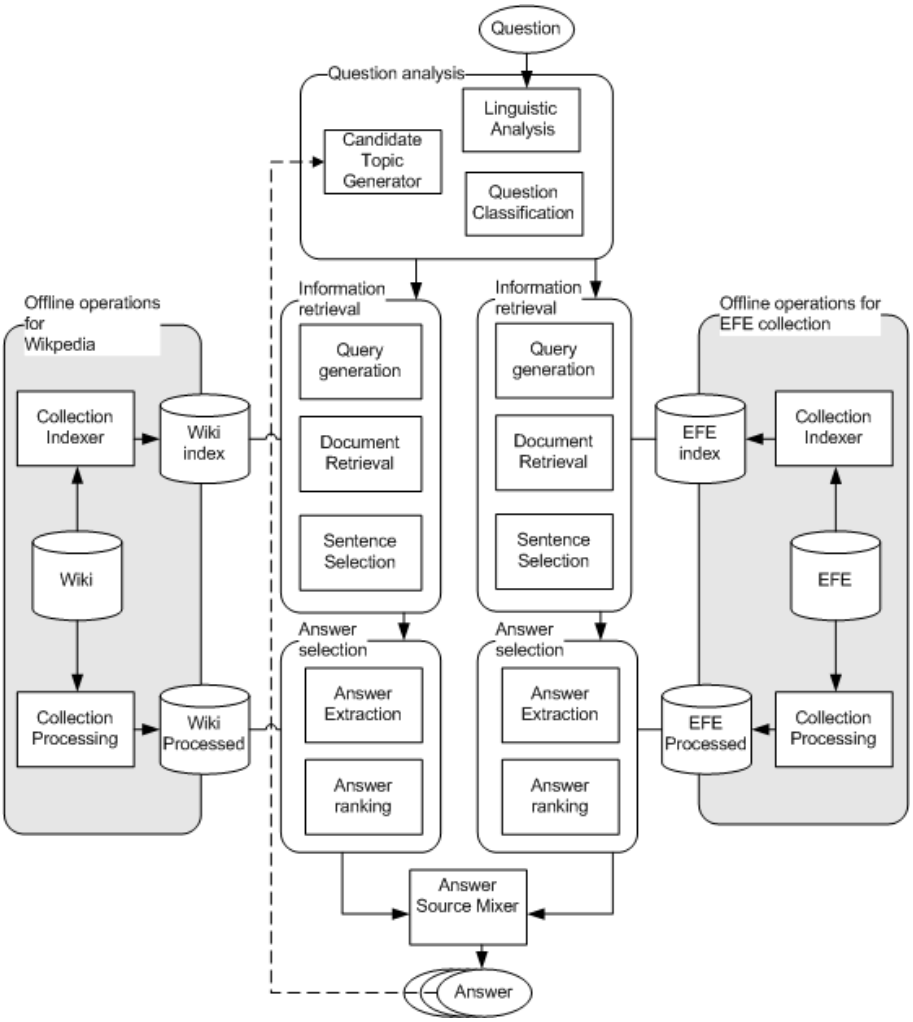

Figure 4.2: MIRACLE 2007 System architecture

4.4.2 Combining EFE and Wikipedia answers

As already mentioned, the setting of the task allowed for finding the right answer to a question in two different collections. Questions could be targeted to the Wikipedia collection, to the EFE news collection or both. A crucial difference is that the two collections have very different timespans, EFE reports events for years up to 1994 and 1995 while Wikipedia was updated to 2006. This allowed us to to test an automatic method to choose which of these sources should be more relevant to extract candidate answers to a given question which was implemented in the Source Mixer component. This module is based on the following heuristics:

- If the verb of the question appears in present tense then preference is given to answers appearing in the Wikipedia collection.
- If the verb is in past tense and the question makes reference to the period covered by the EFE news collection, i.e., 1994 or 1995 years appearing in the question, then preference is given to answers from the EFE news collection.

The preference given to each answer was measured by two parameters, one referring to the verb tense factor and the other referring to the time period factor. In this way, no answer is really dropped from the candidates list but the list is reordered according to this clues. At the output of the Source Mixer component there is a list of candidates ordered according to their source and the information that is present in the question.

4.4.3 Run description and Results

Using the system described above we submitted one monolingual run for the Spanish subtask. Evaluation results and combined measures are outlined in Table 4.15.

Name	R	X	U	*Acc*@1(F)	*Acc*@1(T)	*Acc*@1(D)	*Acc*@1
mira071ESES	30	4	8	18.35	13.95	3.13	**15.00**

Table 4.15: Evaluation results for CLEF 2007

Results were disappointing in general, as they were lower than previous years for almost all types, despite the inclusion of new sources like Wikipedia. Even though almost all modules have been improved, the overall Accuracy remained below 15%. Factual questions with well defined Named Entities achieve a reasonable accuracy and results are comparable to other years. On the other hand, questions with EAT OTHER are much harder and the accuracy dropped. If we ignore question series we obtained an accuracy around 18% for the rest of the questions which is lower but comparable to previous years.

Most of the problems were found for definitional questions where accuracy dropped dramatically. The heuristics that we have defined and tested to extract definitions in EFE did not work for Wikipedia. The system used to signal appositions, nominal phrases before Named Entities and expansion of acronyms as valid definitions for persons and organizations. In contrast, definitional sentences in Wikipedia usually are copulative. There is also a longer distance between the defined object and their definition. Moreover, they are usually placed at the beginning of the document which is a very strong heuristic in this kind of text. Unfortunately, we did not implement any special strategy for these questions. Similar problems with adapting Answer Extraction to the new kind of texts could be the source of errors for other types too.

In a more thorough analysis of the errors we found that the number of document results that contained an answer have dropped. The documents returned were correct in 44% of the results, which is a lower bound to measure the performance of the document retrieval system. Additional errors in Sentence Selection and Answer Extraction cascade to provide the final results. It seems

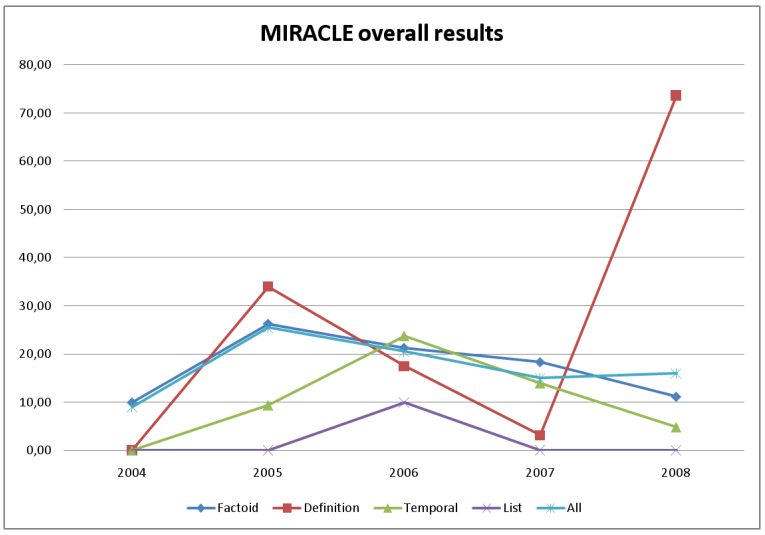

Figure 4.3: Outline of MIRACLE results in CLEF

like the introduction of several collections, in particular Wikipedia, make the IR task harder. For example, a smaller number of document results were retrieved from each collection and they should be fused. A small percentage of the errors could be attributed to the change of IR engine.

A further complication of the 2007 task were questions related by topic. If one of the first questions in the topic is not answered correctly, errors will probably chained and the following questions become harder. There were 20 topic related groups of questions that made a total of 50 questions. The system correctly answered 5 questions of this kind, from three different groups. We analyzed the main source of errors in order to evaluate the co-reference component. Rules for topic detection are the source of errors only for three of the cases, and therefore seems to work reasonably well although there is room for some simple improvements. In one of these cases, the error is due to a question not correctly identified as a NUMEX type. The rest of the errors are due to incorrect identification of the first answer and the subsequent chaining of errors in the question group.

4.5 Overview and analysis of the evaluation results

To conclude we have gather data from our participation along the years in CLEF in Figure 4.3. We include additional results for analysis on the CLEF 2008 task which was similar in scope to 2007, though it included a larger number of topic related question groups. In contrast, we have not describe our CLEF 2008 submission as part of this thesis because the author was involved only in certain parts, like the topic tracking and the co-reference module. Additional information can be found in Martínez-González et al. [2009].

A first look at the graph seems to reveal that there has not been a real quantitative improvement on the performance of the QA system. Our participation in 2005 achieve the best overall results and since then results have slightly going down. Factual questions, the core of the CLEF evaluation, show a similar trend which is also followed for temporally restricted questions with a lag. On the other hand, while doing development we have often confirmed improvements on previous years datasets. This improvements were due to the enrichment of resources but also for the indivual techniques. In fact, previous results have been perused during each year development cycle and it is in part why we do just report official evaluation results.

We believe that the CLEF question sets have become more difficult along the years. Questions following specific templates have been reduced and a wider range of linguistic variance has

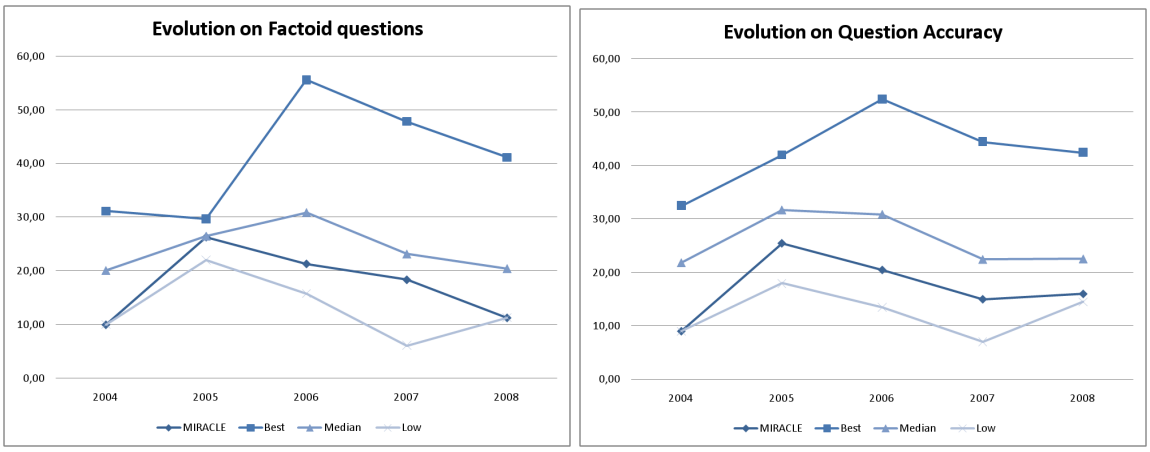

Figure 4.4: Evolution in CLEF and comparative

been explored. It is difficult to support this opinion in quantitative figures because we lack good models to measure question difficulty. In contrast, it has been accepted that the complexity of the QA task has indeed increased a lot. Figure 4.4 shows the evolution of the best, median and lower systems and also MIRACLE's. The general seems to support this conclusion. The introduction of Wikipedia and topic related question groups in 2007 has been a great challenge for all systems as marked with the drop in results. Except for the best performing system, Priberam, that entered in 2006, the median performance has experience just slight improvements from campaign to campaign.

Our results over the years have been below the median performance despite the effort inverted in the system between participation. This effort has been divided between the core task and the adaptation to some of the new challenges which often were required to participate. Certainly, the quantitative results have not been brilliant but fortunately there have been other research outcomes. The system and the architecture has matured along the years and it has become a platform for further work. Our work have also arisen other questions related to the integration of language resources and multilingual applications.

4.6 Conclusions

The approach of the MIRACLE system can be categorized into those that combine IR and IE techniques. This approach has proven useful for Spanish as well as other languages which have been adressed in TREC and CLEF. Several other systems have deployed systems based on similar architectures in the Spanish monolingual track at CLEF. Priberam, the best performer in the track through the years, has been using a similar architecture too which was ported from Portuguese. It has been also effective for answering questions in real time as far as the cost of pre-processing the collection can be assumed. On the other hand, the overall approach has an important limitation. Building a QA system under this approach is a knowledge intensive task. A great amount of effort has to be devoted to develop knowledge resources, linguistic resources and rules for several of the modules. For example, part of the success of our system, and we believe also Priberam's and others, is due to being able to recognize a large number of NE types with certain confidence. Extending the NE types to a full fledged taxonomy like Sekine's would be keystone to QA applications.

Our system make use of relatively shallow linguistic information which has limited us to chose a representation that is based on terms and NE. Advanced and deep language processing has been used in other languages, for example in English, Dutch and German. Clever use of the available tools have reported improvements in Answer Extraction, Answer Validation and

Passage Retrieval among others. On the other hand, their computational cost is still high. Deep techniques have also been limited by low coverage and this teams have backed up with shallower techniques. In our opinion, it is clear that advanced natural language understanding is required for the goal of question answering. However there are important problems at the practical level yet.

One of the problems of QA systems has popped up explicitly during the CLEF evaluations, namely domain overfitting. The introduction of the Wikipedia collection has revealed that a large number of techniques used in QA were too adapted for the news domain. These *tricks* are probably hidden at all levels of the QA architecture. For example, retrieval strategies are dependent on the source document collection and structure. Besides, the recent RespubliQA experiment using a legal collection has revealed the need to answer new questions Forner et al. [2009]. Expanding the hierarchy of NE types is therefore required for specialized domains as part of domain adaptivity.

A very different problem that we have experienced with QA is that a significant improvement in one of the components, does not always translate in similar improvement at the system level. Despite the modularization, there are important dependencies between modules that are not well understood. Moreover, the integration of different tools, specially linguistic analysis tools, have been an important engineering challenge. Though we have describe a Spanish system, a English system has been developed in parallel. We have attempted to integrate analysis from different tools to use the same QA flow with moderate success. We believe that simpler QA methods are required in order to build multilingual QA systems. Otherwise, the increasing number of tools with disparate capabilities, formalisms and formats building would transform building a multilingual system a daunting task.

On the rest of this thesis we explore a different solution for some of these problems. We believe that the solution tries to overcome some of the problems that appear when applying QA to different domains. It is also suitable for multilingual QA systems as it provides a simple model which requires few linguistic analysis. We assume that a large collection of documents is available, otherwise there would be no interest in performing QA. We also assume that a large number of questions can be mapped to requests about a reduced number of types of objects like domain specific NE. We also assume that a large number of questions can be expressed as relations between these reduced set of NEs. This is a proto-ontology of the domain which contain several examples for the different concepts and relations. The QA system is built by combining methods for offline population of the ontology with Knowledge Mining as a backup strategy. The offline population will explore the use of bootstrapping methods to acquire large lists of concepts and relations. Besides indicative patterns that help to identify the concepts and the relations will also be acquired.

Chapter 5

Information Extraction: Semi-supervised and Large Scale approaches

5.1 Information Extraction

The objective of an Information Extraction (IE) system is to identify relevant information in running text and characterize it by adding it a semantic interpretation. Extracted information is structured using a formal model to enhance its organization and access for final users, for instance, by using a relational database. The relevant information is usually defined a priori in the form of templates and stored using formalisms like relational databases or ontologies. Therefore, the whole process is tightly associated to the domain and the application. Besides, the source, style and modality of the text are also known in advance. All these elements configure what has been known as an *extraction scenario*. Popular extraction scenarios have spun off from the Message Understanding Conferences (MUC) [Chinchor, 1997] that focused on extracting information about *terrorist attacks* or *business acquisitions* from news corpora. Other research scenarios have dealt with semi-structured text like the *workshop announcements* scenario in the PASCAL IE Challenge [Ireson et al., 2005]. This challenge consisted on extracting facts that characterize a workshop or conference like the title, acronym, organizers or dates [1].

The architecture of an archetypal IE system is depicted in Figure 5.1. IE practitioners have identified a set of modules that are useful as architectural abstractions across different domains. They are among others: Name Entity Recognition and Classification (NERC), Relation Detec-

[1] http://pascallin.ecs.soton.ac.uk/Challenges/EIRD/

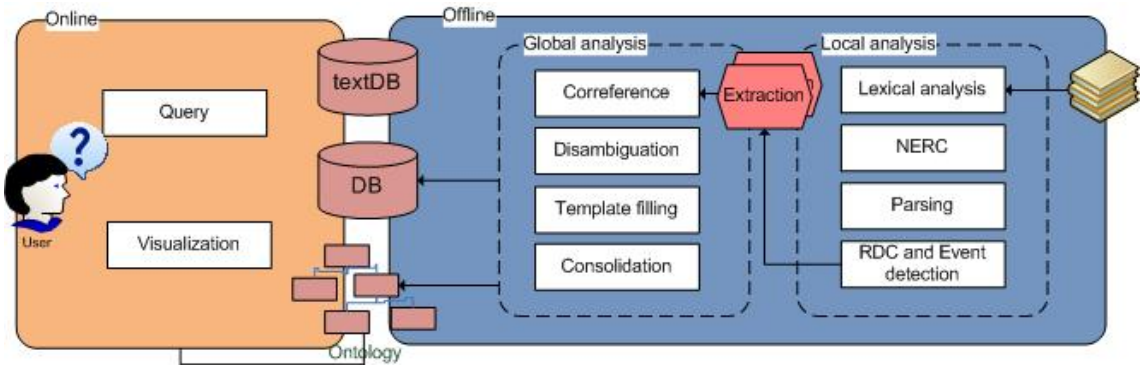

Figure 5.1: The architecture of an Information Extraction system

tion and Characterization (RDC) , Event Detection and Entity Co-reference inside documents and across documents. IE components are also used in other related applications like Summarization, Knowledge Acquisition, QA or Semantic Search Engines. The usefulness of generic modules have made research competitions to transition to the evaluation of individual components. Automatic Content Extraction (ACE) evaluations promoted by NIST [2007, 2008] have organized different tasks to evaluate progress in each of the mentioned modules. Other evaluations, like Conference On Natural Language Learning (CONLL) shared tasks on 2002 and 2003 [Sang, 2002; Sang and Meulder, 2003] have dealt with the recognition of NE in a multilingual context.

Multilinguality has been recognized as a challenge by both evaluation forums because they have considered other languages than English. ACE considered Arabic and Chinese while CONLL provided corpora for Spanish, German and Dutch. Portuguese and Italian have been also received their attention by the local IE community with the HAREM [Mota and Santos, 2008] and EVALITA [Speranza, 2007] evaluations. Besides, ACE has also experimented with other genres beyond newspapers. For example, the English evaluation in 2007 included blogs and speech transcripts.

Then, since the late nineties, the need to adapt IE systems to different languages, text genres, domains and extraction scenarios have triggered an interest into techniques that are capable to adapt specific modules [Turmo et al., 2006]. There are a number of tools and toolkits that help with the engineering process of building and adapting IE systems and components like Proteus [Grishman and Yangarber, 1997], GATE [Cunningham et al., 2008, 2002] or more recently UIMA [Ferrucci and Lally, 2004]. These frameworks provide rule languages or tools that help out to define, debug and evaluate linguistic analyzers. Some of them also assist with the annotation process which is a neccesary step to build systems and evaluate larger architectures. Different approaches have been used to support the creation of IE components [Turmo et al., 2006] and in fact they are often combined in full IE systems.

5.2 Approaches to build Information Extraction Systems

5.2.1 Knowledge engineering

In this approach, a human expert defines, debugs and mantains a set of rules that are used in the different modules of the IE system. Ideally, the expert team must be familiarized with:

(a) the domain of the application

(b) the languages used in source texts

(c) the formalism, languages and tools that are used to specify knowledge like dictionaries, ontologies and rules.

The whole process is usually very expensive and it is difficult to build teams with the desired competences. Nevertheless, best results are usually obtained with this approach after careful rule engineering and debugging processes. In contrast, the cost of adapting a system to a new domain is high. Other problems include how difficult is to maintain this knowledge along time and its re-usability across applications. In research systems, the knowledge engineering approach has been used in classic IE systems like FASTUS [Appelt et al., 1993].

The creation of formal resources like dictionaries, gazetteers, lexical databases like Wordnet [Fellbaum, 1998] and ontologies of different types is in fact another kind of knowledge engineering task. All these resources find their application in a variety of IE subsystems [Cunningham et al., 2008]. Manual and semi-automated techniques are used to generate these resources including the use of lower level IE and NLP techniques.

5.2.2 Supervised Machine Learning

Some of the sub-problems in IE can be casted as a classification task such as NE or Relation Classification. Both involve assigning a label or category to a chunk of text. Even recognition problems can be modelled like a sequential classification task if we model text as a sequence of classifications for each token. For instance, NE Recognition has to assign tags BEGIN, INSIDE, OUTSIDE (BIO tags) or a similar scheme [Sang and Veenstra, 1999] to text tokens. A sequence of tokens tagged like BEGIN or INSIDE would mark a valid Named Entity. When supervised Machine Learning is used in IE, an expert should annotate the desired information in a set of documents and create an annotated corpus for the task. That would serve to obtain the training, development and test set to use for the learning method. This approach has been used with success for tasks like NERC [Bikel et al., 1999; Sang, 2002], RDC [Zelenko et al., 2003] and Coreference Resolution [Ng, 2005; Ng and Cardie, 2002]. The main differential characteristics of machine learning NLP and IE components with respect to other traditional machine learning problems lie on the importance of structure (sequences or trees) and the large number of symbolic features.

The former often implies that problems benefit from using learning algorithms that make use of structure and global information like Hidden Markov Models [Bikel et al., 1999], Conditional Random Fields [Lafferty et al., 2001; McCallum and Li, 2003] and Max-Margin Networks [Taskar et al., 2003]. The latter usually entails that to obtain good results a large number of examples (documents or sentences) must be annotated. Moreover, the annotation process must proceed with a carefully designed methodology and include the work of domain and linguistics experts too. As a consequence, the cost of annotating resources amounts for a great part of the budget of a project. Several alternatives have been researched in order to relieve the annotation problem. Among the alternatives we can find the combination of annotated and unannotated corpora using techniques like Semi-supervised learning, Bootstrapping and Active Learning that are discussed in section 5.3.

5.2.3 Unsupervised Machine Learning

Unsupervised methods have also found widespread use in IE and Text Mining (TM) systems. They are often coupled with advanced visualization techniques [Feldman and Sanger, 2007]. The use of clustering is common as an exploration step in lots of TM applications. Besides, techniques for terminology identification are frequently unsupervised and rely on general statistics [Frantzi and Ananiadou, 1999]. Unsupervised methods could be used to assist end-user in collection navigation or domain experts in the creation of domain ontologies. A known issue with unsupervised methods is that they are designed to capture the structure of an object, but when they are used in large features spaces without knowledge bias, they can learn any kind of structure. For instance, a document clustering can reveal topical distribution or authorship distribution when different attributes are selected. Then an expert would be required to select a wiseful representation of the data to learn. In contrast, semi-supervised methods could help providing some basic structure that guide the learning process by specifying a reduced number of examples.

5.2.3.1 Hybrid methods

Most complete systems use in fact a combination of the above methods to produce a full IE system. The most basic form of hybridization is in fact a practical view. Simple tasks like tokenization, sentence extraction or the recognition of certain classes of entities can achieve high accuracy with simple methods or by reusing widely used tools. Other tasks would benefit from the effort in tagging a training corpus and use it for supervised approaches. Machine learning methods often integrate the use of general and domain specific dictionaries and several works have studied their integration, for example for NERC [Cohen and Sarawagi, 2004]. The Web [Turney

et al., 2006; Wang and Cohen, 2007] and large textual collections like Wikipedia [Kazama and Torisawa, 2007] have been shown useful to acquire such dictionaries using methods that exploit semi-structured and structured text. Semi-supervised pattern acquisition approaches to IE were devised as complementary methods to rule engineering since the beginning. For example, Riloff [1996] filters automatically generated patterns in order to acquire rules for a rule-based system. A different approach has been used in TEG [Rosenfeld et al., 2004] that retrains a Stochastic Context Free Grammar (SCFG) that has been manually specified. In fact, a similar approach is behind the knowledge bias that has been specified in the structure of Hidden Markov Models for NERC [Bikel et al., 1999].

5.3 Reducing annotation work for Information Extraction

Applying Machine Learning to real world problems often implies that human experts should label data for the algorithm to induce useful models. This is the case for most tasks that attempt to learn linguistic structure including IE tasks. Using supervised learning in IE requires several domain experts that define the interesting information, read texts and annotate the desired information. Good results could require a large number of observations which would have great impact on costs. In contrast, unannotated text is easy and cheap to obtain, in particular if a IE solution is demanded. This data can be used to improve the results of an already existing IE module or to reduce the amount of data that is needed to obtain a reasonable performance for a new module. Techniques that combine labeled and unlabeled data are known as semi-supervised in Machine Learning. A complementary alternative consists on modifying the annotation task itself and make it easier for the human annotator. This is the logic behind other approaches like Active Learning, PU Learning and Multi-Instance Learning.

5.3.1 Semi-Supervised Learning

Semi-Supervised Learning (SSL) is a new framework for machine learning that combines the use of labeled and unlabeled data. A simple SSL classification task starts with a set of labeled data points L and unlabeled data points U and the objective is to obtain a function f that predicts the values of the variable Y based on observed values, X. The objective is to find a good predicting function that minimizes the generalized error (ϵ) between the predicted value and the actual value given a cost function $D(x)$.

$$\begin{array}{l} L = (X_l, Y_l) = (x_1, y_1), \ldots, (x_l, y_l) \\ U = X_u = x_{l+1}, \ldots, x_{l+n} \quad \text{with } l << n \\ f : X \rightarrow \hat{Y} \quad \text{such as } arg\,min(\epsilon) \\ \epsilon = \sum_{x \in \Omega} [[\hat{Y}(x) - Y(x)]] D(x) \end{array} \tag{5.1}$$

Unlabeled data in semi-supervised methods helps to find good decision boundaries by avoiding high-dense regions. When dealing with sparse features sets like those founds in NLP or IE this is an important characteristic. Moreover unlabeled data helps to improve features based on vocabularies too. We comment briefly some of the main approaches that have been suggested to achieve the goal of semisupervised learning and discuss their application to tasks dealing with language understanding. A comprehensive view of this subject can be found in Zhu [2007] and [Chapelle et al., 2006] while Abney [2007] presents it from a computational linguistics perspective.

- **Generative models** and in particular, **Expectation Maximization** [Dempster et al., 1977], are probably one of the oldest procedures that has been used for semi-supervised learning. The joint distribution of the data given a model θ is estimated as

$$P(X, Y \mid \theta) = P(Y)p(X \mid Y, \theta)$$

and the unlabeled data is used to iteratively estimate the distribution of labels for unlabeled data. Variants of Expectation Maximization, like the Baum-Welch algorithm, were used in sequential classification tasks like POS-tagging [Elworthy, 1994]. Under the correct independence and identifiability assumptions it has been shown that unlabeled data can help, though it has also been shown that when these assumptions are violated the performance could not improve over supervised results [Elworthy, 1994; Merialdo, 1994].

Algorithm 1 Expectation Maximization algorithm

$ExpectationMaximization(D) \rightarrow \theta$
Input: observed data $D = (X_l, Y_l, X_u)$ and assumes hidden data $H = Y_u$
Output: a model θ that maximizes $p(D \mid \theta) = \sum_H p(D, H \mid \theta)$

Starts with an arbitrary θ_0
while stopping criterion is not met **do**
  E-step: estimates prob. of hidden data $q(H) = p(H \mid D, \theta)$
  M-step: maximizes prob. of observed data $\sum_H q(H) \log p(D, H \mid \theta)$
end while

- **Self-learning** is a semi-supervised wrapper method where a base supervised classifier is trained with the available labeled data. Unlabeled data is labeled using the base classifier and the confidence of each label is also recorded. A subset of the most confident data points in the unlabeled data is added to the training data and the base classifier is retrained. The process is repeated until the stopping criteria is met, usually when no improvement is seen in a small portion of held out data.

Algorithm 2 Self training algorithm

$Self - train(L_0, U) \rightarrow f$
Input: initial labeled data L_0 and unlabeled U
Output: a model f that assigns a confidence value

$f \leftarrow train(L_0)$
while stopping criterion is not met **do**
  $L \leftarrow L_0 + select(label(U, f))$
  $f \leftarrow train(L)$
end while

- **Co-training** is characterized by the use of two base classifiers that are trained in a independent set of features [Blum and Mitchell, 1998]. The process is similar to self-training but the criteria for adding unlabeled data to the training set is based on the agreement between classifiers. Then, agreement is the objective measure to minimize during the learning process. The independence of feature sets is a hard constraint in many practical tasks and is not reuiqred in simpler forms of multi-view learning.

- **Regularization** approaches try to constraint and select the best model under the assumption that a good decision boundary should not be in areas where high density of data points exists. Unlabeled data is used to estimate the distribution of the data points. Several algorithms have been modified to include a regularization term that complements error minimization. An example are Transductive Support Vector Machines (TSVM) [Joachims, 1999] which have been applied with success to text classification.

- **Graph-based models** build a graph with labeled and unlabeled data points where similar data points are linked based on a similarity measure in feature space. Several algorithms

Algorithm 3 Co-training algorithm

$Co - train(L_0, U) \rightarrow f_1, f_2$
Input: initial labeled data L_0 and unlabeled U
Output: two models f_1, f_2

$P \leftarrow random - select(U)$
while stopping criterion is not met **do**
  $f_1 \leftarrow train(view_1(L))$
  $f_2 \leftarrow train(view_2(L))$
  $L_{new} \leftarrow select - by - agreement(label(P, f_1), label(P, f_2))$
  $L \leftarrow L + L_{new}$
end while

that try to exploit the structure of the graph have been proposed. For example, the Graph Mincut algorithm [Blum and Chawla, 2001] tries to find the minimal number of edges that need to be removed to obtain two disconnected components. Soft approximations to the minimum cut graph problem like Label Propagation [Zhu et al., 2003] are also included among these methods.

- **Semi-supervised clustering** methods define labeled data as pairs of elements that *must-link* and that *cannot-link* which help to guide the clustering process. Constraints can be hard or soft, they invalidate or penalize the solution, and can be used to guide the clustering or even to learn similarity metrics between examples. Once a similarity metric is learned we can employ the cluster and label approach, label the cluster with the most frequent label for labeled data.

Algorithm 4 Schema for a semi-supervised clustering algorithm

$Semi - SupervisedClustering(L_0, U) \rightarrow s$
Input: initial labeled data L_0 and unlabeled U
Output: s a similarity function between examples

$ML \leftarrow generate - must - links(L_0)$
$CL \leftarrow generate - cannot - links(L_0)$
$Best_C \leftarrow cluster(L_O \cup U, s)$
while stopping criterion is met **do**
  $modify(s)$
  $C_i \leftarrow cluster(L_O \cup U, s)$
  if $value(C_i) < value(Best_C)$ **then**
    $Best_C \leftarrow C_i$
  end if
end while

5.3.2 Active Learning

Active Learning (AL) pursues the same objective than semi-supervised methods as it seeks to reduce the need for annotated data, but in contrast it interacts with an expert to acquire additional training examples. SSL works mostly in a batch setting, in the sense that training data, whether labeled or unlabeled is given to the algorithm to produce a model. AL requires the expert to provided additional labeled data during the learning process and therefore the learning process is interactive. Nevertheless, the objective of AL is to take as much profit as possible from new labeled examples, maximizing the global improvement with respect to the

number of labels requested. The motivation is that not all data points are equally informative and an algorithm could learn better with less training data if examples are appropiately chosen [Settles, 2009]. AL and SSL are often combined as the query strategy often samples data points from the unlabeled data.

Algorithm 5 Schema for an active learning algorithm

$ActiveLearning(L_0, U) \rightarrow f$
Input: initial labeled data L_0 and unlabeled U
Output: a model f

$f \leftarrow train(L)$
while stopping criterion is met **do**
  $P \leftarrow select - by - uncertainty(label(U, f))$
  $N_i \leftarrow oracle - label(P)$
  $L_i \leftarrow L_{i-1} + N_i$
  $L \leftarrow L + L_{new}$
end while

5.3.3 Learning with Positive and Unlabeled Data

Learning with Positive and Unlabeled data or **PU Learning** is a special case of semi-supervised learning where only positive labeled data is provided. Most typical ML algorithms are thought to be trained with positive and negative points and they need to be adapted to work in these framework. Learning only with positive data is an interesting setting for some IE tasks as it is a natural way to express knowledge for humans. For example, to learn members of an NE class like `Persons`, it is easy to name a number of them. The key difference is that no negative examples are identified and labeled, person names that we have not mentioned could appear in the training data. In a binary classification setting the simpler alternative consists on identifying reliable negative examples. In a multiclass problem, we can use examples from the rest of the classes as negative data points.

If a reliable set of negative examples is found, standard supervised learning algorithms can be employed and even previous SSL techniques are used if unlabeled data is abundant. Association mining techniques are suited to this kind of problem because they can find regularities in data like positive examples. The problem also arises in what IR names *relevance feedback*, deciding whether a set of documents is relevant for a query in order to build a larger query. The Rocchio algorithm [Rocchio, 1971] is an standard algorithm which have a strong resemblance with self-learning.

5.3.4 Multi-instance Learning

Multi-Instance Learning or **MIL** [Dietterich et al., 1997] is a related supervised machine learning framework that specifies labeled bags of examples instead of labeled examples. A bag with a positive label is guaranteed to contain at least one positive instance. A negative bag is guaranteed to contain only negative instances. It is interesting again because it is a compact and useful way of describing knowledge like Names for semantic class acquisions or Relations between names [Bunescu and Mooney, 2007]. The fact *headquartersOf(Google, Mountain View)* could be considered a positive bag where some of the sentences that contain *Google* and *Mountain View* would signal the relation.

5.3.5 Discussion

In the above paragraphs we have briefly outlined several alternatives to reduce the work that experts or linguists need to do in order to effectively solve a IE problem. They provide three different twists to the supervised learning of models that are useful for IE problems in several settings:

(a) use unlabeled data to improve model building or the results of the task

(b) make effective use of labeled examples by requesting most meaningful annotations

(c) specify knowledge in forms that require less work for experts, like enumerating examples instead of annotating documents

All these ideas are interesting research lines that are currently a matter of research. At least two question have arisen by their application in practical large scale IE tasks. Scalability is one of the first issues, in particular scaling with the size of the unlabeled data. Batch semi-supervised techniques like graph-based methods or regularization have worst-time complexity in $O(n^2)$ or $O(n^3)$. Research on reducing this complexity is ongoing but it is premature to use them to exploit very large unlabeled text collections. Besides, other aspects like adaptation to problems with several classes and their behaviour with sparse feature sets are also under basic research. In contrast, methods like EM, self-training and co-training have more widely been used for text collections. Nevertheless, scalability of this applications is also an issue as base learners could be expensive to train. Then, there are open issues on how to make the best use of unlabeled data.

5.4 Large scale IE vs Traditional IE

Traditional IE technology has focused on the task of annotating documents with the set of concepts that are useful for organizing information in a specific domain. The task has been defined as annotating all the possible entities and relations that are found in text. As a consequence the evaluation methodology has been biased to achieve exhaustive and correct annotation of the text. In a broad sense, large scale IE questions that explicit manual text annotation is a prerequisite for useful IE applications. As QA research has pointed out, redundancy is an important source of information at large scale that can overcome errors in textual annotations. We outline some differences and the interdependeces between traditional and large scale views of IE:

- Mention annotation vs Relation expansion. Traditional IE has focused on the problem of annotating sentences. Consider a relation extraction task, the problem is often posed as given two entities that appear in a sentence classify whether the relation holds or not. The classic NE tagging is similar, given a token in context decide whether it belongs to a class. Large scale IE has attempted to build large lists or relations of facts. The RDC task is defined as compile a list of pairs of entities that have these relation. The analogous case for NE can be seen as compile a list of names of the given class.

- Document processing vs Collection processing. Traditional IE has focused on basic understanding and interpretation of single documents by processing each of their sentences. It works scanning and procesing every document to extract all the juice. The goal of large scale IE is at the collection level, regularities and redundance in data are exploited to assess facts. The question is whether the generalization of how these common facts is expressed helps to locate less frequent facts.

- Evaluation methodology. Traditional IE evaluates the correct annotation of mentions while large scale IE evaluates the correct extraction of facts. A fact could be mentioned several

times in the same document or in a whole collection with different forms. The evaluation methodology of Large scale IE is biased towards evaluating extractions and asses their confidence from the whole document collection. Mention annotation in documents is a more complex which is not strictly required. In other words, the correct identification of all redundant information is not required though redundancy certainly helps to achieve better results.

