Towards deep content extraction from specialized discourse: The case of verbal relations in patent claims

Gabriela Ferraro

TESI DOCTORAL UPF / 2012

Prof. Dr. Leo Wanner

Department of Information and Communication Technologies

The court's PhD was appointed by the recto of the Universitat Pompeu Fabra on ....., 2010.

Chairman

Member

Member

Member

Secretary

The doctoral defense was held on ....., 2012,
at the Universitat Pompeu Fabra and scored as .....

PRESIDENT MEMBERS

SECRETARY

Acknowledgements

This thesis would not have been possible without the support of many people. I specially want to express my gratitude to my supervisor Leo Wanner for his constant support and patience during these years. Without his help I would not be able to complete this work.

Many thanks also to the members of my PhD Defense Committee (in alphabetical order): Kim Gerdes, Maarten Janssen, Horacio Rogríguez, Horacio Saggion, and Juan Manuel Torres Moreno, for their time and interest.

During the past five years, I profited greatly from the constant motivation, inspiration, and support from the TALN team: Alicia Burga, Luz Rello, Simon Mille, Gerard Casamayor, Nadjet Bouayad-Agha, Vanesa Vidal, Johannes Graën, Stefan Bott, Joan Codina, Roberto Carlini and Biljana Drndarevic. I also received valuable feedback from the people from the Human Language Technology Group during my research stay in Fondazione Bruno Kessler (especially Emanuele Pianta and Claudio Giuliano).

Thanks to Lydia and Judith for making my life easy at the DTIC.

Thanks also to Sören Brügmann for his selfless help to discover what goes on behind the scenes in the patent world.

Special thanks go to Rogelio Nazar; it was him who made me embark on this PhD journey. I also express my gratitude to the Institut Universitari de Lingüística Aplicada (IULA). I am in debt with Mercè Lorente and María Teresa Cabré, who gave me the opportunity to come to Barcelona and to the UPF. The two years I spent at the IULA were my first days in Barcelona, my first real contact with linguistics and research in general. I am also very grateful to Jorge Vivaldi for his guidance during my stay at the IULA. Thanks to my PhD fellows and IULA's friends: Anna Joan, Alba Coll, Amor Montané, Eufrocina Rojas, Iria da Cuhna, Jenny Azarian, Natalia

Seghezzi, Rejane Bueno, Sabela Fernández and Ross Salazar for their love and friendship .

Thanks to everybody who reviewed this thesis, especially to Evelyn, who did a tremendous job in correcting my English. I also want to express my love to my Barcelona friends: Carola, Martina, Marcos, Mariano, Patri, Silvina, Francesc, Romeu, and to the friends I left in Argentina, for their constant support.

Many thanks to my family for their love, education and encouragement. I cannot finish without saying how grateful I am to Hugo for being by my side.

This research was financially supported by the Agéncia de Gestió d'Ajuts Universitaris i de Recerca and partially by the projects PATExpert (FP6-ICT-028116) and TOPAS (FP7-SME-286639), funded by the European Commission.

Abstract

This thesis addresses the problem of the development of Natural Language Processing techniques for the extraction and generalization of compositional and functional relations from specialized written texts and, in particular, from patent claims. One of the most demanding tasks tackled in the thesis is, according to the state of the art, the semantic generalization of linguistic denominations of relations between object components and processes described in the texts. These denominations are usually verbal expressions or nominalizations that are too concrete to be used as standard labels in knowledge representation forms — as, for example, "A leads to B", and "C provokes D", where "leads to" and "provokes" both express, in abstract terms, a cause, such that in both cases "A CAUSE B" and "C CAUSE D" would be more appropriate. A semantic generalization of the relations allows us to achieve a higher degree of abstraction of the relationships between objects and processes described in the claims and reduces their number to a limited set that is oriented towards relations as commonly used in the generic field of knowledge representation.

Resumen

Esta tesis se centra en el del desarrollo de tecnologías del Procesamiento del Lenguage Natural para la extracción y generalización de relaciones encontradas en textos especializados; concretamente en las reivindicaciones de patentes. Una de las tareas más demandadas de nuestro trabajo, desde el punto vista del estado de la cuestión, es la generalización de las denominaciones lingüísticas de las relaciones. Estas denominaciones, usualmente verbos, son demasiado concretas para ser usadas como etiquetas de relaciones en el contexto de la representación del conocimiento; por ejemplo, "A lleva a B", "B es el resultado de A" están mejor representadas por "A causa B".

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The research on Relation Extraction (RE) from corpora has traditionally focused on the extraction of precise and predetermined relations from texts, e.g., time and location of the seances of a given movie; parts of a whole; the role a person plays in an organization such as member, owner, client, etc. Also, relations are usually extracted from homogeneous corpora, such as, the protein-protein interaction relation from a Biomedical corpus. As a rule, it is required that each relation of interest is specified a priori. Therefore, RE has relied so far on extensive human intervention to come up with manually crafted extraction patterns, extraction rules or manually tagged training examples. As a consequence, application of the developed model to a new domain requires that the user specifies the target relations, manually creates new extraction rules or tags and new training examples. All this implies severe limitations in scaling up the model (the cost of manual intervention multiplies with the number of target relations).

On the other side, enumerating all the potential relations of interest for automatic extraction is highly problematic. Just think, for example, of the enormous number of relations in a corpus as large and as heterogeneous as the web.

Recently, in a number of works, a new view on RE has been proposed, which facilitates the extraction of a large number of relations not specified beforehand. This thesis follows the same path. Its goal is to extract all possible verbal relations from a textual description and generalize them to such an extent that they can serve as abstract labels in a conceptual representation

of the object being described. More precisely, we are interested in the extraction and generalization of diverse *n-ary* verbal relations encountered in the descriptions of complex functional objects, i.e., devices. The corpus on which we focus is a corpus of patent claims; the devices are thus inventions that are described in great detail with respect to both their composition and function.

One of the most demanding tasks of the thesis, according to the state of the art, is the semantic generalization of names of relations between the object components and the processes described. These names are usually predicative lexemes that are too concrete to be used as standard labels in a conceptual map-like representation. Consider, for instance, "A leads to B", where "leads to" is equivalent, in abstract terms, to CAUSE: "A CAUSE B", in the same vein as "A causes B", "B is result of A", "B is the consequence of A", etc. A semantic generalization of the relations allows achieving a high degree of abstraction of the functional relations described in the patents, as well as a reduction of the number and types of relationships to a limited set that is oriented towards the relations used in the generic field of knowledge representation.

1.1 Motivation

Research on the extraction of relations from text corpora is a high priority topic in the Natural Language Processing (NLP) community. This is not surprising since the extraction of relations is at the core of NLP and many of its applications. For example, relations are the backbone for any ontology, and ontologies are being increasingly used in knowledge-based applications. Relations are also essential in applications such as semantic lexicon construction, question answering, textual entailment, text generation, etc.

As already mentioned, we are interested in the extraction of relations between complex objects described in patent claims. The claim section is the most important part of a patent document, as it defines the boundaries of the legal protection of an invention. In the claims, inventions are described in terms of their compositional and functional features. Thus, in order to understand them, it is necessary to identify not only the components of the invention in question, but also how those components interact with each other.

Working with patent claims entails new challenges and motivations. First, patents are a valuable and unique source of up-to-date scientific and technological information. Experts assume that only 10% to 15% of the content described in patents is described in other publications, and that the worldwide stock of patents comprises around 80% to 90% of scientific knowledge (Brügmann and Wanner, 2006).

Second, there is a great commercial demand for automatic patent processing applications. Nowadays, the most important intellectual property or patent-related processing services are offered by private companies.[1] Most of these companies offer solutions that to a considerable extent require manual intervention, such as patent translation, patent abstract compilation, search of similar technologies, etc. All of these solutions presuppose a high cost.

Third, but not less relevant, is that due to language style conventions, patent claims are extremely hard to read and comprehend, especially for laymen but also for NLP applications. Cf. a sample patent claim in (1):

> (1) An automatic focusing device comprising: an objective lens for focusing a light beam emitted by a light source on a track of an information recording medium; a beam splitter for separating a reflected light beam reflected by the information recording medium at a focal spot thereon and through the objective lens from the light beam emitted by the light source; an astigmatic optical system including an optical element capable of causing the astigmatic aberration of the separated reflected light beam; a light detector having a light receiving surface divided, except the central portion thereof, into a plurality of light receiving sections which are arranged symmetrically with respect to a first axis extending in parallel to the axial direction of the optical element and to a second axis extending perpendicularly to the first axis and adapted to receive the reflected beam transmitted through the optical element and to give a light reception output signal corresponding to the shape of the spot of the reflected light beam formed on the light receiving surface; a focal position detecting circuit capable of giving an output signal corresponding to the displacement of the objective lens from the focused position, on the basis of the output signal given by the light detector; and a lens driving circuit which drives the objective lens along the optical axis on the basis of the output signal given by the focal position detecting circuit.

In consequence, the extraction of all functional relations from a given claim, which could be then represented in a relational diagram illustrating its content would be highly desirable. However, as already pointed out above,

[1] The market of patent information and processing is controlled by the Thomson Group, Canada, which has a market share of nearly 80% (Brügmann and Wanner, 2006).

most of the works in RE focus on the detection of a restricted number of prominent relations, in particular *is-a*, *has-part*, *cause*, etc. Our application, which aims to provide comprehensive, easy-to-understand relations between complex functional objects described in patent claims, faces the need to derive a large number of relations that cannot be limited a priori.

All in all, both goals, to extract and to generalize an unlimited number of verbal relations in the domain of patent applications, make our work different from the known state-of-the-art relation extraction techniques.

1.2 Premises and basic assumptions

This research is based on the premise that a content relation can be rendered in a text by a verb, a predicative noun or a predicative adjective. More precisely, a content relation can be considered a relation of the type *label*(A1, A2, A3,...), where 'label' is the relation's name with the morphosyntactic category 'verb', 'noun' or 'adjective' and A_i ($i = 1, 2, \ldots$) are the arguments of the relation. In linguistic terminology, A_i fills a *valency slot* of the label.

Each predicative lexeme possesses a valency structure. The valency structure of a lexeme L specifies how many arguments (or actants) L has. For example, SEPARATE1 has three arguments, *X separates Y from Z*, cf. the sentence *The device separates the husk from the corn.*

Taking into consideration the premises presented above, our approach is based under the following basic assumptions.

- To extract a relation instance labelled by a lexeme L from a corpus we need to identify the argument instances of L and associate them with the specific slots in the valency structure of L. Only then we will be able to obtain the basis for a conceptual representation.
- The valency structure of a predicate can be captured by syntactic dependency relations in the sense that syntactic valency of L can be mapped onto L's semantic valency.
- Due to the fact that surface syntactic dependency relations (such as subject, direct object, indirect object, etc.) are not appropriate to capture the valency structure of the predicates, a deep syntactic dependency structure, which reflects the instantiated valency structures of the predicative lexemes involved, is needed.

As stated above, in natural language, relations are expressed by verbs, nouns and adjectives. In this thesis, we focus only on relations expressed by verbs. The main justification for choosing verbal relations is that they are the most productive predicate form. Note, however, that even if we limit our study to verbal relations, these assumptions are valid for all types of predicative lexemes.

1.3 Objectives

The technical language of patent claims is sufficiently explicit to allow the identification of the functional components of an object or technical process and the relations between them. This thesis attempts to exploit this fact in order to extract relations from the claims by applying NLP technologies.

The objectives of this thesis can be divided into NLP-specific goals and patent claim processing goals. In what follows, a description of both types of these goals is given.

NLP-research oriented goals

From an NLP point of view, the goal of this research is to develop natural language technologies for the extraction of verbal relations from corpora. The nature of the relations to extract is diverse, which means that the types of relations are not restricted to a predefined set. Moreover, our goal is to extract *n-ary* relations, since no restrictions about the arity of the arguments can be made either. In more precise terms, our NLP goals can be summarized as follows:

- Provide a framework for diverse, unlexicalized and *n-ary* verbal relation extraction.
- Provide a framework for the generalization of verbal relations.
- Implement a relation extraction and generalization program.

Patent claim processing oriented goals

The automatic processing of claims is highly demanded and attracts the attention of researchers and patent users with different backgrounds, such

as engineers, patent attorneys, patent searchers, and business managers, among others.

From the patent claim processing point of view, the goal is thus, to facilitate the achievements of the NLP goals:

- Provide a detailed description of the patent claim genre from the linguistic point of view. Patent claims can be considered a genre by themselves, as they have a unique style and a well defined structure. It is this complex linguistic style that makes patent claims difficult to understand. For this reason, it is necessary to study the lexical, syntactic and relational idiosyncrasies of the patent claim genre (as far as we know, there is no linguistic study about patent documents available as yet). Thus, the goals of the linguistic study of patent claims is, on the one hand, to facilitate the achievements of the NLP goals and, on the other hand, to offer new insights on the linguistic nature of patent claims.

- Implement a claim processing strategy and integrate it with the relation extraction and generalization.

1.4 Thesis Outline

This thesis is divided into 8 chapters. The first chapter is this Introduction. The remainder of the thesis is organized as follows.

Chapter 2 offers an overview of the characteristics of patent documents, more precisely, of the claim section of patents, as they are the object of the study in this thesis. Besides, we present an analysis of the genre of patent claims, focusing on the lexical and the syntactic idiosyncrasies of the style of claims.

Chapter 3 describes the theoretical framework of our research. First, we present the philosophy of the linguistic dependency framework and then, we present the linguistic theory on which our work is based on, namely the Meaning-Text Theory.

Chapter 4 review literature related to relation extraction in NLP and, in particular, literature that deals with the extraction of relations from

patents. Moreover, literature related to relation clustering and cluster labeling is presented.

Chapter 5 presents our proposed methodology to distil and generalize verbal relations from patent claims.

Chapter 6 provides a detailed description of our approach to verbal relation extraction and generalization is given. The aim of this chapter is to obtain empirical evidence that supports our basic assumptions.

Chapter 7 presents the experiments, evaluation results and discussion of our approach.

Chapter 8 draws the final conclusions of the thesis. The contributions of the thesis are specified and the limitations are discussed. The interest of the work and possible applications are also presented. Finally, potential directions for future work are examined.

Chapter 2

Patent claims

As mentioned in the Introduction section, the corpus used in this thesis is a corpus of patent claims. Patent claims represent a big challenge for NLP applications, since patents are notoriously difficult to read and comprehend. This is first due to their abstract vocabulary and very complex linguistic constructions. Patents are written vaguely to enlarge the scope of the invention, making it difficult for readers to handle. Second, each patent writer has a unique style and vocabulary, making the language used inconsistent from patent to patent. Third, the claims of patents are particularly challenging because in accordance with international patent writing regulations, each claim must be rendered in a single sentence. As a result, sentences with more than 250 words are not uncommon and such long sentences are difficult to process even for native speakers of English. In the next sections, we explain what claims are and present a detailed analysis of the genre of patent claims in order to better understand the corpus on which our research is based.

2.1 Patent claims

Patents have a predefined document structure that consists of several sections,[1] such as, Title, Abstract, Background of the invention, Summary of

[1]The terminology and requirements, however, may significantly differ from one legislation to another.

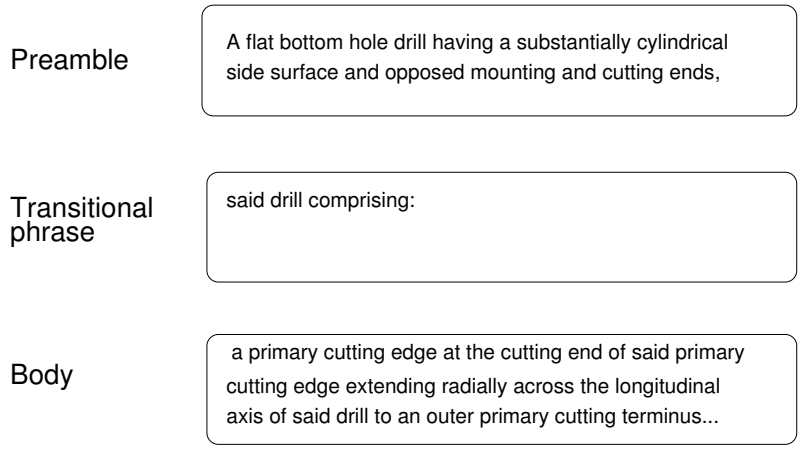

Figure 2.1: Prototypical illustration of a patent claim structure (illustrated on the patent EP0089123A1).

the invention, Description of the drawings and Claims. In this thesis, we focus on the claim section.

The claims are the main section of a patent document as they define the boundaries of legal protection (Pressman, 2006). In most modern patent laws, patent applications must have at least one claim (Pressman, 2006). A claim must define the matter for which the protection is sought in terms of technical features. These technical features can be either structural (e.g. *a disk*, *a recording medium*) or functional (e.g. *fastening means*, *processing unit*). As already mentioned, according to international patent writing regulations, each claim must be rendered in a single sentence.[2] Furthermore, a claim should be composed by, at least, the following parts: (1) a preamble, (2) a transitional phrase and (3) a body, as illustrated in Figure 2.1.

Preamble

Every claim should start with a **preamble**. A preamble is the claim introduction, which describes the class of the invention. The preamble may also include the main properties of the invention as well as its purpose or field.

For example, the preamble in Figure 2.1 introduces the class of the invention (*a flat bottom hole drill*) and its main properties (*having a substantially*

[2]Patent Cooperation Treaty, Rule 6 (Amernick, 1991).

cylindrical side surface and opposed mounting and cutting ends). A preamble that describes the class of the invention is, for example, *A recording method*. A preamble that **mentions** the class of the invention and its purpose is, for example, *A reproducing method for reproducing information audio data*.

Transitional phrase or linking word

A transitional phrase or linking word is a linguistic expression that relates the preamble of a claim to the specific elements set forth in the claim which define what the invention itself actually is. The expressions *comprising*, *containing*, *including*, *consisting of*, *wherein* and *characterized in that* are the most often used.

According to patent regulations, the scope of transitional phrases may differ. Nevertheless, their interpretation may be very close; e.g. for the USPTO (United States Patent and Trademark Office), *comprise* means *having at least the following elements* and therefore considered open or inclusive, and does not exclude additional limitations. Similarly, the EPO (European Patent Office) states that in every day language, the word *comprise* may have the meaning of *include*, *contain*, *comprehend* and *consist of*. However, in patent claims, *comprise* should be interpreted in a broader meaning (e.g. *include*, *contain* or *comprehend*, but not *consist of*). For the USPTO, the transitional phrase *consisting of* has a more limited scope, meaning *having all and only* or *virtually only* and therefore is considered closed or exclusive. Meanwhile, the phrase *consisting essentially of* has an intermediate scope and means that some additional components may be employed without infringement. To the best of our knowledge, these interpretations are not specified in the EPO.[3]

Body

The body of the claim describes the invention and recites its limitations. The elements of the invention should be described as though they interact or cooperate to achieve the desired result (Sheldon, 1992, 13), e.g. *A boring tool comprising a body, a plurality of cutting blades supported by the body so as to be movable along paths*. Optionally, a purpose clause that further describes

[3]For more examples of transitional phrases and how to use them, see (USPTO, August 2001).

the overall operation of the invention, or the goal that the invention achieves, may be included, e.g. *A pressing chuck provided on the sub-chuck head for holding the long bar member by pressing an end face of the bar member.*

Hierarchical structure of the claims

A claim may be an independent or a dependent claim. The dependency between independent and dependent claims defines the hierarchical claim structure. It is assumed that the deeper the claim structure, the more detailed information rendered in the claim. Dependent and independent claims are defined below.

Independent claims: claims which stand on their own and do not refer back to other claims. They generally describe the invention in a broad way. The first claim of a patent must be an independent claim and should reflect the whole picture of the invention.

Dependent claims: claims which cannot stand on their own and depend on single or several claims. A claim in dependent form shall be constructed to incorporate by reference all the limitations of the claim to which it refers. Dependent claims generally describe the invention in a narrower way and must be read in the context of the claims they depend on in order to be fully understood (Pressman, 2006). Dependent claims can be multiply dependent and may contain a reference to more than one claim previously mentioned. As a rule, a multiple dependent claim shall not serve as a basis for any other multiple dependent claim.

The dependency between claims is made explicit by textual references such as *according to Claim 1*, *as set forth in claims 1 to 3*, *as in claims 3-5*, *as defined by claims 1 and 2*, etc. Structures of different complexity can be encountered. Consider, for illustration, Figure 2.2. In this figure, the numbers stand for the individual claims and the arcs connect dependent claims with the claims on which they depend. Thus, 1 and 5 are independent claims since they do not depend on any other claims. Meanwhile, the rest of the claims are dependent; claims 2, 3 and 4 depend on claim 1 and 6, 7, 8 and 9 depend on claim 5.

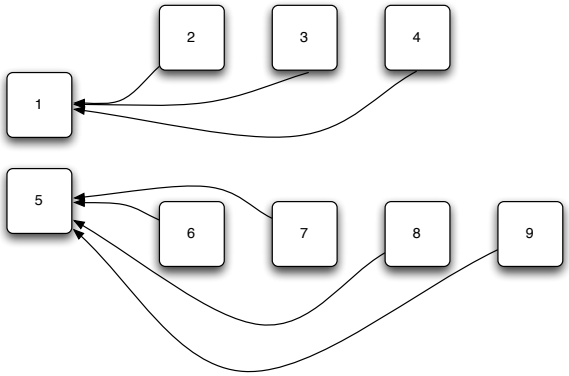

Figure 2.2: Illustration of the hierarchical structure of a patent claim.

Types of claims

Claims can be classified in terms of what they claim. The main two categories are:

- Product or apparatus claims: These claims are used to describe an invention regarding a physical entity, e.g. a product, a new material or an apparatus. The invention structurally recites the pieces of the equipment being claimed, e.g. *A trimming device comprising trimming blade means and a holder.*
- Process or use claims: These claims are used to describe an activity, e.g. a process, a method or a use. The invention recites series of steps or individual operations, where the clauses usually begin with present participle constructions, e.g. *A method comprising the steps of: providing a bevel gear, forming a gear teeth and machining one selected surface.*

In addition to the two basic claim types above, there are also many special types of claims which are used in different circumstances. Further types of claims include:

- Jepson Claim: For the USPTO, a Jepson claim is a method or product claim where one or more limitations are specifically identified as a point of novelty. The phrases *the improvement comprising* or *the*

improvement consisting of or *the improvement consisting essentially of* are generally employed to divide the preamble and the body of the claim.

- Markush Claim: Markush claims are mainly used in chemistry, but not exclusively. A Markush claim is a claim with multiple *functionally equivalent* chemical entities allowed in one or more parts of the compound (Amernick, 1991).

- Omnibus Claim: Omnibus claims include references to the description or the drawings without explicitly stating any technical features of the product or process being claimed. For instance, they may read as *Apparatus as described in the description* or *An X as shown in Figure Y.*

- Product by process Claim: A product by process claim is a product claim where the product is defined by its process of manufacture, especially in the chemical and pharmaceutical industries. They may read, for instance, *Product obtained by the process of claim X*, *Product made by the steps of ...*

- Swiss Claim: In the scope of the European Patent Office (EPO) a Swiss claim is a format intended to cover the first, second or subsequent medical use of a known substance or composition.

The patent corpus compiled for this thesis is composed of patents written in English[4] from the European Patent Office which correspond to product and process claims from two technical domains: *Optical Recording Apparatus* and *Machine Tools*. In addition, we have also studied product and process patents claims from USPTO to ensure that our text understanding approach can be applied to both.

2.2 Analysis of the genre of patents claims

As pointed out in Section 2.1, it is the complex linguistic style that makes patent claims difficult to understand and thus poses a challenge to computational implementations of text understanding. For this reason, it is necessary to review the lexical and syntactic idiosyncrasies of the patent claim genre.

[4]English is the most used language in patents applications around the world **?**.

Lexical idiosyncrasies

One of the most striking features of the patent claim genre is certainly its lexicon or vocabulary. Patent claim vocabulary tends to be very abstract since vagueness lies in the nature of the genre: a claim is intended to prevent competitors from working in a technological area. This is because, the bigger the area, the bigger is one's own freedom to operate in the area one claims. One way of reciting the elements as abstractly as possible, and consequently of making a claim broader, is using the word *means* plus a specific function. In this way, any device or means performing that function would infringe the patent. For example, *amplifying means* is broader than *transistor amplifier*, *tube amplifier* and *maser*.