The two types of tasks have different flavour but are in fact interchangeable. A method that classifies relations between pairs of entities can be used to extract all relations in a large collection. On the other hand, a list of pairs of entities that hold the same relation is a useful a-priori knowledge for detecting if two entity pairs in a given text hold a relation. If we do not only consider factual knowledge but linguistic knowledge that can be aquired in the form of paraphrases or patterns that are associated with a relation we have even a method to generalize beyond extracted facts.

With the advent of the web and large document collections, the focus has been sifted from identifying all mentions to extracting at least one mention that allow to affirm that the relation holds. In this context, IE from large text collections is more similar to Knowledge Acquisition tasks, Ontology Population or even QA where the objective is to compile a large number of facts that are true.

5.4.1 Ontology based IE

Once we question the strategy of how facts and documents should be processed in large scale IE, other questions regarding the order that relations are acquired can be also thought. Ontology driven IE like demonstrated in OntoShyphon [McDowell and Cafarella, 2006] focus on particular parts of an ontology and tries to learn all possible information about particular concepts. In this paradigm, the ontology could be used to select interesting documents but also structure from the ontology could be leveraged in the learning process.

5.4.2 Open IE

Another different dimension of the IE process is the number of different relations that it is able to learn. IE systems can be used to acquire knowledge for specific relations, but, what happens when the number of relations increases? What if it is not limited?. Open Information Extraction has been proposed recently as a method to extract all possible relations of interest from a collection of text in advance. Open IE has been demostrated both in a domain specific corpus [Hasegawa et al., 2004; Shinyama and Sekine, 2006] and in a large sample of the Web by TextRunner [Banko et al., 2007]. A single scan of the collection extract all kinds of facts in a flexible format *(entity1,relation,entity2)* where the relation is formalized in natural language. Redundancy in the extracted arguments or entities is used to obtain descriptions of relations that can be considered synonyms. On-Demand Information Extraction (ODIE) [Sekine, 2006] is a related approach that builds online automatically a relation from text for a query introduced by the user. The URES system [Rosenfeld and Feldman, 2006] combines an Open IE approach that is precise enough with a semi-supervised and large scale IE system (SRES) to acquire information.

5.5 Information Extraction Tasks

Figure 5.1 outlines the main modules of an document oriented IE system. Each document is processed sequentially by a set of local analysis tasks that process sentences. Basic lexical analysis and parsing often support the operation of IE specific modules like NERC and Relation Extraction. These tasks are the focus of the present thesis and therefore their objectives and

definition is analyzed with more detail in the following subsections. Global analysis often requires the analysis of the whole document. Complex IE tasks define templates that integrate information from the whole document. As a prerequisite, it is required to relate pronouns to the discourse objects that are referring to. Pronoun correference as well as linking different entity mentions must be accomplished by the correference module. If the extracted information will be stored in an structured format like a database or a formal ontology, further processing is needed. The cross-document correference subsystem have to decide when elements of templates, relations or entities extracted from different documents are referring to the same real entity world as expressed in the reference ontology. This task is also known as information consolidation, linking or record matching.

5.5.1 Named Entity Recognition and Classification

The objective of the Named Entity Recognition and Classification (NERC) task consists in processing a text and identifying sequences of words that represent entities that can be identified by their name like persons, locations or organizations among others. A similar task in ACE has been named as Entity Detection and Recognition (EDR). This task can be performed in one step or broken into two subproblems. NE Recognition or Detection aims at marking the boundaries of the mention of an entity in running text. NE Classification should assign the correct category to the span of text. Another view of the NERC task, as presented in CONLL, has transformed it into a sequential token classification task by tokenizing the text and assigning a BIO label to each of the tokens. BIO label is just a form to mark the boudaries of an annotation by using `BEGIN`, `INSIDE` and `OUTPUT` labels.

NE are proper nouns that refer to a real world entity and usually are capitalized in most languages. The basic set of classes include Person, Location and Organization but the proper definition is usually binded to a particular domain or task. Finer categories are usually required in some applications, for example the distinction between cities and countries. Sometimes concepts like languages and nationalities have also been considered as NEs because many languages they appear capitalized and refer to concrete concepts. Table 5.1 outlines different taxonomies that have been used in CONLL and ACE as well as Sekine's taxonomy which contain about 200 classes.

Another class of names that defer some treatment are Generalized Names (GN) [Yangarber et al., 2002]. They are concepts that not always appear in capitals but it is a key aspect to treat them as a NE. Examples of GN are names of diseases, drugs and chemical compounds in biomedical applications or names of techniques, products or procedures in strategic surveillance. In that sense, the concept of what is an entity may be affected by language, domain and application characteristics. Finally, temporal expressions, quantities and other numerical and monetary expressions are frequently considered as a component of NERC systems although they rarely qualify as proper NE. There are exceptions though, because certain temporal expression like *Christmas* would be considered as referring to a first class concept. In general, applications like QA would benefit from the ability to recognize and classify entities in larger taxonomies.

The variety of different phenomena in NERC has produced a large number of different approaches with different strengths. Recent surveys with complementary views on the NERC problem are Nadeau and Sekine [2007] and Sarawagi [2008]. On the other hand, almost any NERC system requires knowledge resources as lists of names for a given class. Nevertheless, name dictionaries present some drawbacks:

- Dictionaries do not solve class ambiguity, a mention could represent entities of different classes in different contexts.

- If common words are ambiguous with entity mentions the precision could be seriously affected.

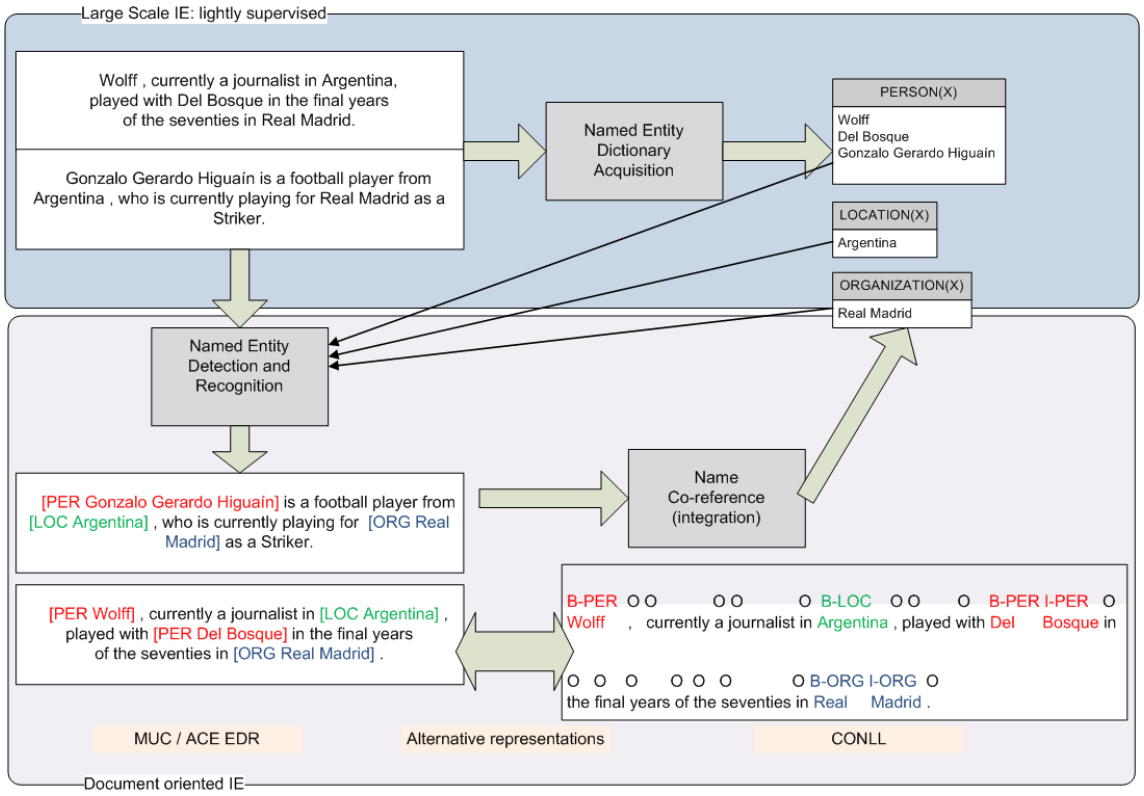

Figure 5.2: An outline of NERC task and NE semantic lexicon acquisition

- No dictionary is complete, particularly those that are manually created. Therefore it is difficult to achieve high recall with only a dictionary based NERC.

Nevertheless, dictionaries can be obtained or completed by mined from large text collections, specialized resources like Wikipedia and the Web. The relation between NERC and the acquisition of lexical NE resources are depicted in Figure 5.2.

Knowledge engineering techniques combine linguistic rules and the integration of handcrafted list of names and other heuristics. Development environments like GATE Cunningham et al. [2008] support the process with an specialized language (JAPE) and other tools. Rules in a knowledge engineered are difficult to maintain because they usually change with different domains and languages. It is also difficult to scale the process of rule creation for very specific classes withouth manually created lexical resources. In a multilingual setting, this approach has been taken by the NewsExplorer system Steinberger et al. [2004, 2005] that has combined the use of multilingual dictionaries, gazzeteers and rules for NE recognition and classification.

Most of the work on using machine learning NERC has focused on the three main classes Person, Location, Organization and sometimes an additional Misc class. The shared tasks at the CONLL 2002 and 2003 Sang [2002]; Sang and Meulder [2003] explored NERC for Spanish, Dutch, English and German. Best performing systems used supervised approaches like Carreras et al. [2002]; Florian et al. [2003] and obtained results between 72% and 89% in F-accuracy. The results really depend on the language and the features used. While some of the features are largely language independent, it is also common that to obtain top results they need to integrate language dependant resources like NE lists, gazzeteers or higher level linguistic analysis.

NERC systems using supervised Machine Learning require a significant amount of annotated corpora in order to estimate parameters for the models. The cost of annotating data turns into a limitation when we approach a new language, a new domain or a different set of classes. In all those cases it is desirable to reduce the amount of supervision that is required. Several alterna-

tives have been investigated in order to complement (and sometimes substitute) annotated data with large quantity of unannotated texts:

- Explore the use of self-supervision, co-training, etc. to build a NERC tagger [Collins and Singer, 1999; Cucerzan and Yarowsky, 1999; Kozareva et al., 2005].
- Employ additional unannotated data to improve an already existing supervised NERC [Ji and Grishman, 2006; Wong and Ng, 2007].
- Reduce annotation requirements by including statistics from larger corpora [Miller et al., 2004] that constraint the model space.
- Use tecniques adapted from Active Learning to reduce the work of annotators and make the best use of their work whether in the context of supervised algorithm [Settles and Craven, 2008] or bootstrapping[Jones, 2005].

The work on pattern learning in contrast is based on semi-automatic construction of domain specific semantic ditionaries from unstructured text. It was pionereed by Riloff and Jones [1999] and Yangarber et al. [2002] with the use of bootstrapping. Once the dictionaries has been extracted it is possible to build a NERC. For example, Turney et al. [2006] evaluates a NERC that uses high precision lists acquired from the web and additional heuristics that help to disambiguate. These works contrast with supervised approaches (and even most of semi-supervised) because it assumes that a very limited supervision is provided. Only examples instances are used and it uses no human annotated data with NE. Most of these works are reviewed in Section 5.6 regarding the algorithms in use. One of the advantages of these kind of NERC is that it may use fine-grained classes provided that lists of names can be acquired.

With the rise of the Web and user generated content their use to acquire knowledge resources has increased. Specialized methods can be devised for semi-structured user generated content like Wikipedia to generate dictionaries [Kazama and Torisawa, 2007; Toral and Munoz., 2006]. Because of their broadness and coverage of different domains it is an interesting alternative.

The Web has also been used in the large scale IE project KnowItAll [Etzioni et al., 2004, 2005]. It combines a range of domain-independent and domain-dependent techniques to acquire lits of cities, companies, actors, etc. The techniques include:

- **Hearst patterns** [Hearst, 1992] adapted for entity extraction like *companies such as NP1, .. and NPn*.
- **Subclass Extraction**, using Wordnet to suggest hyponyms of *company* like *banks* or *telcos*.
- **List Extraction** that learns semi-structured wrappers to locate and extract large lists from deep web resources.
- **Pattern learning** to acquire names from unstructured text. It uses techniques derived from previous bootstrapping approaches.

KnowItAll developers analized the contribution of the different methods and show that the three first are the most productive in a web environment. Nevertheless, pattern learning is interesting because it is a method you could apply to traditional collections where the others three are aimed to obtain few instances of names. Overall, the system relies on the property that correct extractions would be redundant in a huge corpus like the web. Some of these techniques has been recently extended to achieve better results. Kozareva et al. [2008] shows a technique to improve the precision of Hearst patterns by seeding with an additional example of the class that is learnt. SEAL and its improvements [Wang and Cohen, 2007; Wang et al., 2008] have been

used to acquire fine grained NE classes by list extraction from several websites using random walks.

Finally, there are some intersection points between the ML approach to NERC and that based on automatic acquired resources. For example, [Fleischman and Hovy, 2002] has worked on the fine-grained classification of NE which shares common features with work on learning to populate ontologies with NE [Cimiano and Volker, 2005]. Besides, Cohen and Sarawagi [2004]; Wong and Ng [2007] have proposed models for the integration of dictionaries in the process of training a supervised NERC.

5.5.2 Relation Detection and Extraction

The next step for a generic IE system is the recognition of relations. A relation is defined as a type of association between important concepts of the domain, which typically are what we have considered NE. For example, common relations may include associations between NE like the cities where a company has its headquarters (HeadquartersOf(Company,City)) or the capital of a country (CapitalOf(Country,City)). An instance of a relation refers to a particular fact for this type of relation that is true. By understanding the texts in Figure 5.3 we can conclude the fact that HeadquartersOf(`LinkedIn`,`Mountain_View`) is true. We can interpret the relation as a logical predicate that is true when the variables are correctly instantiated.

In addition to the relationships between NEs, it is also common to consider the association with its attributes as a relation. For instance the association between a person and the date where he or she was born (Born(Person,Date)). In conclusion, relations can be defined as a kind of domain focused semantic representation. In the context of IE, documents usually include textual descriptions of this kind of relations which we would like to recognize.

The task of Relation Detection and Characterization (RDC), as formally defined during ACE competitions [LDC, 2008; NIST, 2008], has focused on identifying pre-defined relations between pairs of mentions of entities in text. Examples of the kind of binary relations that are aimed to identify in journalistic texts has been gathered in Table 5.2. The main information components of a relation that they identify are:

- The relation **mention** is the textual extent that express a relation. It is common to mark the extent as the phrase that contains the arguments and all the words that allow to express that the relation holds between them. Most work have considered relation mentions inside sentences, although in fact, relations can span across sentences by using co-reference chains [Stevenson, 2006].

- The **arguments** of a relation are the entities that participate in the given relation. Binary relations are the most common type of relations though N-ary relations could also be defined. Binary relations should define the role of each of the arguments and their correct identification is usually important. For example, in our example `LinkedIn` plays the role of a Company and `Mountain_View` is the Location. Data and Knowledge modelling techniques may help to characterize categories among relations. For example, in symmetric relations like Brother(Person,Person) the order of the arguments will not be important.

- Finally, the relation has **attributes** like their **type** (or their name) and possibly subcategorizations. ACE has attempted to define a set of generic relations between NE that are shown in Table 5.2. The evolution of the taxonomy of relations implies that is difficult to agree in a general definition of relations. Works on domain specific relation extraction has usually defined their specific relations in the table or even concrete subtypes. Other relation mention attributes include their modality, tense or syntactic class which may be useful for their interpretation.

CONLL	ACE class	ACE subclass
PERSON	PERSON	individual group indeterminate
LOCATION	LOCATION	address boundary celestial land-region-natural region-general region-international water-body
	GEO-POLITICAL	continent county-or-district GPE-cluster nation population-center special state-or-province
	FACILITY	airport building-grounds path plant subarea-facility
ORGANIZATION	ORGANIZATION	commercial educational entertainment government media medical-science non-governmental religious sports
MISC	VEHICLE	air land water subarea-vehicle underspecified
	WEAPON	biological blunt chemical exploding nuclear projectile sharp shooting underspecified
	OTHER	

Table 5.1: An outline of NE classes for different schemas

ACE 05 types	ACE 05 subtypes	ACE 08 types	ACE 08 subtypes
ROLE		GEN-AFF (Gen-affiliation)	
	Member		Membership
			Religion
			Ethnicity
		ORG-AFF (Org-affiliation)	
			Sports-Affiliation
			Student-Alumn
	Citizen-Of		Citizen
	General-Staff		Employment
	Management		
	Founder		Founder
	Owner		Ownership
	Affiliate-Partner		Sports-Affiliation
	Client		
	Other		Resident
			Investor-Shareholder
PART		PART-WHOLE (part-whole)	
	Part-Of		
	Subsidiary		Subsidiary
	Other		Artifact
AT	Based-In	GEN-AFF (Gen-affiliation)	Org-Location
	Residence		Resident
	Located		Geographical
NEAR	Relative-Location	PHYS* (physical)	Located
			Near
SOC		PER-SOC* (person-social)	
	Other-Professional		Business
	Other-Personal		Lasting-Personal
	Parent		Family
	Spouse		
	Associate		
	Other-Relative		
	Sibling		
	GrandParent		
		ART (artifact)	User
			Owner
			Inventor
			Manufacturer
		METONYMY*	

Table 5.2: A mapping of different relations taxonomies

The definition of the RDC task is oriented towards the detection and characterization of spans of texts that imply a relation instance. The goal and the evaluation is directed toward the exhaustive detection of every relation mention expressed in a document. Figure 5.3 shows a complementary task of Relation Extraction that aims at extracting all the relation instances. In contrast to RDC, it does not require to recognize every mention in a corpus.

The recognition of relations started using manual patterns defined by linguists and domain experts. This approach was commonly used during MUC evaluations [Aone and Ramos-

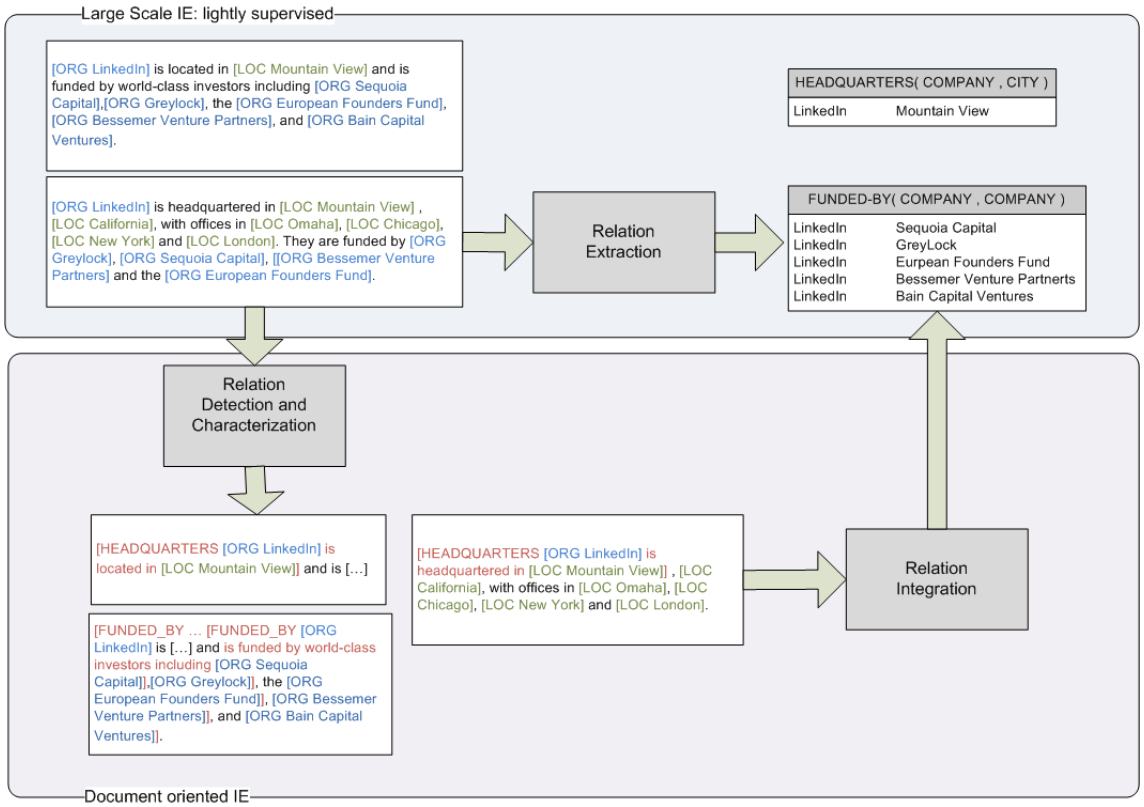

Figure 5.3: An outline of RDC task and the related large scale Relation Extraction task

Santacruz, 2000; Appelt et al., 1993]. Supervised ML methods have gained interest as large corpus like those provided by ACE have been developed with careful annotation of the relation mentions. They usually combine a variety of linguistic analysis tools including shallow and deep parsing with learning algorithms. Generative models were applied to the supervised RDC task by [Miller et al., 2000]. Discriminative methods like SVM are widely used because of the ability to efficiently combine structured representations, like syntactic trees, by means of structured kernels [Bunescu and Mooney, 2005; Culotta and Sorensen, 2004; Zelenko et al., 2003]. In contrast, other approaches that do not require full parsing [Zhao and Grishman, 2005] or that use simpler feature based structures have achieve comparable success[Giuliano et al., 2007; Roth and Yih, 2002; Surdeanu and Ciaramita., 2007].

On the other hand, the need for adaptive methods that require less training data for a new relation has motivated the work on automatically acquiring patterns from unlabelled corpora [Agichtein and Gravano, 2000; Riloff, 1996; Yangarber et al., 2000]. These methods assume that the cost of labelling would not be feasible for most Relation Extraction tasks. Instead of annotations they use examples of relation instances, pairs of entities, as a lighter form of supervision. For example, works like Agichtein and Gravano [2000] have focused on compiling lists of entities that hold the relation. Pattern learning approaches [Feldman and Rosenfeld, 2006; Ravichandran and Hovy, 2002] studied the aquisition of patterns that characterize how the relations are expressed.

Other works have tried to combine learning from mentions with self-learning and bootstrapping [Zhang, 2004] or label propagation [Chen et al., 2006] to reduce the required amount of annotated text. Finally, unsupervised systems that cluster sentences with similar typed arguments have also been applied [Hasegawa et al., 2004] to variants of the task.

NERC and RDC are often performed together, usually in a pipeline architecture, although other methods to combine both tasks have also been studied. For example, Roth and Yih [2002]

perform global inference in the output to classify NEs and Relations simultaneously. Giuliano et al. [2007] have studied how performance of NERC affects the RDC task and whether it is preferably to train in assuming perfect input. Better recall is obtained when training on the Relation task is performed with the output of a real NERC.

5.5.3 Evaluation of IE

The evaluation of an IE system has usually been performed at the system level because IE technologies are part of a larger information access system and not for final users. Document-oriented IE is evaluated by comparing the automatic annotation of the system with a *gold standard*. A gold standard is a benchmark corpus that has been manually annotated with the span of NE mentions and their types, in the case of NERC for example. For RDC the gold standard should mark the arguments and roles of the relations, their type and sometimes the span. The process of generating a gold standard for IE includes agreeing on the definition of the important elements to annotate and also in the specific instructions of what and how should be annotated. The quantitative evaluation is based on measures derived from the contingency table for classification problems.

	Gold standard	
System	class	no class
predicted	tp = true positive	fp = false positive
not predicted	fn = false negative	tn = true negative

Table 5.3: Contingency table for a binary classification task

Accuracy, the ratio of correct predictions to total judgements made by the system is a common measure for classification tasks. The *Error* is its inverse, the ratio of incorrect predictions to total number of judgements. Nevertheless, these two measures are not useful for problems where the number of examples is highly imbalanced and the objective class is the minority one. *Precision* and *Recall* are designed to solve this problem as they reflect better the ability of the system to identify the minority class. Precision measures the proportion of objects marked by the system that have been correctly identified. Recall in contrast is designed to measure the exhaustiveness of the system, how much correct instances are identified among all the manually identified. It is desirable to have a single figure that summarize the results of a system and the *Harmonic mean* or *F-measure* is used to summarize Precision and Recall. A parameter β is often used to decide whether Precision is more important than Recall for a particular application. The usual value of $\beta = 1$ weights both criteria by the same value and it is often the one used in IE applications. *Learning Curves* are another useful measure for systems including a ML component because they are used to reflect the behaviour of an approach with respect to the amount of training data used.

$$Acc = \frac{tp + tn}{tp + fp + fn + tn} \tag{5.2}$$

$$Error = \frac{fp + fn}{tp + fp + fn + tn} \tag{5.3}$$

$$P = \frac{class \bigcap predicted}{predicted} = \frac{tp}{tp + fp} \tag{5.4}$$

$$R = \frac{class \bigcap predicted}{class} = \frac{tp}{tp + fn} \tag{5.5}$$

$$F(\beta) = \frac{(1+\beta^2)}{\frac{1}{P} + \frac{\beta^2}{R}} = \frac{(1+\beta^2)PR}{\beta^2 P + R} \tag{5.6}$$

An important difference between evaluations is what is considered a measureable event for evaluation. For example, the CONLL conferences have adopted a sequence classification model which entails that each token in a sentence is judged. On the other hand, ACE have considered name mentions, which can be composed of several tokens, as the measurable events. In fact, a mention is a complex object which includes their span, class and subclass and the errors can affect only one of these attributes. For example, there are different situations with different severity like partial overlap (the system span and the gold standard span do not match), missclassification (types do not match) or incorrect subcategorization (types match but not subtypes). ACE [LDC, 2008] has opted for assigning different weights to the errors.

The evaluation of RDC faces similar issues. Research that has focused only on this task assumes the correct identification of arguments and simplifies it to a categorization task. Precision and recall can be measured if we consider pairs of candidate arguments as events. A more realistic scenario, like ACE, has considered also relationships as complex objects, with errors inherited from previous step and therefore a more complex metric.

Traditional Document Oriented IE has focused on identifying mentions in the source text. In contrast, Large Scale IE attempts to extract relevant facts from a large collection. The case could be for acquiring a dictionary of names for different semantic classes or extracting a large number facts involving entities. For correct and useful results, it is not compulsory to consider the recognition of every mention. The objective is that as much as possible correct different extractions be acquired. The difference between the two evaluations is clearly marked in [Lin and Katz, 2003] which distinguish between a mention evaluation and an instance evaluation (called type in their work). The name of *Barack Obama* could be mentioned several times in a document but it would be just an instance in an ontology or a lexicon. Likewise, the fact that Barack Obama is the president of the United States PRESIDENT(*Barack Obama*,*United States*) could be stated in one (or several documents) but should be included just one in a relation of presidents. In the case of Large-Scale IE the measurable event is the inclusion on a relation and Precision 5.7 and Recall 5.8 as defined for IR.

The evaluation proceeds by comparing the extracted material with a reference compiled knowledge base. Consider a closed set of concepts like countries, the evaluation can proceed by comparing the extracted results with the complete knowledge base. Though the complete list may contain all countries, it is possible that some of them are unavailable in the collection. We remove those instances and build the definitive *Ideal* list that contain all instances that can be extracted from the current collection.

$$P = \frac{Extracted \bigcap Ideal}{Extracted} = \frac{tp}{tp + fp} \tag{5.7}$$

$$R = \frac{Extracted \bigcap Ideal}{Ideal} = \frac{tp}{tp + fn} \tag{5.8}$$

Nevertheless, the usual case is that we are not interested in extracting names or relations for a predicate that we can already enumerate. In those cases Precision and Recall are more difficult to use. Precision can be estimated by sampling the extracted types and judging them manually. The estimation of the Recall requires sampling documents and manually annotate them. It also requires estimates on the redundancy of mentioned instances. Agichtein et al. [2005] proposes an alternative method for evaluating large scale Relation Extraction that uses an incomplete external list. The external list is expanded using NEs and record matching tools from the collection. Finally, the extracted list is filtered to contain tuples with common arguments. The

filtered extracted list is used to estimate the performance without being punished for missing relations in the incomplete *Ideal* table.

Moreover, it is not clear that a system, that is good in recognising a certain relation, would work equally good for recognising other relations. The evaluation of the patterns or paraphrases that serve to acquire NE or relations is another difficult issue. Manual evaluation is difficult and impractical beyond the qualitative level. The usual approach consists on evaluate the quality of these resources indirectly. Indirect evaluation could be derived from other byproducts like considering extractions or using them in a closely related task like QA [Ravichandran and Hovy, 2002].

A recent survey paper [Lavelli et al., 2008] examines closely the methodology of evaluation in document oriented IE and identifies several problems that hinder a fair comparison among different approaches. Some of them are data problems like different representation and versions of corpora. Other affect the design and presentation of the experiments like the use of different splits of training and test sets or the differences on measuring agreement between systems and gold standard that we outlined above. Large scale IE faces similar problems for comparative evaluation among different approaches. There is no standard corpora and the experimental conditions varied widely between works. Size of the corpora and redundancy of the facts have an important impact to decide what are the most appropriate techniques. Besides, the number of different relations have rarely exceeded a dozen, so in the end no definitive conclusions can be obtained.

5.6 Bootstrapping

In the context of NLP, the idea of using a few examples as labeled data and unannotated text have often been named bootstrapping. The experimental work developed previously or in parallel to the development of the field of semi-supervised learning. For example, the Yarowsky algorithm [Yarowsky, 1995] can be in part described as a kind of self-learning schema.

Bootstrapping has been often associated to a process, not necessarily to an algorithm though the distinction is not always clear and unambiguous. Different semi-supervised techniques, not only self-learning, can be applied to this aim. The common scenario is that the process starts with a small number of seed data and a large quantity of unlabeled data and new knowledge is acquired. This new knowledge could be a model to solve a task, linguistic rules that help to build the model or other resources like dictionaries. In this setting, bootstrapping is an interesting tool to be achieve the goals of large scale IE.

5.6.1 Foundations

The Yarowsky algorithm [Yarowsky, 1995] is probably the first description of an algorithm that exploits annotated and unannotated corpora for an NLP task. It faces Word Sense Disambiguation (WSD) where the objective is to assign the correct sense in a context for a word that could have several meanings, like *bank* or *bass*. The algorithm starts with a word that has been tagged with the correct sense in some examples and with a much larger corpus of raw text. The algorithm proceeds iteratively to learn a decision list classifier (*base learner*) that should disambiguate between senses for a word. The base learner is applied to the raw text to classify new instances and those whose confidence is large enough are added to the training set. There are two key details that make the algorithm works in this context. One is key to the algorithm, a limited number of new examples are added in each iteration. The second can be assumed for this task, it uses the *one sense per discourse* assumption which means that in the context of a discourse (a document, for example) a word will be mentioned only with a sense.

The particular choice of a decision list classifier allows for an additional important feature of the Yarowsky algorithm, duality. Features (or rules) can be seen as dual to instances. It is

NP_x and other NP_y
NP_x or other NP_y
NP_y such as NP_x
Such NP_y as NP_x
NP_y including NP_x
NP_y, especially NP_x

Table 5.4: Examples of Hearst patterns

possible to apply the same self-learning process to discover interesting features beyond those already present in the labeled data. Most confident features can be used to improve the extraction of instances later on. Algorithm 6 shows the modified self-learning version.

Algorithm 6 The schema of Yarowsky algorithm using dual self learning

$Self - train(L_0, U) \rightarrow f$
Input: initial labeled data L_0 and unlabeled U
Output: a model f that assign a confidence value

$f_0 \leftarrow train(L_0)$
while stopping criterion is met **do**
  $L \leftarrow L_0 + select(label(U, f))$
  $f \leftarrow f_0 + select(train(L))$
end while

The Yarowsky algorithm showed competitive results with supervised ML approaches to the WSD task. Recently, Abney [2004] have presented detailed theoretical analysis of variants of the Yarowsky algorithm in order to explain its ability to learn. It has been shown that some of the variants proposed are able to optimize Negative Log Likelihood. Another group of variants, including some that perform sequential updating of features are shown to be able to minimize an upper bound of Negative Log Likelihood.

A seminal work that we can identify as one of the ancestors of bootstrapping is that of Hearst [1992] on using very precise patterns to extract hyponym relations or *IS-A* relations between terms. Patterns like those in Table 5.4 where used to populate the WordNet hierarchy with additional knowledge. In fact, the procedure for dual bootstrapping was suggested in this paper. Extracted terms that participate in an *IS-A* relations could be used to discover new patterns. However, the procedure to select good term-pair candidates or pattern candidates was not implemented automatically.

The interest on using unannotated corpora was pioneered in the IE community by the AutoSlog and AutoSlog-TS systems [Riloff, 1996]. AutoSlog-TS was a system that was able to learn domain specific extraction patterns from a set of relevant documents for that domain. This weak form of supervision contrasted with contemporary approaches that required the explicit annotation of interesting entities and relations. Later on, related works by Riloff and colleagues [Riloff and Jones, 1999; Thelen and Riloff, 2002] explored other ways to specify the knowledge to bootstrap, like the use of a set of seeds to build semantic lexicons.

5.6.2 Applications of bootstrapping

The idea of bootstrapping has been applied in different problems in IE and NLP, specially in the context of the semantic interpretation of language. We have tried to group some of the works regarding their use for producing different types of knowledge useful for text understanding.

- **Lexical and ontological relations** was the initial focus of automatic methods for knowledge acquisition with works like the cited by Hearst [1992] on general patterns that helped to identify hypernym-hyponym pairs or *IS-A* relations. Similar approaches have been attempted later on other lexical relations like meronym [Berland and Charniak, 1999] or synonym. Several of these relationships have been automatically acquired from Wikipedia in Ruiz-Casado et al. [2007].

- **Named Entities**, their recognition and classification, have been the focus of several works using semisupervised methods and large scale IE. We find also significant differences among works that have focused in the acquisition of resources for NERC. NOMEN [Lin et al., 2003] has been used for the acquisition of dictionaries for the basic NE classes, but also in Yangarber et al. [2002] for generalized names, in particular names of diseases. Fine grained NE has been identified as a need for several applications, in particular for QA. Hyponym patterns and set expansion have been used as a base for acquiring large quantities of fine-grained NE lists [Etzioni et al., 2005; Kozareva et al., 2008; Wang and Cohen, 2007]. Riloff and colleagues [Riloff and Jones, 1999; Thelen and Riloff, 2002] have focused in bootstrapping NE lexicons for a specific domain while Lee and Lee [2007] attempted to compile detailed gazzeteers on the geographical domains adapting previous techniques.

  It is important to note that not all works in NE extracted only names. In particular, the original work by Riloff and Jones [1999] was focused on the simultaneous extraction of names and patterns. Besides the work by Talukdar et al. [2006] also builds relevant context as Finite State Automata (FSA) that could be useful to generalize the context of NE classes.

- Detection, extraction or classification of **relations between NE** started with the work of DIPRE [Brin, 1999] and Snowball [Agichtein and Gravano, 2000]. The first was focused in semistructured documents on the web, while the second formalized and made the practical idea for unstructured text. Similar to work on NE the focus has varied between those systems that compile a large number of facts for a relation [Banko et al., 2007; Etzioni et al., 2004; Pasca et al., 2006], those that compile patterns like Ravichandran and Hovy [2002] or ESPRESSO [Pantel and Pennacchiotti, 2006] or models that help to identify the relation [Bunescu and Mooney, 2007; Suchanek et al., 2006], or both [Feldman and Rosenfeld, 2006; Yangarber, 2003].

- **Identification of attributes for NEs** has been pursued from a bootstrapping perspective in Ghani et al. [2006] and in Pasca and Durme [2007]. In this case, the objective is to obtain names of attributes that are common for a class of objects. For example a person would have a name, age, nationality, etc. A camera instead would have a size, a number of pixels, etc. The acquisition of attribute values, like the age of a given person, could be seen as the aquisition of domain specific relations, and specialization of the previous task.

- **Paraphrase acquisition and entailment relations**. A paraphrase is just a different form of expressing the same idea with other words. For example, the phrases *X prevents Y* and *X lowers the risk of Y* convey almost the same meaning. Entailment relations include a directionality constraint that means that if one expression is true the other must also hold like in *X acquired Y* entails *X owns Y*. Both are useful knowledge for semantic reasoning with natural language. Large scale acquisition of paraphrases was pioneered in DIRT [Lin and Pantel, 2002] using dependency parsing. Systems like TEASE [Szpektor et al., 2004] have tried the acquisition of a large number of entailments relations from the web. These systems starts for example from verbs and the restriction that X and Y must co-occur often and use bootstrapping techniques but are not restricted to specific relations. In contrast, systems like Ravichandran and Hovy [2002] and ESPRESSO [Pantel and Pennacchiotti, 2006] focus on acquiring patterns or paraphrases for specific relations.