The use of evasive words is another way of making a claim as abstract as possible. For example, the words *substantially*, *about* or *approximately* are used to avoid limiting a claim to specific dimensions. Consider the following examples:

(1) *said correction of spherical aberration is carried out* ***substantially*** *by the focusing/collecting arrangement*

(2) *said paraxial curvature radius is* ***approximately*** *infinite on* ***at leas****t one of the surfaces of the first optical element*

(3) *said diameter of the apertures ranges from* ***about*** *100 microns to about 2000 microns*

In example (1), the action of correction is carried out mainly by the *focusing/collection arrangement*, but it is not limited to it. In example (2), there are two expressions that make the statement abstract: on the one hand the dimension of the radius of the paraxial curvature is not fixed or determined and, on the other hand, the expression *at least* means that the mentioned radius may refer to any of the surfaces of the first optical disk. In example (3), the word *about* is used to avoid specifying the dimension of the diameter.

On the contrary, other words that can lead to a narrow interpretation of the claim are avoided, e.g. statements including words like *critical*, *must*, *required*, *necessary*, *only*, *always* and *never*.

In addition, the patent vocabulary is highly specialized and there are plenty of complex terminological units, usually long noun phrases; some of them composed by four or five words. Consider the examples presented in (4):

(4)

- *magneto optical recording unit*,
- *tool retainer collect means*,
- *long bar member machining apparatus*,
- *rotary/percussive portable power-operated drill*.

Syntactic idiosyncrasies

The syntactic structure is another peculiarity of patent claims. As already pointed out in Section 2.1, due to legal regulations, each claim is rendered in one single sentence, which results in unusually long sentences with a highly complex syntactic structure.

Perhaps the most singular aspect of a patent claim, from the point of view of syntax, is that, unlike general language sentences, claim sentences are not usually governed by a verb, and they do not follow the subject predicate construction as the basic syntactic structure. In contrast, claim sentences are mainly composed of complex noun phrases (NPs) or sequences of complex noun phrases that recite the characteristics of an invention. In addition, the noun phrases encountered in patent claims tend to be extremely long, a fact which makes the patent claim style unique.

In what follows we present a description of NP constructions and the way they are used in patent claims. This is necessary because NPs are the main, and sometimes the only, syntactic structure in a claim sentence. Moreover, we present subordinate and coordinate constructions and describe the anaphoric reference style as well as the main punctuation conventions found in patent claims.

Noun phrases

As mentioned above, NPs are the main syntactic structure in claim sentences, in contrast to the general discourse where clauses with finite verbs are the main type of constructions. NPs are composed by a noun head combined with modifiers. A modifier can be a single word, a phrase or a clause; its structure can be extremely varied and combined in several manners, see, for example, Table 2.1.

Noun modifiers	
Modifier category	**Example**
adjective	*blue beam*
noun	*chip face*
verb	*light detector* ***having*** *a light*
preposition	*primary cutting edge* ***at*** *the cutting end*
pronouns	*a plug* ***which*** *backs a frusto-conical pressure*

Table 2.1: Noun modifiers examples.

As shown in Table 2.1, a noun can be modified either by adjectives, other nouns, verbs, prepositions or pronouns. In the following, each type of modifier and how they are used in claim sentences is explained in detail.

Adjectives as noun modifiers

In English, adjectives are considered the most frequent and the most prototypical grammatical form that functions as noun phrase modifier. When several adjectives appear before a noun, they should follow a particular order Quirk et al. (1985). Adjectives are usually enumerated in the following order: *number, judgements, attitudes, size, length, height, colour, origin, material, purpose* and the noun, as in *second planar reflection surface.*

As already mentioned, patent vocabulary is highly specialized and tends to communicate knowledge through complex noun phrases, thus plenty of adjectives in the pre-modification position are not uncommon. Consider the noun phrases with several adjectives as pre-modifiers in (5):

(5)

- *first acute angle*
- *top mayor flat surfaces*
- *planar upper surface*

Nouns as noun modifiers

Nouns as pre-modifiers denote a syntactic relation between two nouns. This relation can be composition or apposition. In a composition structure, a head noun modifies another noun, e.g., *cutter blades.* In an apposition structure, the elements in the NP are not in a head-modifier relationship,

but in a relation of equality, e.g. in *element box* there is an apposition relation between *element* and *box*.

At the semantic level, two nouns may be related by an actantial relation, as in *unit gate*, where *gate* is the first actant of *unit*.

Participles as noun modifiers

A participle is a verb form which can be used as an adjective to modify a noun. Present and past participle constructions can be employed as noun pre-modifiers or noun post-modifiers, as in the examples in (6):

(6)
Participles as pre-modifiers:
working tool
additional clamping force
clamped device
cemented carbide support
Participles as post-modifiers:
an astigmatic optical system ***including*** *an optical element*
one or more optical components ***arranged*** *in said optical path*
a cutting edge ***disposed*** *at the cutting end of said drill*

Observe that, in claim sentences, participles as post-modifiers are usually used to specify the composition of the invention. For example: *an astigmatic optical system including an optical element*, *a light detector having a light*, etc. Especially in independent claims, we find verbs in present participle modifying sequences of coordinated complex nominal phrases. Commonly, present participles function as linking words expressing a part-whole relation between an NP and the sequences of coordinated NPs. The coordinated NPs are usually separated by a semicolon. Cf., the following claim fragment in (7) where the initial phrase *An optical disk drive comprising* is followed by a sequence of coordinated NPs (emphasized in boldface).

> (7) *an optical disk drive comprising:* ***a laser light source*** *for emitting a laser beam;* ***an optical system*** *for conversing the laser beam from the laser light source* [...]; ***one or more optical components*** *arranged in the optical path between the laser light source* [...]; ***a detection means*** *for detecting the light reflected from the optical disk; and* ***a signal processing circuit*** *for generating a secondary differential signal* [...]

The participle phrases are also found as sequences of embedded phrases, as in the example (8); participles constructions are emphasized in boldface.

(8) *an anvil member* ***characterized*** *in that said anvil is mounted to yield by means of a coupling* ***comprising*** *a number of axially extending springs* ***surrounding*** *and guided by pins* ***located*** *in corresponding counter bores* [...]

Another construction involving participles encountered in the claims are the to-infinitive constructions that modify a participle. These constructions are used to express the purpose of a noun, e.g.: *said gate* ***adapted to receive*** *the reflected beam*, where the to-infinitive phrase is a syntactic actant of the participle, thus *to receive* modifies the verb form *adapted*, not the noun.[5]

Prepositions as noun modifiers

In patent claims, we found plenty of prepositional phrases, most of them introduced by the preposition *for*. The *for*-prepositional constructions are post-modifying phrases with present participle constructions that are clearly used to express the purpose of a component or a functionality, as in (9):

(9) *an objective lens* ***for*** *focusing a light beam*

In the following sections, we present another two important syntactic relations found in the claims sentences, namely, subordinate and coordinate constructions.

Subordinate constructions

Subordinate constructions allow the writer to emphasize the most important aspects of an statement while still including relevant, even less important, information, as shown in example (10).

(10) *a coated cutting tool*
wherein said outermost layer of the coating has a surface roughness

[5]Note that in semantics, there is a relation between the noun *gate* and the verb *receive*, as *gate* acts as the first actant of *receive*.

Subordination of dependent clauses is usually introduced by subordinate conjunctions, adverbs or relative pronouns. Some of these dependent words have a specific meaning in patent claims and may be used and interpreted according to it, as illustrated and described in examples from (11) to (13).

(11) *an axial flow gas turbine engine arranged about a central axis comprising: a rotor mounted* ***such that*** *it can rotate about the central axis*
(12) *the method further comprising the step of: determining the reference signal* ***so that*** *it is based in part on tool operation with respect to the fluid*
(13) *a pre-treating system further comprising means for heating said sample holding means to* ***thereby*** *heat the sample*

The subordinate conjunctions *such that* and *so that* are used to restrict a part to a defined function, as in examples (11) and (12). The adverb *thereby* is used to specify a result or connection between an element and what it does, as shown in example (13).

(14) *a method further comprising the step of: determining the reference signal so that it is based in part on tool operation with respect to said fluid axis* ***whereby*** *said fluid axis is controlled in a closed loop manner*
(15) *said tool* ***wherein*** *the remote signal comprises an optical signal*

The subordinate conjunction *whereby* is used to introduce a function or result at the end of a claim (example 14) and the conjunction *wherein* is used to recite an element or mechanism more specifically, as shown in example (15). Other subordinated dependent clauses are introduced by conjunctions that do not have a special interpretation in the patent domain, for instance, examples (16), (17) and (18).

(16) *system as claimed in claim 12* ***where*** *said openings in said opaque mask are disposed in the pattern of an array*
(17) *said tilt sensor of claim 4, wherein said tilt detector senses the phase difference by detecting and comparing the levels of the two push-pull signals* ***while*** *the light beam focused by the objective lens is following the track*
(18) *said optical information reproducing apparatus according to claim 22, wherein each beam transmitting portion generates a plane-wave beam with a different out-of-plane angle* ***when*** *said beam transmitting portion is moved at an out-of-plane position*

Dependent clauses introduced by a relative pronouns like *which* and *that* are found very often as they serve as a means to connect a subordinated clause with a main clause. Consider, for example, the sentences below, where relative pronouns are emphasized in boldface.

(19) *a lens driving circuit* ***which*** *drives said objective lens along said optical axis*
(20) *a first planar reflection surface* ***that*** *reflects light*

As a claim takes the form of a single sentence, long sentences with several subordinations clauses are encountered, as they enable the elaboration of a clause already mentioned. Subordinations are usually found in sequences, that is, one after the other. When subordinate clauses are dependent, the question about to which main clause each dependent clause depends on arises. Sometimes, the context of the dependent clause is clear enough to determine its governor, but sometimes the governing clause remains ambiguous. Consider, for illustration, the claim fragments of examples (20), (21) and (22), where clauses are enumerated by new line breaks.

(21)
1. *a coated cutting tool*
2. *where in the outermost layer of said coating has an average* [...]
3. *for making contact with a workpiece except the vicinity* [...]
4. *when measured by a method*
5. *that observes said cross section of the tool*

(22)
1. *a tool damage detection device comprising a plunger type contact sensor*
2. *which is configured to detect damage of an edge of said tool,*
3. *said plunger type contact sensor being disposed in an outboard end*

(23)
1. *and a control means*
2. *for rotating said first motor in a given rotating direction with a given ratio of rotating speed*
3. *relative to a rotation of the second motor*
4. *when said second motor is driven*
5. *for indexing the rotary table*

Example (21) corresponds to a claim fragment consisting of five clauses (one clause per line), where the fifth clause is subordinated to the fourth clause and in turn the fourth clause is subordinated to the third clause, and so on. This subordination structure is considered right branching, because the subordination clause appears immediately to the right of its governor. Example (22) consists of three clauses, where both, the third and the second clauses, are subordinated to the first clause. In the claim fragment presented in example (23), the antecedent of the last clause remains ambiguous, as it is not clear whether the clause *for indexing the rotary table* has its antecedent in the forth clause or in the third clause.

Coordinate constructions

Coordination is the process of combining elements of equal importance by means of coordinating conjunctions or punctuation marks. The elements of the coordination construction can be NPs, subordinate clauses, main clauses, etc. See, for example, the following claim fragments:

(24) *said method comprising the steps of:*
a) providing a bevel gear blank having a gear-head;
b) forming gear teeth on said gear-head of said bevel gear blank [...];
and c) machining at least one selected surface of said unfinished [...]

(25) *an optical system*
for conversing said laser beam from the laser light source [...]
and for transmitting said light reflected from the signal plane [...]

(26) *and a head abutting inner ends of said blades*
for advancing
and retracting said blades on their paths consequent upon rotation [...]

Example (24) shows a coordination of present participle verbal phrases and example (25) shows a coordination of prepositional phrases. Observe that, as shown in example (26), in a coordination of prepositional phrases, the conjunction can be elided (the preposition *for* of the last clause is not present).

In claim sentences, we also find coordination of coordinated phrases, which means that a coordination can contain another embedded coordination, as in (27):

(27)
1. *a turret tool rest*
2. *wherein said clutch unit* ***includes***
3. *a clutch shaft*
4. *coupled to an output shaft of said electric motor*
5. ***and*** *a rotatable annular clutch member*
6. *arranged coaxially with said clutch shaft*
7. ***and*** *connected to said turret,*

In the previous example, the second and the fourth clauses are coordinated; moreover, the fifth clause, which is subordinated to the fourth clause, is coordinated with the sixth clause. In patent claims, coordinate constructions are usually used to enumerate the components of an invention, to describe the functionalities of a component or method, and to determine the spatial position of the components. Usually, each of the coordinated elements is a complex phrase with its own internal structure, which means that the elements of the coordination can be separated in the text by a long distance.

Anaphoric references

Even though patent claims have a vague style, minimization of ambiguity at the syntactic level is an important feature. To minimize ambiguity, a claim can not contain anaphoric references. Instead full NP repetition is a norm, even for complex NPs, sometimes at the cost of readability.

In patent claims, *said* is used as a determiner to indicate an explicit anaphoric reference. As a norm, every time there is a reference to a previously mentioned NP, it should be recited exactly by the same words and preceded by the expression *said*. Consider, for illustration, the claim fragment in (28):

(28) ***a recording media storage and player unit****, comprising: playback means for playing back data retrieved from a recording medium maintained at* ***said recording media storage and player unit***

Note that when the NP *recording media storage and player unit* is mentioned for the first time, it is preceded by the indefinite article *a*, but later in the same sentence, the mentioned NP is fully repeated, preceded by *said*.

Finally, it is important to note that, apart from the conventional types of constructions mentioned here, a large number of other *ad hoc* constructions can be also found.

Claim punctuation conventions

Because a claim is a single sentence, special punctuation conventions have been developed and are being increasingly used by patent attorneys. Modern Claims follow a format in which the preamble is separated from the transitional phrase by a comma, the transitional phrase is separated from the body by a colon and each of the elements of the invention in the body is separated by a semicolon Radack (1995). Other specifications regarding punctuation are the following:

- A comma should be used in all natural pauses.
- The serial comma[6] should be used to separate the elements of a list, e.g., *the unit process CDs, DVDs, and magnetic tapes*, where the comma indicates that *DVDs* and *magnetic tapes* are not mixed.
- A claim should contain a period only at its end.
- Dashes, quotes, parentheses or abbreviations should be avoided.

As we will see in the rest of the thesis, the linguistic idiosyncrasies of patent claims as discussed in the course of this chapter can be exploited for relation extraction and representation. The predominance of intra-sentential syntactic constructions, such as subordinate and coordinate constructions, with clear punctuation and lexical markers favour the segmentation of the patent claims and thus also the simplification of the original claim sentences (see Section 6.1). Similarly, explicit inter-claim references allow for the detection of the claim dependency structure. Furthermore, explicit anaphoric references make accurate coreference resolution possible (see Section 6.1).

[6]The serial comma (also known as the Oxford comma) is the comma used immediately before a coordinating conjunction (usually *and* or *or*, and sometimes *nor*) preceding the final item in a list of three or more items. http://oxforddictionaries.com

Chapter 3

Theoretical framework

As stated in the Introduction, the aim of this thesis is to obtain content relations via the valency structure of predicative lexical units. Our basic assumption is that the valency structure of the predicates can be captured by syntactic dependency relations, because syntactic valencies can be mapped onto semantic valencies. However, to better understand these assumptions, it is convenient to present the theoretical framework that sustains it, namely linguistic dependency.

In this Chapter, we first present the main ideas underlying linguistic dependency theories. Then, we present the Meaning-Text Theory, the linguistic dependency theory on which this thesis is based. Finally, we discuss why an intermediate syntactic representation, a deep syntactic representation, is needed between the conventional syntactic and semantic representations.

3.1 Linguistic dependency

Linguistic dependency has a long tradition in the description of linguistic constructions. More recently, it has generated a widespread interest in computational linguistics, where it is used for applications as diverse as information extraction, machine translation, ontology induction, and so on.

Dependency has been used as a formal means for representing the syntactic structure of a sentence by traditional syntacticians for centuries, especially in Europe, and particularly in Classical and Slavic linguistics. In modern

linguistics, the works by Tesnière (Tesnière, 1966), Hays (Hays, 1960, 1964), Robinson (Robinson, 1970) and Mel'čuk (Mel'čuk, 1988) are to be especially mentioned.

The dependency tradition comprises a large family of grammatical theories and formalisms that share some basic assumptions about dependency structures; in particular, the assumption that a dependency structure consists of terminal lexical elements linked by binary asymmetrical relations called *dependencies*. A study about variants of dependency grammars can be found in (Iordanskaja and Mel'čuk, 2009).

A dependency relation can be a morphological, a syntactic or a semantic relation; it is defined as a binary relation that takes two arguments. One argument is the *head* and the other argument is its *dependent*. Thus, $A \rightarrow B$ means "A governs B" or "B depends on A". A dependency relation has the following central properties,

- antisymmetry: the dependency relation between a *head* and its *dependent* is antisymmetric; if L_1 depends on L_2, then L_2 cannot depend on L_1.
- antireflexivity: dependency relations are antireflexive, which means that an L_1 cannot dependent on itself.
- antitransitivity: if L_1 is the head of L_2 and L_3 is the direct dependent of L_2, then L_3 can not directly depend on L_1.

To illustrate linguistic dependencies, we briefly describe the notion of syntactic dependency structure, which is one of the structures we use.

For dependency grammarians, a syntactic structure of a sentence is a rooted tree. In an syntactic dependency tree, each node is labelled by a lexeme and the links are labelled by grammatical function relations such as subject, direct object, indirect object, etc., as shown in Figure 3.1.

The structure in Figure 3.1 corresponds to the sentence *The main component, a disc device, comprises an optical pickup*. The root node of the syntactic tree is the verb *comprise*. The root node has two children, *component* and *pickup*, which are related to the root node by the relations *subject* (subj) and *direct object* (dobj), respectively. The node *pickup* has two dependants: *optical* which is related to *pickup* via the relation *modifier* (modif) and *a*, which is related to it via the relation *determiner* (det). The

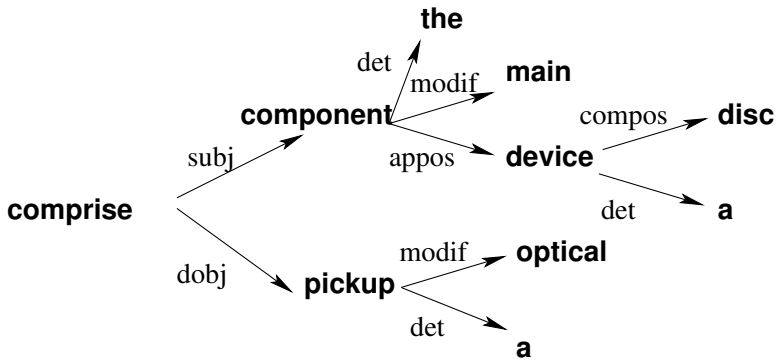

Figure 3.1: Example of a dependency syntax analysis.

node *component* has three dependants, *the*, *main*, and *device*; the is the *determiner* (det) of *component*, *main* is a *modifier* (modif) of *component* and between *component* and *device* an *apposition* (appos) relations holds. Furthermore, *device* governs two nodes, *disc* and *a*, which are related to *device* by the relations *modifier* and *determiner* (det) respectively.

By definition, syntactic dependencies link all words of a sentence. Therefore, sentences with unlinked elements are ungrammatical. Moreover, linguistic dependencies are labelled, which means that the relations are typed.

Nowadays, with state-of-the-art parsing technologies, it is possible to automatically obtain the syntactic dependency analysis of a sentence with reasonable accuracy.[1] With the syntactic structure at hand, it is possible to reach the semantic level, a representation that captures the valency structure of the predicates. This is possible because there is a projection from syntactic dependencies to semantic dependencies. See, for instance,

[1] An advantage of dependencies is that a syntactic dependency tree contains one node per word, so the job of a syntactic dependency parser is to connect existing nodes, not to postulate new ones, which makes the task of parsing in some sense easier. Another advantage of dependencies from the parsing perspective has been pointed out by (Covington, 2001), who said that syntactic dependency links are close to the semantic relationships needed for the next stage of interpretation, because it is not necessary to "read off" head-modifiers or head-complement relations from the syntactic tree that does not show them directly. Besides, syntactic dependency structures delivered by dependency parsers abstract over the word order of the sentence. Therefore, they are more appropriate to analyse language with more order variation such as German, Korean or Chinese (in contrast to constituent parsing, which performs best on languages with a fixed order like English). For more details on the state-of-the-art on dependency parsing, see (Nivre, 2005).

Figure 3.2: Example of the mapping from a syntactic to a semantic structure.

Figure 3.2, which shows a mapping between the syntactic relations and the semantic relations of the predicate *comprise*. Note that the syntactic dependency relation *subject* is mapped to the argument relation *1*, and the syntactic relation *object* is mapped to the argument relation *2*. Therefore, *subject* becomes the first argument (1) of the predicate *comprise* and *object* becomes its second argument (2).

However, the mapping from syntax to semantics is rather complex. At the syntactic level, dependency relations capture the grammatical dependencies of the lexical units, while, at the semantic level, dependency relations capture the predicate argument structure of the semantemes.

In the next Section, we present the Meaning-Text Theory (MTT) linguistic dependency theory that offers an intermediate representation between surface syntax and semantics, called deep syntactic representation.

3.2 The Meaning-Text Theory (MTT)

The Meaning-Text Theory (Mel'čuk, 1988) is a theoretical framework for the description of natural language. The MTT operates on the principle that language lies in a mapping from the meaning to the spoken text.

In the MTT, meanings are captured in a semantic representation and texts in a phonetic representation. In order to define the correspondence between these representations, a set of intermediate representation levels is defined. These levels go from deep to surface and the correspondence between meaning and text is bidirectional, as Figure 3.3 shows. A citation by Kahane (2003) sheds some further light on the nature of MTT:

> *Apart from the use of dependency rather constituency, MTT can be characterized by the massive relocation of syntactic information into the*

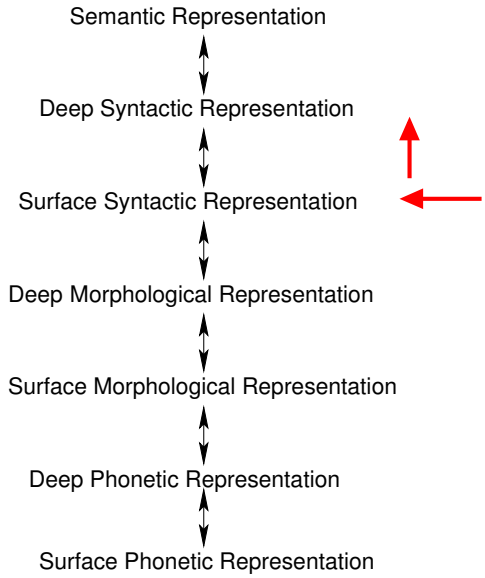

Figure 3.3: MTT representation levels (the unidirectional arrows at the right hand side mark aspects of MTT we deal with in the thesis).

> *lexicon—anticipating on that characteristic, most of the contemporary linguistic theories—and a transductive, rather than a generative, presentation, which favours the direction of (speech) synthesis, rather than analysis* (Kahane, 2003).

In Figure 3.4, we present the structures at different levels of representation in the MTT: networks for semantic representations, trees for syntactic representations, chains for morphological representations and strings of phone for phonetic representations.

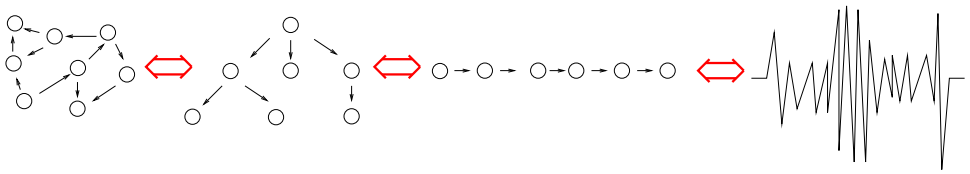

Figure 3.4: Degrees of freedom of the structures at different levels of the MTT model.