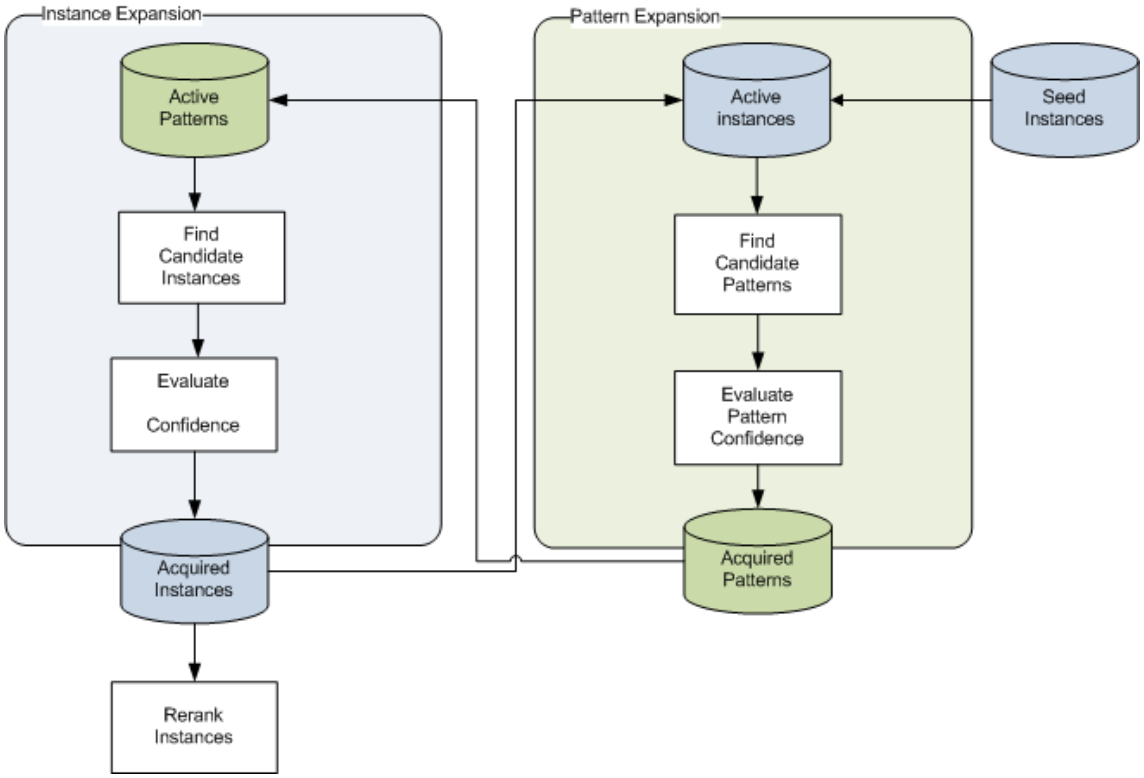

Figure 5.4: The schema of a Dual Bootstrapping algorithm

5.6.3 Dual Bootstrapping architecture

The idea of dual bootstrapping was implemented for IE in Riloff and Jones [1999] for the acquisition of domain specific dictionaries and patterns. The algorithm starts with a category of entities (*Web Location*) and a set of seeds for that category (*australia canada china england...*). Using a corpus of the domain, the seeds and a base IE system, the bootstrapping algorithm is able to discover patterns. The patterns are evaluated and used to signal new instances or seeds that could belong to the dictionary. This basic idea was named by authors as **Mutual Bootstrapping**. A similar idea was used by Brin [1999] in DIPRE, *Dual Iterative Pattern Relation Extraction*, for acquiring relations with two arguments and patterns from the Web. They use a BOOK-AUTHOR relation and learn patterns for specific URLs. Nevertheless, there are significant differences between the two approaches. While Riloff and Jones [1999] focused on a well-formed small text collection from one domain that and use deep language analysis, Brin [1999] relied on vast amounts of semi-structured HTML markup text.

A common problem that was pointed in the first dual bootstrapping algorithms was that the accuracy of the learned seeds could be seriously affected when incorrect seeds are taken as correct. Incorrect or spurious seeds would be employed in successive iterations and would have a negative effect in precision. The same situation reproduces in the dual process of acquiring patterns. The common solution is **throttling**, i.e. selecting only a limited of new seeds per iteration. In Riloff and Jones [1999] they used **Meta-bootstrapping** to obtain a global evaluation measure for candidate seeds. New candidate seeds are ranked by the number of patterns that extract them and only a small subset is added in each iteration. A similar strategy, called Yarowsky-cautious was also used in NE Classification experiments and shown effective in Collins and Singer [1999].

The basic bootstrapping algorithm relabels unlabeled data every iteration. A practical variation is **indelibility** of classifications [Abney, 2007], where classifiers can abstain but once a label is assigned it cannot change. Indelibility also favors some classifiers which do not need to

be retrained from scratch every time. **Persistence** is a weaker form of indelibility where the data remains labeled but it can change afterwards. Jones [2005] shows that indelibility did not affect learning results for NE. Then, it is a desirable property when learning from large unlabeled corpora as it improves performance.

Basilisk [Thelen and Riloff, 2002] is a modification of previous work by Riloff that introduces the idea that learning several classes simultaneously could improve the precision of the acquired lexicons. The same idea was used in NOMEN [Yangarber, 2003] for the extraction of generalized of different classes with good results. This feature of the algorithm has been named as **learning N semantic classes at once** or **counter-training**. In counter-training when the algorithm learns a class, positive instances of the rest of the classes are used as negative examples. This idea could be seen as a trick to transform a PU learning problem into a standard LU learning problem. **Balancing** is another common technique when using several classes which uses a-priori knowledge on the proportion of examples of each class to balance the acquisition process.

In the following subsections we analyze and compare other differences among bootstrapping approaches to IE tasks. Techniques and measures to select and evaluate seeds are analyzed. Analogously, for patterns that can be seen as rules for classifiers we analyze evaluation measures too. The issue of how linguistic information has been used in bootstrapping algorithm is analyzed. The use of other alternatives for representing knowledge beyond patterns or rules are also considered. Finally, issues on scaling bootstrapping algorithms are also considered.

5.6.4 Pattern confidence

An important step in the process of bootstrapping is assesing the confidence of the acquired intermediate results, pattern and instances. Several heuristics have been used to select adequate patterns. Blohm et al. [2007] have classified some of the heuristics used to assess patterns into:

- **Syntactic assesment**. The score is only based on syntactic criteria, like the length of the pattern which is a good indicator for a precise pattern [Brin, 1999].

- **Support-based assigment**. The quality of a pattern can be estimated by the number of examples that have generated it. DIPRE [Brin, 1999] used a count on the number of examples that extracted a pattern to filter them. An alternative used in [Blohm et al., 2007] and named as *merge* consists on counting the number of different instances that support a pattern and require a minimum threshold.

- **Performance-based assesment**. In this case, the pattern is evaluated using typical performance measures in IE like Precision or Accuracy [Ravichandran and Hovy, 2002]. Iterative algorithms compute the performance of patterns using the results of previous iterations like [Agichtein and Gravano, 2000; Thelen and Riloff, 2002; Yangarber et al., 2002]. Different proposals have been gathered under a common notation in Table 5.6 which is clarified in Table 5.6.

- **Instance-pattern correlation**. If a pattern frequently correlates with instances of document collection is a good indicator of correctness. Pointwise Mutual Information (PMI) is one the association measures that has been used for pattern confidence assesment. PMI is defined as the information that is gained by considering the values of two variables (X,Y) as joint or independent as defined by:

$$PMI(X,Y) = log\frac{P(X,Y)}{P(X)P(Y)} \tag{5.9}$$

  The way that systems have estimated the probabilities using counts of instances or patterns from text or from the Web differ. For example, the ESPRESSO system [Pantel and

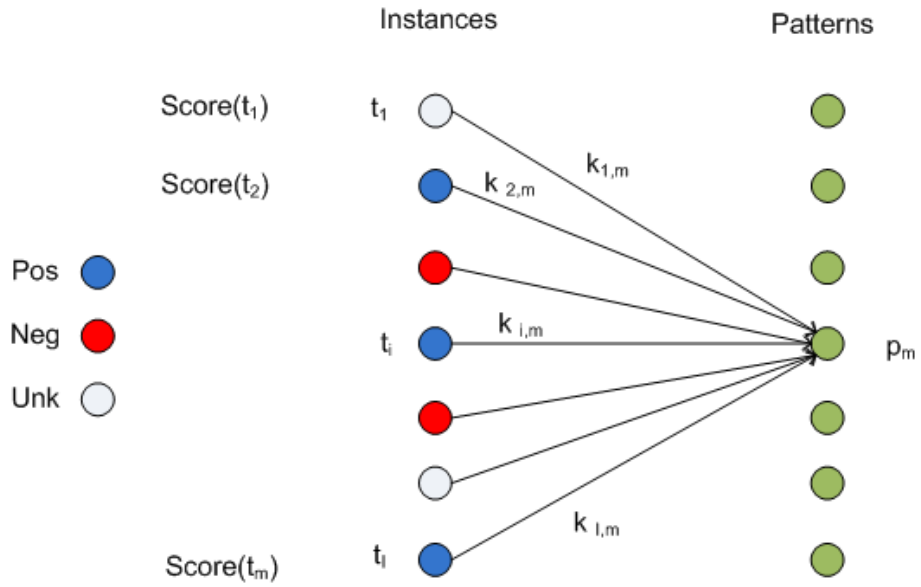

Figure 5.5: A instance-pattern graph depicting the concepts used in pattern scoring functions

Pennacchiotti, 2006] have defined it for learning relations with two arguments as

$$PMI(t_i, p_m) = log\frac{P(t_i, p_m)}{P(t_i)P(p_m)} = log\frac{count(t_i, p_m)}{count(t_i)count(p_m)} = log\frac{count(t_{i,1}, p, t_{i,2})}{count(t_{i,1}, *, t_{i,2})count(*, p, *)} \tag{5.10}$$

Besides, each system has defined slightly different scoring metrics based on PMI. As with performance-based assesment, the scores based on PMI are re-estimated to improve in sucessive iterations.

- **Inter-pattern agreement**. Patterns that produce similar output to previous good patterns are probably good extractors too. In an iterative setting a good productive pattern, one that produces many new tuples, can be found. Nevertheless, there is no way to distinguish it from a bad productive pattern. Confidence measures, as used in Yangarber et al. [2002] for example, try to quantify this effect and delay the acceptance of very productive patterns until more evidence is found. This can be seen as another way of throttling. Stevenson and Greenwood [2005] introduce the idea that patterns that are similar to those previously acquired are good patterns. A similarity measure for patterns is defined and distance to the centroid used to measure pattern quality.

Some systems have combined several measures in order to achieve the desired effect and set thresholds to filter out unprecise or unconfident patterns. For the Snowball system [Agichtein et al., 2005] a method to find the appropiate thresholds and parameters values has been described. It adapts the EM-Spy method [Liu et al., 2002] to leave some examples instances out of the query and generation process. These seed instances are then used to evaluate patterns in an independent manner, in a way that is inspired by the separation of test and training sets in supervised ML.

Name	Description
t_i	An instance proposed for the relation
$t_{i,1}$	The first argument of the instance
$t_{i,2}$	The second argument of the instance
$k_{i,m}$	an estimate of the coocurrences of t_i and p_m, also $count(t_i, p_m)$
p_m	A pattern proposed to extract the relation
Pos	Number of positive instances in the relation
Neg	Number of negative instances in the relation, appear in other relations
Unk	Number of unknown instances
N_p	Maximum number of Pos

Table 5.5: Definitions for scoring models

	$Strength(p_m) = l * \sum_{i=1}^{I} k_{i,m}$	Brin [1999]
	$Precision_1(p_m) = \frac{\sum_{i \in Pos} k_{i,m}}{\sum_{i=1}^{I} k_{i,m}}$	Ravichandran and Hovy [2002]
	$Support(p_m) = \sum_{i \in Pos} = \lvert Pos(p_m) \rvert 1$	Blohm et al. [2007]
	$Conf(p_m) = \frac{\sum_{i \in Pos} 1}{\sum_{i=1}^{I} 1} = \frac{\lvert Pos(p_m) \rvert}{\lvert Pos(p_m) \rvert + \lvert Neg(p_m) \rvert + \lvert Unk(p_m) \rvert}$	Agichtein and Gravano [2000]
	$RLogF(p_m) = Conf(p_m) * log_2(Support(p_m))$	Riloff and Jones [1999]
$Acc(p_m) = \frac{\lvert Pos(p_m) \rvert}{\lvert Neg(p_m) \rvert}$		Yangarber et al. [2002]
$Acc(p_m) = \frac{\lvert Pos(p_m) \rvert}{\lvert Neg(p_m) \rvert} * \frac{1 - \frac{1}{\lvert Pos(p_m) \rvert}}{1 - \frac{1}{max(\lvert Pos(p_m) \rvert)}}$ $Conf_2(p_m) = \frac{\lvert Pos(p_m) \rvert - \lvert Neg(p_m) \rvert}{\lvert Pos(p_m) \rvert + \lvert Neg(p_m) \rvert + \lvert Unk(p_m) \rvert}$	$RLogF(p_m) = Conf_2(p_m) * \frac{log_{10}(\lvert Pos(p_m) + 1)}{log_{10}(\lvert max(\lvert Pos(p_m) \rvert) + 1)}$	Lee and Lee [2007]
	$Score(p_m) = \frac{\lvert Pos(p_m) \rvert}{(\lvert Neg(p_m) \rvert + 1)^2}$	Rosenfeld and Feldman [2008]
	$PMI - score(p_m) = \frac{\sum_{i=1}^{I} \frac{pmi(t_i, p_m)}{max(pmi)} * score(p_m)}{I}$	Pantel and Pennacchiotti [2006]

Table 5.6: Scoring models for patterns

5.6.5 Instance confidence estimation

The issue of how to measure the confidence in the extractions under limited evidence is central to bootstrapping and the main cause behind the use of throttling. The issue of evaluation of instances (or tuples) arises at least in two points in the bootstrapping process. After each iteration the algorithms needs to evaluate extractions. Besides, at the end extractions can be judged using evidence accumulated in all the process and for example they can be reranked. The evaluation could make use of prior knowledge and constrainsts about the task or to be completely general.

5.6.5.1 Relation constraints

There are natural constraints in the data that can be used to assess the confidence in extracted data. For example, constraints used in data modelling like primary keys or **functional dependencies**[Codd, 1970, 1990] could be useful in learning certain predicates. If we consider the predicate *CapitalOf(City,Country)* we know that there is a functional dependency between a country and its capital as a country only has one capital. These assumption could be not always true in reality. Example 5.1 shows a case where this assumption does not hold. A text describing the change of a capital could mention two capitals in different times. If time is out of our scope in the predicate both could be considered true. Nevertheless, functional dependence is a useful assumption as soon it is not often violated.

The same idea is in fact used in counter-training, a **exclusivity constraint** is believed to hold between different predicates. Another important objection is that there would be predicates which do not have useful constraints that you can use to assess facts.

Example 5.1 An example where an apparent functional dependency constraint does not hold because time is not considered

---

... Elisabeth and Ferdinand married in Valladolid, the capital of the Kingdom of Spain. ...
The capital of the Kingdom of Spain was moved to Madrid by Philip II. ...

CapitalOf	
City	Country
Valladolid	Kingdom of Spain
Madrid	Kingdom of Spain

Most works in Relation Detection have considered some form of **type constraints** for arguments. For example, for the relation *HEADQUARTERS_OF(COMPANY,CITY)*, the first argument of a valid extraction should be compatible with an organization and the second with a location. Snowball [Agichtein and Gravano, 2000] starts the bootstrapping with a NE tagged corpus. Rosenfeld and Feldman [2007] showed that NE corpus statistics can be used to filter and improve the extracted instances. REALM [Downey et al., 2007] also uses type-checking of the arguments based on a HMM.

5.6.5.2 Confidence scores

Global confidence methods address the issue of estimating confidence for extracted tuples in a general setting which do not take into account particular constraints. The confidence of an extraction is calculated taking into account global statistics like the number of times the same value was proposed or the quality of patterns that served to extract it. Several confidence metrics used by previous systems are summarized in Table 5.8 with a common notation that is shown in Table 5.7.

Variable	Meaning
t	a tuple/instance that is evaluated
m	an index for different patterns that extract a tuple
M	$\|m\|$ the number of patterns types that extract a tuple
p_m	the pattern with index m
k_m	number of instances of tuple t that extracted p_m
C	the set of correct tuples
E	the set of erroneous tuples
$num(D)$	a function that returns the number of instances in a corpus D
$num(t)$	returns the number of instances of tuple t
$num(C)$	returns a multiset of the number of instances of each tuple $t \in C$
$s = \|num(C \cap E)\|$	the number of tuples in D, correct or incorrect
$score(p_m)$	a function that assigns an score to a p_m
$P(p_m)$	a function that assigns a probability to a p_m
n	number of extraction events

Table 5.7: Description of variables and functions for tuple scoring models

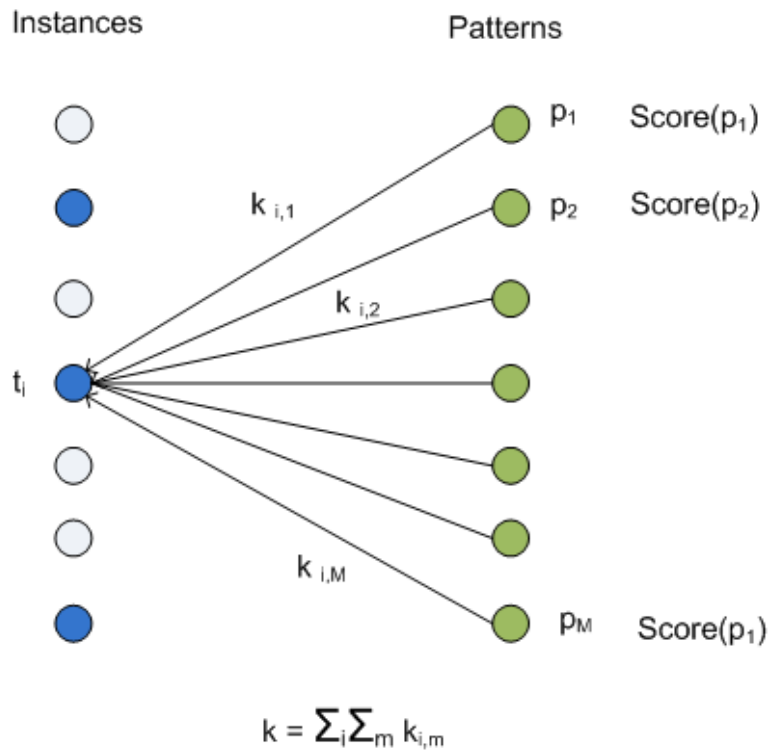

Figure 5.6: A instance-pattern graph depicting the concepts used in the process of selecting instances

If a tuple is extracted several times it is a good basic indicator of their correctness that has been often used in large scale IE and QA. A minimum frequency is often required in combination with other quality scores [Brin, 1999; Etzioni et al., 2005; Yangarber et al., 2002]. The number of patterns that extract the same value has also been used in several of the first bootstrapping systems [Riloff and Jones, 1999; Thelen and Riloff, 2002]. The Snowball system [Agichtein and Gravano, 2000] extended DIPRE in several forms and it proposed the Noisy-OR model as a score for confidence estimation.

Model	Score	Author
Frequency	$Freq(t) = \sum_{m=1}^{M} 1 = M$	
Frequency with tiebreak	$Freq - tie(t) = \sum_{m=1}^{M} 1 + (0.1 * score(p_m)))$	Riloff and Jones [1999]
Strength	$Strength(t) = \sum_{m=1}^{M} k_m$	McDowell and Cafarella [2006]
Strength-norm	$Strength - norm(t) = \frac{\sum_{m=1}^{M} k_m}{num(t)}$	McDowell and Cafarella [2006]
AvgLog	$AvgLog(t) = \frac{\sum_{m=1}^{M} \log_2(k_m + 1)}{M}$	Thelen and Riloff [2002]
PMI	$PMI = \frac{\sum_{m=1}^{M} \frac{pmi(t,p_m)}{max_{pmi}} * score(p_m)}{M}$	Pantel and Pennacchiotti [2006]
PMI-NB	$PMI_{NB} = P(t \in C) = \frac{P(C)\prod_{m=1}^{M} P(p_m\|C)}{P(C)\prod_{m=1}^{M} P(p_m\|C) + P(\neg C)\prod_{m=1}^{M} P(p_m\|\neg C)}$	Etzioni et al. [2005]
Noisy-OR	$Noisy - OR(t) = 1 - \prod_{m=1}^{M} (1 - P(p_m))^{k_m}$	Agichtein and Gravano [2000]; Lin et al. [2003]
Noisy-OR v2	$Noisy - OR(t) = 1 - \prod_{m=1}^{M} (1 - P(p_m))$	
URNS (one urn)	$URNS - s(t) = \frac{\sum_{r \in num(C)} \left(\frac{r}{s}\right)^k \left(1 - \frac{r}{s}\right)^{n-k}}{\sum_{r' \in num(C \cup E)} \left(\frac{r'}{s}\right)^k \left(1 - \frac{r'}{s}\right)^{n-k}}$	Downey et al. [2005]
URNS (multiple urns)	$URNS(t) = \frac{\sum_{c_i \in num(C)} \prod_{m=1}^{M} P(A_m(c_i, k_m, n_m))}{\sum_{x \in num(C \cup E)} \prod_{m=1}^{M} P(A_m(x, k_m, n_m))}$	Downey et al. [2005]

Table 5.8: Scoring models for entities

KnowItAll [Etzioni et al., 2005] uses Pointwise Mutual Information (PMI) to estimate the degree of association between an extraction and a pattern. A Naive Bayesian classifier is trained to use PMI estimations to assess extractions. KnowItAll estimates PMI using several queries to estimate hit counts for different phrase combinations which is a costly process. Moreover, PMI provides too low estimation for unfrequent events. Later on, the URNS model [Downey et al., 2005; McDowell and Cafarella, 2006] has been proposed to consider the frequency of a tuple extraction in a principled probabilistic method. It uses the idea of modelling the large scale IE process as extracting balls from an urn with replacement. The model needs to estimate frequencies for the ocurrence of a fact in a corpus for which it assumes a Zipf distribution for facts.

All these methods are unsupervised or weakly supervised, they need at most estimates on pattern quality and seeds quality to provide a confidence. As the cost of producing a new model is moderate they can be employed as a constructive block in algorithms that iterate several times in the data.

5.6.5.3 Reranking

Some systems apply an additional final reranking step to the complete relation of tuples extracted. The SRES system [Feldman and Rosenfeld, 2006] uses a regression model to learn a scoring function that could be used for ranking a set of extractions. The model is trained in a set of triplets formed by the sentence, the extraction and the pattern matched that are tagged as correct and incorrect. The feature vectors contain predicate independent features like the frequency of the extraction, the number of stopwords in the patterns or the number of stopwords in the sentence among others. Although Feldman's method for reranking is supervised, they have shown that the model can be trained in one relation and used in other relations.

With the objective of ranking a large number of extractions, REALM [Downey et al., 2007] combines type checking with relational language modelling on the contexts of extractions. In this case, frequent extractions for a relation are used to build a language model that is used to assess less frequent extractions. Parameter estimation has been performed with some of the methods presented in Table 5.8. The final objective has also been to rerank a large number of extractions.

5.6.6 Linguistic information and pattern models

An important issue in bootstrapping algorithms is how much linguistic information is used. Any kind of linguistic analysis is a desirable form of generalization that is useful when we deal with symbolic information like language. On the other hand, making use of linguistic information does not come for free. Linguistic analysis of large corpora requires robust and effective tools that scale [Pasca et al., 2006]. The conclusion is that there is an interplay between the detail of analysis that we can obtain and the size of the corpora that we can use. Fortunately, it seems that in some cases more available data could compensate for the lack of advanced resources when used wisely [Banko and Brill, 2001].

The first approaches to bootstrapping relied on patterns that used grammatical roles like Subject, Verb and Objects [Riloff and Jones, 1999; Stevenson and Greenwood, 2005; Yangarber et al., 2000]. They are often named SVO patterns or Predicate-Argument patterns and have been used often in the RDC and related tasks. Different dependency parsers like Sundance or MINIPAR have been used to obtain the parse tree. They have in common that the collections used for bootstrapping are relatively small. They have also used additional semantic analysis like NERC. Two main lines of research have evolved from these initial choice, the use of simpler patterns in larger collections [Agichtein et al., 2005; Agichtein and Gravano, 2000] and design issues related to the different forms that patterns can be generated from parse trees [Stevenson, 2006].

Agichtein et al. [2005]; Agichtein and Gravano [2000] experiment with different simpler patterns and the use of NERC in Relation Extraction. They borrow from DIPRE the use of sequential token based patterns. DIPRE was targeted to use semi-structured web text which is much noisier and where it is well known that syntactic parsers fail. Snowball introduces the concept of *Vector Acceptors* that is a vector based representation of surrounding tokens that accumulates global information using *tf-idf* weighting. Besides [Agichtein et al., 2005] proposes the use of *Classifier based acceptors* that are based on a ML classifier that matches the sentence or not. Considering classifier based acceptors is possible to see a unified framework for pattern-based systems and those base in feature representations.

Greenwood and Stevenson [2006]; Stevenson and Greenwood [2006] analyzed different alternatives for the generation of patterns from parse trees besides SVO patterns. A well known limitation of SVO patterns is that they can only extract relations between the verb and their arguments. They proposed the following pattern generation models:

- **Chains**: A chain is a path between the verb node and another descendant node, not necessarily a direct path. It encodes the relation between the verb and another argument. It is able to represent nominalization or prepositional phrases because in a correct parse tree this phenomena appear as a direct child of the argument.
- **Linked chain**: A linked chain is a path between two chains that share the same verb but they are not direct descendants. They can encode relations between arguments like SVO and also nominalizations and prepositional phrases.
- **Subtrees** Any subtree of the dependency parse tree is proposed as candidate, excluding single nodes. That includes chains and linked chains as possible subtrees.

The productivity, or the number of potential patterns that can be generated with each of the approaches for a single sentence, increases with different rates being exponential for subtrees and linear for the rest of the models. Besides, the generation of all subtrees have been shown to be a #P-complete problem [Stevenson and Greenwood, 2006]. They also measure coverage, the number of arguments from a corpus of identified relations that are identified using the different pattern models. Each of the more complex models (SVO, chains, linked chains and subtrees) have better coverage but the use of subtrees do not compensate the increase in cost with respect to linked chains. Experiments in an semi-supervised IE task [Greenwood and Stevenson, 2006] showed that SVO patterns are able to achieve high precision but low recall, while Subtrees and Linked Chains obtain high recall but lower precision.

The interpretation of lexical patterns (sequential tokens) and tree patterns can be seen as a rule based classifier. A pattern is a classifier that assigns a label class if it covers the example. Rules can be generalized by the use of wildcard tokens that can cover a set of different symbols. Semantic generalization based on NE classes [Agichtein and Gravano, 2000; Rosenfeld and Feldman, 2007] or clustering semantically related words [Pasca et al., 2006] could be seen as a kind of semantic wildcards. Other softer interpretations of rules are possible beyond strict matching. [Ruiz-Casado et al., 2007] use edit-distances between patterns and sentences to allow for a number of errors. In contrast, feature based representation of linguistic structures have been used in combination with more advanced learning techniques like SVM [Agichtein et al., 2005; Suchanek et al., 2006]. Structured kernels allow to incorporate complex linguistic information in these learners which have been shown effective in supervised NLP tasks recently. Nevertheless, there are two main drawbacks with SVM and other large margin algorithms in a large scale self-learning setting. It is difficult to generate queries from a feature based representation to explore the document collection and these algorithms need to be trained in a batch of examples.

Task	Approach	Information Level	Pattern type/Classifier	Author
NE	Linguistic	Lexical	Character sequence	Cucerzan and Yarowsky [1999, 2002]
		POS-tagging	Token sequence w/ wildcards	Lin and Katz [2003]; Yangarber [2003]
		Syntax-Dependency	SVO tuples - Cover	Riloff and Jones [1999]; Thelen and Riloff [2002]
		Syntax-Dependency	Decision List	Collins and Singer [1999]
	Feature	Lexical	Token/Character sequence	Kozareva et al. [2005]
		Lexical	Token sequence	Murphy and Curran [2007]
RE	Linguistic	Lexical	Token sequence	Agichtein and Gravano [2000]
		Lexical	Token vector -Cosine Similarity	
		Lexical	Token sequence -wildcards	Feldman and Rosenfeld [2006]; Ravichandran and Hovy [2002]
		POS-tagging	Token sequence	Etzioni et al. [2004]
		Syntax	SVO vector - Cosine Similarity	Stevenson and Greenwood [2005]; Yangarber et al. [2000]
		Syntax -Dependency	Subtrees	Greenwood and Stevenson [2005, 2006]; Sudo et al. [2003]
		Syntax -Dependency	Chain	Greenwood and Stevenson [2005, 2006]
		Syntax -Dependency	Linked chains	
		Syntax -Dependency	Linked chains-	Greenwood and Stevenson [2005]
		Semantic	Semantic-class sequence	Pasca et al. [2006]
	Feature	Lexical	Token vectors - NaiveBayes	Agichtein and Gravano [2000]
		Lexical	Token vectors - SVM	
		Lexical	Token vectors - SVM-MIL	Bunescu and Mooney [2007]
		Syntax -Dependency	(word,function,direction) vectors - k-NN	Suchanek et al. [2006]
		Syntax -Dependency	(word,function,direction) vectors - SVM	
		Syntax -Constituent	Feature vectors - SVM	Zhang [2004]

Table 5.9: An outline of representation and linguistic information used in bootstrapping systems

5.6.7 Sources for bootstrapping

Algorithms for bootstrapping knowledge have used different sources in order to acquire semantic lexicons, paraphrases, relations or tagging models. Bootstrapping from unstructured documents have been the main issue of the works that we have analyzed. Riloff and Jones [1999] use a relatively small and domain-specific collection combined with deep linguistic analysis. Larger text collections but shallower linguistic analysis have been used by Snowball [Agichtein and Gravano, 2000]. The power of the Web as the larger and most redundant text collection has been tappered by several works too. Ravichandran and Hovy [2002] and the KnowItAll system [Etzioni et al., 2004] queried an external Web search engine. In contrast, Pasca et al. [2006], Google researchers, have used large chunks of Web text. Recently, Google has released a large corpus of n-grams and their counts which have also been used in semantic dictionary acquisition [Murphy and Curran, 2007].

The Web is not only a vast resource of unstructured text, but there are also a large quantity of databases that are published as semi-structured documents. The DIPRE system [Brin, 1999] targeted the acquisition of relations from these kind of documents. Other works in list extraction like KnowItAll or SEAL have focused in semi-structured documents.

Not only documents but also web search engine query logs have shown very useful to extract information and build knowledge resources. Pasca and colleagues [Pasca, 2007; Pasca and Durme, 2007; Pasca et al., 2007] have shown that query logs often convey semantic relations between terms that are useful to discover for example attributes of objects.

5.6.8 Execution strategies for Collection Processing

Another issue that arises as a consequence of large scale IE is whether the process should proceed by analyzing each of the documents or are there methods to per-use the collection that provide similar results at a smaller cost. Ipeirotis et al. [2006] analyze different strategies for text-centric task processing including IE from the perspective of query processing in databases. They provide a cost and time analysis with respect to the completeness of the desired results. The IE system is abstracted as a document processor P which processes a document d and extracts a set of tokens $Tokens(P, d)$. For example, in a large scale task of extracting tuples of the HeadquartersOf(Company,City) predicate, the objective would be to obtain as much distinct correct tuples that appear in a collection D. Is it better to scan each document or try to locate finance documents and apply the extractor to them?. The completeness of the process is measured in terms of Recall for a collection D and an execution strategy S (see Table 4.10):

$$Recall(S, D) = \frac{|Tokens(S, D_{proc})|}{|Tokens(S, D)|} \tag{5.11}$$

The execution model assumes constant time for the basic operations, which include querying the document collection, retrieving, processing and filtering a document. Filtering is assumed to be a cheaper process than processing the document with the IE system. Some of the operations of the execution strategy could need a training step like training a filter or learning a set of queries for a domain. Note, that the model assumes that the IE system (the processor P) is fixed, so it has been already trained. The execution time assuming constant time for each document is:

$$Time(S, D) = t_T(S) + t_Q|Q_{sent}| + (t_R + t_F)|D_{retr}| + t_P|D_{proc}| \tag{5.12}$$

Four strategies are analyzed using this basic set of primitive operations. A model that dinamically select the most appropiate strategy depending on the target recall is also proposed. The four strategies are summarized here:

Parameter	Description
t_T	time to train the needed filter,query generation, etc to execute strategy
t_Q	time to generate queries
t_R	time to retrieve documents
t_F	time to filter documents
t_P	time to process documents (extract information)
Q_{sent}	queries generated
D_{retr}	retrieved documents
D_{proc}	processed documents

Table 5.10: Terminology for IE execution strategies

- **Scan**. The Scan strategy processes each document in the collection exhaustively until the target recall is achieved. This is the simplest strategy and most traditional IE systems have used it. Assuming the IE system is fixed it does not require any training phase.

- **Filtered Scan**. The Filtered Scan is a variation of the basic Scan that uses a filter to decide if a document is useful before processing it with the IE system which would be an expensive operation. A classifier has to be trained before applying this strategy. Quick rejection of documents with no useful extractions would speed the processing of the large collection significantly.

- **Iterative Set Expansion**. The Iterative Set Expansion is a query-based strategy that starts with a small number of user provided seeds or tuples. Seeds are transformed to queries that locate relevant documents. This documents are processed to extract new tokens or extractions that expand the set. The extractions are used in succesive iterations to discover new documents. This strategy has no training phase as the queries are derived in a task specific way from extractions. This strategy stalls if documents contain only single extractions in isolation. It has been used by the KnowItAll system to extract names and relations from the web. It has also been used for semantic class expansion from the Web [Wang et al., 2008].

- **Automatic Query Generation**. This strategy is query based but it works in two phases, query generation and execution. In the first stage a classifier is trained that categorize documents as useful or not for the task and then queries are derived from the classifier. During the execution stage the document database is searched and the retrieved documents are processed with the IE system. The QXtract system [Agichtein et al., 2005; Agichtein and Gravano, 2003] uses this strategy for several extraction task.

Scan and Filtered Scan are crawling-based strategies, they crawl the complete file structure or the whole document to process documents. Iterative Set Expansion and Automatic Query Generation are query-based strategies that only access a subset of the documents in the collection. Analysis of the cost makes use of the concept of the *Reachibility Graph*, which is a directed graph that connects a tuple t_i with a document d_i through a query q_i and the document d_i to another tuple t_j when it is extracted from it. In this case we will say that t_i is linked to t_j. The reachability graph and its connectivity define the maximum recall achievable by a query-based strategy. The upper limit of the recall is given by the set of connected components that are targeted by the initial conditions, the set of seed tuples for Iterative Set Expansion or the generated queries for the Automatic Query Generation.

So far the analysis in Ipeirotis et al. [2006] has considered a fixed IE system that does not improve with increasing quantities of text seen. For example, when using self-learning most approaches use the scan strategy, so they iterate over the whole document collection and train a

new semi-supervised model each time. Using large quantities of data is still a matter of research for most batch approaches (like regularization and SVMs). Online learners, which adapt the model with each new instance, would benefit by training only in useful documents in order to enlarge the amount of unlabeled data they can employ. Therefore, the use of different execution strategies is also meaningful for the task of training an IE system. There is an interplay between at least three factors; collection size, linguistic analysis and execution strategy whose trade-offs have not been widely explored yet. Traditional pattern learning algorithms [Agichtein and Gravano, 2000; Riloff and Jones, 1999; Yangarber, 2003] have characterized by the use of a few scan iterations on small collections of unlabeled data (in the order of megabytes) enriched with linguistic analysis. Works on using more complex linguistic patterns continue to use scan strategies [Stevenson, 2006]. Shallower analysis has been applied by Pasca et al. [2006] in a terabyte collection to improve recall with only few scan iterations. Iterative Set Expansion has been employed with no analysis in KnowItAll using an external Web Search Engine and using shallow POS analysis with the Bindings Engine [Cafarella and Etzioni, 2005].

5.6.9 Indexing strategies for Large Scale IE

Query-based strategies issue queries to indexing engines in order to retrieve documents or parts of them, like passages or sentences. Then they try to extract relevant information or infer patterns. Traditional IR engines could be used to access documents based on token queries. Web Search Engines help to leverage large collections like the Web via APIs like Google Search API [Google] or Yahoo! Boss [Yahoo!]. However, structured queries that exploit linguistic analysis is not possible with most search engines. Specialized indexes or adaptations of current indexing strategies are needed to leverage the ability to include linguistic information in the query language. This capability would have a broad application in large scale IE and QA, among other uses.

Adding type information for indexing terms have been made available in open-source IR engines like Lucene [**?**] using payloads that allow indexing POS tags or NE tags associated to tokens. Type restrictions have been used in QA with success to restrict the results to only certain types in predictive annotation [Prager et al., 2006a; Radev et al., 2000]. Chakrabarti et al. [2006] propose methods to index large hierarchies of NE classes. They also provide ranking methods to queries with type restrictions which help to select promising passages. A query like `type=distance NEAR Hamburg Munich` try to find passages that contain which is the distance from Hamburg to Munich. There are space and efficiency issues when indexing with a large hierarchy index like Wordnet. The trade-off between full type annotation and partial annotation complemented with a reachability index is studied.

The Bindings Engine [Cafarella and Etzioni, 2005] use and extended posting structure to store Part of Speech and Syntactic Chunk structure. It shows a large efficiency improvement for queries that contain a typed variable. Queries that are improved with this index are for example `powerful` ⟨`NOUN`⟩ that retrieve nouns that are preceded by `powerful` in a phrase context. Hearst patterns can be evaluated quickly with expressions like *cities such as* ⟨*PROPER_NOUN(HEAD(NOUN_PHRASE))*⟩ using the indexed information.

Indexing and querying general linguistic annotations and patterns is a challenging task because there is a wide range of phenomena to model. Besides, errors in annotations tools require complex graph structures [Bird et al., 2000]. Nevertheless, it is practical in many situations to restrict it to simpler tree models. With the advancement on XML DBMS and related technologies it has become an alternative to store annotated text as XML trees. For example, the XQUESTA system [de Rijke et al., 2006] has implemented a full QA system using a XML indexing engine. Query languages for XML like XQuery have also recently included an extension for keyword search which would facilitate to combine lexical and structure restrictions. Mayo et al. [2006] describe alternatives to adapt a language for querying stand-off annotations to XQuery. Despite their interest, query efficiency is still an important issue when dealing with large collections and

the performance of XML engines is far from full-text engines.

5.7 Conclusion and Discussion

In this chapter we have outlined the landscape of Information Extraction research and practice. Supervised machine learning methods for IE have been proposed for several tasks like NERC and RDC. However, one of the problems with these methods is their need of annotated data which requires a significant effort. Although task like NERC and RDC may be considered generic they require being adapted to the specific concepts and characteristic of each domain of application. Similarly, when IE requires to be applied to other languages additional annotation effort is required. There is an increasing need for IE tools in different languages and domains as the digitization of unstructured text increases but applications that use semantics are hindered because the need to create training annotations. Nevertheless, the effort to develop tools for other languages require annotated corpora, dictionaries, analysis tools and other language resources. The generation of these resources, especially when the application targets several languages, becomes one of the most expensive steps in the whole IE process.

Semi-supervised IE may take profit of the large amount of raw corpora available. Under the hood of semi-supervised IE we can distinguish between those using the unannotated text to improve performance and those aimed at requiring less annotation effort. Among the alternatives we focus on large scale IE that focus on knowledge acquisition from large text collections using only instance examples. We briefly compare this approach with the most traditional one based on explicit annotation of documents.

The chapter also analyzes different bootstrapping algorithms that have been used in the context of large scale IE to acquire knowledge for IE system from dictionaries of entities, trigger words or relation patterns. We analyze their main components and proposed a common framework or architecture for their comparison and analysis. Besides, the architecture is useful to find the common traits between systems aimed at acquiring NE and relations. We also describe some of the problems of these systems regarding semantic drifting and scalability to large collections. Furthermore, another important point that serves to compare different proposals is the use of linguistic resources.

While the bulk of research on IE has focused in English, there has been evidence that bootstrapping techniques could be useful if adapted for other languages. Bootstrapping for multilingual applications and languages other than English would benefit from the use of as few language analysis tools as possible. Although the Web is an invaluable resource and it should be used whenever it is possible, it is not always as useful as in English. Web extraction systems clearly benefit from redundancy, but unfortunately even the Web does not present the same level of redundant information in other languages than English. However, large text collections are common for the scenarios using this kind of technologies like IE or QA. Therefore, we believe that it is an interesting line of research to consider how useful bootstrapping techniques for other languages than English are. In order to make bootstrapping useful for several languages is required to analyze if the ablation of language resources can be compensated in realistic scenarios. Besides, it also allows to research if the assumptions used in bootstrapping algorithms extends to other languages that show complex linguistic phenomena.

We aim at scratch the surface of these questions in the rest of the chapters. Next chapter outlines a proposal for using bootstrapping for detecting NE and relations as part of Question Answering system. On the following chapter we present a bootstrapping algorithm for medium-sized collections that may be re-targeted for different languages and domains and aims at using few linguistic resources. We focus on the analysis of their use for the acquisition of useful resources for the problem of NERC.

Chapter 6

A practical proposal for building adaptive QA systems based on Large Scale IE

6.1 Motivation

During our participation in the CLEF-QA evaluation we have confirmed that the adaptation of a QA system is a difficult task. Here, by adaptation, we consider the point of view of the designer, as the required modifications to carry out in a generic QA system to work in a new scenario. It could be argued that our approach should be better named as *re-targeting*, but the term *adaptation* has found widespread use among NLP communities. It is important to distinguish it from adaptation from the user point of view, which in our context we would named as *personalization*.

Adaptation of systems that involve natural language processing may be required along different axes like domain, language, style or mode (written text vs. speech transcription) among others. In QA systems a change on any of these dimensions would probably impact every module from Question Analysis to Answer Selection including the Information Retrieval components. Nevertheless, we believe one of the key modules are those included in Answer Extraction, in particular those that perform the recognition of NE and the relationships between them. In general, a change along any of this dimensions require to develop new rules in knowledge engineered systems or to retrain with an annotated corpora. Quite often that implies the development of the resource. Although there may be specialized strategies to exploit in particular cases, it is interesting to devise techniques that allow to bootstrap specialized resources with less effort. As we have seen, bootstrapping techniques are maturing to the level they can become useful. On the other hand a problem that is specific to language adaptation and it is frequent in some potential domains of applications is the availability of language tools and resources. In this case it is interesting to reduce to a minimum their use and investigate whether those language and knowledge resources may be acquired from large text collections for particular applications.

Therefore, in the rest of this thesis, we focus on the first two axes of adaptation, language and domain. We aim at exploring whether the application of bootstrapping techniques may be applied to the production of components that helps to Answer Extraction in different domains and languages. We aim at exploring if resources for different languages and collections may be created with minimal supervision and without the need of expensive language resources. We not only focus on the acquisition of particular knowledge (names and the main relationships) but also if specialized language patterns and may be reused. Finally, it is also interesting to unify relation extraction and NE acquisition to come out with a common procedure that simplifies the overall process.

The following section introduces how these resources may be integrated in a QA system and

what are the assumptions that we believe make the approach sensible. Implicitly, in our attempt to reduce the startup requirements of QA systems for different languages, we aim at enabling easier development of multilingual QA systems in those situations where parallel, comparable or complementary collections in different language may exists.

6.2 Architecture

The following scenario describes what we believe is a common starting point for a text based Question Answering system. We presume that these resources may be available:

- A **large corpus of documents** that someone would like to leverage as a source of answers for a broad base of users. Usually, the documents have been created with other uses in mind and not necessarily with the goal of answering concrete questions (like FAQs or other archives of questions and answers). The documents contain a large number of sparse facts which users would like to query without the need of a formal query language.
- A task to fulfill that provide us with an idea of common user needs and some **example questions**. Though not always the requests are about concrete facts, in this work, we focus only in those questions that are factoids. Usually the questions have not been categorized and, definitely, they are too few in number to directly apply supervised Machine Learning methods to QA.
- There is **domain knowledge** like taxonomies, database schemas or ontologies that help to model the type of information in the collection and the questions. This knowledge reveals the important concepts in the domain, including the typical entities and the relationships among them. If this high level conceptualization is not available, at least it could be created with the help of experts in the domain of application.
- There are **no language resources** and tools, or at least not particularly tailored to the domain. On the other hand there may be taxonomies and ontologies that contain examples of instances (even lots of them) for the most important concepts.

The approach that we propose focuses so far on the use of large scale IE in order to bootstrap knowledge and language processing resources for answering factoid questions. An overview of the general process and components is provided in Figure 6.1. The first step consists on the development (or reuse) of a *lightweight domain ontology* [Gómez-Pérez et al., 2004] of the domain that helps to support the QA system. This ontology should define at least the main classes of entities in the domain as well as the relations that should be modelled between them. It is expected that such ontology would have a good coverage of the kind of questions that could be accurately solved by using the information on the document collection. The coverage of questions may be improved once the QA system is working. For instance, by analyzing the logs of the questions that are submitted we could complete the ontology. This ontology defines what we called the semantic model in our process of bootstrapping and acquiring lexical and pattern resources.

In addition, the bootstrapping process requires that some instances or examples of the entities and the relations are also specified. In order to acquire this knowledge we can make use of example questions, existing domain knowledge or elicit additional examples from domain experts. We believe that the process of producing instances is less expensive than proceed with formal text annotation as it would be required by supervised methods given a new language or a new domain.

Chapter 6 introduces a bootstrapping algorithm aimed at the acquisition of instance knowledge and pattern knowledge from textual collections. The proposed algorithm has been implemented in the SPINDEL system that is able to acquire knowledge for unary relations (NE classes) and binary relations.

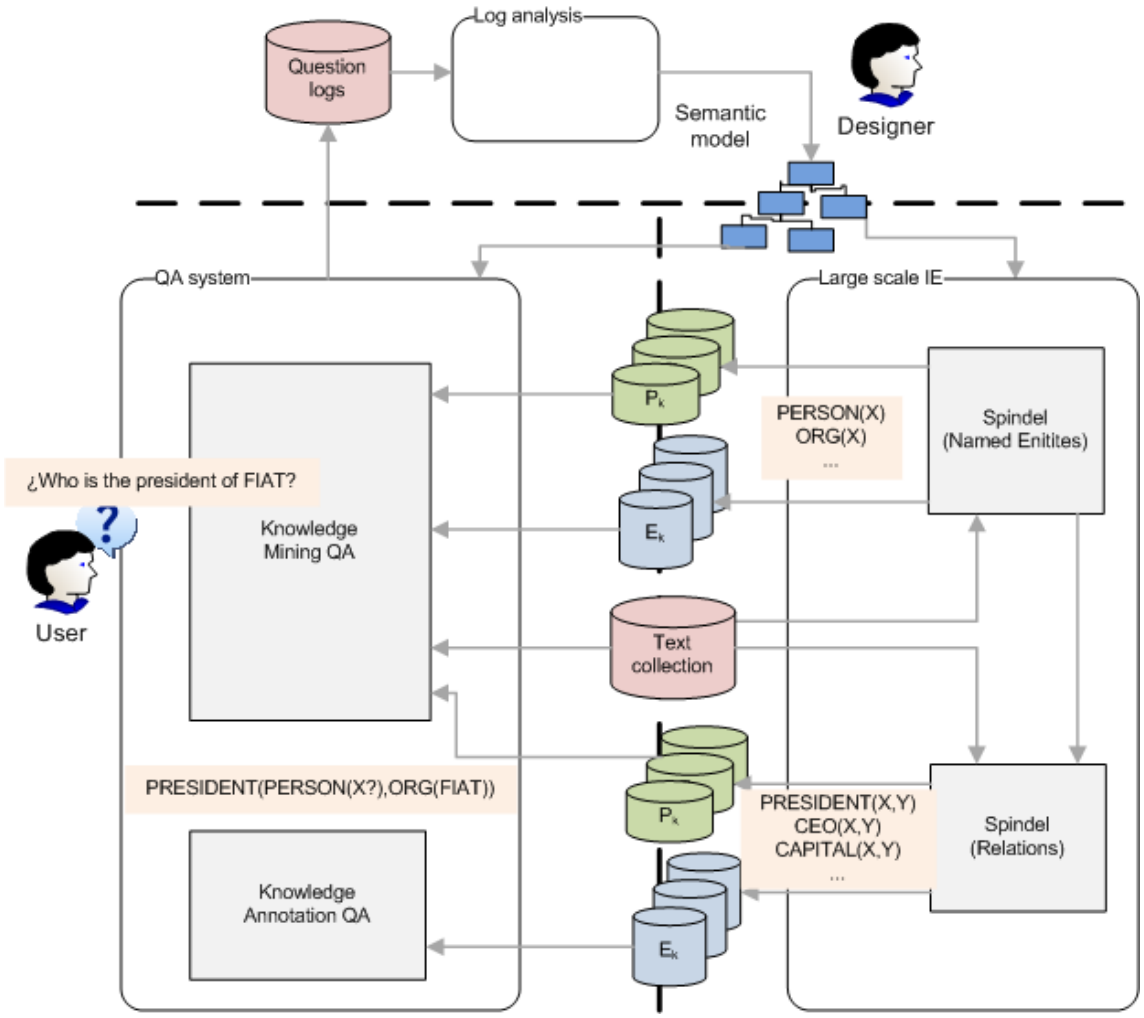

Figure 6.1: Overview of the process of QA system by using Large Scale IE

In our experiments in Chapter 8, we have focused on the acquisition of NE related resources. We study the use of a reduced set of examples for large entity classes which is one of the most unfavourable starting points for a bootstrapping algorithm. Using corpora from CLEF, Wikipedia and the CONLL shared task on NERC we evaluate the acquired resources for Spanish and English-

In a more realistic scenario, the quantity of the seeds may be larger because of additional existing resources. Besides, it is also possible to use certain amount of manual supervision at the beginning until the amount and quality of the seeds is enough to leave the automatic process works autonomously.

Different knowledge and language resources may be bootstrapped from the instances in the ontology from the document collection. The process of instance expansion makes use of existing redundancy in the collection in a different manner than QA systems described in Section 2.10.1. Redundancy-based QA systems assess the correctness of a single fact that appears in several sentences. Here we are using the property that similar facts, consider for instance PERSON or LOCATION names, appear in similar contexts. Therefore, the redundancy is exploited at the concept level and makes use of Harris' distributional hypothesis, that is outlined as "words that appear in similar contexts tend to have similar meanings". Something similar could be said about relations between entities, there are many forms to express where a person was born but all of them should be common across persons and their born places. As we have seen in the

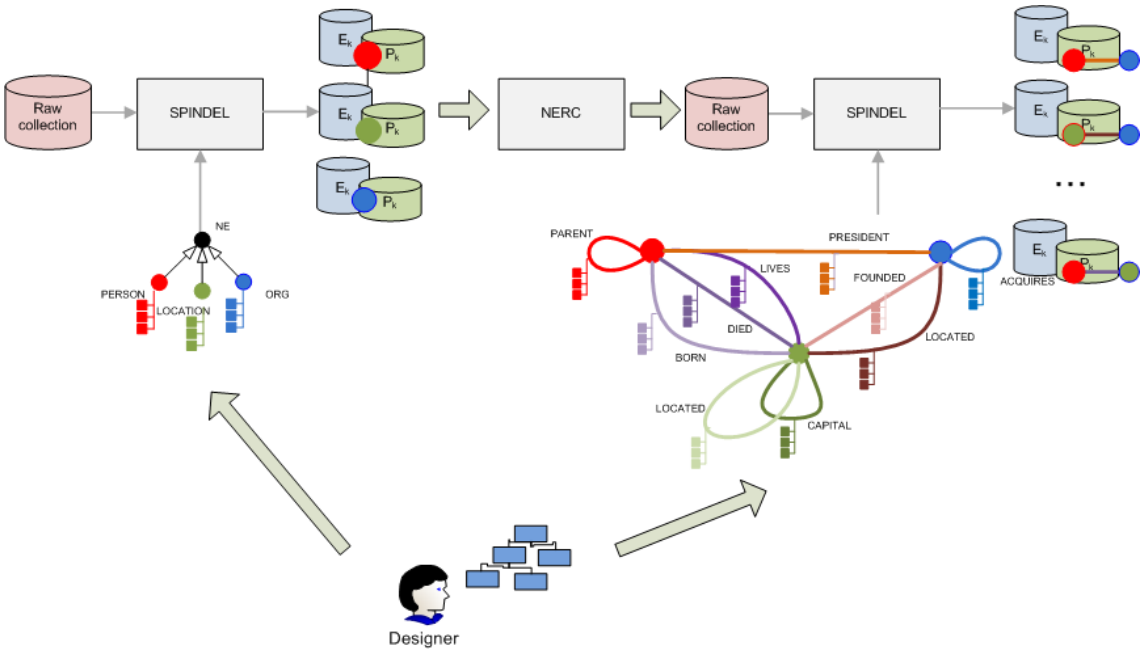

Figure 6.2: Process for building resources for QA

previous chapter it should be possible to discover the most prevalent paraphrases. On its turn the most common paraphrases should help to locate similar relations.

In addition, we will also take advantage that is easier to learn several concepts at once than separately. This is also the case for relations, specially if their arguments are of similar types. In both cases, we minimize the risk of drifting to other concepts and relations. The process is so analogous for entities and relations that both could be abstracted as the expansion of relations with different number of arguments. We devise that the recognition of concepts should help the accurate recognition of relations. In the end we expect to obtain the following resources:

- A **list of entity instances (names)** associated with each of the entity classes. Those can be used as a lexicon or dictionary in order to help in the recognition and classification of entities in text (NERC). If the dictionary is successfully expanded with correct terms we can expect to obtain a high precision NERC. In contrast, its recall would vary with the size. By mining names from the same collection where they should be recognized, we can expect a higher overlap than with external dictionaries.

- A **list of textual patterns** that are associated with each of the entity classes. These patterns may help to trigger the recognition of the class of an entity. We would attempt to use them to improve the recall of NERC beyond those names in the dictionary.

- A **list of relation instances** mined from the document collection offline. This list should be directly useful to answer some of the questions posed to the QA system. Ideally, this list would contain those relation instances that have a high frequency in the collection.

- A **list of textual relation patterns** which are paraphrases of how a certain relation class is usually expressed in text. These can be also useful to pinpoint less frequent relation facts based on those that are expressed with more redundancy in the collection. In this case, questions would typically instantiate one of the arguments in the collection and we could try to retrieve less frequent facts given the argument using the paraphrase correlation with more assessed pairs.

Once the resources are generated they can be integrated into different QA pipelines. Figure 6.1 describes how different byproducts of the bootstrapping may be integrated into a Knowledge Mining and a Knowledge Annotation QA system. For instance, the set of extracted relation instances may be useful to answer questions (almost directly) in a Knowledge Annotation based QA system. In contrast, the acquired relation paraphrase patterns, the names and their associated patterns may be deployed to answer questions using the Knowledge Mining approach. Moreover, we can leverage entity instances and patterns to build NERC systems that at least pinpoint correctly typed answer beyond the predefined relations.

Chapter 7

SPINDEL: Crawling the Text Graph for the Dual Acquisition of Patterns and Extractions

7.1 Introduction

We explore an alternative to acquire Named Entities for NERC using a monolingual document collection using minimal supervision and few language-dependant resources. The algorithm has been designed with the main objective of being easy to adapt to new languages and new domains. A set of semantic categories like PERSON, LOCATION and ORGANIZATION are defined and a few examples are provided for each of them. The bootstrapping algorithm uses a query based exploration strategy to retrieve frequent patterns that co-occur with the examples. A subset of the most connected and promising patterns are selected to retrieve new entity names. The result is a list of categorized names and indicative patterns that are adapted to the domain of the original collection.

The key differences with previous bootstrapping approaches that have been presented in Chapter 5 are:

- The use of language dependant tools (such as a POS tagger for example) has been reduced to a minimum.
- Only a small set of seeds is needed with the aim of acquiring firstly the most frequent sense.
- A useful list of contextual patterns is acquired along with the list of entity names.
- The solution scales to larger collections by using a query based exploration and an incremental acquisition process.

The objective of the algorithm is to build simultaneously two different lists or relations for every predicate that we intend to learn.

- The first list is formed by NE (instances or occurrences) of an entity class. If we think of a NE class as a unary predicate like PERSON(x), that list will be composed of person names like `Barack_Obama`, `Bill Clinton` or `José_Luís_Rodríguez_Zapatero`. In fact, different variant names of the same real entity could be included in the list (`Zapatero`, `ZP`, etc.). We simply treat them as different instances and we do not address the problem of resolving them. Our focus is on acquiring large relations of categorized entity instances from a text collection that could be used for a NERC tool for example.

- The second list contains textual patterns that frequently co-occur with valid entity instances. These patterns would be useful for locating, classifying or ranking new names for that entity class. For example, consider instantiations of Hearst patterns like `persons_such_as` or `presidents_like` but also more general patterns like `told_reporters`. All of them are highly associated with entities of the class PERSON. Similar patterns for Spanish could be `actual_presidente` or `dijo_hoy_que`. Textual patterns are not sufficiently defined by their textual extent, in contrast, their relative position with respect to the entity mention is also required. Being more complete, the pattern `(text="dijo_hoy_que", pos=right)` would be a valid tuple for the relation storing patterns for predicate PERSON(X). Patterns are useful knowledge to build several information extraction subsystems. The recall of a NERC module can be improved using patterns while in Relation Extraction they are useful paraphrases. But patterns are often just indicative, and not exclusive of a given predicate like PERSON. For example, in the context *el presidente del Gobierno dijo hoy que* given a pattern like `dijo_hoy_que` is extrictly preceded by an ORGANIZATION (*Gobierno*). Nevertheless, the complete nominal group refers to a person though it is not a proper entity.

The algorithm starts with a collection of unannotated text D, a set of predicates to learn, and a set of seed tuples for each of the predicates. It proceeds in an iterative fashion, tuples are used to find mentions in the text collection from which the context is extracted. Each context produces a set of pattern types that are stored in a pool of candidates. Most frequent pattern types are selected as candidates for being good indicative patterns and are evaluated. Those that are confident enough would be used for finding new entities in the next step. This process is depicted in Figures 7.2, 7.3 and 7.4 as the iterative construction of a k-partite graph. The loop is simultaneously repeated for each of the predicates that we acquire in a experiment or scenario. The flow is synchronized at certain steps as tuples acquired for one predicate serve as negative examples for the rest.

The main ideas that the algorithm combines are summarized in these principles:

- **Dual bootstrapping** of names and patterns. Names are used to discover new frequent patterns and these patterns help to discover new names. The process is iteratively repeated. Furthermore, the assessment of new patterns is based on previous seeds and vice versa.

- **Exclusivity or *one sense per name*** principle. A name is allowed to fulfil only one predicate. This is a crude simplification of reality. It holds for a large number of instances in a domain although a few instances are ambiguous. For instance, `Madrid` in a general news domain could be considered as a LOCATION (city), a PERSON (surname) or even an ORGANIZATION (in reference to Real Madrid, the football team). However, only the context would help to distinguish the correct sense. This simplification can be seen as acquiring the most frequent sense among the entity mentions, but overall, it allows to bootstrap lists from few seeds.

- **Counter-training**. Examples that have been acquired for one entity class are used as counter-examples for the rest of the classes. The process is carried out in parallel for all classes at the same time so they compete to acquire a coherent meaning for names.

- **Query based exploration** of the collection. Full scanning of the document collection at each iteration is impracticable. We assume large collections and few new seeds and patterns in each iteration.

Table 7.1: Examples of textual context with mentions for `Bill_Clinton`

S_l	S_e	S_r	
	Bill Clinton	and his wife	as part
it is	time for President	Bill Clinton	to be as	big a
President	Bill Clinton	often laments that	his achievements
The past	week may convince	Bill Clinton	that his most	recent predecessors
	Bill Clinton	told reporters that	there

7.2 Definitions

The notation for different IE and bootstrapping systems is introduced before presenting the algorithm:

Document collection or **corpus** is denoted as D and represents a collection of documents in a language. The document collection is indexed in order to scale for efficient access.

Entity class is each of the semantic categories that we have identified as useful for our application. The most general entity classes are PERSON, LOCATION and ORGANIZATION. Entity classes are interpreted as logical predicates $PERSON : x \rightarrow \{true, false\}$. A predicate PERSON(x) will define a set of objects that belong to a set like {x: x is a person}. Different subclasses or other classes may be interesting depending on the application. Subclasses of PERSON, like ACTOR or SCIENTIST have shown to be useful for applications like QA while other classes like DISEASE or GENE would be of interest in specific domains.

Semantic model is the set of entity classes that the algorithm is using. We try to name semantic models with initials of the entity class (or their predicates) when it is possible. For example the PLO semantic model considers PERSON, LOCATION and ORGANIZATION classes. When dealing with several predicates in an abstract form we will use an index k to refer to them and the predicates would be denoted as $R_k(x)$ or R_k for short.

Entity instance is used for each of the named members of an entity class. Valid entity instances that fulfil the predicate PERSON(x) could be `Bill_Clinton` and `Clinton`. We will consider them as two different entity instances, ignoring whether they refer to the same real world entity or not. The set of entity instances is a relation in the mathematical sense and as it is used in relational databases. An example of the relation $E_{PERSON}(x)$ is shown in Table 7.2. For each entity class we will aim at obtaining a relation $E_k(x)$ of entity instances that satisfy the predicate R_k.

$E_{PERSON}(x)$
`Hillary_Clinton`
`Barack_Obama`
`Clinton`
`Obama`
`José_Luís_Rodríguez_Zapatero`
`Zapatero`
`ZP`

Table 7.2: An example of a relation with entity instances for the entity class $PERSON$

Entity mention refers to the concrete occurrence of an entity instance in a document collection, that is, the different realizations of instances of an entity. Table 7.1 show examples

Table 7.3: Text contexts for the pattern `p(and_his_wife,→)`

	S_e	S_r	
	Bill Clinton	and his wife	as
assertion by	President Clinton	and his wife	that
Long before	President Clinton	and his wife	began
his friend	Vince Foster	and his wife	
consul general	Alberto Boniver	and his wife	Suzy
Mao invited	Lin Piao	and his wife	to
expenses for	Davis	and his wife	Christina

of mentions for the entity instance `Bill_Clinton`.

Pattern instances represent unique sequences of tokens that are frequently adjacent to an entity instance of a given class. Pattern instances are also relations $P_k(text, dir)$ associated with a predicate R_k. An example of a pattern instance that would recall the class PERSON is `(and_his_wife,→)` where the arrow represents that, when the pattern is mentioned appears to the right of person names like in the examples in Table 7.3. In contrast, patterns that occur to the left of an entity mention are marked with $dir = \leftarrow$.

Pattern mention is therefore the ocurrence of a pattern instance inside a document.

Text Context is used to produce the graph that links pattern instances and entity instances. The distinction between instances and mentions is important. The algorithm acquires and classifies instances but it uses mentions to link entity instances and pattern instances when they are adjacent in the same text context. Two different types of text contexts are used. The contexts of entity mentions are used to generate left and right patterns instances which can be seen as links. If the context of a pattern mention matches an entity mention it also generates a link.

The *text context for an entity type* (C_e) is formed by three text chunks, $C_e = S_l S_e S_r$, where the substring S_e will match the text of the entity type and the left (S_l) and right (S_r) substrings will be considered pattern mentions.

The *text context for a pattern type* (C_p) is similar but its interpretation depends on the directionality. For a right pattern $p(S_r, \rightarrow)$ the context would be a substring $C_p = S_e S_r$. For a valid context the substring S_e must match a descrition of entity types, usually in the form of a regular expression, $match_{entity}(s_e)$. An analogous interpretation for left patterns exists. Table 7.3 shows several examples text context for the pattern `p(and_his_wife,→)` that produce candidate entity instances.

7.3 Architecture

The architecture of the bootstrapping algorithm is depicted in Figure 7.1 and is further detailed in Algorithm 9. The process begins with an indexed collection of documents D and a semantic model M that is formed by a set of predicates R_k from which we want to get the resources. For each predicate R_k we will obtain a list of entity instances E_k and a list of pattern instances P_k. For each of the predicates we have to define a set of initial seeds together with a regular expression to help us identify candidate entity mentions. For each of the predicates a set of initial seeds and a regular expression that help to identifying candidates for the list are needed. In our experiments, we have used a regular expression that requires the entity mention to be capitalized. The initial seeds are assigned full confidence. The algorithm proceeds in an iterative way repeating two phases, Pattern Expansion and Entity Expansion for each of the predicates.

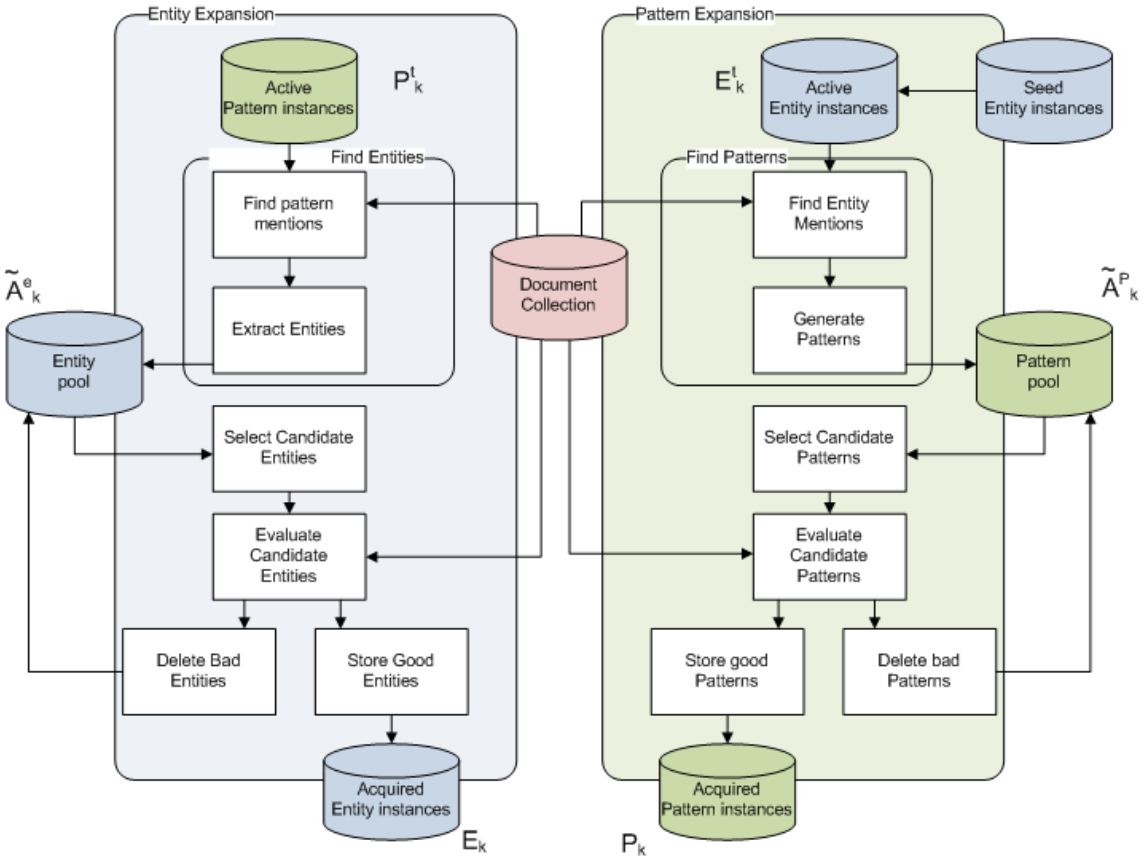

Figure 7.1: SPINDEL architecture

This algorithm produces a bipartite graph or bigraph $G(E, P, A)$ where E represent entity instances, P are pattern instances and A are the directed arcs between E and P vertices[1]. An entity instance is linked to a pattern instance, or viceversa, when a query locates an entity mention next to a pattern mention in a text context. In contrast with other bootstrapping algorithms, nodes are not only labelled but also discovered as iterations are performed. For example, shaded vertices (Figure 7.2) are not yet materialized, they need to be discovered via queries. The query exploration strategy could be classified as an Automatic Query Generation (AQG) strategy in the terminology of Ipeirotis et al. [2006]. In fact, the property of exclusivity combined with the learning of k predicates would end up building a $k + 1$-partite graph in the form $G(E_1, E_2, \ldots, E_k, P, A)$.

It has been shown and tested in different works [Lee and Lee, 2004; Thelen and Riloff, 2002; Yangarber, 2003] that bootstrapping several classes at once has a positive effect for improving the precision of the whole process. Entity types from one of the classes or predicates are used as counter-examples for the rest of the predicates. This property of exclusivity combined with the learning of k predicates would end up building a $k + 1$-partite graph in the form $G(E_1, E_2, \ldots, E_k, P, A)$.

7.3.1 Document Collection Pre-processing and Indexing

In order to scale the algorithm for larger collections, we have decided to implement a query-based exploration of the collection. An scan-based exploration of the collection would work with

[1]We would use A, from arcs, to denote edges instead of the more common E as the notation clashes in our context

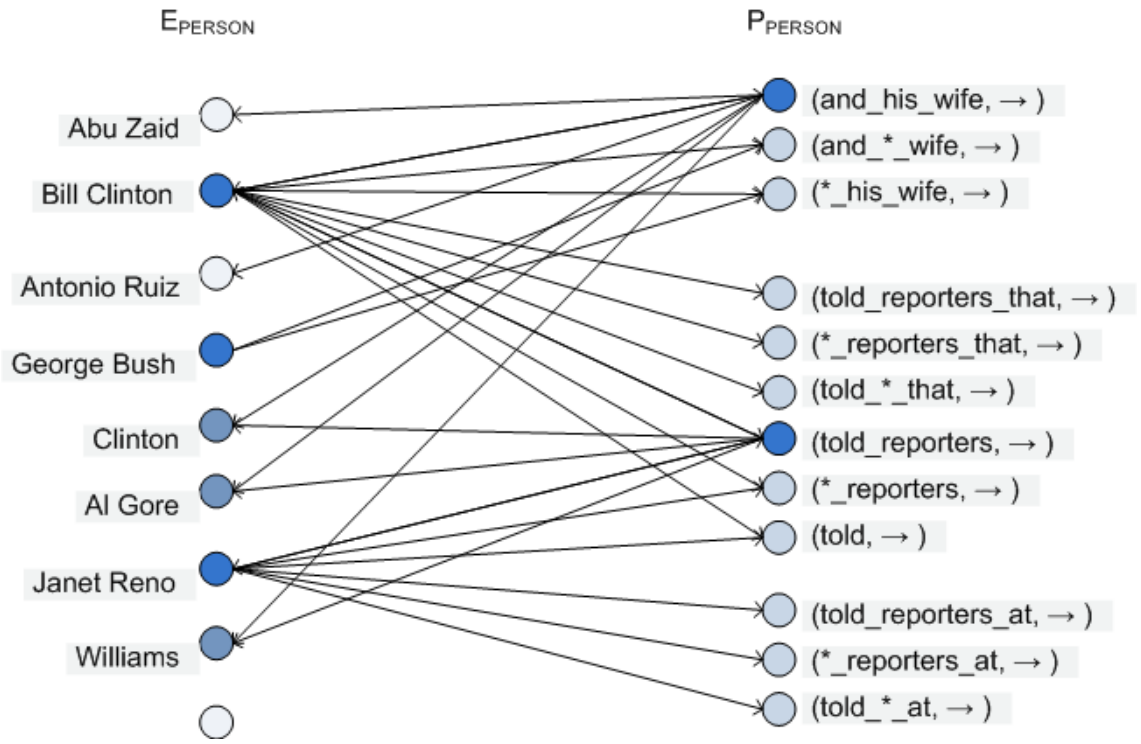

Figure 7.2: A graph for one Entity class relating entity and pattern instances

the whole document collection in each iteration of the algorithm. In contrast, a query based exploration uses queries to select only a subset of documents from the whole collection each iteration. This strategy allows to make use of larger collections, while in contrast, it requires the collecion to be preprocessed and indexed for efficient access based on queries.

For most of the experiments, the collection has been indexed at the sentence level, then each sentence is considered a document in the IR subsystem. Before indexing, each document is preprocessed, in the case of the experiments presented here documents have been splitted in sentences and tokenized. Stopwords have been indexed too as they are useful components in entity types and pattern types. In contrast, we have ignored punctuation marks.