In what follows, we explain in detail the deep and surface syntactic levels of the MTT, i.e., the levels we use directly in this thesis. Moreover, we justify why a deep syntactic representation is needed for our purposes.

Syntax in the MTT

Syntactic structures in MTT are represented by dependency trees. There are two levels of syntax in MTT, the deep syntactic representation (DSyntR) and the surface syntactic representation (SSyntR); see Figure 3.3.

The **Deep Syntactic Structure** (DSyntS) is the main structure of the DSyntR; more details about the DSyntR can be found in (Milićević, 2006). A DSyntS is a syntactic dependency structure represented by an unordered tree. The nodes of the tree are labelled by generalized lexemes and the links or arcs between the nodes are labelled by a very small set of deep syntactic relations. A generalized lexeme can be one of the following items:

- a complete lexeme: a complete lexeme is a lexeme which is not semantically empty as, e.g., governed prepositions and conjunctions, as well as auxiliary verbs (which are introduced only in the surface-syntactic structure).
- a multilexemic phraseological unit: a multilexemic phraseological unit is a unit composed by more than one lexeme, but which is semantically a whole, e.g., an idiom.
- a lexical function: a lexical function is an abstract lexical unit F that associates a lexical unit L, called key word, to a collection of more or less synonymous lexical units (Li), called value of F, which express, in relation to L, a meaning represented by F.[2]

In this thesis, we do not use multilexemic phraseological units and lexical functions as nodes in DSynts.

[2] An example of an LF is *Magn*, which stands for any concrete lexical unit meaning 'intensification'. Thus, Magn associates L = RAIN, with {HEAVY, POURING, ...}, L = WIND with {STRONG, HEAVY, STIFF, ...}, L = BOMBARDMENT with {INTENSE,...}, etc. Usually, LFs are written as functions: Magn(RAIN) = HEAVY, Magn(WIND) = STRONG, Magn(BOMBARDMENT) = INTENSE. Note, however, that in mathematical terms LFs are not functions (but, rather, maps); see (Kahane and Polguére, 2001).

As we mentioned above, the set of deep syntactic relations is small; it consists of the following relations:

- Actantial relations: the actantial relations (I, II,..., VI) mark the actants of the lexical units.
- Attributive relation: the attributives relations (ATTR) cover all types of modifiers (circumstantials and attributes).
- Coordinative relation: the coordinative relation (COORD) covers all the coordinative constructions.
- Appenditive relation: the appenditive relation (APPEND) refers to parenthetical and intersection constructions.

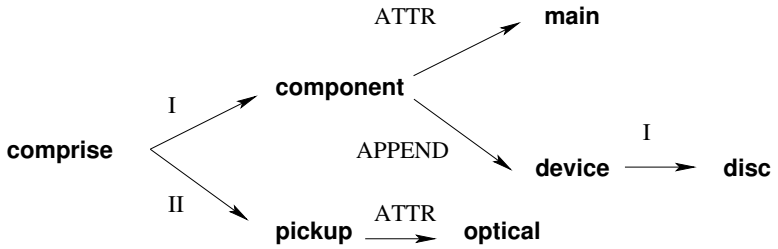

Figure 3.5: Example of a deep syntactic structure.

Figure 3.5 shows an example of the DSyntS. In this example, which corresponds to the sentence *The main component, a disc device, comprises an optical pickup*, the main verb, *comprise*, has two actantial relations. The first actant of *comprise* is *component* and, the second actant is *pickup*. The node *component* has two children, *main* and *device*; *main* is related to *component* by an attributive (ATTR) relation and *device* is related to *component* by an *appenditive* (APPEND) relation. Furthermore, the noun *device*, which is a predicate, has an actantial relation *I* with the noun *disc*. Finally, the noun *pickup* has an *attributive* (ATTR) relation with the adjective *optical*.

The **Surface Syntactic Structure** (SSyntS) is the main structure of the SSyntR, defined by (Mel'čuk, 1988, 68) as:

> *The Surface Syntactic Structure of a sentence is also a tree whose nodes are labelled with all the lexemes of the sentence (including all auxiliary*

> *and structural words), again there being a one-to-one correspondence between the SSynt-nodes and the lexeme; the arcs of this tree, also called branches, are labelled with names of language-specific Surface Syntactic Relations, each of which represents a particular construction of the language (their number, in an average language, is somewhere around 50).*

The definition establishes that the structure at the surface syntactic level is a dependency tree where the nodes are lexemes and the branches are labelled by grammatical functions (subject, object, modifiers, complements, etc.), which describe the syntactic constructions of particular languages.[3] Figure 3.1 at the beginning of Section 3.1, illustrates the SSyntS that corresponds to the DSyntS example presented in Figure 3.5.

As illustrated by the examples, there are some differences between the nodes in a surface-syntactic structure and those in a deep-syntactic structure (Kahane, 2003), which are the following,

- in a SSyntS, all lexemes of the sentence are present, including semantically empty (i.e. structural) words, while in a DSyntS, only meaningful lexemes are included;
- in a SSyntS, all idioms are expanded into actual surface syntactic trees, while in a DSyntS, they are considered a single unit, which is semantically a whole;
- the values of all lexical functions are computed (using the dictionary) and spelled out as actual lexemes, replacing the lexical function symbols in DSyntSs;
- all fictitious lexemes in a DSyntS are expressed in the SSyntS by the corresponding surface syntactic relations and thus disappear;
- in a SSyntS, the pronominalization is carried out, so that a surface syntactic node can be a pronoun, while in a DSyntS, only open class lexemes are used.[4]

[3] In the framework of MTT, lists of surface-syntactic relations have been proposed for Russian (Apresjan et al., 2006), English (Mel'čuk and Pertsov, 1987, 85-156), French (Iordanskaja and Mel'čuk, 2009) and Spanish (Mille et al., 2009, 2011).

[4] The exceptions are demonstrative and possessive pronouns. However, we refrain from entering into a deep linguistic discussion here.

There are also differences between the dependency relations in the SSyntS and those in the DSyntS. The main difference is that, while the inventory of SSynt relations comprise around fifty language-dependent relations, the inventory of DSynt relations comprise only nine relations, which are language-independent.

3.3 Why deep syntax?

The stratified model of the MTT offers the possibility to reach out to the semantic level from the syntactic level one, using an intermediate representation, namely the deep syntactic representation.

In the MTT model, this implies two mappings/transductions. A mapping from SSyntSs to DSynSs and a mapping from DSyntS to SemSs. However, as already stated, DSyntS are sufficient to capture the valency structure of predicate units. In this thesis, we took advantage of this fact and distilled relation tuples from deep syntactic structures, thus, our approach comprises one mapping, the one from SSyntSs to DSyntSs. To obtain DSyntSs, we parse our data with an off-the-shelf dependency parser and then map the obtained SSyntS to DSyntS from which relation tuples are distilled. The mapping from SSyntSs to DSyntSs is carried out using a transition grammar, which, somewhat simplified, can be defined as follows:

A transition grammar $TG_{\mathcal{S}_i \to \mathcal{S}_{i+1}}$ is a quadruple of the following kind:

$$TG_{\mathcal{S}_i \to \mathcal{S}_{i+1}} = (\mathcal{S}_i, \mathcal{S}_{i+1}, P_{\mathcal{S}_i \to \mathcal{S}_{i+1}}, \sqcup)$$

where

- $\mathcal{S}_i$ is a triple (Σ_i, R_i, C_i), with Σ_i as the node alphabet, R_i the arc alphabet and C_i the set of well-formedness criteria according to which structures over Σ_i and R_i are defined;
- $\mathcal{S}_{i+1}$ is a triple $(\Sigma_{i+1}, R_{i+1}, C_{i+1})$, with Σ_{i+1} as the node alphabet, R_{i+1} the arc alphabet and C_{i+1} the set of well-formedness criteria, according to which structures over Σ_{i+1} and R_{i+1} are defined;
- $P_{\mathcal{S}_i \to \mathcal{S}_{i+1}}$ is a set of transition rules that maps a partition of the fragments of any well-formed structure S_j of $\mathcal{S}_i$ onto a forest of well-formed structures of $\mathcal{S}_{i+1}$;

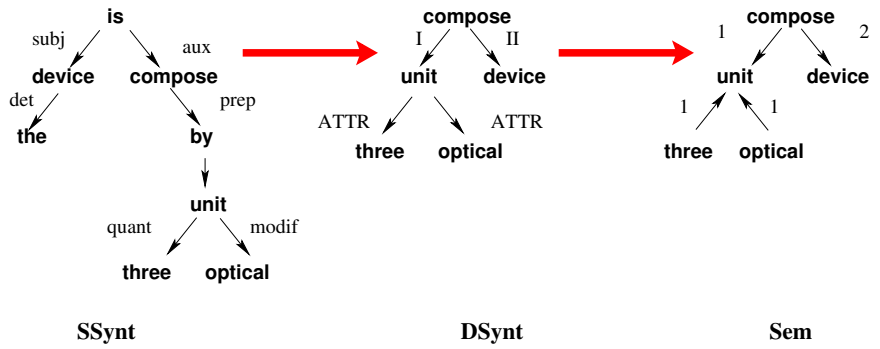

Figure 3.6: Example (1) of a mapping from a SSynt to Sem in the MTT model.

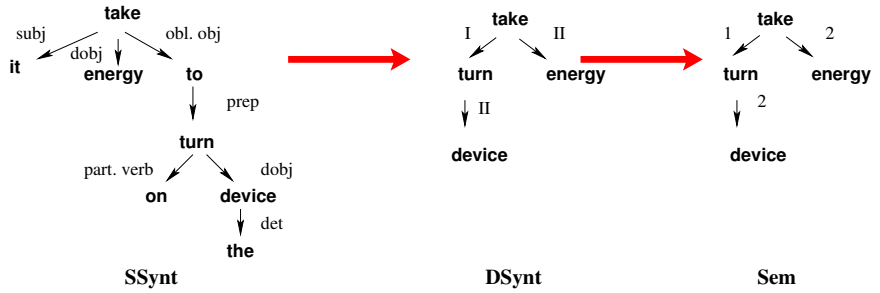

Figure 3.7: Example (2) of a mapping from a SSynt to Sem in the MTT model.

- $\sqcup$ is a unification operator that unifies the forest obtained by $P_{\mathcal{S}_i \to \mathcal{S}_{i+1}}$ when applied to S_j to a well-formed structure S_k.

For further details, see Kahane (2000); Bohnet (2006).

For illustration of why a DSyntS is preferred to a SSyntS, consider Figure 3.6 and Figure 3.7.

Figure 3.6 shows that the mapping from surface syntax to semantics is not straightforward. Observe that the *subject* of the verb *compose* (in its passive form) is *device*. But, the SSynt relation *subject* does not correspond to the first argument in the semantic structure, rather to the second one.

Consider also the mapping example of Figure 3.7. At the surface level, the verb *take* is the head of three dependent nodes, while at the semantic level, *take* requires only two arguments to fill its valency slots. Note also that the particle *it* is not mapped onto deep syntax, because it is semantically

empty. Thus, the SSynt relation *subject* again does not correspond to the first argument in semantics (rather, the *obl.obj* (oblique object) relation does it).

Note that the actantial relations in DSyntS are analogous to the arguments or valency structure of a predicate in a semantic structure. Certain non-actantial relations are reversed in semantics. For example, the ATTR-relation is reversed because its dependent is, at the semantic level, a predicative unit whose argument is the semantic node corresponding to its syntactic head. See, for example, the mapping from DSynt to Sem of the tuple *optical unit* in Figure 3.6.

As shown by the two examples, many nodes that appeared at the surface level are not mapped to the deep level. Thus, as already pointed out above, in surface syntax, all the lexemes from a sentence are present, while in deep syntax only those lexemes which are not semantically empty appear, which results in an abstract and simpler representation. Another advantage of the deep level is, certainly, its language independent nature, given that it allows us to work with any language using the same relations.

In summary, linguistic dependencies give us a formal means to work with natural language content. As explained above, the MTT provides elaborate and formal basics for the linguistic description. The formal structures for different levels of sentence representation and the possibility to map representations from one level to another are particularly useful for computer applications. Nowadays, dependency parsing technologies, which perform surface syntactic parsing, are very popular and their performance is acceptable. However, using surface syntactic structures to correctly identify the arguments of a predicate is not appropriate. To cope with this problem, we take advantage of the deep syntactic structure defined in the MTT, which is an intermediate representation level between the surface syntactic structure and the semantic structure.