For the concrete details of the implementation, we have used Lucene [**?**] as the IR engine for indexing and OpenNLP [Baldridge and Morton] for sentence splitting. Different configurations are possible wich we believe implies different trade-offs in results and time performance, but we have not experimented extensively with them. For example, sentence splitting is not strictly needed though we believe it helps to improve accuracy.

7.3.2 Pattern Expansion

In this phase, the seed entity instances are used to acquire new candidate patterns and a subset of them are evaluated and consolidated. The process is depicted in Figure 7.3 and Table 7.4 outlines the definition of the different parameters.

7.3.2.1 Find Entity Mentions

The first step consists in finding text context for the selected seeds. For each instance it generates a query and a filter that should be applied to the retrieved documents. For example, for the entity instance `Bill_Clinton` we can generate a phrase query `"Bill Clinton"` and a similar filter `Bill\s+Clinton`. The use of filters is useful depending on normalization and tokenization

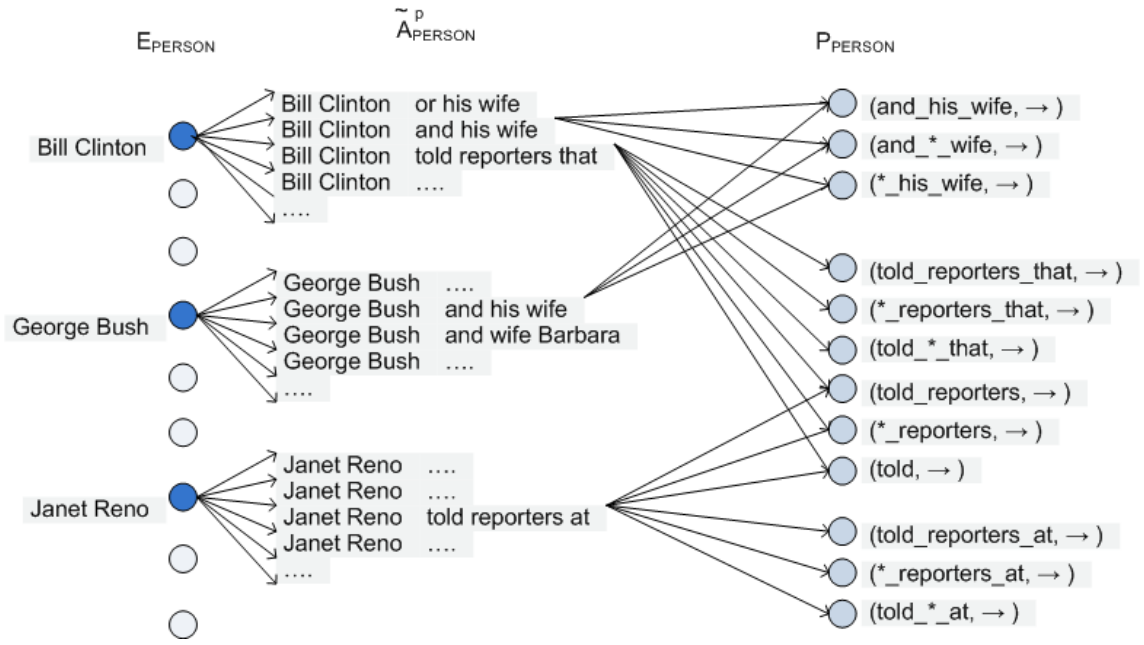

Figure 7.3: The process of building the graph: Pattern expansion

details of the collection text. For homographs of common names like *Bush* in English it could help to locate proper names correctly.

Once we have located mentions of entities and their associated contexts, the algorithm should generate pattern instances from the surrounding contexts. The whole process of finding entities and generating patterns is outlined in the Algorithm 7.

Algorithm 7 FindPatterns

Function: $FindPatterns(E, D)$
Input: E a set of entity instances, an indexed document collection D
Parameters: N_d^e the maximum number of documents retrieved in a query, w_l is the window size for a pattern
Output: $\tilde{A}_p(e,p) = \{\}$ a set of textual contexts that link entities e in E with candidate patterns p

for each e in E **do**
  $q \leftarrow generateQuery(e)$;
  $f \leftarrow generateFilter(e)$;
  $T \leftarrow filterDocs(retrieveDocs(q, N_d^e), f)$;
  for each t in T **do**
    $\tilde{A}_p(e,p) \leftarrow \tilde{A}_p(e,p) \bigcup generatePatterns(e,t,w_l)$
  end for
end for

7.3.2.2 Generate Patterns

There are several alternatives regarding the level of linguistic information that the method that generates patterns could use. We have used only lexical information in our patterns because our goal is to design and evaluate an algorithm that should be useful for several languages. For this reason no linguistic processing, including stemming is used. Our patterns are generated

Name	Description	Range
	General	
w_l	window length for patterns	1 -
t_w	time window before cleaning the candidate pool	1 -
t_{offset}	time offset before cleaning the candidate pool	0 -
	Patterns instances	
N_l^p	Number of maximum pattern instances acquired in each iteration, p superindex is used for parameters that affect patterns	1 -
N_d^p	Maximum number of documents retrieved per query	1 -
$\tau_{support}^p$	Threshold support	2 -
p_{rep}	Probability of select pattern instances that have been already used as queries	0.0 - 1.0
τ_{acc}^p	Threshold accuracy	0.5 - 1.0
τ_{conf}^p	Threshold confidence	0.0 - 1.0
	Entity instances	
N_l^e	Number of maximum entity instances acquired by iteration, e superindex is used for parameters that affect entities	1 -
N_d^e	Maximum number of documents per query	1 -
$\tau_{support}^e$	Threshold support	2 -
τ_{conf}^e	Threshold confidence	0.0 - 1.0

Table 7.4: Outline of parameters used in SPINDEL

Table 7.5: Generation of pattern instances for a context to the right of an entity mention, c is for candidate patterns and d is for discarded ones

c	w_1	w_2	w_3	and	his	wife
c	*	w_2	w_3	*	his	wife
c	w_1	*	w_3	and	*	wife
c	*	*	w_3	*	*	wife
d	w_1	w_2		~~and~~	~~his~~	
d	*	w_2		~~*~~	~~his~~	
d	w_1			~~and~~		

from the tokens adjacent to a entity mention, the text context. One parameter, w_l, controls the length of the window of tokens.

Several patterns are produced from a single text context by including wildcards. The function that generates candidate patterns substitutes each token with a wildcard. Wildcards that appear on the edge of a pattern will always be matched, therefore it is simpler to represent these patterns as shorter patterns. Besides, those patterns that are only composed of stopwords and wildcards are filtered. Therefore the stopword list is the only specific language resource that we use.

Examples of the pattern generation function are outlined in Table 7.5 for the context *and his wife* which is composed of three tokens. Assuming that the tokens *and* and *his* are in the stopword list the last three patterns are discarded (d) because they are just formed by stopwords and wildcards. Another example for Spanish and a textual context for a left pattern is presented in Table 7.6.

Table 7.6: Generation of pattern instances for a context to the left of an entity mention

c	w_{-3}	w_{-2}	w_{-1}	del	escaño	de
d	w_{-3}	*	w_{-1}	~~del~~	*	~~de~~
c	w_{-3}	w_{-2}	*	del	escaño	*
d	w_{-3}	*	*	~~del~~	*	*
c		w_{-2}	w_{-1}		escaño	de
c		w_{-2}	*		escaño	*
d			w_{-1}			~~de~~

7.3.2.3 Select Candidate Patterns

From the pool of patterns the algorithm should select the most confident ones to continue the discovery process. The evaluation of candidate pattern instances is a costly process and only a subset N_l^p of all the candidates is evaluated in each iteration. N_l^p stands for the number of instances (l) of patterns (p) that are selected for evaluation in eacg iteration (i) (see Table 7.4). The selection must avoid a type of bias that appears as a consequence of retrieving candidates by querying text. Consider using the entity instance `Bill_Clinton`, it should sieve those patterns that are associated only to a particular instance (`(and_Hillary_Clinton,`$\rightarrow$`)`) from those that are more closely associated with the predicate (`(and_his_wife,`$\rightarrow$`)`).

In order to select the candidate subset, the patterns are ranked in a two step process. First, the support of a pattern is used. According to Equation 7.1 the support of a pattern p for a predicate R_k is given by the number of different entity instances e of the predicate R_k that have been discovered in a shared textual context. Therefore entity instances e are labelled nodes that form part of E_k from previous iterations.

$$Support(p, R_k) = Pos(p, R_k) = degree(p, R_k) = \|\{(p, e) : e \in E_k^{t-1}\}\| \tag{7.1}$$

In our algorithm, *Support* is calculated as the association between entity instances and patterns in contrast to entity mentions (which we named as *Strength* in our comparison of the state of the art in Table 5.6). We believe that this definition avoids biasing the selection towards patterns that only co-ocurr with one single instance. A threshold value that requires a minimum support $\tau_{support}^p$ is defined to preselected the most frequent patterns first and avoid that bias. In the second step, the accuracy of each of the candidates is estimated using information from previously acquired instances. Equation 7.2 defines the accueracy as the ratio between the number of entities that support the pattern and the total number of entities. Such number is estimated from those entity instances that have been assigned to competing classes as defined in Equation 7.3. In order to avoid semantic drifting, a second parameter τ_{acc}^p that applies to pattern instance defines the minimum accuracy that is required for further evaluation.

$$Acc(p, R_k) = \frac{Pos(p, R_k)}{Pos(p, R_k) + Neg(p, R_k)} \tag{7.2}$$

$$Neg(p, R_k) = \|\{(p, e) : e \in E_j^{t-1}; \text{ where } j \neq k\}\| \tag{7.3}$$

The pool of patterns instances is conserved along iterations and its management resembles the front of exploration in a web crawler. As noted by Thelen and Riloff [2002], after some iterations the candidate pattern pool will saturate with repeated pattern instances, those that are more frequently associated with certain entity class. The problem with these behaviour is that only a part of the graph would be explored. The simplest solution consists in using only new candidate instances in each iteration and avoid repeated instances altogether. However, due to the iterative nature of the algorithm, patterns are evaluated with limited evidence and it could be beneficial to re-evaluate patterns periodically. On the other hand, it is required to consider

Algorithm 8 SelectCandidatePatterns

Function $SelectCandidatePatterns(\tilde{A}_p, R_k, E_j^t)$
Input: $\tilde{A}_p$ the pool of candidate pattern instances, E_j^t extracted entities until iteration t for all predicates j
Parameters: N_l^p the max number of candidates
Output: $P^t = \{\}$ the set of candidates at iteration t

r : generates random numbers $r : r \in [0, 1]$
$\tilde{A}_p^t \leftarrow order(\tilde{A}_p^t, Pos(p, R_k))$

while $(\|P^t\| < N_l^p \wedge \|\tilde{A}_p^t\| > 0)$ **do**

  if $p \in P_k \wedge r > p_{rep}$ **then**
    {Skip repeated patterns}
    $\tilde{A}_p \leftarrow \tilde{A}_p - p$
  else if $(Pos(p, R_k) > \tau_{support}^p \wedge Acc(p, R_k) > \tau_{acc}^p)$ **then**
    {Select frequent, accurate patterns and sometimes repeated}
    $P^t \leftarrow P^t \bigcup p$
    $\tilde{A}_p \leftarrow \tilde{A}_p - p$
  end if

end while

new patterns to explore new connections in the graph and acquire new instances. In order to enable this requirements, another parameter p_{rep} (see Table 7.4) is used to fix the probability that a repeated pattern instance is considered again. The global effect is that in each iteration the system is a mix of repeated pattern instance to re-evaluate and new pattern instances that help to discover new entities.

7.3.2.4 Evaluate Candidate Pattern

Candidate patterns are evaluated by querying the document collection again and retrieving associated entity instances. Note that this step is not redundant, candidate pattern instances are suggested by accepted entity instances. In this step, we query the document collection D for the mention of patterns in association with possibly new instances. In addition to consider the extraction of positive (Pos) and negative (Neg) entity instances we may have instances that have not been assigned to any entity class. The last ones are named as unknown (Unk) and defined in Equation 7.4. To evaluate pattern instances we use two measures, pattern Accuracy (as defined in 7.2) and Confidence (defined in Equation 7.5).

$$Unk(p, R_k) = \|\{(p, e) : e \notin E_j^{t-1}\}\| \tag{7.4}$$

$$Conf(p, R_k) = \frac{Pos(p, R_k) - Neg(p, R_k)}{Pos(p, R_k) + Neg(p, R_k) + Unk(p, R_k)} \tag{7.5}$$

An additional threshold for confidence (τ_{conf}^p) is defined to discard ambiguous patterns or those for which the algorithm cannot assess a strong association yet. In other words, it controls the aggressiveness of the acquisition process. Patterns instances that are accurate and confident are added to the definitive list P_k and removed from the pool. Patterns that are not accurate enough are also removed but those that the system is not confident enough are kept.

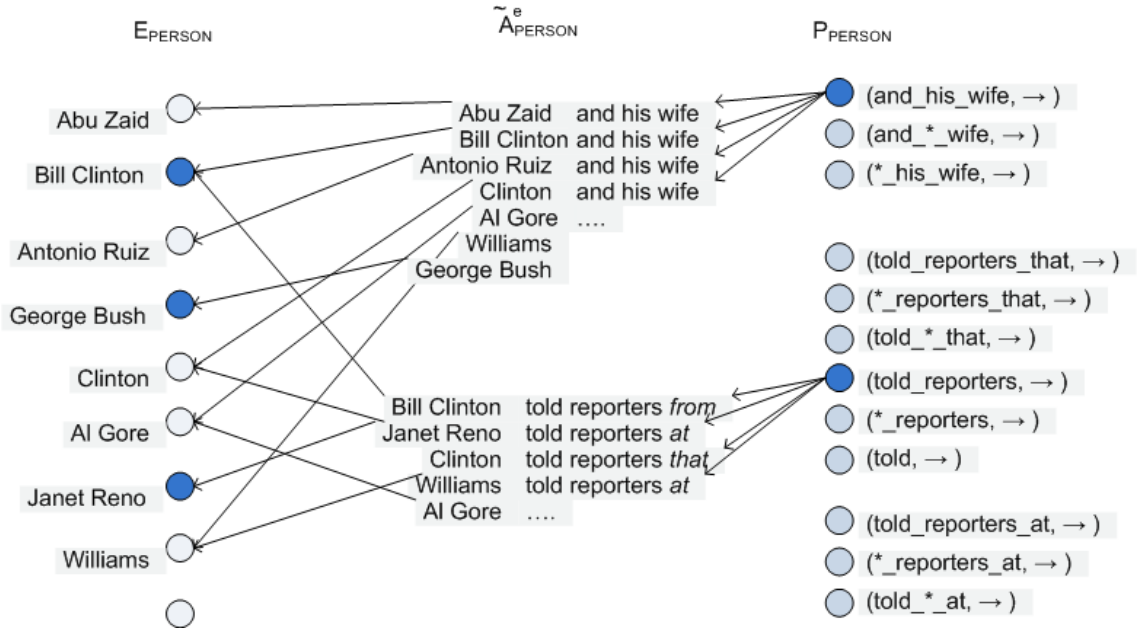

Figure 7.4: The process of building the graph 2: Entity expansion

7.3.3 Entity Expansion

Entity Expansion is dual to the process of Pattern Expansion, the process is at a large extent analogous (see Figure 7.4. Acquired pattern instances are used to query and filter textual context that carry a pattern mention and a candidate entity mention. For example, for the pattern `p(and_his_wife.→)` we generate a phrase query `"and his wife"` and the filter `and\s+his\s+wife\s+NERegExp`. All the candidate entity instances located are added to a pool.

Another subtle differences arise for those patterns that contain wildcards characters. The interpretation of the wildcard in the filter allows to match zero or one token. The query transformation process depends on the query language and the operations that the search engine provides efficiently.

The selection of candidate entity instances is simpler and is carried out in one step based on the degree of connectivity to pattern instance. Evaluation starts by querying the text and using patterns to score the hypothesis that an entity fulfils a predicate. The confidence of a seed is measured in terms of a variation of the NoisyOR model [Agichtein and Gravano, 2000]. In our use of NoisyOR we aim at estimating the confidence that an entity instance e fulfils the predicate R_k. In this context, confidence may be also interpreted as probabilities. In order to estimate that we take into account the different pattern instances $p_{i,slot}$ that co-occur in the text in a given direction or slot. Each of them is treated as independent evidence as states Equation 7.6.

$$Conf_{slot}(e, R_k) = 1 - \prod_i (1 - Conf_{pattern}(p_{i,slot}, R_k))) \tag{7.6}$$

$$Conf_{NE}(e, R_k) = Conf_{\leftarrow}(e, R_k) * Conf_{\rightarrow}(e, R_k) \tag{7.7}$$

A different confidence score is considered for left patterns and right patterns. The motivation of the Equation 7.7 is to correct the bias introduced by particular queries and require a correct entity instance co-occur with patterns from both sides. Entity instances with a confidence over the threshold τ^e_{conf} are considered correct for a given class.

7.3.4 Efficiency considerations

In principle it would be possible to store all the arcs that have not been evaluated. In practice, due to the Zipf distribution of arcs there would be a large number of low frequent candidates in pools $\tilde{A}_k^p$ and $\tilde{A}_k^e$. These infrequent candidates slow down the selection of promising ones. We have decided to periodically trim the long tail of the pool. If after a number of iterations (t_w) and a proportional number of queries the candidate is still unfrequent it is possibly of no help in the acquisition process and therefore may be discarded. Another parameter (t_{offset}) is designed to postpone the pruning process for a number of iterations. It allows to be flexible at the beginning of the algorithm when few seeds when the frequency of candidates is expected to be low.

Another consideration for efficiency is the number of queries that we issue for the acquisition of mentions and their contexts. With a proper management of the textual contexts in the pool, it is possible to intertwine both steps to reduce the number of queries. Though our current experiments use indexed local collections, this is especially interesting if the query engine is an external service which usually imposes a limit in the number of queries issued per day.

The complete algorithm is presented in Algorithm 9 and an outline of the different parameters includig their range of values is presented in Table 7.4.

Algorithm 9 SPINDEL meta-algorithm

Input:
D an indexed corpus,
R_k entity classes to learn
E_k entity instances seeds for class R_k,
$P_k = \{\}$ an empty list of patterns for class R_k,
$\tilde{A}_k^p = \{\}$ pool of contexts extracted from entities
$\tilde{A}_k^e = \{\}$ pool of contexts extracted from patterns

Output: E_k seeds, P_k patterns

$t = 0$ {Start iterations in time}
for each k **do**
  $\tilde{A}_k^p \leftarrow findPatternMentions(E_k^0)$
  $E_k \leftarrow E_k^0$
end for
repeat
  $t = t + 1$
  {Pattern Expansion: Find new patterns instances from entities}
  for each k **do**
    $P_k^i \leftarrow selectCandidatePatterns(\tilde{A}_k^p)$
    $\tilde{A}_k^e(P_k^t) \leftarrow findEntityMentions(P_k^t, D)$
  end for
  for each k **do**
    $evaluatePatterns(P_k^t, E_k^{t-1})$
    $storeGoodPatterns(P_k^t, \tilde{A}_k^e(P_k^t)) \wedge removeBadPatterns(P_k^t, \tilde{A}_k^e(P_k^t))$
  end for

  {Entity Expansion: Find new entity instances from patterns}
  for each k **do**
    $E_k^t \leftarrow selectCandidateEntities(\tilde{A}_k^e)$
    $\tilde{A}_k^p(E_k^t) \leftarrow findPatternMentions(E_k^i, D)$
  end for
  for each k **do**
    $evaluateEntities(E_k^t, P_k)$
    $storeGoodEntities(E_k^t, \tilde{A}_k^p(E_k^t)) \wedge removeBadEntities(E_k^t, \tilde{A}_k^p(E_k^t))$
  end for

  {Delete old candidates from the pool for efficiency}
  if $(t - t_{offset})\%t_w = 0$ **then**
    $deleteVeryOldPatternMentions(\tilde{A}_k^p)$
    $deleteVeryOldEntityMentions(\tilde{A}_k^p)$
  end if
until $(|goodPatterns(P_k^t)| = 0 \wedge (|goodSeeds(E_k^t)| = 0)$

Chapter 8

Acquiring Named Entity Resources in Different Languages and Domains

We have used SPINDEL for aquiring resources for unary predicates which could be useful for the task of Name Classification. Name Classification is one of the substeps in NERC systems whose performance can be improved by the use of gazzeteers or lists of names. Besides, lexical patterns that model the context of Named Entities are also useful for classifying NE mentions.

Basic generic NERC systems use a few simple categories like PERSON, LOCATION or ORGANIZATION to classify proper names. An additional miscellaneous classs (MISC) is sometimes used to group smaller categories for other objects like special dates (*Christhmas*, *New Years Day*), awards (*Nobel Prize*, *Oscar*), etc. Other schemes like those used in ACE [NIST, 2007] or HAREM [Mota and Santos, 2008] define a much more complex semantic model for the general domain. Other semantic models like the one proposed by Sekine et al. [2002] extends the number of classes and subclasses up to 200. Domain specific applications require new classes or refinements of the open domain, for example an application in the biomedical domain that detects interactions between drugs would need to identify the names of drugs and possibly distinguish between families or classifying them in active principles, generic or brand names for medicines.

While the question about what is a Named Entity could be discussed, a frequent feature for most of them in many languages is that they are marked by capitalization. However in some applications, interesting NE are not capitalized, like in the case of diseases or active principles. Moreover, sometimes this decision depends on language conventions or editorial guidelines. As the focused of our work is classification, we have ignored these problems by limiting our scope to capitalized NE. We believe that techniques can be adapted to other types in future work.

We have performed experiments in two languages, Spanish and English, and using three large collections of documents. In a group of experiments we have used the collection of Spanish and English news from the CLEF forum. Another experiment has been performed in an snapshot of the Spanish Wikipedia. Collection statistics are summarized in Table 8.1. All these collections are open-domain and unannotated collections where only raw text has been used. Besides, Wikipedia is a user-generated encyclopedia with its own conventions and stylistics and often much less curated content than a typical newspaper. The goal of the three experiments has been to bootstrap a large list of Named Entities and associated patterns from a few example seeds which could be useful for the development of NERC systems.

The experiments have initially used three entity classes PERSON, LOCATION and ORGANIZATION which we call the PLO semantic model. In some of the experiments we have added additional entity classes in order to test their effect on the accuracy of lists. Some experiments have tried to exploit the fact that our algorithm requires exclusivity between entity classes. On one hand this is an unrealistic assumption. On the other hand it could be used to improve accuracy by adding more entity classes. For example, we added an additional class like MIS-

Name	Source	Date	Lang	# Docs	# Tokens
EFE9495	EFE newswire	1994-1995	ES	454,000	136 466,251
LAT94GH95	Los Angeles Times	1994	EN	113,005	72 410,429
	Glasgow Herald	1995	EN	56,467	25 015,576
WIKI-ES	Wikipedia snapshot	10-2006	ES	279,195	69 515,149

Table 8.1: Description of the boostrapping document collections

CELLANEOUS to contrast the introduction of an ill-defined (but practical) class. For English we used new predicates like LANGUAGE or NATIONALITY as valid instances that appear always capitalized and are frequently considered NE in different annotated corpora.

8.1 Evaluation

To assess the usefulness of the acquired lists we have carried out two different kind of evaluations, that we have called direct and indirect evaluation. The direct evaluation aims at measure the quality of the list of entity instances. Large list of names that are correctly classified are desirable. In contrast, the evaluation of such large lists is very expensive because we do not have a gold standard to compare and should be carried manually. Moreover, list of patterns are even more difficult to judge as they can be associated with several entity classes.

For these reasons, we have decided to include an indirect evaluation too. This evaluation uses the acquired lists of resources to classify NE mentions in annotated text. The main advantage of this evaluation is that it can be performed automatically by reusing annotated corpora. We discuss more in detail the two evaluations.

8.1.1 Direct Evaluation

The direct evaluation targets the relations with entity instances that have been acquired for each of the entity classes like PERSON or LOCATION. For certain classes of entities which contain a finite number of elements, we may find or build the ideal list of instances, ext_I, from other sources of knowledge. Consider for example classes like COUNTRIES, STATE PRESIDENTS or CHEMICAL ELEMENTS. In contrast, for classes with a great number of instances there is no gold standard to compare. There is no large list of persons that overlap with the unannotated corpora.

Due to these limitations we have opted for manually judging a sample of the extractions to estimate the *Precision* of the lists. We can create a random sample of extractions and judge them to create a gold standard set of instances ext_G. This lists of instances may contain positive and negative judgements. This set may be created from the results of a single algorithm or by pooling the extensions of several algorithms.

Precision is extended to be calculated at a given rank point in Equation 8.1. $ext_S(c, r)$ provides the set of concept in a system list up to a rank r, the list of instances extracted at a given rank. $ext_G(c)$ is a function that is true if the instance t is included in the sampled gold standard.

$$Precision(r) = \frac{|ext_S(t, r) \bigcap ext_G(t)|}{|ext_S(t, r)|} \tag{8.1}$$

A ranked list of results may be evaluated using the Average Precision metric (Equation 8.2), a standard evaluation measure use in IR for judging ranked lists. $corr(r)$ is an indicator function that is 1 if the instance at rank r is correct, otherwise is 0. This measure evaluate that correct extractions are given before than incorrect extractions in the list.

$$AveragePrecision = \frac{\Sigma_{r=1}^{|ext_S(c)|} Precision(r) * corr(r)}{\Sigma_{r=1}^{|ext_S(c)|} corr(r)} \tag{8.2}$$

In general, our lists are ranked by the iteration in which they are extracted and confidence, which implies certain similarity to the initial seeds. The main motivation is also due to the iterative nature of the algorithm and their tendence to degenerate with bad previous results. In a practical situation is possible to trim the ranked lists if better precision (in contrast to larger recall) is favoured.

True *Recall* estimation is not possible for open lists of entities which is another significant problem with the direct evaluation scheme. In order to build a complete list of entity instances we should annotate each of their mentions which practically means solving perfectly NE Recognition and Classification or manually annotate a large unannotated corpora. For practical reasons, we use the number of extracted instances as the simplest practical substitute of *Recall*. We will plot $Precision(r)$ against the number of instances to compare the results of different classes and run configurations as for instance in Figure 8.3.

8.1.1.1 Criteria for evaluation

Manual evaluation requires some more explanation regarding the guidelines followed in the decisions to accept a correct entity extraction. We have decided to use only binary judgements, whether the entity instance represented by a string of text is a valid NE of the class or not. The decision is easy in commmon situations, like deciding if `José Luis Rodríguez Zapatero` or `Barack Obama` are valid instances for PERSON. The basic criteria has been to judge an entity instance as correct for a class if we can find at least a mention in context that supports the classification. However, difficult situations arise with high frequency and a detailef protocol is needed. This are some of the most common problems:

- A **String represents entity instances that belong to several classes**. The same string could represent several classes, like *Madrid* a location, a surname and a valid referece to an organization. In the proposed evaluation framework we have considered that as far as one mention support the class we would judge it as correct. Note that the more strict option of enumerating all the valid classes for a mention would be much more expensive to carry as evaluation.

- **Require common world or expert knowledge**. Some types maybe difficult to judge withouth the appropiate world knowledge. Consider for example the entity represented by `J.P. Morgan` as a valid ORGANIZATION. While any English speaker could consider this as a valid person name, to judge if it is an organization requires world knowledge about banking institutions. To cope with this type of problem the evaluation tool should provide access to mentions in context, for example by searching the collecction. This is enough in most cases, but there are further problems with those entities that are homonyms with common words (`Como`,the lake, against Spanish conjunction *como*), acronyms or minority senses where even these results would reflect useless context. Access to world knowledge like Wikipedia or Google could help at least to rule out an incorrect case.

- **Incomplete mentions**. Sometimes the extraction could be incomplete or partially incorrect. This is the case of for example considering `Rodríguez` as a partial extraction from a sentence that mentions *José Luis Rodríguez Zapatero*. Using a search engine would be difficult to judge if any mention refers to a person because the query language would not help to locate mentions of an isolated *Rodríguez* without careful re-indexing. Our criteria, for this situation has relied on knowledge that the judge is able to carry or acquire from Wikipedia to locate valid contexts.

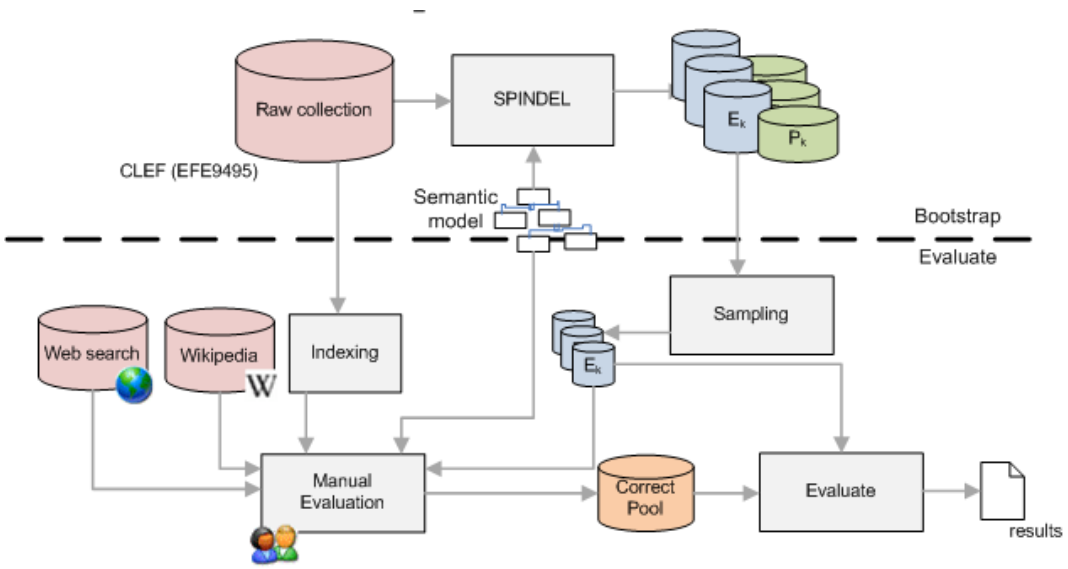

Figure 8.1: Process diagram for the direct evaluation of acquired list of NE

- **Ambiguous class definitions**. Using a limited number of entity classes (3 in most of our experiments) often implies some ambiguity in their definition. For example, a judge should decide whether a type like *Jonas Brothers* or *Hermanos Calatrava* are in fact PERSON or should be considered an ORGANIZATION as a group. A lightly supervised method dealing with huge quantities of text is more amenable to find these kind of difficult cases than the manual annotation of a hundred of texts. In fact, it is not easy to anticipate and provide sensible guidelines for these kind of problems.

The direct evaluation of the acquired patterns for classifying NE is even more complicated to perform manually. In order to decide if a pattern like `(el_presidente_electo,←)` is useful or not for recognizing instances of class PERSON would require to judge a large number of sentences. Besides, small details like whether there is a comma or not at the end that indicates an apposition could be meaningful but difficult to distinguish even for a trained judge. For this reason, we have decided to use an indirect evaluation procedure to judge the acquired pattern instances. Patterns acquired for a predicate are included in a simple NE classifier to improve coverage beyond the acquired lists of names.

In order to perform the manual evaluation we have developed a tool that aid manual judges. The tool helps to sample from the list, capture judgements and integrate search aids to decide in ambiguous cases like those mentioned above. It include automated searchs in the unannotated collections use for acquiring as well as programmatic Google and Wikipedia queries. Finally, in order to relief manual evaluators it also pools results from the same category to reuse judgements across experiments. The architecture of the evaluation tool is depicted in Figure 8.1.

8.1.2 Indirect Evaluation

We have found several problems with direct evaluation of the acquired relations of names and patterns. In particular it is difficult to evaluate the recall of the Name list (or entity instances) and we cannot evaluate the quality of the patterns. Moreover, it is an expensive procedure that cannot be automated.

For these reason, we have attempted to define an alternative evaluation procedure that we called indirect evaluation. The evaluation of the acquired resources is performed by using them in a related task like Name Entity Recognition and Classification. Fortunately, existing annotated

Name	Source	Date	Lang	type	# tokens	# NE tokens
CONLL-ES (2002)	EFE newswire	2000	ES	train	273,037	32,794
				testa	54,837	7,567
				testb	53,049	6,178
CONLL-EN (2003)	Reuters	1996-1997	EN	train	204,567	34,044
				testa	51,578	8.603
				testb	46,666	8,112

Table 8.2: Description of the CONLL evaluation collections

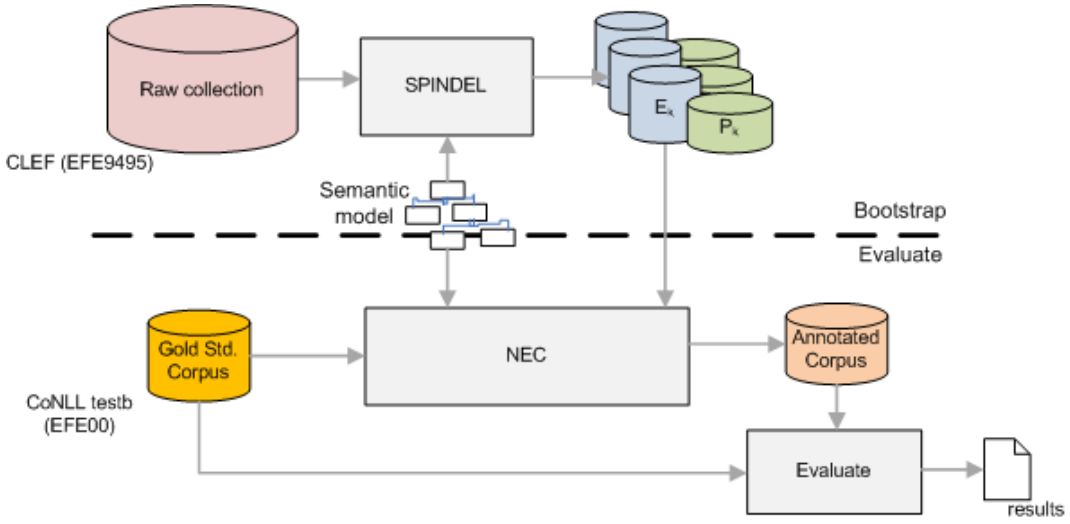

Figure 8.2: Process diagram for the indirect evaluation of acquired list of NE and patterns

NERC corpora could be used in our case to test the performance of the bootstrapping algorithm. Besides, it may also provide an insight into the trade-offs between supervised machine learning algorithm and those that use lighter supervision.

In the indirect evaluation we have used annotated corpora from the 2002 and 2003 CONLL shared evaluation [Sang, 2002] which is summarized in Table 8.2. These collections have been designed for supervised machine learning NERC and include a training set (train), a development set (testa) and a test set (testb). The CONLL collections use four different classes: PER (Person), LOC (Location), ORG (Organization) and MISC (Miscellanea). We use the test set (testb) for our evaluation because it allows comparing our results with those of supervised systems. However, it is important to recall that in our case we have used a large raw text collection (CLEF or Wikipedia) and lists of seeds for training instead of annotated data.

In the indirect evaluation we have restricted to the NE Classification task as depicted in the Figure 8.2. The objective is assigning the correct class to a chunk of text that have already been identified as a NE. The input is the text with the boundaries of NE mentions annotated and the goal is to use the lists to assign the correct class to each mention. We use the traditional measures of *Precision* and *Recall* (as defined in 5.5.3) because a classifier based on lists of rules (names or pattern instances) may abstain to produce a decision for mentions that are not in their ruleset. In some of the results we have used the assumtion that unclassified names belong to the majority class and in these cases *Accuracy* is used instead.

High precision entity lists should assign the correct class to unambiguous mentions. However, *Recall* will depend on the size of the entity lists and its overlap with the evaluation corpora. Even with large manual dictionaries it is not always possible to obtain good *Recall* numbers in

General			
w_l		3	
t_w		200	
t_{offset}		100	
Entity instances		Patterns instances	
N_l^e	40	N_l^p	40
N_d^e	400	N_d^p	400
$\tau_{support}^e$	2	$\tau_{support}^p$	2
		τ_{acc}^p	0.5
τ_{conf}^e	0.1	τ_{conf}^p	0.01
		p_{rep}	0.01

Table 8.3: Parameters for experiment PLO in EFE9495

open classes. However it is clear that the comprehensiveness of the dictionaries will affect the final numbers. The list of patterns could be used for improving recall under the assumption that patterns should co-occur with entities of the same class.