Chapter 4

Related work

We reviewed the literature related to our work belonging to three fields: relation extraction, relation clustering and cluster labeling.

4.1 Relation extraction

Relation extraction addresses the problem of the recognition of a relation between two or more entities in a text. The identification and further extraction of relations in a text is at the core of Natural Language Processing and many of its applications, including ontology engineering, semantic lexicon construction and knowledge representation, among others.

In this section, we first present and discuss the most prominent techniques on relation extraction. Then, we give an overview of relation extraction techniques applied to patent data.

General discourse relation extraction

The problem of relation extraction has been commonly addressed drawing upon one of the following three sources of linguistic information:

- lexico-syntactic patterns,
- purely syntactic patterns,
- word co-occurrences.

The techniques that use this information are rule-based, machine learning-based or statistical.

Traditionally, **lexico-syntactic patterns** are mostly used for the extraction of a restricted set of relations identified beforehand. The patterns are determined either manually (Brin, 1998; Hearst, 1992; Berland and Charniak, 1999; Ogata and Collier, 2004), or derived from a corpus using machine learning techniques (Agichtein and Gravano, 2000; Snow et al., 2005; Bunescu and Mooney, 2005; Siddharth and Ellen, 2006; Rosenfeld and Feldman, 2006; Girju et al., 2006).

In some works, the lexico-syntactic patterns have been enriched by named entity tags or Wordnet features. For instance, the Snowball system (Agichtein and Gravano, 2000), which focusses on the extraction of the relation LOCATION - ORGANIZATION, include named entity tags in the extraction patterns, such that LOCATION matches only the tokens identified by an named entity tagger as entities of the type LOCATION and ORGANIZATION. The influence of named entity tags on relation extraction has been studied in detail (Giuliano et al., 2007). Girju et al. (2006) enriches relation patterns by hyperonymy features from Wordnet (Fellbaum, 1998) to distinguish between correct and incorrect PART-OF relations. Other extensions of pattern-based relation extraction include filters for selecting only the most prominent patterns; cf., e.g., (Blohm and Cimiano, 2007; Etzioni et al., 2008, 2011).

Purely syntactic patterns facilitate the extraction of any type of relation that matches them. Both dependency patterns and constituency patterns have been used. Ciaramita et al. (2005) extract "subject — verb — object" and "subject — verb — indirect object" pattern instances, which are then filtered for significance using the χ^2 test. The significant instances are

assumed to express conceptual relations. A manual evaluation is reported to have revealed 0.83 precision.

Cimiano et al. (2006) extract "$NP_1 - V - NP_2$" and "$NP_1 - V - PP - NP_2$" pattern instances and derive from them relations with the *Vs* as their labels and the heads of NP_1 and NP_2 as their arguments. As a result of an informal evaluation, the authors report that 15 of the 100 most frequent relations are considered inappropriate.

An important amount of work on relation extraction is based on **word co-occurrences** in restricted contexts. In general, co-occurrence identification techniques are based on n-gram frequency and statistical association measures. Yamaguchi (2001) presents an approach to learn non-taxonomic relations using high frequency 4-gram co-occurrences. Each 4-gram is represented as a vector with features such as the context around the appearance place of the terms in the 4-gram, the WN synset of each term in the 4-gram, etc. The similarity between pairs of 4-grams is calculated as the cosine of the angle between their vectors. If the similarity is above a certain threshold, a strong but unknown relation between both words is assumed. With a threshold of 0.9993 on a legal corpus, Yamaguchi extracts 90 different relations with a precision rate of 0.58).

Relation extraction via supervised machine learning techniques has been recently explored in breadth in such competitions as the ACE-competition (Doddington et al., 2004) and SemEval-competition (Girju et al., 2007). Both competitions provide a corpus tagged with a predefined set of relations for training. In ACE, the following five relations are used, some of them are further sub-classified such that a total of 24 relations is tagged: (i) ROLE (the role a person plays in an organization, with subrelations including Management, Member, Owner, Founder, Client, etc.); (ii) PART (the part of a whole); (iii) AT (location, with the sub-relations Located, Based-in and Residence); (iv) NEAR (relative location); and (v) SOCIAL (cognation, with the sub-relations Parent, Sibling, Spouse, Grandparent, Other-relative, etc.).

Among the works that use the ACE corpus are (Culotta and Sorensen, 2004; Reichartz et al., 2009; Surdeanu and Ciaramita, 2007). Culotta and Sorensen (2004) present a kernel-based relation extraction technique in which each relation instance is represented as a dependency tree and a set of features assigned to the nodes of the tree. The features include POS-tags, chunking tags, Wordnet-hyperonyms of the node, etc. Their dependency tree kernel shows an improvement of 20% over a bag-of-words kernel (in

which the tree is encoded as a vector of features over nodes, disregarding any structural information). Reichartz et al. (2009) propose a combination of kernels for phrase grammar parse trees and for dependency parse trees, which perform better than a single type parse tree, suggesting that both types of trees contain complementary information for relation extraction. Surdeanu and Ciaramita (2007) achieve an F-score of 0.33 with a technique that uses a Perceptron algorithm (Rosemblatt, 1958) and surface linguistic information.

The SemEval-2007 competition focuses on the extraction of a predefined set of seven semantic relations between nominals. The relations are:

1. Cause-Effect (e.g., VIRUS - FLU);
2. Instrument-Agency (e.g., LASER - PRINTER);
3. Product-Producer (e.g., HONEY -BEE);
4. Origin-Entity (e.g., RYE - WHISKY);
5. Theme-Tool (e.g., SOUP - POT);
6. Part-Whole (e.g., WHEEL - CAR);
7. Content-Container (e.g., APPLE - BASKET).

Many of the presented techniques in SemEval-2007 are kernel-based; e.g., (Giuliano et al., 2007), who achieves an overall F-score of 0.71. Beamer et al. (2007) use Support Vector Machines to train binary classifiers for each of the seven semantic relations. They report an average F-score of 0.72.

More ambitious works try to extract higher arity or *n-ary* relations, instead of restricting the extraction to binary relations. The work of (McDonald et al., 2005) proposed a method to extract *4-ary* relations between entities from biomedical texts. First, all possible binary links, no matter whether they are positive or negative relations, are extracted. Then, a standard classifier is trained to recognize pairs of positive related entities. Finally, a graph is constructed from the output of the classifier and the *n-ary* relations are determined from the maximum cliques of this graph.[1] Using a corpus

[1] A clique in an undirected graph is a subset of its vertices such that every two vertices in the subset are connected by an edge. A maximum clique is a clique that cannot be extended by including one or more adjacent vertex, that is, a clique which does not exists exclusively within the vertex set of a larger clique.

of 4773 entities and 1218 relations for training a binary Maximum entropy classifier, they obtained an F-score of 0.69.

Sekine (2005) deals with unsupervised relation discovery. He clusters pairs of named entities according to the similarity of context words between the named entities. The obtained clusters are supposed to group pairs of entities that entail the same relation. If most of the entities in a cluster had context words in common, these common words are used to labelled the cluster. The clustering evaluation is performed against a set of manually classified instances of two different pairs of entity domains: PERSON-GEOPOLITIC ENTITIES and COMPANY-COMPANY, getting an F-measure of 0.80 and 0.74 respectively. Regarding the evaluation of the cluster labeling, the authors mentioned that all the large clusters were accurately labelled in contrast to smaller clusters that were less accurately labelled.

Discussion

Relation extraction based on lexico-syntactic pattern techniques are restricted to binary relations and the relations to extract are usually a small set that tend to be very general. Nevertheless, these approaches are especially useful in applications where the types of relations of interest are known in advance, for example, for the population of an existing ontology with instances of genes that predispose a disease. The systems that use a small set of seed examples or a few hand-crafted lexico-syntactic patterns to start a semi-supervised or bootstrapping learning process are more suitable for RE as they require less manual intervention.

Purely syntactic pattern based approaches to relation extraction have the advantage of identifying all the relations that match the syntactic patterns. Syntactic patterns from syntactic dependency analysis are preferred to syntactic patterns from chunking analysis. Above all, because chunking structures are more ambiguous. A very well known problem is the so called, PP-attachment ambiguity, which arises from the fact that a prepositional phrase can either modify the preceding noun phrase or the verb phrase, as in *The recording unit controls the optical system with the recording control device*, in which *recording control device* can be either a modifier of *optical system* or it can be the instrument used by *the recording unit* to control *the optical system*.

We think that the SemEval and the ACE Programs provide valuable resources for the evaluation and comparison of current supervised approaches

to relation classification, although they are restricted to binary relations. The main drawback of these approaches is that they only focus on certain types of predefined information; thus they depend too much on the training data and limit the domain of application. For instance, in the patent domain it would be useful to have a category *LocationOf*, which is not in the SemEval or ACE relations inventory. Other relations used in these approaches are irrelevant for the analysis of patent documents due to the way patents are written. For example, some entities are never mentioned in a patent document such as Organization, Location or Geopolitics, among others. Moreover, the SemEval and the ACE data sets are difficult to extend to new relation types and to higher arity relations. In both competitions, tree kernel methods are preferred because they can use structural information such as syntactic dependency tree structures. Tree kernel methods take as input rich structural representations like parse trees in contrast to feature-based supervised approaches that take as input a set of features without structure. Consequently, the results obtained from tree kernel methods are usually better. However, even though supervised approaches like tree kernel methods perform well, they require a large amount of labelled data which takes a great deal of time and effort to be prepared.

The problem of binary relation extraction has received much attention in the last years, but its seems that the problem of extracting *n-ary* relations has only been scratched at the surface. As mentioned by McDonald et al. (2005), the higher the arity of a relation, the more likely it is that the arguments of it are spread out along the text, making long range dependencies especially important. This is certainly true in the case of patent claims, where the arguments of a relation may be distributed over a long distance from the relation itself (see Chapter 2). The work of (McDonald et al., 2005) is the first to use binary relation classifiers, which have been largely studied, to extract 4-ary relations. Nevertheless, this approach is restricted to *4-ary* relations and the authors did not mention whether it is possible to extend it to higher order relations. As tree kernel methods are the most accurate in the classification of binary relations, it would be interesting to investigate whether the combination of tree kernels and (McDonald et al., 2005) method improves accuracy.

Relation extraction in the patent processing domain

The work of Yang et al. (2007) proposes an interesting technique to extract semantic relations between components of the claims and the claim struc-

ture. The aim of this work is to facilitate the comparison between patent claims. The method of semantic relation extraction is based on domain-specific regular expressions. According to the authors, as claim sentences tend to be very long, regular expressions perform more efficiently than a natural language parser. They identified eight types of regular expressions to extract semantic relations, which are:

- Claim: three regular expressions, one for split claims according to the claims numbers, another to extract the claim type (independent/dependent) and a content type of the claim regular expression, if a method or an apparatus is being claimed.
- Component: regular expressions to extract components in the claims, with patterns including POS-tags.
- Reference: regular expressions to extract the reference links between components in a claim and the reference links between an independent claim and its dependent claims in order to capture the components that appear later in a claim refer back to the component that appears for the first time.
- Attribute: regular expression to extract what the authors call "the attribute description of a component". They define six classes:
    1. Property: component property, e.g., temperature and weight.
    2. Assignment: the relation between a property and its value, e.g., *greater than*, *equals to*.
    3. Range: the value range of an attribute, e.g., *from 1 to 4*.
    4. Unit: the unit of an attribute, e.g., *kcal*, *mm*.
    5. Value-unit: it represents the relation between the unit and the value, e.g., (Value + Unit) or (Range + Unit).
    6. Property value: it represents the relation between the unit and the value, expressed as (Property + Assignment + Value-unit).
- Functionality: regular expressions matching the preposition *for* to extract functional descriptions.
- Containment: regular expressions to extract *part-of* relation between components, with patterns such as *comprising*, *consisting of*, *essentially consisting of*, *including*, *having* and *containing*.

- Spatial: regular expressions to extract spatial relations among components such as *in*, *on*, *at*, *onto*, *opposite*, *surrounding*, etc., and also spatial relations among components that are related to positions like, *bind*, *adhere*, *form*, etc.