8.2 Experiments for Spanish with the CLEF corpus

We have used the PLO semantic model to learn from the whole EFE-9495 corpus. Each relation has been initialized with a handful of seeds that guaranteed the iterative process to start and be maintained for some iterations. For these experiments, seeds have been selected by a native speaker who knows the collection and its domain. The introspection process takes no longer than an hour and for each class no more than 40 seeds were used. Only the following rules have been considered:

- Seeds should appear several times in the collection.
- Avoid ambiguous seeds that could belong to several classes or predicates.
- The set should match entities with both genres because these could affect lexical patterns that do not use any kind of linguistic analysis.

The parameters of the experiment are summarized in table 8.3. The most critical parameters are the support for patterns ($\tau_{support}^p$) and entities ($\tau_{support}^e$) which are set to a value of 2. In the case of $\tau_{support}^p$ it requires that at least two entities instances co-occur with the same pattern in order to be considered a candidate. This value is justified because of the small number of inital seeds (40) and the low redundancy in the collection which means there will be few common contexts in the initial iterations.

8.2.1 Direct evaluation

Results of the direct evaluation of the PLO experiment are shown in Figure 8.3 and summarized in Table 8.4. The algorithm is able to extract very high precision lists of about 800 hundred elements. After that, precision goes down at different rates extracting about 30.000 entity instances per class. Precision for the PERSON predicate is fairly high and resulting dictionaries are directly useful. For the LOCATION and ORGANIZATION classes, the head of the lists achieves reasonable quality but the tail could be fairly improved. Manual randomized evaluation or indirect evaluation in a separated development testset could help to select appropiate sublists.

During the evaluation we identified different sources of errors:

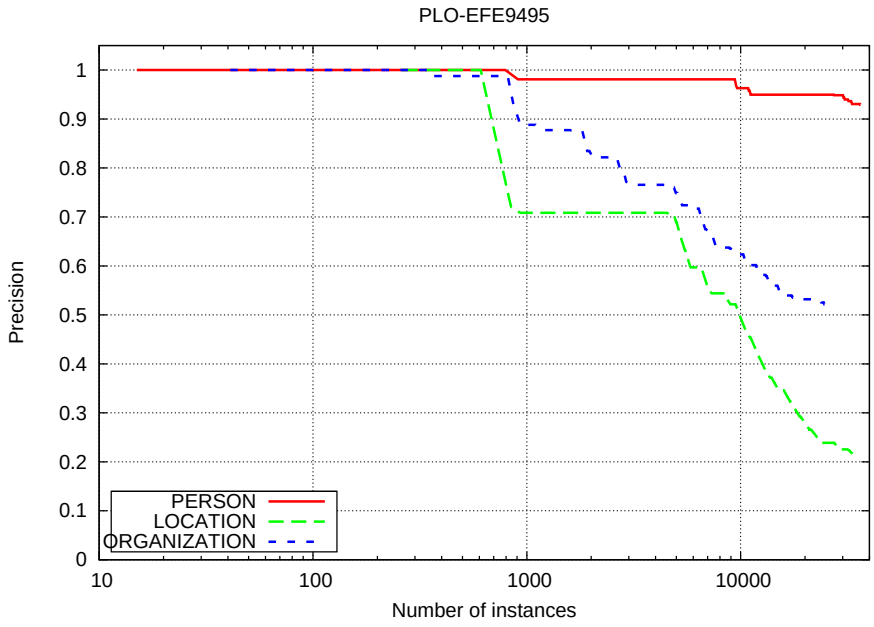

Figure 8.3: Direct evaluation of precision for PLO-EFE9495

	Precision			
# instances	PER	LOC	ORG	Mean
100	1.0	1.0	1.0	1.0
500	1.0	1.0	0.988	0.996
1,000	0.981	0.709	0.888	0.859
2,000	0.981	0.709	0.835	0.841
5,000	0.981	0.690	0.749	0.806
10,000	0.963	0.497	0.624	0.694
20,000	0.950	0.280	0.532	0.587
30,000	0.949	0.226	–	0.568*
at end	0.929	0.217	0.522	0.556
AvgPrec	0.948	0.527	0.671	0.715
	total # instances			
	36,316	33,335	24,673	

Table 8.4: Direct evaluation of precision for PLO-EFE9495

- The ORGANIZATION class acquires instances that are part of classes not in our present semantic model. Examples of these errors are events (*Juegos Olímpicos*) or prizes (*Premio Príncipe de Asturias*). One approach is to consider them to include additional classes on its own. Another approach that is compatible with existing NERC corpora is to model them as part of a MISCELANEOUS class.

- Regarding the errors of LOCATION, the main source of confussion are sport teams, which in Spanish are often ambiguous with locations, specially city names. For example, it is common to refer to the `Real Madrid` team as `Madrid`. These errors are rooted, at least in part, in the assumption that one particular mention is associated only with one entity and transitively with one class or predicate.

- Another common source of errors is produced by early semantic drift. By the design of the algorithm, it explores the space of names that are semantically similar and related to

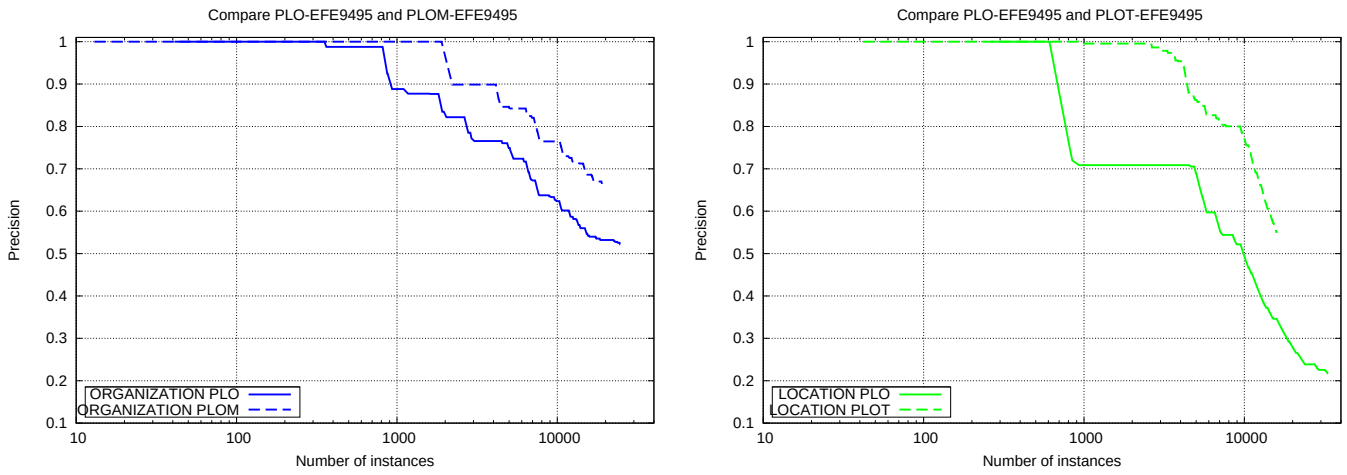

Figure 8.4: Comparison of Precision for ORG between PLO and PLOM (left) and for LOCATION in PLO and PLOT experiments (right)

Model	PER	LOC	ORG	M / T	Mean
PLO	0.948	0.527	0.671	–	0.715
PLOM	0.930	0.448	0.793	0.750	0,730
PLOT	0.948	0.874	0.811	0.409	0.760

Table 8.5: Average Precision for different Semantic Models

previous seeds. Considering several seeds enlarge the radius of exploration and usually enhances the results of the algorithm. In contrast, sometimes a predicate could burst onto a neighbouring area. That area would be better assigned to other class that has not explored it yet because not enough related entity instances have been acquired. The undesired semantic drift results in a loss in precision.

- The last important cause of errors is cascaded from the processing requirements. For example, errors in the simple recall oriented NE Recognition regular expression includes the beginning of phrases followed by a proper name. Sentence recognition and especially tables, like football results in EFE corpus, also account for some errors. Incorrect table segmentation chunks several names as one. Simple filtering rules that avoid incorrect segmented phrases could be used to improve results.

In order to mitigate some of the source of errors, we have explored the use of different semantic models that include additional classes or predicates. In one experiment we added a MISCELANEOUS class, which yielded the PLOM series of experiments. In another experimet we included a TEAM class in the semantic model, named PLOT, trying to ameliorate the problem between cities and teams. Experiments with PLOM and PLOT model have used the same parameter set than the PLO configuration outlined in Table 8.3.

In both cases, using additional entity classes help to improve the precision of the lists of entity instances. The TEAM class violates some of the modelling assumptions as it is usually considered a kind of ORGANIZATION. Their effect in *Precision* is only moderated and also imply a decrease on the number of extracted instances which indirectly may affect the recall. Finally, the overall results for the different experiments have been summarized in Table 8.5. Curves are summarized by using *AveragePrecision* which are averaged after that over the range of classes considered in the semantic model. These overall results also show that using a larger number of predicates help to improve results.

PERSON(x)		
Num	Round	Name
15	0	Fernando Arrabal
64	2	Teodoro Obiang
68	2	Salvador Allende
128	3	Peres
156	4	Edouard Balladur
157	4	Juan Antonio Samaranch
332	9	Grachov
423	11	Calderón
450	12	Colom
522	14	Joaquín Almunia

LOCATION(x)		
Num	Round	Name
262	6	Alicante
309	7	Castellón
380	9	Melbourne
390	9	La Rioja
500	12	Adén
569	14	Salónica
573	14	San Lorenzo
609	15	Querétaro
828	21	Real Madrid B
846	21	Atlético Marbella

ORGANIZATION(x)		
Num	Round	Name
41	0	Endesa
54	1	Renfe
59	1	CNT
78	1	OLP
80	1	Cancillería
100	2	Fiscalía
106	2	FSM
140	3	Procuraduría General de la República
201	4	ACB
270	6	Democracia Cristiana

Table 8.6: Random examples of entity instances in PLO-EFE9495

A small sample of acquired entity and pattern instances are outlined for the three predicates in Tables 8.6 and 8.7. The *Round* column represents the iteration number when the instance was extracted. The *Num* column is the rank in the extracted list. In the same iteration instances are ranked by confidence as a result of the evaluation. Finally, it is important to highlight the potential of pattern instances as useful triggers for NE Recognition and Classification.

8.2.2 Indirect Evaluation: Name Classification in CONLL 2002

In the indirect evaluation we solve the simpler task of NE classification as a mean of evaluating the lists of entities instances and patterns. We use two different baselines for comparison. The

Left patterns			Right patterns		
Num	Round	Name	Num	Round	Name
0	1	Gobierno del presidente	6	1	, ### esta tarde
1	1	Gobierno del ###	9	1	, vencedor
12	2	gobierno del presidente	21	2	y el ex
13	2	presidente del país ,	26	2	, viajará
29	3	actual presidente	34	3	, y su colega
30	3	jefe del Estado ,	42	3	, visitará
47	4	palabras de	42	3	, visitará
50	4	cuyo ### ,	49	4	, y el líder
60	5	presidente ,	63	5	y el presidente
61	5	reunión con	65	5	se entrevistó'

Left patterns			Right patterns		
Num	Round	Name	Num	Round	Name
0	1	México y	1	1	y México
2	1	embajador de	4	1	, Venezuela
38	2	embajada de	49	2	y España
39	2	Embajada de	50	2	y Francia
77	3	estancia en	75	3	es un país
78	3	embajador en	76	3	e Italia
111	4	llegará a	134	4	procedente
112	4	regreso a	163	5	para asistir
149	5	regresará a	165	5	para entrevistarse
150	5	regresar a	201	6	, adonde

Left patterns			Right patterns		
Num	Round	Name	Num	Round	Name
0	1	sede de	10	1	de Cataluna
1	1	fuentes de	17	1	de España
31	2	acuerdo con la	35	2	en Bosnia
32	2	intervención de la	48	2	española
69	3	añadió que la	93	3	estadounidense
70	3	disposición de la	96	3	de Colombia
107	4	pidió a la	108	4	norteamericana
109	4	ONU y la	126	4	chilena
146	5	solicitó a la	182	5	de la República
147	5	respaldo de la	185	5	argentina

Table 8.7: Random examples of pattern instances in PLO-EFE9495

first baseline (ORG) assigns the most common class to each name mention, in this collection, i.e, ORGANIZATION. The second baseline was proposed in the CONLL shared task and it assigns the most common label for a NE token as is seen in the training data. This baseline would be roughly equivalent to produce a dictionary from the annotated training data.

We assume perfect NE recognition and each of the entity classes from the bootstrapping algorithm are mapped to their equivalent in the CONLL corpus. For instance, in experiments with PLOT semantic model both ORGANIZATION and TEAM are mapped to ORG. The first NE classifier (*entities*) uses a naive pure dictionary approach to classify names. Only if the exact mention appears in one of the list of names, it is tagged with the corresponding label.

Table 8.8: Name classification in CONLL-ES collection

	baseline		entities			entities+patterns		
	CONLL	ORG	PLO	PLOM	PLOT	PLO	PLOM	PLOT
P	26.27	–	77.33	**78.85**	78.72	66.12	73.65	66.35
R	56.48	–	54.34	51.53	41.58	57.97	**61.73**	56.62
F	35.86	–	63.83	62.36	54.42	61.78	**67.17**	61.10
Acc	–	39.34	64.04	66.24	62.18	63.17	**71.29**	62.50

	baseline		entities		entities+patterns	
	CONLL	ORG	PLO	PLOM	PLO	PLOM
P	26.27	–	**78.89**	77.82	73.42	73.86
R	56.48	–	47.34	46.64	**53.86**	53.75
F	35.86	–	59.17	58.33	62.14	**62.22**
Acc	–	39.34	61.30	61.01	62.60	**63.05**

Table 8.9: Name classification in CONLL-ES collection with Wiki dictionaries

Precision, Recall and F-measure for this dictionary-based classifier are presented in Table 8.8 for the CONLL-ES test set. Accuracy results are produced when we add the majority rule, that is, for those names in no list, the majority class, ORG, is assigned. A second NE classifier (*entities+patterns*) uses the list of acquired patterns. When a name mention is not found in the NE lists, their contexts are matched with the lists of patterns and the name is classified with the most voted class.

Results for the first classifier show a significant improvement over the baselines. Precision is also enhanced by modelling more NE classes. On the contrary, we observe that when the modelling assumptions are too restrictive, Recall could be seriously affected. If we compare the performance of the classifiers using the PLO model and the PLOM model, the decrease in Recall is not compensated for the increase in Precision. Therefore, the overall performance of the classifier is not improved. Regarding the use of patterns, results show that there is a consistent improvement in Recall while the global improvement (F-measure) is not always guaranteed.

Nevertheless, best results have been achieved when the PLOM semantic model is used and dictionaries with entities and patterns are integrated. Our overall results are lower than most supervised approaches but it is important to recall that our approach is only lightly supervised. It does not require annotated text but, only a few name instances for every class. We believe that this aspect makes up for the small loss of accuracy.

8.3 Experiments with the Spanish Wikipedia

Similar experiments have been carried out by learning the dictionaries from the snapshot of the Spanish version of Wikipedia (WIKI-ES). The experiments were conducted with the same parameters for the PLO and PLOM semantic models. Table 8.9 presents the results for NE Classification using the CONLL test corpus. Precision for the classifier using entity dictionaries is comparable to the models trained on EFE corpora. In contrast, Recall for the two models is lower than their respective counterparts.

Although patterns from Wikipedia help to achieve higher recall, their contribution is lower than EFE experiments. We believe that both effects are due to the test and bootstrapping corpus being from different domains, which has an impact on the overlap of NEs and patterns.

	baseline		entities		entities+patterns	
	CONLL	LOC	PLOM	PLONT	PLOM	PLONT
P	71.91	–	63.30	**66.32**	62.96	65.48
R	50.90	–	40.07	41.87	47.17	**48.90**
F	59.61	–	49.07	51.33	53.93	**55.99**
Acc	–	29.45	60.08	61.97	60.69	**62.61**

Table 8.10: Name classification in CONLL-EN collection with LAT94GH95

8.4 Experiments for English with the CLEF corpus

In order to test the usefulness of the approach for other languages we have decided to perform similar experiments for English. The setting and the parameters are similar to the Spanish collection. We use the English CLEF collections for NE and pattern acquisition and the CONLL 2003 English collection for the evaluation. Some adaptations are required because the conventions differ from Spanish language. For example, nationalities and other demonyms are capitalized and therefore tagged as NE. Days and months are often capitalized too. In this case, the CONLL corpus does not tag them as NE but it is convenient to avoid errors for other predicates.

We have experimented with two semantic models, PLOM that groups these classes in the single Misc entity class and PLONT which uses two additional separate entity classes, NATIONALITY and TIME. Results are reported in Table 8.10.

As it was already observed for Spanish, the use of patterns in the classifier improves recall figures. Comparison of results for PLOM and PLONT models, supports that modelling additional entity classes in the bootstrapping process helps to achieve slightly higher Precision.

Nevertheless, the most significant fact is the difference between results for Spanish and English. The CONLL baseline built from the annotated training performed better than our approach. A possible cause is the degree of NE overlap among the test and bootstrapping collections. They belong to different time spans and different sources. This is supported by the fact that higher recall is obtained for locations, which tend to change slowly along time in news, while this is much lower for persons and organizations. However, further research is needed to clarify this aspect. Another hypothesis for this decrease in performance could be the quality of the initial seeds.

8.5 Discussion

We have presented a new bootstrapping algorithm aimed at the acquisition of useful NE resources for multilingual information access applications. In our opinion it addresses a common situation in these kind of applications. A large document collection is available but there is no annotated training data in the domain. It is feasible to specify a few seeds for each of the entity classes and extend them to build larger NE and pattern lists. Both lists are directly useful to build a prototype NERC, even if their performance is lower than those based on supervised Machine Learning or Knowledge Engineering. The trade-off is interesting in multilingual applications because of the costs associated with annotating training corpora or developing rules in several languages. The initial seeds are easy to generate and they require only a few persons/hour work and basic understanding of the target language.

The experiments carried out for the traditional NE classes show that the algorithm is very productive. It expands the number of seeds by a factor larger than 500 with a reasonable precision. Besides, the contextual patterns are useful to generalize beyond the acquired names, in particular if they are obtained from the same corpus. This is achieved by the combination of dual bootstrapping of several exclusive classes, query based exploration, throttling and the

incremental assessments of NE and patterns based on previously acquired types introduced in SPINDEL. A larger number of seeds would improve the quality of the final results and external resources like Wikipedia classifications can be leveraged to use as seeds.

Our bootstrapping algorithm does not assume the use of language-specific NLP tools like POS taggers, chunkers or parsers which could not be available or tuned to the domain. It is fairly language independent, it relies in a simple regular expression that uses formatting cues like capitalization and general heuristics for European languages to locate candidate NE. However, it uses a language specific list of stopwords to filter and generalize candidate patterns which in contrast is a common requirement for large collections that include an IR component. Future work should explore alternatives to other languages like German or Arabic where formatting is not so useful.

Chapter 9

Conclusions and Future Work

9.1 A QA system for Spanish and its evaluation

One of the contributions of this thesis is the development of a textual QA system for Spanish. The MIRACLE QA system has followed a Knowledge Mining approach to QA by combining techniques from IR and IE. A cornerstone of our QA system has been the use shallow language analysis tools for Spanish including morpho-syntactic analysis, lemmatization and NERC as provided by STILUS tools. As a result, our architecture uses a shallow representation of information requests that has paid special attention to Named Entities. Additional knowledge required for QA has been included in several forms, for example we use linguistic rules for question classification and common heuristics for ranking, like frequency or proximity between terms and candidate answers. The main resource for Answer Extraction is based on typing based on the NERC output which is based on compiled knowledge resources (list of categorized Named Entities) and additional rules to improve recall and include some additional types. Besides, during the last years additional semantic knowledge based on ontological relations has been included. With the years, the QA system has become more intensive in the use of knowledge resources, linguistic resources and rules for several modules.

Our system has been evaluated since 2004 in the main QA@CLEF track but also in related tracks like the Real Time QA Exercise or the WiQA pilot on novelty detection in Wikipedia. As a result we have contributed to the advancement of the Spanish QA community almost since their beginning. Being our system and the task evolving objects it is difficult to measure the improvement with a single figure . Nevertheless, the range of new scenarios that have been addressed gives an idea of the qualitative improvements and the limitations of our QA system and current research. In order to highlight some of our modest achievements:

- In CLEF 2005 the architecture described in Chapter 3 was setup and results were on part with other QA systems for Spanish as well as other European languages. With no doubt an important addition was the proper recognition and classification of Named Entities using STILUS and the adapted filters.
- That year some additional cross-lingual experiments were send and the study highlighted also the importance of the proper translation of Named Entities.
- The focus of our runs in the main QA task in 2006 was in the generation of queries for multiwords and Named Entities. Though our results in the main task were lower than the previous year, it contrasted with our participation in the Real Time QA experiment where the system was highlighted among the top performers in MRR and response time.
- The system presented in CLEF 2007 started to handle new kind of questions. It was particularly challenging the introduction of topic related questions simulated as dialogues between the user and the QA system.

- Besides, the CLEF 2007 evaluation also innovated in the use of a different text collection, Wikipedia. We adapted the MIRACLE system to include several streams, though the evaluation uncovered issues with the new genre and the combination of answers from different sources.

9.2 Reflections on the evaluation of QA systems

It is difficult to extract a single clear quantitative conclusion from our evaluation in CLEF. This aspect requires to reconsider our methodological approach and also the evaluation. I believe that one of the problems is that the CLEF@QA task has become a moving target. It started with a clearly established closed task but progressively included new challenges every year. The challenges have broaden the QA task and make it more realistic. On the other hand, it made difficult to compare the evolution along the years. At the same time, our goal has been to improve the QA system and introduce new techniques, modules and resources that could have an impact on results. Such a complex system may face software bugs under the tight schedule of a yearly evaluation. Moreover, some of the module improvements, like the NERC module, may blend algorithmic and resource improvements. In those cases the individual contribution of each improvement have been difficult to assess in a quantitative way in the current evaluation.

As a consequence in such an dynamic scenario it has been difficult to account for a single cause of improvements. Beyond our own experience, we believe that this problem may plague other QA systems and it is therefore an issue to take into account in designing future evaluations. In general, systems are too complex, include several modules and different execution flows which cannot be described in detail in scientific papers. The overall result is that despite the common agreement that the QA community have advanced in the task, it is difficult to pinpoint without doubt to the key requirements of a QA system for achieving certain performance and provide real value to final users. This is in fact a strong self-critique to the current evaluation presented in this work. Fortunately, resources created at CLEF may lead to a better evaluation in the future.

In conclusion, I have the conviction that further advance in QA should continue by further research in our current evaluation methodology. First of all, the QA community need to segment the task in more comprehensive and manageable parts. For example, QA is fairly segmented with respect to the question types and specialized techniques are devised for factual, definitions or procedure questions. In addition, such segmentation may be applied to specific linguistic problems and key functional modules too. Applying the *divide and conquer* metaphor would provide, in our opinion, a deeper understanding of more focused problems. For example, segmentation may be carried for questions (or requests) along language phenomena typologies, like those including ambiguous names, word sense disambiguation, frequent vs infrequent facts. That would help to study if specific techniques whose usefulness is frequently disguised in larger question sets are useful for certain difficult cases. In the same line, applying evaluation to specific modules may improve the modules itself and their combination. In our experience, a current problem with QA is that the composition of functionalities is not trivial and there are hidden dependencies in the combination of modules. For example, the distribution of facts among retrieved documents may have a deep impact for extracting answers. Making a simile with electrical components, modules may have *operational ranges* that if are not considered may harm overall performance.

In the same analogy, models that provide guidance on how to connect heterogeneous language processing models would be required. Evaluation should move to solutions that help to integrate specialized approaches, support the repeatability of experiments and the access to final and intermediate results. Ideally, systems composed of best-of-breed specific solutions may be harmonized and provide value for final users. Feedback and logs from users would enable to propose *live labs* with a continuous stream of questions that help to obtain more confident

evaluation results.

9.3 Towards adaptive QA systems

Our MIRACLE system has adapted the approach based on the use of IE components and relatively shallow language analysis. This approach was already common to English systems that took part in TREC evaluations. With small differences, it has been reproduced during CLEF evaluations in other European languages and it has been common (if not the dominant one) also in NTCIR. Their widespread use and the constant improvement in results are enough to justify the following statement. The knowledge intensive approach is right now the most sensible approach to build effective QA systems for factoid questions. This is in spite the interestingness and advances of redundancy based QA systems, those based on deeper language analysis and even those using end-to-end machine learning.

However, knowledge intensive QA systems have also their problems and some of them have reflected in our participation in CLEF. We believe that they can be tracked to other QA systems too. They require a large amount of manual work to achieve competitive results. At great extent, those teams that have introduced additional knowledge and tune results obtain better results. This is true whether the approach consist on the manual acquisition of rules or indirectly use annotated corpora to train certain components. It would be desirable if we could have methods to reduce the amount of work required. We have looked at bootstrapping methods as an alternative for the acquisition of useful resources for the QA process. In particular, we focus on the acquisition of NE knowledge from large textual collections. Though the methods may apply to huge collections like the Web, the challenge in this case is that CLEF and other domain collections would contain much less redundancy than the Web.

As a consequence of the heavy knowledge requirements, QA systems have shown difficult to adapt to different domains and languages. Few teams have been able to replicate QA systems in languages other than English and its own native. Though the ideas are portable, resources and tools are not. This hinders multilingual applications at large and multilingual QA systems in particular.

Adapting or re-targeting a textual QA system from one domain to a different one is also difficult. Research systems are typically tailored to an specific domain. This has been our experience with CLEF@QA tasks where changes in the collection from news to Wikipedia, and then to the JRC-Acquis collection have introduced modifications in almost every module. Though it is naive to believe that no modifications should be carried on, the required expertise to achieve reasonable performance need to decrease.

Domain adaptation implies changes on the collection, on the types of questions that are relevant, how the information is conveyed, etc. Different collections may require different retrieval strategies on how to find relevant documents and passages. Relevant information is also expressed in different ways across domains, as we have seen with definitional sentences in news and Wikipedia. Finally, language analysis tools are also tailored for the language of certain domain (typically, news) and may be adapted or rebuilt for others. This is particularly true for language analysis tools that deal with semantics like the type of answers.

I believe that language and domain adaptation represent a challenge for knowledge intensive QA systems. Being able to acquire useful knowledge resources without the deluge of large annotated data or requiring too much work from human experts would help to popularize the use of QA solutions. An additional complication is that language analysis tools may be scarce for certain languages and applications too. In our approach, we have looked again towards the option of bootstrapping knowledge from textual collections.

Our proposal for factual QA systems, that goes beyond the work presented in this thesis, consist on discovering the main concepts that are part of questions and elicit the important relations that are subject to be asked. Once, the main generic concepts, particularly Named

Entities, and their relations have been identified, knowledge resources may be acquired including lists of NE, list of relations between NE and patterns that help to recognize both in texts using almost no language processing tools. Some works have started to show the feasibility of these approach for relations and therefore we decided to focus on the acquisition of NE resources. However, we have attempted to analyze and provide a common definition for both tasks because both are required for such a QA system. On the other hand, in our experiments we have focus on how domain and language adaptation affect the recognition and categorization of Named Entities. NERC is too tailored to well studied classes where annotated corpora exist. Alternative techniques that require less supervision are needed because most languages and domains cannot count upon the use of annotated corpora.

9.4 A common architecture for bootstrapping systems

We have studied several bootstrapping algorithms devised for acquiring knowledge from text collections under different conditions. Our study have explored their use in several tasks, but particularly for the acquisition of NE and their relationships. We have attempted to provide an unified view of both task as Large Scale Information Extraction tasks that use a collection to complete relations (or lists) of instances (or tuples) with different arity. We have also started to develop a unified view of algorithms based on a graph description that links relation instances and pattern instances through their mentions in texts. Our research has compiled a number of techniques that allow to reduce one of the main problems with bootstrapping and lightly supervised algorithms, semantic drifting. Among them, different measures for the evaluation of extracted instances and patterns have been included in our graph view of bootstrapping. In our analysis we take into account other factors like language, knowledge sources for bootstrapping and exploration techniques. Among them, language and the use of language analysis tools are of particular interest. Most works on bootstrapping have focused on English. Most works on Relation Extraction require syntactic analysis or at least NERC for typing arguments. In turn, Named Entity extraction uses shallower language analysis tools but often requires POS-tagging. The trend is to reduce analysis as the size of the collection increases, but on the other hand, few attention has been provided to other languages and the language independence issue.

Finally, we present a bootstrapping algorithm aimed at the acquisition of relevant knowledge for different domains and languages that we evaluate on the acquisition of large Named Entity resources. Moreover, the algorithm is suited to large collections but not huge, as the Web where plain redundancy is enough to provide reasonable results. Our efforts is directed to the typical collections that are used in textual QA in particular domains. We have attempt to include the learned lessons from state of the art. The characteristic features of the algorithm are:

- it is a graph-based algorithm that incrementally explores text. It establishes links between instances and patterns that help to discover the extensional description of the semantics (list of instances) as well as their characterization in text (list of patterns). The process resembles a focused web crawler that follows links and estimates their relevance to explore specific webs.

- It is a dual bootstrapping algorithm, in other words, it is able to extract not only factual knowledge like instances but also to acquire patterns or rules that may help to generalize concepts.

- We have attempted to reduce language tools requirements to a minimum, in order to be able to apply it to various domains and language.

- In order to reduce the risk of semantic drifting it uses simultaneous and exclusive acquisition of different predicates (Named Entity classes or relations).

- It uses a query based exploration strategy which is more appropriate when there is a small proportion of seeds with respect to the size of the collection.

9.5 Bootstrapping resources for language and domain adaptation

We have carried experiments on different collections and languages in order to assess the results of the algorithm. The experiments attempt to acquire basic NE dictionaries or unary relations like PERSON, LOCATION and ORGANIZATION from a few initial seeds. We have evaluated the quality of the lists manually but also on standard hand-labelled datasets for NERC that are available in Spanish and English. While results are not as good as supervised systems we believed that they could be useful for the task. We also found that instance patterns may be also used in a NERC and they will improve recall considerably. Finally, experiments for different type of collections like news collection and Wikipedia show that patterns are more useful when extracted from the same domain.

Several open questions remain regarding the difference in performance across domains, languages and collections. We have performed experiments in Spanish and English as both languages have resources where we could perform an indirect evaluation. So far, our results are better for Spanish than English in a similar corpus (CLEF). Nevertheless, corpus size and the quality of the seeds may be the cause of the difference. The experiments have to be extended to other languages and collections in order to draw additional conclusions along the longitudinal axes. For example, at what extent can we apply the algorithm to other languages? from those closer (Portuguese or German) to those falling under a different family (Chinese, Japanese or Hindi).

Our experiments have started with a fairly basic semantic model in a broad domain, like that defined by CLEF corpora or Wikipedia. This corpora was selected because our exploration require available resources for its evaluation too and CONLL corpora shared a similar domain and was available in several languages. We have attempted to apply our algorithm in other domains, like clinical notes [Pérez-Laínez et al., 2009], with certain degree of success. NE lists were bootstrapped and useful for a de-identification task which is close to NERC. Nevertheless, further methodical analysis is required in at least two directions. The first one is on the adaptation to different semantic models and the second on how the features of a given entity class affect performance. Consider the acquisition of instances for the class PERSON against other like DRUGS or DISEASE from the biomedical domain. How features like the size of the expected relation and redundancy of mentions affect the acquisition process? Another line for future work consists on the inclusion of complex taxonomies of classes and the most effective method to proceed. The population of complex taxonomies (or ontologies) are required for their application in Question Answering.

Finally, several features of the bootstrapping algorithm are well suited to extend the acquisition of names and patterns from the web as it is based on query exploration and it makes an economic use of queries. In fact, the redundancy of web information may be helpful to acquire Name Entity instances. In fact, in lots of cases it could be a complementary source combined with the domain corpus.

9.6 Results dissemination

The main bulk of this thesis have been carried out in the context of our participation in the CLEF QA task. We have presented our results and experiments at the annual CLEF workshop since 2004, though we focus on the period from 2004 to 2007. An initial description has been always presented first in the Working Notes. Then, revised versions including quite often additional experiments and explanations have been published in the LNCS Springer Proceedings.

- CLEF 2004: initial system described in de Pablo-Sánchez et al. [2005b] (Working notes: [de Pablo-Sánchez et al., 2004])
- CLEF 2005 included cross-lingual experiments in de Pablo-Sánchez et al. [2006e] (Working notes: [de Pablo-Sánchez et al., 2005a])
- in CLEF 2006 we also participated in the Real Time pilot and QA on Wikipedia (WiQA) described in de Pablo-Sánchez et al. [2007a]. Besides, we help in the coordination of WiQA for the Spanish language. (Working notes: de Pablo-Sánchez et al. [2006b] for the main task and de Pablo-Sánchez et al. [2006c] for WiQA).
- our CLEF 2007 participation is outlined in Martínez-González et al. [2009] (Working Notes:[Martínez-González et al., 2008])

In addition, our work on multilingual retrieval and multilingual processing of Named Entities has been simultaneous to QA. It help to contribute and shape the vision presented in this thesis. [de Pablo-Sánchez et al., 2006e] describe our participation in WebCLEF. It shows that the addition of a NE index contributed to improve precision in multilingual web retrieval even if the recognition of NE was carried out with language independent automata. [de Pablo-Sánchez et al., 2006d] was presented at the *Workshop on New Directions in Multilingual Information Access* at *SIGIR 2006* and presents the idea of using lightly supervised NE processing modules required for Multilingual Information Access applications.

The bootstrapping algorithm proposed in Chapter 7, SPINDEL, has been initially presented in de Pablo-Sánchez and Martínez [2009a] in the *European Conference for Information Retrieval 2009* and it has been extended and submitted for journal publication. Related work on Web People Search [del Valle-Agudo et al., 2007] and the extraction of attributes from people pages [de Pablo-Sánchez and Martínez, 2009b] has been carried out in WePS shared task (Web People Search) in *SemEval 2007* and *World Wide Web 2009*. Besides, our participation in Knowledge Base Population (KBP) task at the *Text Analysis Conference 2009* complements our work on bootstrapping NE resources [de Pablo-Sánchez et al., 2009] as we use the OpenEphyra system that acquires patterns for relations in a task related to QA. However, a complete QA or KBP system following the approach outlined in Chapter 6 is still missing but work is under development. Related work on the use of SPINDEL in the clinical domain to help in the semantic annotation of clinical notes and their anonymization has been initially presented in Pérez-Laínez et al. [2009] at the *International Conference on Knowledge Discovery and Information Retrieval*.

Finally, additional work has been carried from two lines that are not central to the thesis but we started in the context of the QA system, temporal processing and anaphora resolution. Collaborations in these areas led to the a Spanish temporal annotation system [Vicente-Díez et al., 2007] and works on the analysis of anaphora resolution in drug-drug interactions [Segura-Bedmar et al., 2009, 2010; **?**].

9.7 Future Work

Several research works are envisaged as a result of the ideas and experiments of this thesis:

- **Build a complete QA system based on bootstrapped knowledge and language resources**. In this thesis, we have developed a QA system for Spanish. We also explored the use of large documents collections to acquire resources that help to recognize and categorize Named Entities. This is an important step for building QA systems using lightly supervised IE systems. To complete our proposal for a QA system using lightly supervised IE components, we should integrate relation extraction and NERC in a complete system. The comparison for a new domain with no available annotated corpora would help to asses the approach outlined in Chapter 6. For comparison, news and Wikipedia collections may

provide a reference against human engineered knowledge (like in STILUS tool) or systems based on supervised machine learning.

- **Explore the acquisition of additional knowledge from unannotated text collection and their use in QA systems**. In our last experiments with QA, the STILUS tool have been improved to include additional ontological information that help to perform shallow reasoning. It would be interesting to test if conceptual knowledge can be acquired and used in QA systems too. The use of Ontology Learning from text may help to create taxonomies to improve in question classification and query formulation among others.

- **Reuse relation patterns in other parts of the QA process**. Relation patterns, and at less extent, patterns associated with entities, may be useful in other steps beyond Answer Extraction. For example, their use in order to reformulate questions, expand queries and re-rank answers may be considered.

- **Study the combination of QA in multiple languages**. One of the long term aims of this work consists on simplifying the development of systems that make use of language analysis. Once the systems are relatively language independent it is possible to devise QA systems that treat each language as an independent source of evidence. It would be interesting to compare whether the combination of different languages may bring benefit over a single one.

- Carry on a wider **study of the acquisition of knowledge and linguistic patterns for different domains**. There are a number of features that may have an impact on the ability to aquire knowledge about concepts or relations. Some of these feature may vary from one domain to another. Among these features we believe that the number of classes, the complexity of a taxonomy, the morphological clues to use, the degree of ambiguity, the number of instances per class or their distribution in text may be important for concepts or NE. Relations may have different complexity too, for example being more frequent the use of negation, speculation or modal language in scientific or technical domains. A common framework to study and evaluate along different domains and collections (news, encyclopedia, legal, biomedicine, clinical, etc.) may provide insight on the properties of different algorithms.