- Utility: common utility of regular expressions to be used by other regular expressions such as *match a white space character*.

Using a corpus of 40 patent claims from the CMP[2] domain, twenty patent claims as a training set and the rest as a test set, domain experts assigned the system a precision of 0.86 and a recall of 0.69 on average.

Parapatics and Dittenbach (2009) present a pattern-based approach to decompose patent claims into several parts in order to improve parsing performance for further automatic processing. The split parts are used to form a new dependency structure where each node is considered an independent sentence. In this work, there are different types of claims: Method claims, Use claims and Physical objects claims are analysed separately from each other. For example, the Method claims analysis is composed by the following patterns:

- Claim subject: the claim subject is usually the syntactic subject of the claim which is used as the root node in the new dependency structure.

- Characterized-pattern: the characterized-pattern is used to separate the preamble from the rest of the claim. Once identified, these chunks of text are attached as nodes to the root with the relation CHARACTERIZE.

- Description-pattern: the description pattern is identified by segments starting with the keywords *for*, *to* or *of* and ending with the Composition-pattern or the By-pattern. The identified chunk is appended to the root as a node with the METHOD-DESCRIPTION relation.

- By-pattern: the by-pattern starts with the particle *by* and is followed by a gerund verb; the identified chunk is appended to the root with the METHOD-BY relation.

[2]CMP is a global planarization process technology used in semiconductor manufacturing.

- Composition-pattern: the composition pattern extracts chunks starting with the words *comprises*, *comprising* and *including*, which are then attached to the root with the COMPOSITION relation.

They evaluate the extraction method by applying the Stanford parser (Klein and Manning, 2003b) to the extracted parts (the nodes of the new dependency structure) and to the original claims, improving parsing results by approximately 10% percent.

Pianta (2007) presents a content analysis module of patent claims with a shallow and a deep approach. The shallow content analysis module is implemented as key-concept extraction, using basic linguistics analysis (POS-Tagging and lemmatization) combined with basic statistical measures. The deep content analysis module is implemented as ontology learning and population, applying deeper linguistic analysis such as syntactic parsing. The method is as follows:

- Syntactic parsing of the patent claims with the Minipar (Klein and Manning, 2003a) syntactic dependency parser.
- Frame recognition, understanding as a frame a structure that describes an event or a state and the roles of its participants. They define six types of frames: *part whole*, *cause increase of*, *cause decrease of*, *has figure*, *has label*, *is of drawing type*, *conveys information subject*.
- Knowledge Base (KB) triple extraction, consisting of mapping frames to KB-relations and frame elements to KB-concepts.

To recognize a frame instance some semantic restrictions are applied to the frame elements. The semantic restrictions are expressed in terms of Wordnet synsets, for example, the elements of an *inclusion* relation has to be of the type *artifact*.

The PAT-Analyser system (Cascini and Rissone, 2003) is, according to its authors, a methodology and a software tool capable of extracting the functional model of a patent automatically, by means of semantic analysis of patent texts. PAT-Analyser works as follows: first, the components of the invention are identified. This is done taking advantage of the list of drawings included in the patent document, where the components are enumerated.

They also apply a lemmatizer and a filter to improve results (although they do not give details about how the filter is). Second, the components identified in the first step are classified according to their detail level of abstraction, where abstract components are over less abstract components, similar to the compositional structure of the invention. Finally, relations between components are identified with a syntactic parser (although the kind of syntactic parser is not mentioned), using the most frequent verbs as relations.

Discussion

The works of Yang et al. (2007) and Parapatics and Dittenbach (2009) are strongly domain dependent because regular expressions have to be specified first by human experts. Moreover, regular expressions, as other patterns, are difficult to define, error-prone and often produce poor recall when doing text analysis. We agree with Yang et al. (2007) that syntactic parsers performed poorly in patent claims as claim sentences are very long, although we think regular expressions have several limitations, for example, they are not suitable for parsing arbitrarily nested text. One advantage of (Yang et al., 2007) approach is the identification of the patent claim structure, which is a useful feature that may be exploited when doing patent claim analysis since the logic behind the structure of claims express import facts about the knowledge encoded in the claims.

Regarding the work of Pianta (2007), we consider that the deep analysis, carried out as frame instantiation, offers poor recall with respect to the content of the documents because it is focussed on a certain pre-defined set of relations. However, the extracted frames are represented as formal assertions in a knowledge base that can be further used, for example, in semantic search.

The idea in (Cascini and Rissone, 2003) is similar to our objectives, as they intented to also draw a visual representation of the objects described in patents. Concerning PAT-Analyser limitations we refer to the assessment done by Brügmann and Wanner (2006), where most evident errors are mentioned:

- Limited lemmatization, there is no distinction among singular and plural, e.i. *jaw* vs. *jaws*, *side* vs. *sides*.

- POS-Tagging errors, e.i. *lower* is interpreted as a noun, possible cause by an error on the tagger or for the impossibility to resolve coordinative constructions, like, *lower and upper case jaws*.

- Chunking errors: the expression *opposite flat* is interpreted as a noun phrase and consequently is identified as a component. This error leads to a non-existent component included in the diagram. The same expression, *opposite flat*, is recognized as a modifier in the expression *opposite flat surfaces*.

- There is no semantic association of the verbs as an strategy to minimize the number of relations, both relational and functional, e.i. *make with* and *use*.

We observed that the state-of-the-art RE techniques are not yet fully explored in the patent processing domain. In part, because patents are freely available, or available at a reasonable price, from not long ago. Nevertheless, due to the particular style of patent claims, some authors prefer to adopt *ad hoc* strategies to capture the peculiarities of the claims, as (Yang et al., 2007; Parapatics and Dittenbach, 2009).

4.2 Relation clustering

Relation clustering deals with grouping relations with respect to a predefined typology or are based on the similarity between relations. Grouping similar relations is often called relation *synonym resolution* or *paraphrases discovery* or acquisition (Barzilay and Lee, 2003; Sekine, 2005). In this work, we focus on the relations expressed by verbs.

Most of the recent work on the semantic classification of verbs draw upon the verb taxonomy of Levin (1993) using supervised and unsupervised machine learning techniques (Brew and Schulte im Walde, 2002; Schulte im Walde, 2006; Korhonen et al., 2006; Sun et al., 2008; Sun and Korhonen, 2009; Vlachos et al., 2009). Somewhat simplified, this taxonomy captures the projection of semantic valency structures of verbs onto their syntactic valency structures. Since our goal is a grouping (or clustering) based on purely semantic criteria, we make no use of Levin's taxonomy.

A number of proposals focus on the classification of (already extracted) relation instances, for example, using a semantic distance measure. Plenty

of works undertaken to calculate the similarity distance between words. One way to know how similar or different a word is with respect to other words is to measure their similarity in a Vector Space Model (VSM) (Salton and McGill, 1986; Salton and Buckley, 1988; Salton, 1989). Turney and Littman (2005) extends the VSM to define a new technique called Latent Relation Analysis, a method for measuring the similarity of the semantic relation between pairs of words. For example, the pair *cat:meow* and *dog:bark*.

Regarding unsupervised systems for the identification of paraphrases we refer to the work of (Sekine, 2005), who uses a heuristic similarity measure to cluster similar relations. Davidov and Rappoport (2008) use a heuristic clustering method to find groups of relation patterns that can be used to extract instances. Hasegawa et al. (2004) applies the Cosine Similarity Metric (Salton and McGill, 1986) and a hierarchical clustering technique to group similar relations. The DIRT system (Lin and Pantel, 2001) applies a similarity measure based on mutual information to identify relations that are similar to a given one. For example, given the tuple *Y is solved by X*, the system found similar tuples like *X resolves Y*, *X finds a solution to Y*, *X deals with Y*, etc. To identify relation and noun synonyms, the Resolver system (Yates and Etzioni, 2009) uses a probabilistic model based on string similarity and on the similarity of the instances where relations appear (inspired on the distributional similarity metrics (Lee, 1999)). Finally, a hierarchical clustering technique is applied on the basis of obtained similarities.

Another way to calculate the similarity of relations is to use the Wordnet's hierarchical structure by comparing the hyperonymy chains of the relations. The work of Yang and Powers (2006) use the Wordnet noun hierarchy to measure the semantic similarity between nouns based on a variation of edge counting. They also applied these techniques to classify verbs. To counterbalance the shallowness of Wordnet's verb hierarchy, they attempt to take advantage of the noun hierarchy and definitional Wordnet glosses. For this, they do verb nominalization and stemming. Despite these auxiliary measures, they conclude that the Wordnet verb hierarchy is too shallow for assessing the similarity between verbs.

Discussion

The most evident drawback of grouping relations with respect to a predefined typology is that, sometimes, instances of relations are hard to classify,

either because they fall into more than one category or because they do not fit under any of them.

We think the major drawback is that it is not realistic to assume that one can define an exhaustive typology beforehand, at least not with restricted effort, as is the case with broad domain as patent documentation (see, e.g., WordNet or Cyc, into which a lot of money and a lot of time, over 10 years now, with a lot of manpower have been invested, without that we can say that they are complete and serve our purposes).

In general, similarity metrics used in relation clustering produces similarity scores for relations by comparing the distributions of the relation arguments. Moreover, in some works, the arguments of the relations are classified in terms of named entities such as PERSON, LOCATION, etc. But, comparing the distributions of the arguments, in terms of named entities, is not adequate nor enough to group relations in the patent claim domain, as the relation arguments are too homogeneous. The arguments of the relations are usually complex terminological units that refers to mechanical or electronic devices. Thus, the application of a classical named entity tagger may be useless, as most of the arguments will be classified as DEVICE or PHYSICAL-OBJECT, making the distribution of the relation arguments too homogeneous and consequently insufficient to characterize different types of relations.

The mentioned works make an important simplifying assumption: they assumed that every relation belongs to exactly one cluster, thus polysemy is not taken into account. To take polysemy into account soft clustering techniques may be a solution, allowing a lexeme to be assigned to as many different clusters as senses it has. Although some solutions deal with polysemous nouns, as far as we know, polysemous relations or verbs have not been taken into account. Our approach to deal with polysemy is described and justified in Section 6.2.

4.3 Cluster labeling

Cluster labeling deals with finding an appropriate name for a given group (= cluster) of similar elements. Two main strategies can be distinguished in the literature: (i) cluster internal labeling and (ii) differential cluster labeling. In the first, the label of a given cluster is chosen drawing solely

on the cluster itself. In the second, the label for a cluster is chosen by contrasting this cluster with other available clusters.

The proposal by Pantel and Ravichandran (2004) is an algorithm for automatically inducing names for semantic classes of nouns. The classes consist of instances grouped according to their attributional similarity (not relation similarity). The input of their system are semantic classes (cluster of nouns) and the output of the system is a ranked list of label names for each semantic class. First, for each member of a cluster, grammatical signatures that capture its prototypical semantic context in different occurrences are computed. In other words, each word of a cluster is represented by a feature vector where each feature corresponds to a context in which the word occurs, understanding as context the grammatical relationship outputs by the Minipar parser. For example, "catch —" is a verb object context. If the word *wave* occurred in this context, then the context is a feature of *wave*. Then, among these signatures, simple hyperonymy patterns, such as "Noun–apposition–Noun" (e.g., H1N1, the disease) are searched. At last, the mutual information scores for each hyperonymy candidate are calculated and the highest scoring hyperonymy is chosen as the name of the cluster.

Further similar proposals are (Carmel et al., 2009) and Manning et al. (2008). The proposal by Dias et al. (2009), which addresses the problem of clustering of web page results and the subsequent labeling of the obtained clusters, is an example for differential cluster labeling. It chooses as a label of a cluster a noun or a noun compound that (i) occurs in most of the URLs of the cluster in question, and (ii) discriminates the cluster sufficiently well from the other clusters.

The more complex problem of labeling nodes in a hierarchy (which requires distinguishing more general labels for parents from more specific labels for children) is tackled by (Glover et al., 2002) and (Treeratpituk and Callan, 2006). Glover et al. (2002) defines a set of features that according to its frequency in a set of clusters are useful to identify three different types of terms for labeling a hierarchical clustering solution. The frequency features are:

- **Self terms** that describe a cluster as a whole: if a term is very common in a cluster but relatively rare in the collection, then the term is considered a good self term.
- **Parent terms** that describe more general concepts: if a term is common in a cluster but also common in the entire collection, then the

term is considered more general and appropriate as a good parent feature.