- In a similar sense, a **broader study of the acquisition algorithms across languages and language families** is also required. Our experiments have only been applied to a couple of languages. Experimentation should be applied to other languages in order to draw additional conclusions. Closer languages like Portuguese may require few changes but those with different typographical conventions (like German) should need to improve how to recognize current NE candidates. Work should also be extended for other languages beyond Romance and Germanic ones like Chinese, Japanese, Hindi or Arabic. Due to the challenges (particularly in evaluation) it may entail, we have decide to follow the bottom-up path, or in other words, from few familiar languages increase the generalization slightly in each step.

- **Comparative and exhaustive evaluation of different techniques and metrics**. Though Relation Extraction and NE acquisition have already been studied, there is no shared benchmark for their formulation as Large Scale IE task. Few bootstrapping and large scale algorithms have been evaluated in similar conditions. The use of different collections, different relations and different seeds (in number and quality) do not allow to extract firm conclusions regarding the most adequate algorithms for a given scenario. Moreover, in lightly supervised scenarios algorithm stability need to be defined and evaluated because is far more important for their use in a large number of relations.

- **Extend the acquisition of knowledge for larger and more complex taxonomies of concepts**. In our experiments we have explored the acquisition of few simple classes and compared it to classic NERC systems used in the news domain. Other domains involve more complex taxonomies that are formed by more classes, several levels of *is-a* relations. In that taxonomies, the criteria for inclusion of instances in given classes may be more complex. Models that are able to handle the hierarchy of the taxonomy are required. On the other hand, QA systems would benefit from detailed semantic resources generated this way.

- **Study how coupling and combining different process for the acquisition of knowledge**. In our work we have proposed a pipeline for the acquisition of Named Entities and their relationships. This is the simplest way to proceed because the acquisition of NE resources may help to locate arguments for the extraction of relations. It contrast, the simultaneous coupling of several processes may help to improve results, in particular the precision. There are several alternatives for coupling that may be considered, coupling the acquisition of NE and relations, coupling the acquisition across collections or across languages.

- **Evaluate the contribution of language analysis** is an important contribution once the evaluation framework is well defined. We have adopted an agnostic view on the use of language analysis because of their limitation for multilingual applications and domain adaptation. However, if NLP tools are available they will probably improve the bootstrapping of semantic resources.

Nomenclature

ACE Automatic Content Extraction

AI Artificial Intelligence

AL Active Learning

API Application Programming Interface

AVE Answer Validation Exercise

BRUJA-QA

CL-QA Cross Lingual Question Answering

CLEF Cross Language Evaluation Forum

CLQA Cross Lingual Question Answering Task at NTCIIR

CMU Carnegie Mellon University

COLE

CONLL Conference On Natural Language Learning

CWS Confidence Weighted Score

DUC Document Understanding Conference

EAT Expected Answer Type

EDR Entity Detection and Recognition

ENE taxonomy Extended Named Entity Taxonomy

EVALITA Evaluation of NLP and Speech Tools for Italian

FAQ Frequent Answered Questions

FASTUS Finite State Automata-based Text Understanding System

FSA Finite State Automata

GATE General Architecture for Text Engineering

HAREM Reconhecimento de Entidades Mencionadas em Português

HMM Hidden Markov Model

HTML Hyper Text Markup Language

IBM International Business Machines

IE Information Extraction

INAOE Instituto Nacional de Astrofísica, Óptica y Electrónica, México

IR Information Retrieval

JNI Java Native Interface

KA Knowledge Annotation

KM Knowledge Mining

KR Knowledge Representation

LCC Language Computer Corporation

MIRACLE Multilingual Information Retrieval at CLEF (Universidad Autonoma de Madrid, Universidad Politécnica de Madrid, Universidad Carlos III de Madrid, DAEDALUS S.A.)

ML Machine Learning

ML-QA Multilingual Question Answering

MRR Mean Reciprocal Rank

MT Machine Translation

MTRR Mean Total Reciprocal Rank

MUC Message Understanding Conference

NE Named Entity

NERC Named Entity Recognition and Classification

NLP Natural Language Processing

NUS National University of Singapore

ODIE On-Demand Information Extraction

POS Part of Speech

Priberam Priberam Informatica

QA Question Answering

QAC Question Answering Challenge at NTCIIR

QF Question Focus

QT Question Type

RDC Relation Detection and Characterization

RDF Resource Description Framework

RE Relation Extraction

SCFG Stochastic Context Free Grammar

SGML Standard Generalized Markup Language

SMU Southern Methodist University

SRES Self-supervised Relation Extraction System

SRL Semantic Role Labelling

SSL Semi-Supervised Learning

SVM Support Vector Machine

TALP-QA Centro de Tecnologías y Aplicaciones del Lenguaje y del Habla, Universidad Politécnica de Cataluña

TC Text Classification

TE Textual Entailment

TEG Trainable Exraction Grammar

TM Text Mining

TREC Text REtrieval Conference

TSVM Transductive Support Vector Machine

TU Tokio University

UA Universidad de Alicante

UIMA Unstructured Information Management Architecture

UPM Universidad Politécnica de Madrid

UPV Universidad Politécnica de Valencia

URES Unsupervised Relation Extraction System

VSM Vector Space Model

WSD Word Sense Disambiaguation

XML eXtensible Markup Language

Bibliography

Abney, S. (2004). Understanding the Yarowsky algorithm. *Computational Linguistics*, 30(3):365–395.

Abney, S. (2007). *Semisupervised Learning for Computational Linguistics*. Chapman and Hall/CRC.

Abney, S., Collins, M., and Singhal, A. (2000). Answer Extraction. In *Sixth conference on Applied Natural Language Processing*.

Aceves-Pérez, R., Montes-Y-Gómez, M., and Villaseñor Pineda, L. (2007). Enhancing Cross-Language Question Answering by Combining Multiple Question Translations. *Lecture Notes In Computer Science*, 4394:485–493.

Aceves-Pérez, R., Montes-y Gómez, M., Villaseñor Pineda, L., and Ureña López, L. A. (2008). Two Approaches for Multilingual Question Answering: Merging Passages vs. Merging Answers. *Computational Linguistics and Chinese Language processing*, 13(1):27–40.

Agichtein, E., Burges, C., and Brill, E. (2007). Question Answering over Implicitly Structured Web Content. In *IEEE/WIC/ACM International Conference on Web Intelligence*, pages 18–25.

Agichtein, E., Cucerzan, S., and Brill, E. (2005). Analysis of factoid questions for effective relation extraction. In *28th annual international ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 567–568, New York, NY, USA. ACM Press.

Agichtein, E. and Gravano, L. (2000). Snowball: Extracting relations from large plain-text collections. In *Fifth ACM conference on Digital Libraries*, pages 85–94. ACM New York, NY, USA.

Agichtein, E. and Gravano, L. (2003). Querying Text Databases for Efficient Information Extraction. In *IEEE International Conference on Data Engineering (ICDE)*.

Ahn, D., Jijkoun, V., Müller, K., de Rijke, M., Schlobach, S., and Mishne, G. (2005). Making Stone Soup: Evaluating a Recall-Oriented Multi-stream Question Answering System for Dutch. *Multilingual Information Access for Text, Speech and Images*, pages 423–434.

Al-Maskari, A. and Sanderson, M. (2006). The Affect of Machine Translation on the Performance of Arabic-English QA System. In *Workshop on Multilingual Question Answering–MLQA'06*.

Aone, C. and Ramos-Santacruz, M. (2000). REES: A Large-Scale Relation and Event Extraction System. In *Sixth conference on Applied Natural Language Processing*, pages 76–83.

Appelt, D., Hobbs, J. R., Bear, J., Israel, D., and Tyson, M. (1993). FASTUS: A finite-state processor for information extraction from real-world text. In *International Joint Conference on Artifical Intelligence*, pages 1172–1178.

Baker, C., Fillmore, C., and Lowe, J. (1998). The berkeley framenet project. In *36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics*.

Baldridge, J. and Morton, T. S. OpenNLP.

Banko, M. and Brill, E. (2001). Scaling to Very Large Corpora for Natural Language Disambiguation. In *39th Annual Meeting of the Association for Computational Linguistics (ACL)*, Toulouse, France.

Banko, M., Brill, E., and Dumais, S. (2002a). An analysis of the AskMSR question-answering system. *Empirical methods in natural language processing - EMNLP '02*, pages 257–264.

Banko, M., Broadhead, M., Cafarella, M. J., Soderland, S., and Etzioni, O. (2007). Open information extraction from the web. In *Proceedings of the IJCAI 2007*.

Banko, M., Dumais, S. T., Brill, E., Lin, J., and Ng, A. (2002b). Web question answering: Is more always better? In *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 291–298. ACM New York, NY, USA.

Bar-Haim, R., Dagan, I., Dolan, B., Ferro, L., Giampiccolo, D., Magnini, B., and Szpektor, I. (2006). The Second PASCAL Recognising Textual Entailment Challenge. In *Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment, Venice, Italy.*

Berland, M. and Charniak, E. (1999). Finding Parts in Very Large Corpora. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, pages 57–64, College Park, Maryland, USA. Association for Computational Linguistics.

Bikel, D., Schwartz, R., and Weischedel, R. M. (1999). An algorithm that learns what's in a name. *Machine learning*, 34(1):211–231.

Bilotti, M. W., Katz, B., and Lin, J. (2004). What Works Better for Question Answering: Stemming or Morphological Query Expansion? In *IR4QA workshop in SIGIR' 04*.

Bilotti, M. W., Ogilvie, P., Callan, J., and Nyberg, E. (2007). Structured retrieval for question answering. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in information retrieval*, pages 351–358. ACM New York, NY, USA.

Bilotti, M. W., Zhao, L., Callan, J., and Nyberg, E. (2008). Focused retrieval over richly-annotated collections. In *SIGIR 2008*, Singapore.

Bird, S., Day, D., Garofolo, J., Henderson, J., Laprun, C., and Liberman, M. (2000). ATLAS A Flexible and Extensible architecture for Linguistic Annotation. In *LREC 2000*.

Bizer, C., Auer, S., Kobilarov, G., Lehmann, J., Cyganiak, R., and Ives, Z. (2007). Dbpedia: A nucleus for a web of open data. *Lecture Notes in Computer Science*, 4825:722.

Bizer, C., Heath, T., and Berners-Lee, T. (2009). Linked Data - The Story So Far. *International Journal on Semantic web and Information Systems*, (to appear.

Blair-Goldenshon, S. (2007). *Long-Answer Question Answering and Rhetorical-Semantic Relations*. Phd, Columbia University.

Blohm, S., Cimiano, P., and Stemle, E. (2007). Harvesting relations from the web-quantifiying the impact of filtering functions. In *Proceedings of the AAAI 2007*, volume 22, page 1316. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999.

Blum, A. and Chawla, S. (2001). Learning from labeled and unlabeled data using graph mincuts. In *19th International Conference on Machine Learning (ICML-2001)*.

Blum, A. and Mitchell, T. (1998). Combining Labeled and Unlabeled Data with Co-Training. In *Computational Learning Theory*.

Bollacker, K., Evans, C., Paritosh, P., Sturge, T., and Taylor, J. (2008). Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, pages 1247–1250. ACM New York, NY, USA.

Bouma, G., Mur, J., Van Noord, G., and Plas, L. V. D. (2006). Question answering for Dutch using dependency relations. *Lecture Notes in Computer Science*.

Bowden, M., Olteanu, M., Suriyentrakor, P., Clark, J., and Moldovan, D. (2007). LCC's Power Answer at QA@ CLEF 2006. *Lecture Notes in Computer Science*, 4730(1):310.

Brin, S. (1999). Extracting patterns and relations from the world wide web. *Lecture Notes in Computer Science*, pages 172–183.

Bunescu, R. and Mooney, R. J. (2005). A shortest path dependency kernel for relation extraction. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, number October, pages 724–731. Association for Computational Linguistics Morristown, NJ, USA.

Bunescu, R. and Mooney, R. J. (2007). Learning to extract relations from the web using minimal supervision. In *45th Annual Meeting of the Association for Computational Linguistics*, volume 45, page 576.

Burger, J. D., Cardie, C., Chaudhri, V., Gaizauskas, R., Harabagiu, S., Israel, D., Jacquemin, C., Lin, C., Maiorano, S., Miller, G., and Others (2001). Issues, tasks and program structures to roadmap research in question \& answering (Q\&A). *Document Understanding Conferences*, pages 1–35.

Burke, R. D., Hammond, K. J., Kulyukin, V. A., Lytinen, S. L., Tomuro, N., and Schoenberg, S. (1997). Question Answering from Frequently-Asked Question Files: Experiences with the FAQ Finder System. *AI Magazine*, 18:57–66.

Cafarella, M. J. and Etzioni, O. (2005). A search engine for natural language applications. In *Proceedings of the 14th international conference on World Wide Web*, page 452. ACM.

Carreras, X. and Márquez, L. (2005). Introduction to the CoNLL-2005 Shared Task: Semantic Role Labeling. In *Proceedings of the CONLL 2005*.

Carreras, X., Márquez, L., and Padró, L. (2002). Named Entity Extraction using AdaBoost. In *CONLL '02*.

Cassan, A., Figueira, H., Martins, A., Mendes, A., Mendes, P., Pinto, C., and Vidal, D. (2006). Priberam's Question Answering System in a Cross-Lingual Environment. In *Accessing Multilingual Information Repositories*.

Celikyilmaz, A., Thint, M., Telecom, B., and Huang, Z. (2008). A Graph-based Semi-Supervised Learning for Question-Answering. In *46th Annual Meeting of the Association for Computational Linguistics (ACL) :Human Language Technology*, number August, pages 719–727.

Chakrabarti, S., Das, S., Krishnan, V., and Puniyani, K. (2006). *Text Search Enhanced with Types and Entities*. Chapman and Hall/CRC.

Chapelle, O., Schölkopf, B., and Zien, A. (2006). *Semi-Supervised Learning*. MIT Press.

Chen, H., Sasaki, Y., Chen, K., and Lin, C. (2005). Overview of the NTCIR-5 cross-lingual question answering task (clqa1). In *Proceedings of the Fifth NTCIR Workshop Meeting*.

Chen, Y., Zhou, M., and Wang, S. (2006). Reranking answers for definitional QA using language modeling. In *21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, number July, pages 1081–1088, Morristown, NJ, USA. Association for Computational Linguistics.

Chinchor, N. (1997). Overview of MUC-7/MET-2. In *MUC-Proceedings*.

Chu-Carroll, J., Czuba, K., Prager, J., and Ittycheriah, A. (2003). In question answering, two heads are better than one. *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - NAACL '03*, (June):24–31.

Chu-Carroll, J. and Prager, J. (2007). An experimental study of the impact of information extraction accuracy on semantic search performance. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management - CIKM '07*, page 505, New York, New York, USA. ACM Press.

Chu-Carroll, J., Prager, J., Czuba, K., Ferrucci, D., and Duboue, P. (2006). Semantic search via XML fragments: a high-precision approach to IR. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, page 452. ACM.

Cimiano, P., Haase, P., Heizmann, J., and Mantel, M. (2008). Orakel: A portable natural language interface to knowledge bases. *Data & Knowledge Engineering*, 65(2):325–354.

Cimiano, P. and Volker, J. (2005). Towards large-scale, open-domain and ontology-based named entity classification. In *Proceedings of RANLP*, volume 5, pages 66–166.

Clarke, C. L. A., Cormack, G. V., R.Lynam, T., and Terra, E. L. (2006). *Question Answering with Passage Selection*, chapter Question A, pages 259–285. Springer.

Cleverdon, C. W., Mills, J., and Keen, E. M. (1966). Factors determining the performance of indexing systems.

Clifton, T. and Teahan, W. (2005). Bangor at TREC 2004: Question Answering Track. In *The Thirteenth Text Retrieval Conference (TREC 2004)*.

Codd, E. F. (1970). A relational model of data for large shared data banks. *Communications of the ACM*, 13(6).

Codd, E. F. (1990). *The relational model for Database Management. Version 2*. Addison Wesley.

Cohen, W. W. and Sarawagi, S. (2004). Exploiting Dictionaries in Named Entity Extraction: Combining Semi-Markov Extraction Processes and Data Integration Methods. *Learning*, pages 89–98.

Collins, M. and Singer, Y. (1999). Unsupervised Models for Named Entity Classification. In *Proceedings of EMNLP*.

Croft, B. and Metzler, D. (2005). Analysis of Statistical Question Classification for Fact-Based Questions. *Information Retrieval*, 8(3):481–504.

Croft, B. and Murdock, V. (2004). Simple translation models for sentence retrieval in factoid question answering. In *Proceedings of the Information Retrieval for Question Answering Workshop at SIGIR*, volume 2004.

Cucerzan, S. and Yarowsky, D. (1999). Language Independent Named Entity Recognition Combining Morphological and Contextual Evidence. In *Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, pages 90–99.

Cucerzan, S. and Yarowsky, D. (2002). Language independent NER using a unified model of internal and contextual evidence. In *Proceedings of CoNLL 2002*, pages 171–174.

Cui, H., Kan, M.-Y., and Chua, T.-S. (2005a). Generic soft pattern models for definitional question answering. *28th annual international ACM SIGIR conference on Research and development in information retrieval*, page 384.

Cui, H., Sun, R., Li, K., Kan, M.-Y., and Chua, T.-S. (2005b). Question answering passage retrieval using dependency relations. In *ACM Conference on Research and Development in Information Retrieval*.

Culotta, A. and Sorensen, J. S. (2004). Dependency tree kernels for relation extraction. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, number Table 2.

Cunningham, H., Maynard, D., Bontcheva, K., Tablan, V., Ursu, C., Dimitrov, M., Dowman, M., Aswani, N., Roberts, I., Li, Y., Shafirin, A., and Funk, A. (2008). Developing Language Processing Components with GATE Version 5 (a User Guide).

Cunningham, H., Wilks, Y., and Gaizauskas, R. J. (2002). GATE - a General Architecture for Text Engineering. Technical report, University of Sheffield.

Dang, H. T. (2006). Overview of DUC 2006. In *Document Understanding Conferences*.

de Pablo-Sánchez, C., González-Ledesma, A., Martínez-Fernández, J. L., Guirao, J., Martínez, P., and Moreno, A. (2005a). MIRACLE's Cross-Lingual Question Answering Experiments with Spanish as a Target Language. In *Cross Language Evaluation Forum (CLEF 2005)*.

de Pablo-Sánchez, C., González-Ledesma, A., Martínez-Fernández, J. L., Guirao, J., Martínez, P., and Moreno, A. (2006a). MIRACLE's Cross-Lingual Question Answering Experiments with Spanish as a Target Language. *Accessing Multilingual Information Repositories*, pages 488–491.

de Pablo-Sánchez, C., González-Ledesma, A., Martínez-Fernández, J. L., Moreno, A., and Martínez, P. (2006b). MIRACLE at the Spanish CLEF@QA 2006 Track. In *Cross Language Evaluation Forum (CLEF 2006)*.

de Pablo-Sánchez, C., González-Ledesma, A., Moreno, A., and Vicente-Díez, M. T. (2007a). MIRACLE experiments in QA@CLEF 2006 in Spanish: main task, real-time QA and exploratory QA using Wikipedia (wiQA). *Evaluation of Multilingual and Multi-modal Information Retrieval*, 4370:463–472.

de Pablo-Sánchez, C. and Martínez, P. (2009a). Building a Graph of Names and Contextual Patterns for Named Entity Classification. In *European Conference in Information Retrieval 2009*.

de Pablo-Sánchez, C. and Martínez, P. (2009b). UC3M at WePS2-AE: Acquiring Patterns for People Attribute Extraction from Webpages. In *Second Workshop on Web People Search at WWW 2009*.

de Pablo-Sánchez, C., Martínez-Fernández, J. L., González-Ledesma, A., Samy, D., Martínez, P., Moreno, A., and Al-Jumaily, H. (2007b). MIRACLE Question Answering System for Spanish at CLEF2007. In *Cross Language Evaluation Forum (CLEF 2007)*.

de Pablo-Sánchez, C., Martínez-Fernández, J. L., and Martínez, P. (2006c). MIRACLE at the Spanish WiQA Pilot: Using Named Entities and Cosine Similarity to extend Wikipedia articles. In *Cross Language Evaluation Forum (CLEF 2006)*.

de Pablo-Sánchez, C., Martínez-Fernández, J. L., Martínez, P., and del Valle-Agudo, D. (2006d). Named Entity Processing for Cross-Lingual and Multilingual IR applications. In *Workshop on New Directions in Multilingual Information Access, SIGIR 2006*.

de Pablo-Sánchez, C., Martínez-Fernández, J. L., Martínez, P., and Villena-Román, J. (2004). miraQA: Initial Experiments in Question Answering. In *Cross Language Evaluation Forum (CLEF 2004)*.

de Pablo-Sánchez, C., Martínez-Fernández, J. L., Martínez-González, A., and Villena-Román, J. (2006e). MIRACLE at WebCLEF 2005: Combining Web Specific and Linguistic Information. *Accessing Multilingual Information Repositories*, pages 869–872.

de Pablo-Sánchez, C., Martínez-Fernández, J. L., Villena-Román, J., and Martínez, P. (2005b). miraQA: Experiments with Learning Answer Context Patterns from the Web. *Multilingual Information Access for Text, Speech and Images*, pages 494–501.

de Pablo-Sánchez, C., Perea, J., Segura-Bedmar, I., and Martínez, P. (2009). The UC3M team at the Knowledge Base Population task. In *Text Analysis Conference (TAC 2009)*.

de Rijke, M., Ahn, D., Fissaha, S., Müller, K., Jijkoun, V., and Sang, E. T. K. (2006). Towards a Multi-Stream Question Answering-As-XML-Retrieval Strategy. In *Fourteenth Text Retrieval Conference (TREC 2005)*.

de Rijke, M., Ahn, D., Jijkoun, V., Mishne, G., Müller, K., and Schlobach, S. (2005). Using Wikipedia in the TREC QA Track. In Voorhees, E. M. and Buckland, L. P., editors, *The Thirteenth Text Retrieval Conference (TREC 2004)*.

del Valle-Agudo, D., de Pablo-Sánchez, C., and Vicente-Díez, M. T. (2007). Disambiguation of Person Names Based on the Composition of Simple Bags of Typed Terms. In *SEMEVAL-2007*.

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum Likelihood from Incomplete Data via the EM Algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1):1 – 38.

Dietterich, T. G., Lathrop, R. H., and Lozano-Pérez, T. (1997). Solving the multiple instance problem with axis-parallel rectangles. *Artificial Intelligence*, 89:31–71.

Downey, D., Etzioni, O., and Soderland, S. (2005). A probabilistic model of redundancy in information extraction. In *18th International Joint Conference on Artificial Intelligence (IJCAI-05)*, volume 19, page 1034. Citeseer.

Downey, D., Schoenmackers, S., and Etzioni, O. (2007). Sparse information extraction: Unsupervised language models to the rescue. In *45th Annual Meeting of the Association for Computational Linguistics*, volume 45, page 696.

Echihabi, A., Hermjacob, U., Hovy, E., Marcu, D., Melz, E., and Ravichandran, D. (2006). How to Select an Answer String? *Advances in Open Domain Question Answering*.

Elworthy, D. (1994). Does Baum-Welch Re-estimation help taggers? In *4th ACL Conf on ANLP*, pages 53–58, Stuttgart.

Etzioni, O., Cafarella, M. J., Downey, D., Kok, S., Popescu, A.-m., Shaked, T., Soderland, S., Weld, D. S., and Yates, A. (2004). Web-scale information extraction in KnowItAll:(preliminary results). In *Proceedings of the 13th international conference on World Wide Web*, pages 100–110. ACM New York, NY, USA.

Etzioni, O., Cafarella, M. J., Downey, D., Popescu, A.-m., Shaked, T., Soderland, S., Weld, D. S., and Yates, A. (2005). Unsupervised named-entity extraction from the web: An experimental study. *Artificial Intelligence*, 165(1):91–134.

Feldman, R. and Rosenfeld, B. (2006). Boosting unsupervised relation extraction by using NER. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, number July, pages 473–481. Association for Computational Linguistics.

Feldman, R. and Sanger, J. (2007). *The Text Mining Handbook: Advanced Approaches in Analyzing Unstructured Data*. Cambridge University Press.

Fellbaum, C. (1998). *WordNet: An electronic lexical database*. The MIT Press, Cambridge, MA.

Ferrández, O., Izquierdo, R., Ferrández, S., and Vicedo, J. L. (2009). Addressing ontology-based question answering with collections of user queries. *Information Processing & Management*, 45(2):175–188.

Ferrández, S., López-Moreno, P., Roger, S., Ferrández, A., Peral, J., Alvarado, X., Noguera, E., and Llopis, F. (2006). AliQAn and BRILI QA systems at CLEF 2006.

Ferrés, D., Kanaan, S., Ageno, A., González, E., Rodríguez, H., Surdeanu, M., and Turmo, J. (2005). *The TALP-QA System for Spanish at CLEF 2004: Structural and Hierarchical Relaxing of Semantic Constraints*.

Ferrucci, D. and Lally, A. (2004). UIMA: an architectural approach to unstructured information processing in the corporate research environment. *Natural Language Engineering*, 10:327–348.

Ferrucci, D., Nyberg, E., Allan, J., Barker, K., Brown, E., Chu-Carroll, J., Ciccolo, A., Duboue, P., Fan, J., Gondek, D., Hovy, E., Katz, B., Lally, A., McCord, M., Morarescu, P., Murdock, B., Porter, B., Prager, J., Strzalkowski, T., Welty, C., and Zadrozny, W. (2009). Towards the Open Advancement of Question Answering Systems.

Fleischman, M. and Hovy, E. (2002). Fine grained classification of named entities. In *Proceedings of the Conference on Computational Linguistics, Taipei, Taiwan, August 2002*, pages 1–7.

Fleischman, M., Hovy, E., and Echihabi, A. (2003). Offline Strategies for Online Question Answering: Answering Questions Before They Are Asked. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1*, pages 1–7. Association for Computational Linguistics Morristown, NJ, USA.

Florian, R., Ittycheriah, A., Jing, H., and Zhang, T. (2003). Named Entity Recognition through Classifier Combination. In *HLT-NAACL '03*.

Forner, P., Sutcliffe, R., Giampiccolo, D., Moreau, N., and Osenova, P. (2009). Overview of ResPubliQA 2009: Question Answering Evaluation over European Legislation. In *CLEF 2009*.

Frantzi, K. T. and Ananiadou, S. (1999). The C-value/NC-value domain-independent method for multi-word term extraction. *Journal of Natural Language Processing*, 6:145?179.

Friedland, N. S., Allen, P. G., Matthews, G., Witbrock, M., Baxter, D., Curtis, J., Shepard, B., Miraglia, P., Staab, S., Moench, E., Oppermann, H., Wenke, D., Israel, D., Chaudhri, V., Porter, B., Barker, K., Fan, J., Chaw, S. Y., Yeh, P., Tecuci, D., and Clark, P. (2004). Project Halo: Towards a Digital Aristotle. pages 1–31.

García-Cumbreras, M. A., Ureña López, L. A., Martínez-Santiago, F., and Perea-Ortega, J. M. (2006). BRUJA System. The University of Jaén at the Spanish task of CLEFQA 2006. In *CLEF 2006*.

Ghani, R., Probst, K., Liu, Y., Krema, M., and Fano, A. (2006). Text mining for product attribute extraction. *SIGKDD Explor. Newsl.*, 8:41–48.

Giampiccolo, D., Peñas, A., Ayache, C., Cristea, D., Forner, P., Jijkoun, V., Osenova, P., Rocha, P., Sacaleanu, B., and Sutcliffe, R. (2007). Overview of the clef 2007 multilingual question answering track. In *CLEF 2007*.

Gildea, D. and Jurafsky, D. (2002). Automatic labeling of semantic roles. *Computational Linguistics*, 28(3):245–288.

Giuliano, C., Lavelli, A., and Romano., L. (2007). Relation extraction and the influence of automatic named-entity recognition. *ACM Transactions on Speech and Language Processing (TSLP)*, 5(1):2.

Goñi Menoyo, J. M., González-Cristóbal, J. C., and Villena-Román, J. (2006). MIRACLE at Ad-Hoc CLEF 2005: Merging and Combining Without Using a Single Approach. In *Accessing Multilingual Information Repositories*, pages 44–53.

Gómez-Pérez, A., Fernández-López, M., and Corcho, O. (2004). *Ontological Engineering*. Springer, London.

Gómez-Soriano, J. M., Bisbal-Asensi, E., Buscaldi, D., Rosso, P., and Sanchís-Arnal, E. (2005). Monolingual and Cross-language QA using a QA-oriented Passage Retrieval System. In *Cross Language Evaluation Forum Workshop (CLEF 2005)*, Vienna, Austria.

Gonzalo, J., Clough, P., and Vallin, A. (2006). Overview of the CLEF 2005 Interactive Track. In *Accessing Multilingual Information Repositories*, pages 251–262.

Gonzalo, J. and Oard, D. W. (2005). *iCLEF 2004 Track Overview: Pilot Experiments in Interactive Cross-Language Question Answering*.

Google. Google Ajax Search API.

Grau, B., Ferret, O., Hurault-Plantet, M., Jacquemin, C., Monceaux, L., Robba, I., and Vilnat, A. (2006). Coping with Alternate Formulations of Questions and Answers. *Advances in Open Domain Question Answering*, pages 189–225.

Green, C. (1968). Theorem proving by resolution as a basis for question-answering systems. *Machine Intelligence*, 4:183–205.

Green Jr, B., Wolf, A., Chomsky, C., and Laughery, K. (1961). Baseball: an automatic question-answerer. *AFIPS Joint Computer Conferences*, 19pp:219–224.

Greenwood, M. A. (2008). Proceedings of the 2nd workshop on Information Retrieval for Question Answering.

Greenwood, M. A. (2009). IR4QA: An Unhappy Marriage.

Greenwood, M. A. and Stevenson, M. (2005). A task-based comparison of information extraction pattern models. In *21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, volume 100, pages 81–88.

Greenwood, M. A. and Stevenson, M. (2006). Improving semi-supervised acquisition of relation extraction patterns. In *ACL 2006 Workshop on Information Extraction Beyond the Document*, number July, pages 29–35.

Grishman, R. and Yangarber, R. (1997). Customization of information extraction systems. *International Workshop on Lexically Driven Information Extraction*, pages 1–11.

Hacioglu, K. and Ward, W. (2003). Question classification with support vector machines and error correcting codes. *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology companion volume of the Proceedings of HLT-NAACL 2003–short papers - NAACL '03*, pages 28–30.

Harabagiu, S., Clark, C., Moldovan, D., Bensley, J., Williams, J., and Bowden, M. (2003a). Answer Mining by Combining Extraction Techniques with Reasoning. In *The Twelfth Text Retrieval Conference (TREC 2003)*.

Harabagiu, S. and Hickl, A. (2006). Methods for using textual entailment in open-domain question answering. In *44th annual meeting of the Association for Computational Linguistics*, number July, pages 905–912, Morristown, NJ, USA. Association for Computational Linguistics.

Harabagiu, S. and Moldovan, D. (2003). *Oxford Handbook of Computational Linguistics*, chapter Question A, pages 560–582. Oxford University Press.

Harabagiu, S., Moldovan, D., Clark, C., and Maiorano, S. (2003b). Cogex: A logic prover for question answering. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, number June, pages 87–93. Association for Computational Linguistics Morristown, NJ, USA.

Harabagiu, S., Moldovan, D., Pasca, M., Mihalcea, R., Surdeanu, M., Bunescu, R., Girju, R., Rus, V., and Morarescu, P. (2001). The role of lexico-semantic feedback in open-domain textual question-answering. In *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics - ACL '01*, volume 39, pages 282–289, Morristown, NJ, USA. Association for Computational Linguistics.

Harabagiu, S. and Strzalkowski, T. (2006). *Advances in Open Domain Question Answering*. Springer.

Hartrumpf, S. (2005). Question Answering Using Sentence Parsing and Semantic Network Matching. In *Multilingual Information Access for Text, Speech and Images*, pages 512–521.

Hasegawa, T., Sekine, S., and Grishman, R. (2004). Discovering relations among named entities from large corpora. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, page 415. Association for Computational Linguistics.

Hearst, M. A. (1992). Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the 14th conference on Computational linguistics-Volume 2*, pages 539–545. Association for Computational Linguistics Morristown, NJ, USA.

Hendrix, G. G. (1977). Human engineering for applied natural language processing. In *Fifth International Joint Conference on Artificial Intelligence (IJCAI-77)*.

Hermjacob, U. (2001). Parsing and Question Classification for Question Answering. In *Workshop on Open-Domain Question Answering Systems at ACL 2001.*

Hersh, W. (2006). Evaluating Interactive Question Answering. *Advances in Open Domain Question Answering*, pages 431–457.

Hinton, G. E. and Sejnowski, T. J. (1986). Learning and relearning in Boltzmann machines. *Mit Press Computational Models Of Cognition And Perception Series.*

Huang, Z., Thint, M., and Qin, Z. (2008). Question Classification using Head Words and their Hypernyms. In *Proceedings of the 2008 Conference on Empirical*, number October, pages 927–936.

Ipeirotis, P., Agichtein, E., Jain, P., and Gravano, L. (2006). To search or to crawl?: towards a query optimizer for text-centric tasks. In *Proceedings of the 2006 ACM SIGMOD international conference on Management of data*, page 276. ACM.

Ireson, N., Ciravegna, F., Califf, M. E., Freitag, D., Kushmerick, N., and Lavelli, A. (2005). Evaluating machine learning for information extraction. In *ICML '05: Proceedings of the 22nd international conference on Machine learning*, pages 345–352, New York, NY, USA. ACM.

Ittycheriah, A. (2006). A Statististical Approach for Open Domain Question Answering. *Advances in Open Domain Question Answering*, pages 35–71.

Ji, H. and Grishman, R. (2006). Data selection in semi-supervised learning for name tagging. *ACL 2006 Workshop on Information Extraction Beyond the Document*, 55(July):48–55.

Joachims, T. (1999). Transductive Inference for Text Classification using Support Vector Machines. In *International Conference on Machine Learning.*

Jones, R. (2005). *Learning to Extract Entities from Labeled and Unlabeled Text.* Phd., Carnegie Mellon University.

Jurafsky, D. and Martin, J. H. (2008). *Speech and Language Processing. An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition.* Second edition.

Kaisser, M. and Lowe, J. B. (2008). Creating a Research Collection of Question Answer Sentence Pairs with Amazon's Mechanical Turk. In *LREC 08.*

Kaisser, M. and Webber, B. (2007). Question Answering based on Semantic Roles. In *45th Annual Meeting of the Association for Computational Linguistics.*

Kato, T., Fukumoto, J., Masui, F., and Kando, N. (2005). Are open-domain question answering technologies useful for information access dialogues?—an empirical study and a proposal of a novel challenge. *ACM Transactions on Asian Language Information Processing*, 4(3):243–262.

Katz, B. (1988). Using English for Indexing and Retrieving. In *Proceedings of the 1st RIAO Conference on User-Oriented Content-Based Text and Image Handling (RIAO '88).*

Katz, B. (1997). Annotating the World Wide Web Using Natural Language. In *Proceedings of the 5th RIAO Conference on Computer Assisted Information Searching on the Internet (RIAO '97).*

Katz, B., Lin, J., and Felshin, S. (2001). Gathering Knowledge for a Question Answering System from Heterogeneous Information Sources.

Kazama, J. and Torisawa, K. (2007). Exploiting Wikipedia as external Knowledge for Named Entity Recognition. In *EMNLP-CONLL 2007*.

Khalid, M. A., Jijkoun, V., and Rijke, M. D. (2008). The Impact of Named Entity Normalization on Information Retrieval for Question Answering. In *ECIR 2008*, pages 705–710.

Ko, J., Si, L., and Nyberg, E. (2007a). A Probabilistic Framework for Answer Selection in Question Answering. In *HLT/NAACL 2007*.

Ko, J., Si, L., and Nyberg, E. (2007b). A probabilistic graphical model for joint answer ranking in question answering. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval - SIGIR '07*, page 343, New York, New York, USA. ACM Press.

Kozareva, Z., Bonev, B., and Montoyo, A. (2005). Self-training and co-training applied to spanish named entity recognition. *Lecture notes in computer science*, 3789:770.

Kozareva, Z., Riloff, E., and Hovy, E. (2008). Semantic Class Learning from the Web with Hyponym Pattern Linkage Graphs. In *Proceedings of ACL-08: HLT*, pages 1048–1056, Columbus, Ohio. Association for Computational Linguistics.

Krishnan, V., Das, S., and Chakrabarti, S. (2005). Enhanced answer type inference from questions using sequential models. *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing - HLT '05*, pages 315–322.

Kupiec, J. (1993). MURAX: A robust linguistic approach for question answering using an online encyclopedia. In *Proceedings of the 16th annual international ACM SIGIR conference on Research and development in information retrieval*, volume pp, pages 181–190. ACM New York, NY, USA.

Kwok, C., Etzioni, O., and Weld, D. (2001). Scaling question answering to the Web. *ACM Transactions on Information Systems*, 19(3):242–262.

Lafferty, J., Pereira, F. C. N., and Mccallum, A. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, volume pages, pages 282–289.

Lavelli, A., Califf, M. E., Ciravegna, F., Freitag, D., Giuliano, C., Kushmerick, N., Romano, L., and Ireson, N. (2008). Evaluation of machine learning-based information extraction algorithms: criticisms and recommendations. *Language Resources and Evaluation*, 42:361–393.

LDC (2008). ACE English Annotation Guidelines for Relations (version 6.2). Technical report, Linguistic Data Consortium.

Lee, S. and Lee, G. G. (2004). A bootstrapping approach for geographic named entity annotation. In *Asia Information Retrieval Symposium,(AIRS2004)*, pages 178–189. Springer.

Lee, S. and Lee, G. G. (2007). Exploring phrasal context and error correction heuristics in bootstrapping for geographic named entity annotation. *Information Systems*, 32:575–592.

Lehnert, W. G. (1977). A conceptual theory of question answering. In *Fifth International Joint Conference on Artificial Intelligence (IJCAI-77)*.

Lehnert, W. G. (1981). A computational theory of human question answering. In Joshi Aravind K.and Webber, B. L. and Sag, I. A., editors, *Elements of Discourse Understanding*. Cambridge University Press, Cambridge, England.

Li, X. and Roth, D. (2002). Learning question classifiers. *Proceedings of the 19th international conference on Computational linguistics* -, pages 1–7.

Li, X. and Roth, D. (2005). Learning question classifiers: the role of semantic information. *Natural Language Engineering*, 12(03):229.

Lin, C.-Y., Hovy, E., Gerber, L., Junk, M., and Hermjacob, U. (2001). Question Answering in Webclopedia. In *Proceedings of the Ninth Text REtrieval Conference (TREC-9 (2000)*, pages 655–664.

Lin, D. (1998). Dependency-based Evaluation of MINIPAR. In *Workshop on the Evaluation of Parsing Systems*.

Lin, D. and Pantel, P. (2002). Discovery of inference rules for question-answering. *Natural Language Engineering*, 7(04):343–360.

Lin, D. and Pinchak, C. (2006). A Probabilistic Answer Type Model. In *11th Conference of the European Chapter of the Association for Computational Linguistics: EACL 2006*.

Lin, J. and Katz, B. (2003). Question answering from the web using knowledge annotation and knowledge mining techniques. In *Proceedings of the twelfth international conference on Information and knowledge management*, pages 116–123. ACM New York, NY, USA.

Lin, J. and Katz, B. (2006). Building a reusable test collection for question answering. *Journal of the American Society for Information Science and Technology*, 57(7):851–861.

Lin, W., Yangarber, R., and Grishman, R. (2003). Bootstrapped learning of semantic classes from positive and negative examples. In *Proceedings of ICML-2003 Workshop on The Continuum from Labeled to Unlabeled Data*, pages 103–111. Citeseer.

Lita, L. V. and Carbonell, J. (2004). Instance-Based Question Answering: A Data-Driven Approach. In Lin, D. and Wu, D., editors, *Proceedings of EMNLP 2004*, pages 396–403, Barcelona, Spain. Association for Computational Linguistics.

Lita, L. V. and Carbonell, J. (2007). Cluster-Based Selection of Statistical Answering Strategies. In *20th International Joint Conference on Artificial Intelligence (IJCAI-07)*.

Litkowski, K. C. (2005). Evolving XML and Dictionary Strategies for Question Answering and Novelty Tasks. In Voorhees, E. M. and Buckland, L. P., editors, *The Thirteenth Text Retrieval Conference (TREC 2004)*.

Liu, B., Lee, W. S., Yu, P. S., and Li, X. (2002). Partially Supervised Classification of Text Documents. *ICML '02: Proceedings of the Nineteenth International Conference on Machine Learning*, pages 387–394.

Llopis, F., Escapa, A., Navarro, S., and Noguera, E. (2009). How long can you wait for your QA system? In *SIGIR 2009 Workshop on The Future of IR Evaluation*.

Llopis, F., Noguera, E., Ferrández, A., and Escapa, A. (2007). Evaluación de Sistemas de Búsqueda de Respuestas con restricción de tiempo. *Procesamiento del Lenguaje Natural*, 39:45–52.

Lopez, V., Motta, E., and Uren, V. (2006). PowerAqua: Fishing the Semantic Web. *The Semantic Web: Research and Applications*, pages 393–410.

Lowe, J. B. (2000). What's in store for question answering? Prognostications based on corpus analysis of several hundred million questions. In *Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*.

Magnini, B., De Rijke, M., Romagnoli, S., Vallin, A., Herrera, J., Peñas, A., Peinado, V., and Verdejo, F. (2003). The multiple language question answering track at CLEF 2003. In *Comparative Evaluation of Multilingual Information Access Systems, 4th Workshop of the Cross-Language Evaluation Forum, CLEF 2003*. Springer.

Magnini, B., Giampiccolo, D., Forner, P., Ayache, C., and Jijkoun, V. (2006). Overview of the CLEF 2006 Multilingual Question Answering Track. In *Proceedings of CLEF 2006*, CLEF 2006.

Magnini, B., Negri, M., Prevete, R., and Tanev, H. (2002). Is it the right answer?: exploiting web redundancy for Answer Validation. *40th Annual Meeting of the Association for Computational Linguistics*.

Magnini, B., Vallin, A., Ayache, C., Erbach, G., Peñas, A., de Rijke, M., Rocha, P., Simov, K., and Sutcliffe, R. (2005). Overview of the CLEF 2004 Multilingual Question Answering Track. In *Multilingual Information Access for Text, Speech and Images*, pages 371–391.

Malouf, R. and van Noord, G. (2004). Wide coverage parsing with stochastic attribute value grammars. In *In IJCNLP-04 Workshop: Beyond shallow analyses - Formalisms and statistical modeling for deep analyses.*

Manning, C. D. and Schütze, H. (1999). *Foundations of Statistical Natural Language Processing*. MIT Press.

Martínez-González, A., de Pablo-Sánchez, C., Polo-Bayo, C., Vicente-Díez, M. T., Martínez, P., and Martínez-Fernández, J. L. (2008). The MIRACLE Team at the CLEF 2008 Multilingual Question Answering Track. In *CLEF 2008 Working Notes*.

Martínez-González, A., Vicente-Díez, M. T., Martínez, P., de Pablo-Sánchez, C., Polo-Bayo, C., and Martínez-Fernández, J. L. (2009). The MIRACLE Team at the CLEF 2008 Multilingual Question Answering Track. *Lecture Notes in Computer Science*, 5706(Evaluating Systems for Multilingual and Multimodal Information Access):409–420.

Mayo, N., Kilgour, J., and Carletta, J. (2006). Towards an Alternative Implementation of NXT's Query Language via XQuery. In *Proceedings of the 5th Workshop on NLP and XML (NLPXML-2006)*.

McCallum, A. and Li, W. (2003). Early Results for Named Entity Recognition with Conditional Random Fields, Feature Induction and Web-Enhanced Lexicons. In *CONLL'03*.

McDowell, L. K. and Cafarella, M. J. (2006). Ontology-driven information extraction with ontosyphon. In *Science*. Springer.

Merialdo, B. (1994). Tagging English text with a probabilistic model. *Computational Linguistics*, 20:155–172.

Merkel, A. and Klakow, D. (2007). Language Model Based Query Classifcation. In *ECIR 2007*, pages 2–5.

Miller, S., Fox, H., Ramshaw, L., and Weischedel, R. (2000). A novel use of statistical parsing to extract information from text. In *Sixth Applied Natural Language Processing Conference*, pages 226–233.

Miller, S., Guinness, J., and Zamanian, A. (2004). Name tagging with word clusters and discriminative training. In *Proceedings of HLT-NAACL*, volume 4. Citeseer.

Mitkov, R. (2003). *The Oxford Handbook of Computational Linguistics*. Oxford University Press.

Mollá, D. (2006). Learning of Graph-based Question Answering Rules. In *Workshop on Graph Algorithms for Natural Language Processing (HLT/NAACL 2006)*.

Mollá, D., Berri, J., and Hess, M. (1998). A Real World Implementation of Answer Extraction. *Proceedings of the 9th International Workshop on Database and Expert Systems*.

Mollá, D. and Pizzato, L. A. (2008). Indexing on Semantic Roles for Question Answering. *IR4QA workshop at COLING '08*, (August):74.

Mollá, D., Schwitter, R., Rinaldi, F., Dowdall, J., and Hess, M. (2003). ExtrAns: Extracting answers from technical texts. *Intelligent Systems, IEEE [see also IEEE Intelligent Systems and Their Applications*, 18:12–17.

Mollá, D. and Vicedo, J. L. (2007). Question Answering in Restricted Domains: An Overview. *Computational Linguistics*, 33(1):41–62.

Monz, C. (2004). Minimal Span Weighting Retrieval for Question Answering. In *IR4QA workshop in SIGIR' 04*.

Morton, T. S. (2000). Coreference for NLP Applications. In *38th Annual Meeting of the Association for Computational Linguistics*.

Mota, C. and Santos, D., editors (2008). *Desafios na avaliação conjunta do reconhecimento de entidades mencionadas: Actas do Encontro do Segundo HAREM*. Linguateca.

Murphy, T. and Curran, J. R. (2007). Experiments in mutual exclusion bootstrapping. In *Proceedings of the Australasian Language Technology Workshop (ALTW)*, pages 66–74. Citeseer.

Nadeau, D. and Sekine, S. (2007). A survey of named entity recognition and classification. *Linguisticae Investigationes*, 30:3–26.

Negri, M., Kouylekov, M., Magnini, B., and Tanev, H. (2003). ITC-irst at TREC 2003: the DIOGENE QA System. In *The Twelfth Text Retrieval Conference (TREC 2003)*.

Ng, V. (2005). Machine Learning for Coreference Resolution: From Local Classification to Global Ranking. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, page 0.

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

NIST (2007). The ACE 2007 (ACE07) Evaluation Plan.

NIST (2008). Automatic Content Extraction 2008 Evaluation Plan (ACE 2008). Assesment of Detection and Recognition of Entities and Relations Within and Across Documents. Technical report, National Institute of Standards and Technology.

Nivre, J., Hall, J., Nilsson, J., Chanev, A., Eryigit, G., Kübler, S., Marinov, S., and Marsi, E. (2007). MaltParser: A language-independent system for data-driven dependency parsing. *Natural Language Engineering*, 13(January):1.

Nyberg, E. and Frederking, R. E. (2003). JAVELIN: A Flexible, Planner-Based Architecture for Question Answering. In *HLT-NAACL*.

Ogden, W., McDonald, J., Bernick, P., and Chadwick, R. (2006). Habitability in Question Answering Systems. *Advances in Open Domain Question Answering*, pages 409–431.

Palmer, M., Gildea, D., and Kingsbury, P. (2005). The Proposition Bank: An Annotated Corpus of Semantic Roles. *Computational Linguistics*, 31(1):71–106.

Palomar, M., Moreda, P., Llorens, H., and Saquete, E. (2008). The infuence of Semantic Roles in QA: A comparative analysis. *Procesamiento del Lenguaje Natural*, 41:55–62.

Pantel, P. and Pennacchiotti, M. (2006). Espresso: Leveraging generic patterns for automatically harvesting semantic relations. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, number Hindle 1990, pages 113–120. Association for Computational Linguistics Morristown, NJ, USA.

Pasca, M. (2003). *Open-Domain Question Answering from Large Text Collections*. CSLI Publications.

Pasca, M. (2007). Organizing and Searching the World Wide Web of Facts - Step Two: Harnessing the Wisdom of the Crowds. In *Proceedings of the 16th International World Wide Web Conference (WWW-07)*, pages 101–110.

Pasca, M. and Durme, B. V. (2007). What You Seek is What You Get: Extraction of Class Attributes from Query Logs. In *20th International Joint Conference on Artificial Intelligence (IJCAI-07)*, pages 2832–2837.

Pasca, M., Durme, B. V., and Garera, N. (2007). The Role of Documents vs. Queries in Extracting Class Attributes from Text. In *Proceedings of the 16th ACM Conference on Information and Knowledge Management (CIKM-2007)*, pages 485–494.

Pasca, M., Lin, D., Bigham, J., Lifchits, A., and Jain, A. (2006). Names and similarities on the web: Fact extraction in the fast lane. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, number July, pages 809–816. Association for Computational Linguistics Morristown, NJ, USA.

Peñas, A., Rodrigo, A., Sama, V., and Verdejo, F. (2006). Overview of the Answer Validation Exercise 2006. In *Cross Language Evaluation Forum (CLEF 2006)*.

Peñas, A., Rodrigo, A., Sama, V., and Verdejo, F. (2007). Testing the Reasoning for Question Answering Validation. *Journal of Logic and Computation*, 18(3):459–474.

Pérez-Coutiño, M., Montes-y Gómez, M., López-López, A., Villaseñor Pineda, L., and Pancardo-Rodr\'\iguez, A. (2006). A Shallow Approach for Answer Selection based on Dependency Trees and Term Density. In *Cross Language Evaluation Forum Workshop (CLEF 2006)*. Citeseer.

Pérez-Coutiño, M., Solorio, T., Montes-y Gómez, M., López-López, A., and Villaseñor Pineda, L. (2005). Question Answering for Spanish Supported by Lexical Context Annotation. In *Multilingual Information Access for Text, Speech and Images*, pages 502–511.

Pérez-Laínez, R., de Pablo-Sánchez, C., and Iglesias, A. M. (2009). Anonimytext: Anonimization of Unstructured Documents. In *Proceedings of the International Conference on Knowledge Discovery and Information Retrieval*.

Perret, L. (2005). *A Question Answering System for French*.

Pizzato, L. A., Mollá, D., and Cécile, P. (2006). Pseudo Relevance Feedback Using Named Entities for Question Answering. In *Australasian Language Technology Association Workshop*, pages 83–90.

Pomerantz, J. (2005). A linguistic analysis of question taxonomies. *Journal of the American Society for Information Science and Technology*, 56(7):715–728.

Porter, M. (2001). Snowball: A language for stemming algorithms.

Pradhan, S., Ward, W., Hacioglu, K., Martin, J. H., and Jurafsky, D. (2004). Shallow Semantic Parsing using Support Vector Machines.

Prager, J. (2004). Question answering using constraint satisfaction: QA-by-Dossier-with-Constraints. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL'04*, pages 574–581. ACL.

Prager, J., Brown, E., Chu-Carroll, J., and Czuba, K. (2006a). Question Answering by Predictive Annotation. *Advances in Open Domain Question Answering*, pages 307–347.

Prager, J., Chu-carroll, J., and Czuba, K. (2002). Use of WordNet Hypernyms for Answering What-Is Questions.

Prager, J., Chu-Carroll, J., Duboue, P., and Czuba, K. (2006b). IBM's PIQUANT II in TREC 2005. In Voorhees, E. M. and Buckland, L. P., editors, *The Thirteenth Text Retrieval Conference (TREC 2005)*.

Prager, J., Duboue, P., and Chu-Carroll, J. (2006c). Improving QA accuracy by question inversion. In *44th annual meeting of the Association for Computational Linguistics*, number July, pages 1073–1080, Morristown, NJ, USA. Association for Computational Linguistics.

Rabiner, L. R. (1989). A tutorial on hidden markov models and selected applications in speech recognition. In *Proceedings of the IEEE*, pages 257–286.

Radev, D. R., Prager, J., Brown, E., and Coden, A. (2000). Question-answering by predictive annotation. In *SIGIR 2000*, pages 184–191.

Radev, D. R., Qi, H., Zheng, Z., Blair-Goldenshon, S., Zhang, Z., Fan, W., and Prager, J. (2006). Query Modulation for Web Question Answering. *Advances in Open Domain Question Answering*, pages 285–305.

Ramakrishnan, G., Chakrabarti, S., Paranjpe, D., and Bhattacharyya, P. (2004). Is Question Answering an Acquired Skill?

Ravichandran, D. and Hovy, E. (2002). Learning Surface Text Patterns for a Question Answering System. *Proceedings of ACL*, (July):41–47.

Ravichandran, D., Ittycheriah, A., and Roukos, S. (2003). Automatic derivation of surface text patterns for a maximum entropy based question answering system. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, pages 85–87, Morristown, NJ, USA. Association for Computational Linguistics.

Riloff, E. (1996). Automatically Generating Extraction Patterns from Untagged Text. In *13th National Conference on Artificial Intelligence, Vol. 2*, pages 1044–1049.

Riloff, E. and Jones, R. (1999). Learning dictionaries for information extraction by multi-level bootstrapping. In *Sixteenth national conference on Artificial Intelligence and the eleventh Innovative Applications of Artificial Intelligence*, pages 474–479, Menlo Park, CA, USA. American Association for Artificial Intelligence.

Roberts, I. and Gaizauskas, R. (2002). Evaluating Passage Retrieval Approaches for Question Answering. In *Proceedings of 26th European Conference on Information Retrieval*, volume 2001. University of Sheffield.

Robertson, S. E. and Spärck Jones, K. (1976). Relevance weighting of search terms. *Journal of the American Society for Information Science and Technology*, 27:129–146.

Robertson, S. E., Walker, S., Jones, S., Hancock-Beaulieu, M., and Gatford, M. (1994). Okapi at TREC-3. In *Text Retrieval Conference (TREC 3)*, Gaithersburg.

Rocchio, J. J. (1971). Relevance Feedback in Information Retrieval. *The SMART Retrieval System: Experiments in Automatic Document Processing*.

Roger, S., Tomás, D., Ferrández, S., Ferrández, A., Peral, J., Llopis, F., and Aguilar, A. (2006). AliQAn, Spanish QA System at CLEF-2005. In *Accessing Multilingual Information Repositories*, pages 457–466.

Rosenfeld, B. and Feldman, R. (2006). URES : an Unsupervised Web Relation Extraction System. In *COLING/ACL 2006*.

Rosenfeld, B. and Feldman, R. (2007). Using corpus statistics on entities to improve semi-supervised relation extraction from the web. In *45th Annual Meeting of the Association for Computational Linguistics*, volume 45, page 600.

Rosenfeld, B. and Feldman, R. (2008). Self-supervised relation extraction from the web. *Knowledge and Information Systems*, 17:17–33.

Rosenfeld, B., Feldman, R., Fresko, M., Schler, J., and Aumann, Y. (2004). TEG—a hybrid approach to information extraction. In *ACM international conference on Information and Knowledge Management CIKM '04*, volume 9. Springer.

Rosso, P. and Buscaldi, D. (2006). Mining knowledge from wikipedia for the question answering task. In *Fifth International Conference on Language Resources and Evaluation (LREC'06)*, pages 727–730.

Rosso, P., Montes-y Gómez, M., Villaseñor Pineda, L., Pérez-Coutiño, M., Gómez-Soriano, J. M., and Sanchis-Arnal, E. (2005). INAOE-UPV Joint Participation at CLEF 2005: Experiments in Monolingual Question Answering. In *Cross Language Evaluation Forum Workshop (CLEF 2005)*, pages 21–23. Citeseer.

Roth, D. and Yih, W.-t. (2002). Probabilistic reasoning for entity and relation recognition. In *Proc. of COLING*, pages 835–841.

Ruiz-Casado, M., Alfonseca, E., and Castells, P. (2007). Automatising the learning of lexical patterns: An application to the enrichment of WordNet by extracting semantic relationships from Wikipedia. *Data & Knowledge Engineering*, 61:484–499.

Saggion, H., Gaizauskas, R., Greenwood, M. A., Hepple, M., and Roberts, I. (2004). Exploring the Performance of Boolean Retrieval Strategies for Open Domain Question Answering. In *IR4QA workshop in SIGIR' 04*.

Salton, G., Wong, A., and Yang, C. (1975). A vector space model for automatic indexing. *Communications of the ACM*, 18(11):620.

Sang, E. T. K. (2002). Introduction to the CoNLL-2002 Shared Task: Language-Independent Named Entity Recognition. In *Proceedings of CoNLL-2002*, pages 155–158. Taipei, Taiwan.

Sang, E. T. K., Forner, P., Peñas, A., Aguirre, E., Alegría, I. n., Forascu, C., Moreau, N., Osenova, P., Prokopidis, P., Rocha, P., Sacaleanu, B., and Sutcliffe, R. (2008). Overview of the CLEF 2008 multilingual question answering track. In *Cross Language Evaluation Forum Workshop (CLEF 2008)*, pages 19–21. Springer.

Sang, E. T. K. and Meulder, F. D. (2003). Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition. In *CONLL '03*.

Sang, E. T. K. and Veenstra, J. (1999). Representing Text Chunks. In *EACL'99*.

Santos, D. and Cardoso, N. (2008). GikiP: Evaluating Geographical Answers from Wikipedia. In *GIR'08*, pages 2–3.

Sarawagi, S. (2008). Information Extraction. *Foundations and Trends in Databases*, 1.

Schlaefer, N., Gieselmann, P., and Sautter, G. (2006). The Ephyra QA system at TREC 2006. In *Proceedings of the Fifteenth Text REtrieval Conference*.

Schlaefer, N., Ko, J., Betteridge, J., Sautter, G., Manas, P., and Nyberg, E. (2007). Semantic extensions of the Ephyra QA system for TREC 2007. *Sixteenth Text REtrieval Conference (TREC 2007)*.

Schlobach, S., Ahn, D., De Rijke, M., and Jijkoun, V. (2007). Data-driven type checking in open domain question answering. *Journal of Applied Logic*, 5(1):121–143.

Schuler, K. K. (2005). *VerbNet: A BroadCoverage, Comprehensive Verb Lexicon.* PhD thesis, University of Pennsylvania.

Séguéla, P., Laurent, D., and Nègre, S. (2006). QA better than IR? *EACL 2006, Workshop MLQA'06*.

Segura-Bedmar, I., Crespo, M., and de Pablo-Sánchez, C. (2009). Score-based Approach for Anaphora Resolution in Drug-Drug Interaction Documents. In *Proceedings of the 14th International Conference on Applications of Proceedings of the 14th Conference of Applications of Natural Language to Information Systems (NLDB 2009)*.

Segura-Bedmar, I., Crespo, M., de Pablo-Sánchez, C., and Martínez, P. (2010). Resolving anaphoras for the extraction of drug-drug interactions in pharmacological documents. *BMC bioinformatics*, 11 Suppl 2(Suppl 2):S1.

Sekine, S. (2006). On-demand information extraction. In *Proceedings of the COLING/ACL on Main conference poster sessions*, page 738. Association for Computational Linguistics.

Sekine, S. (2008). Extended named entity ontology with attribute information. In *Sixth International Language Resources and Evaluation (LREC'08)*.

Sekine, S. and Nobata, C. (2004). Definition, dictionaries and tagger for extended named entity hierarchy. In *Proceedings of the Language Resources and Evaluation Conference (LREC)*, pages 1977–1980. Citeseer.

Sekine, S., Sudo, K., and Nobata, C. (2002). Extended Named Entity hierarchy. In *Proceedings of the LREC-2002 conference.*

Settles, B. (2009). Active Learning Literature Survey.

Settles, B. and Craven, M. (2008). An analysis of active learning strategies for sequence labeling tasks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1070–1079. Association for Computational Linguistics.

Shen, D. and Klakow, D. (2006). Exploring correlation of dependency relation paths for answer extraction. In *21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, number July, pages 889–896, Morristown, NJ, USA. Association for Computational Linguistics.

Shen, D. and Lapata, M. (2007). Using Semantic Roles to Improve Question Answering. In *EMNLP 2007*, number June, pages 12–21.

Shinyama, Y. and Sekine, S. (2006). Preemptive information extraction using unrestricted relation discovery. In *NAACL '06*.

Sleator, D. and Temperley, D. (1993). Parsing English with a Link Grammar. In *Third International Workshop on Parsing Technologies*.

Solorio, T., Perez-Coutino, M., Montes-y Gómez, M., Villaseñor Pineda, L., and López-López, A. (2005). Question classification in Spanish and Portuguese. *Lecture Notes in Computer Science*, (1).

Soubbotin, M. M. (2001). Patterns of Potential Answer Expressions as Clues to the Right Answers. In *Proceedings of TREC-10*.

Speranza, M. (2007). Evalita 2007: The Named Entity Recognition Task. In *Proceedings of EVALITA*.

Srihari, R. K., Li, W., and Li, X. (2006). Question Answering Supported by Multiple Levels of Information Extraction. *Advances in Open Domain Question Answering*, pages 349–383.

Steinberger, R., Bruno, P., and Ignat, C. (2004). Exploiting Multilingual Nomenclatures and Language-Independent Text Features as an Interlingua for Cross-lingual Text Analysis Applications. In *Proceedings of the 4th Slovenian Language Technology Conference. Information Society 2004 (IS'2004). Ljubljana, Slovenia, 13-14 October 2004.*

Steinberger, R., Pouliquen, B., and Ignat, C. (2005). Navigating Multilingual News Collections Using Automatically Extracted Information. *Journal of Computing and Information Technology*, pages 257–264.

Stevenson, M. (2006). Fact distribution in information extraction. *Language Resources and Evaluation*, 40(2):183–201.

Stevenson, M. and Greenwood, M. A. (2005). A semantic approach to IE pattern induction. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, number June, pages 379–386. Association for Computational Linguistics Morristown, NJ, USA.

Stevenson, M. and Greenwood, M. A. (2006). Comparing information extraction pattern models. In *ACL 2006 Workshop on Information Extraction Beyond the Document*, number July, pages 12–19.

Strzalkowski, T. (1999). *Natural Language Information Retrieval*. Springer.

Suchanek, F. M., Ifrim, G., and Weikum, G. (2006). Combining linguistic and statistical analysis to extract relations from web documents. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, number April, pages 712–717. ACM New York, NY, USA.

Sudo, K., Sekine, S., and Grishman, R. (2003). An improved extraction pattern representation model for automatic IE pattern acquisition. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (ACL-03)*, number July, pages 224–231.

Sun, R., Jiang, J., Tan, Y. F., Cui, H., Chua, T.-s., and Kan, M.-y. (2005). Using Syntactic and Semantic Relation Analysis in Question Answering. In *Fourteenth Text Retrieval Conference (TREC 2005)*.

Sun, R., Ong, C.-H., and Chua, T.-S. (2006). Mining dependency relations for query expansion in passage retrieval. *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval - SIGIR '06*, page 382.

Surdeanu, M. and Ciaramita., M. (2007). Robust Information Extraction with Perceptrons. In *Proceedings of the NIST 2007 Automatic Content Extraction Workshop (ACE07)*.

Surdeanu, M. and Turmo, J. (2005). Semantic Role Labeling Using Complete Syntactic Analysis. In *CONLL 05*.

Suzuki, J., Taira, H., Sasaki, Y., and Maeda, E. (2003). Question classification using HDAG kernel. *Proceedings of the ACL 2003 workshop on Multilingual summarization and question answering* -, pages 61–68.

Szpektor, I., Tanev, H., Dagan, I., and Coppola, B. (2004). Scaling web-based acquisition of entailment relations. In *Proceedings of EMNLP*, volume 4, pages 41–48.

Talukdar, P. P., Brants, T., and Pereira, M. (2006). A context pattern induction method for named entity extraction. In *Tenth Conference on Computational Natural Language Learning*, page 141.

Tanev, H., Negri, M., Magnini, B., and Kouylekov, M. (2005). The DIOGENE Question Answering System at CLEF-2004. In *Multilingual Information Access for Text, Speech and Images*, pages 435–445.

Taskar, B., Guestrin, C., and Koller, D. (2003). Max-Margin Markov Networks. In *NIPS*.

Tellex, S., Katz, B., Lin, J., Fernandes, A., and Marton, G. (2003). Quantitative evaluation of passage retrieval algorithms for question answering. In *ACM Conference on Research and Development in Information Retrieval*.

Téllez-Valero, A., Juarez, A., Hernández, G., Montes, M., Denicia, C., Villatoro, E., Montes-y Gómez, M., and Villaseñor Pineda, L. (2007). INAOE's Participation at QA@CLEF 2007. In *CLEF 2007*, pages 1–6.

Téllez-Valero, A., Juárez-González, A., Montes-y Gómez, M., and Villaseñor Pineda, L. (2008). INAOE at QA@CLEF 2008: Evaluating Answer Validation in Spanish Question Answering. In *CLEF 2008 Working Notes*.

Thelen, M. and Riloff, E. (2002). A bootstrapping method for learning semantic lexicons using extraction pattern contexts. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing*, number July, pages 214–221.

Tiedemann, J. (2005). Integrating linguistic knowledge in passage retrieval for question answering. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing - HLT '05*, pages 939–946, Morristown, NJ, USA. Association for Computational Linguistics.

Tiedemann, J. and Mur, J. (2008). Simple is Best: Experiments with Different Document Segmentation Strategies for Passage Retrieval. In *Workshop IR4QA: Information Retrieval for Question Answering (SIGIR2004)*.

Tomás, D., Vicedo, J. L., Saiz, M., and Izquierdo, R. (2006). An XML-Based System for Spanish Question Answering. In *Accessing Multilingual Information Repositories*, pages 347–350.

Toral, A. and Munoz., R. (2006). A proposal to automatically build and maintain gazetteers for Named Entity Recognition using Wikipedia. In *EACL '06*.

Turmo, J., Ageno, A., and Català, N. (2006). Adaptive information extraction. *ACM Computing Surveys (CSUR)*, 38(2):4.

Turmo, J., Atserias, J., Carmona, J., Castellón, I., Cervell, S., Civit, M., Màrquez, L., Martí, M., Padró, L., Placer, R., Rodríguez, H., and Taulé, M. (1998). Morphosyntactic Analysis and Parsing of Unrestricted Spanish Text. In *First International Conference on Language Resources and Evaluation*.

Turney, P. D., Nadeau, D., and Matwin, S. (2006). Unsupervised Named-Entity Recognition: Generating Gazetteers and Resolving Ambiguity. *Advances in Artificial Intelligence (LNCS)*, 401:266–277.

Vallin, A., Magnini, B., Giampiccolo, D., Aunimo, L., Ayache, C., Osenova, P., Peñas, A., de Rijke, M., Sacaleanu, B., Santos, D., and Sutcliffe, R. (2006). Overview of the CLEF 2005 Multilingual Question Answering Track. In *Accessing Multilingual Information Repositories*, pages 307–331.

Van Zaanen, M., Pizzato, L. A., and Mollá, D. (2005). Classifying Sentences Using Induced Structure. *Lecture notes in computer science*, 3772:139.

Verberne, S., Boves, L., Coppen, P.-a., and Oostdijk, N. (2007). Discourse-based answering of why-questions. *Traitement Automatique des Langues*, 47(2):21–42.

Vicedo, J. L. (2002). *SEMQA: Un modelo semántico aplicado a los sistemas de Búsqueda de Respuestas*. Phd, Universidad de Alicante.

Vicedo, J. L. (2003). La Búsqueda de Respuestas: Estado Actual y Perspectivas de Futuro. *Revista Iberoamericana de Inteligencia Artificial*, 20:34–52.

Vicedo, J. L. and Ferrández, A. (2006). Correference in Q & A. *Advances in Open Domain Question Answering*, pages 71–96.

Vicedo, J. L., Saiz, M., Izquierdo, R., and Llopis, F. (2005). Does English Help Question Answering in Spanish? In *Multilingual Information Access for Text, Speech and Images*, pages 552–556.

Vicente-Díez, M. T., de Pablo-Sánchez, C., and Martínez, P. (2007). Evaluación de un sistema de reconocimiento y normalización de expresiones temporales en español. In *SEPLN2007*.

Voorhees, E. M. (2000). The TREC-8 Question Answering Track Report.

Voorhees, E. M. (2002). Overview of the TREC 2002 question answering track. In *Twelfth Text REtrieval Conference (TREC 2002)*, volume 142. Citeseer.

Voorhees, E. M. (2003). Overview of the TREC 2003 question answering track. In *The Twelfth Text Retrieval Conference (TREC 2003)*, volume 142. Citeseer.

Voorhees, E. M. (2006). *Evaluating Question Answering Performance*, chapter Evaluating, pages 409–431. Springer.

Wang, R. C. and Cohen, W. W. (2007). Language-independent set expansion of named entities using the web. In *Proceedings of IEEE International Conference on Data Mining*.

Wang, R. C., Schlaefer, N., Cohen, W. W., and Nyberg, E. (2008). Automatic Set Expansion for List Question Answering. In *EMNLP '08*, number October, pages 947–954.

Warren, D. H. D. and Pereira, F. C. N. (1982). An efficient easily adaptable system for interpreting natural language queries. *Computational Linguistics*, 8(3-4):110–122.

Whittaker, E. W. D., Chatain, P., Furui, S., and Klakow, D. (2005a). TREC2005 Question Answering Experiments at Tokyo Institute of Technology. In Voorhees, E. M. and Buckland, L. P., editors, *The Thirteenth Text Retrieval Conference (TREC 2004)*.

Whittaker, E. W. D., Furui, S., and Klakow, D. (2005b). A Statistical Classification Approach to Question Answering using Web Data. In *Proceedings of the International Conference on Cyberworlds*.

Whittaker, E. W. D., Novak, J., Chatain, P., Dixon, P., Heie, M., and Furui, S. (2007). CLEF2006 Question Answering Experiments at Tokyo Institute of Technology. *Lecture Notes In Computer Science*, 4730:351.

Wilensky, R., Chin, D. N., Luria, M., Martin, J. H., Mayfield, J., and Wu., D. (1989). The Berkeley UNIX Consultant project. Technical report, Technical Report CSD-89-520, Computer Science Division, the University of California at Berkeley.

Winograd, T. (1972). Understanding natural language. *Cognitive Psychology*, 3(1):1–191.

Wong, Y. and Ng, H. T. (2007). One class per named entity: Exploiting unlabeled text for named entity recognition. In *IJCAI-07*.

Xu, J., Licuanan, A., and Weischedel, R. (2003). TREC2003 QA at BBN: Answering Definitional Questions. In *The Twelfth Text Retrieval Conference (TREC 2003)*.

Yahoo! Yahoo! Search BOSS.

Yangarber, R. (2003). Counter-training in discovery of semantic patterns. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (ACL-03)*, pages 343–350.

Yangarber, R., Grishman, R., Tapanainen, P., and Huttunen, S. (2000). Unsupervised discovery of scenario-level patterns for information extraction. In *Proceedings of the sixth conference on Applied natural language processing*, pages 282–289. Association for Computational Linguistics Morristown, NJ, USA.

Yangarber, R., Lin, W., and Grishman, R. (2002). Unsupervised learning of generalized names. In *Proceedings of the 19th international conference on Computational linguistics*, number Coling, pages 1–7. Association for Computational Linguistics.

Yarowsky, D. (1995). Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of the 33rd annual meeting on Association for Computational Linguistics*, pages 189–196. Association for Computational Linguistics Morristown, NJ, USA.

Zelenko, D., Aone, C., and Richardella, A. (2003). Kernel methods for relation extraction. *Journal of Machine Learning Research*, 3:1083–1106.

Zhang, D. and Lee, W. S. (2003). Question classification using support vector machines. In *SIGIR '03: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 26–32, New York, NY, USA. ACM Press.

Zhang, Z. (2004). Weakly-supervised relation classification for information extraction. In *Proceedings of the thirteenth ACM international conference on Information and knowledge management*, pages 581–588. ACM New York, NY, USA.

Zhao, S. and Grishman, R. (2005). Extracting relations with integrated information using kernel methods. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, number June, page 426. Association for Computational Linguistics.

Zhu, X. (2007). Semi-Supervised Learning Tutorial.

Zhu, X., Ghahramani, Z., and Lafferty, J. (2003). Semi-supervised learning using Gaussian fields and harmonic functions. In *Twentieth International Conference on Machine Learning ICML-2003*, volume 20, page 912.

Zukerman, I. and Raskutti, B. (2002). Lexical query paraphrasing for document retrieval. In *Proceedings of the 19th international conference on Computational linguistics*, pages 1–7.