- **Child terms** that describe specializations of a cluster: if a term is common in a cluster but very rare in the general collection, then the term is considered a good child feature because it only describes a subset of the positive documents.

In the work of (Treeratpituk and Callan, 2006) the goal is also to produce appropriate category labels for a cluster of documents in a cluster hierarchy. Given a cluster and its parent cluster, the algorithm selects labels for the cluster according to the following four steps:

1. Collect phrase statistics: document frequency and term frequency of a phrase p with respect to the given cluster, the parent cluster and a general English corpus; the number of documents in the cluster that contain p and the total number of occurrences of p in the cluster.
2. Select label candidates: select only phrases that occurred in at least 20% of the documents in the cluster.
3. Calculate the descriptive score: the algorithm computes how descriptive a phrase is as label for a given cluster S, with parent cluster P with a formula that combines several features such as the normalized document frequency, TFIDF, phrase length, etc.
4. Calculate the cutt-off point: by default the algorithm displays five labels as the cluster descriptor, but if most of the label candidates have low descriptive scores more labels are displayed.

Some clustering algorithms attempt to find a set of labels first and then build (often overlapping) clusters around the labels, thereby avoiding the problem of labeling altogether (Osinski and Weiss, 2005; Zamir and Etzioni, 1999; Mika, 2005). For example, the Lingo algorithm (Osinski and Weiss, 2005) combines phrase discovery and latent semantic indexing techniques to separate search results into meaningful groups. The algorithm looks for frequent phrases, extracted from the document snippets to use as labels, and then assigns documents to the labels to form clusters.

Similarly, Zamir and Etzioni (1999) propose an algorithm for clustering similar documents that uses phrases extracted from the document snippets to identify similarities between documents and hierarchically construct

clusters. The work of Mika (2005) aims to filter search results based on automatically computed categories. Given a set of documents from a search result, the categorization algorithm selects the most common words and phrases and uses them as categories. These categories are displayed to the users as a list next to the actual results.

Discussion

Despite the importance of making the results of clustering useful, via cluster labeling, comparatively little work has been done on cluster labeling. In the literature, most of the works are related to IR problems and clusters are usually composed by documents, in contrast to our clusters, that are composed by lexical units. In any case, we considered the possibility of extending and/or adapting IR cluster labeling solutions to our cluster labeling scenario.

Most of the work on cluster labeling offers a list of terms as labels, which are often less useful than a single category label, because it requires the user to infer the concept implied by the terms. Usually, the list of terms offered by cluster labeling techniques are ranked by its confidence on how descriptive the label is, emphasizing that this list should be as short as possible. However, the advantage of a list of terms to describe a cluster is that a user can often infer the content of a cluster even when some of the selected terms are poor choices.

Despite that a list of labels to describe a cluster can be useful in some applications, it is certainly not useful for our purposes. Our clusters are populated with similar lexical units (not with documents) and the size of the clusters tend to be small, usually composed by 4 or 5 lexical units (see Section 6.2). Therefore, a good cluster descriptor would be a single concept that characterize all the lexical units in a cluster. Our purpose is to use cluster labels as labels on conceptual representations like block diagrams or concept map drawings, thus we must achieve abstraction over the particular terms inside a cluster. For instance, a cluster populated with terms such as MODIFY, CHANGE, ALTER, etc. may be well represented by the concept 'modify'.

Even that the work of Pantel and Ravichandran (2004) deals with clusters of similar nouns, we can not use or adapt this technique to group similar verbs because we can not retrieve label candidates from hyperonym pattern when working with verbs. Meanwhile, the probability that a noun and its

hyperonym co-occurrence (at a short distance) in the same sentence is high, it is very infrequent that a verb and its hyperonym co-occurr close to each other.

The mentioned algorithms that deal with finding a set of labels first and then builds clusters around the labels have been applied to IR problems. However, as pointed out by Manning et al. (2008), has no comprehensive study that compares the quality of such *label-based* clustering to the classical clustering algorithms is known. Regarding labeling cluster of similar verbs, to the extent of our knowledge, no work has been devoted to it yet.

4.4 Taking the state-of-the-art a step further

Over the last decade, RE has been intensively studied by the NLP community. To overcome scalability problems, semi-supervised learning methods and abstract patterns that rely on grammatical functions have been applied. More recently, standard evaluation datasets have been proposed to study binary relation extraction and classification of a predefined set of relations. In consequence, a large number of supervised learning methods and feature sets have been proposed.

Open or diverse relation extraction approaches based on abstract relation paths, as the ones in a syntactic dependency tree, are more promising because they rely on abstract labels to identify relations. For instance, they assume that every verb entails a relation. They also use structural information, as the shortest path between the verb and two noun phrases, to identify the arguments of the relation, instead of word order of the sentences.

The mayor problem of surface syntactic structures, as the one delivered by the state-of-the-art parsing tools, is that the order of the arguments of a predicate does not always correspond to the valency structure of them (see Chapter 3).

Furthermore, syntactic dependency analysis has not been fully exploited for relation extraction. For example, it can be used to extract *n-ary* relations without mayor efforts. In this research, we take this fact into account and do not restrict our application to the extraction of binary relations.

Regarding relation classification, unsupervised methods are preferred to supervised ones because unsupervised methods like clustering do not depend on manually created training data. Even tough, they usually depend on fea-

tures that are difficult to obtain and prone to errors like sub-categorization frames or characteristics about the arguments of the relations. Thus, it would be interesting to investigate whether simpler features can be used to group similar relations.

Moreover, unsupervised relation classification outputs group of similar relations instead of relation classes. Hence, if the goal is to propose a model for relation classification, it is necessary to label each group of similar relations with a concept that represents them.

Chapter 5

Methodology

The goal of capturing all verbal relations that contribute to the description of a complex functional object implies that our task cannot be restricted to a limited number of indicators of relations such as lexico syntactic patterns. Nor can we rely upon the co-occurrences of words that engage in a relation since the idiosyncratic style of patent material leads to a distribution of these words across a long distance. Cf., e.g., claim (1), where the arguments of *comprising* (in *An automatic focusing device comprising:* [...]) appear at a distance of 22, 56, 76, 167, and 203 tokens respectively from *comprising* itself.

Furthermore, the straightforward use of the verbal lexical units (LUs) as labels of content relations as done, e.g., in (Cascini and Russo, 2007) is not appropriate. For instance, the LUs *comprise*, *have*, *contain*, *consist of* and the like must be mapped onto the concept 'part of'; the LUs *cause*, *lead to*, *result in*, etc. are captured by the concept 'cause', and so on. Thus, we must thus abstract over concrete terms, mapping, e.g., (quasi-) synonyms onto one common concept.

For the sake of more uniform and simple verbal relations, the relation abstraction should go even further. Thus, we should also aim to find a common conceptual equivalent for more heterogeneous (but still sufficiently semantically similar) LUs such as, e.g., *adjust*, *arrange*, *place*, *position*, and *set*, *encode*, *encrypt*, *inscribe*, just to give a few examples. This semantic abstraction should be sought in order to reduce the number of labels, selecting labels of which each subsumes a number of sufficiently similar concrete la-

bels.

Taking the considerations presented above into account, we come up with the following generic two-stage procedure to retrieve content relations from patent claims:

1. Distillation of relation tuples via dependency parsing.
2. Generalization of the obtained relations in the previous stage via clustering and cluster labeling techniques.

In accordance with the two-stage procedure presented above, we propose the following methodology.

5.1 Distilling relation tuples via dependency parsing

In order to capture all verbal relations from patent claims we use syntactic dependency analysis. In a dependency structure a predicate and its arguments can be easily identified, as the structure represents, in fact, a projection of the (semantic) predicate-argument structure. The condition is that we do not use surface-oriented dependency structures (even though most of the state-of-the-art parsers deliver them), but instead abstract syntactic structures (DSyntSs) in the Meaning-Text Theory (Mel'čuk, 1988) described in Chapter 3.

To distil relation tuples via deep parsing we face two problems:

(1) Most of the state-of-the-art dependency parsers are not able to parse patent claims with sufficient quality given that patent claims sentences are very long and their structures are very complex. Therefore, in order to obtain deep syntactic structures, it is necessary to apply a preprocessing step consisting of:

(a) simplifying the original sentences with the aim to improve of-the-shelf parsing performance on patent claims;

(b) parsing the simplified sentences with an of-the-shelf dependency parser;

(c) mapping the parser output onto deep syntactic structures.

(2) The state-of-the-art dependency parser technologies cannot be used as such because they tend to provide surface-syntax like structures as output rather than deep-syntactic structures.

5.2 Relation generalization

Relation generalization is carried out via clustering techniques. For this task, two important questions arise: (i) how to asses the similarity of verbal relations? and (ii) how to label with a common concept groups of similar relations?

Assessing the similarity of verbal relations

To asses the similarity of verbal relations we use the lexical database WordNet (Fellbaum, 1998). WordNet (WN) follows the relational paradigm of the lexicon, which means that the meaning of a given lexeme is specified in terms of relations to other lexemes.[1] The most important relations are hyperonymy, hyponymy, synonymy, antonymy, holonymy and meronymy. For instance, in WN the lexemes *display1*$_V$ and *display1*$_N$ obtain the description of hyperonym and synonym relations as presented in Table 5.1, and *show1*$_V$ and *show1*$_N$, the description shown in Table 5.2.

Comparing the descriptions of both *display1*$_{V/N}$ and *show1*$_{V/N}$, we can thus deduce—at least with a certain probability—how similar or different the meanings of these two lexemes are. Hyperonymy chains (see Chapter 4) from WN have been used to determine the similarity between nouns. Verbal hyperonymy chains are much less extended in WN than nominal hyperonymy chains. This makes their use as semantic descriptors much less reliable; cf. also the findings of (Yang and Powers, 2006) already mentioned in Chapter 4. Therefore, we use synonymy instead of hyperonymy, representing each verbal relation label as a vector of its WN-synonyms.[2]

Consider, for illustration, the vectors of *construct1*, *create6*, *make24*, and *produce2*, which are intuitively semantically rather close:

[1] WordNet has been repeatedly used in previous works as a source of fine-grained semantic information—also in the context of semantic lexeme clustering; cf. Chapter 4.

[2] In WN, the synonyms of a lexeme are summarized in a *synset*.

Table 5.1: WN description of *display1*$_V$ and *display*$_N$.

display1$_V$		
	Definition:	To show, make visible or apparent
	Synonyms:	expose3, exhibit2
	Hyperonyms definition:	Make visible or noticeable
	Hyperonyms:	show4
display1$_N$		
	Definition:	Something done in order to communicate a particular impression
	Synonyms:	show2
	Hyperonyms definition:	A visual presentation showing how something works
	Hyperonyms:	demostration5, demo1

Table 5.2: WN description for *show1*$_V$ and *show1*$_N$.

show1$_V$		
	Definition:	Show or demonstrate something to an interested audience
	Synonyms:	demo1, exhibit2, present1, demonstrate1
	Hyperonyms definition:	Make visible or noticeable
	Hyperonyms:	show4
show1$_N$		
	Definition:	A public entertainment or exhibition
	Synonyms:	—
	Hyperonyms definition:	An activity that entertains
	Hyperonyms:	entertainment1, amusement2

construct1:	(construct1, build1, make17)
create6:	(create6, produce2, make6)
make24:	(make24, buil1, construct1)
produce2:	(produce2, make6, create6)

the vectors of *differentiate1*, *separate3*, *distinguish1*:

differentiate1:	(differentiate1, distinguish1, secern1, secernate1, separate3, severalize2, tell8, tell apart2, identify1, place1, recognize as being1)
separate3:	(differentiate1, distinguish1, secern1, secernate1, separate3, severalize2, tell8, tell apart2, identify1, place1, recognize as being1)
distinguish1:	(differentiate1, distinguish1, secern1, secernate1, separate3, severalize2, tell8, tell apart2, identify1, place1, recognize as being1)

and the vectors of *cut1*, *reduce1*, and *trim2*:

cut1:	(bring down1, cut2, cut back2, cut down1, cut down-on1, make areduction in1, reduce1, trim2, trim back1, trim down1, decrease2, lessen2, make smaller1, minify1)
reduce1:	(bring down1, cut2, cut back2, cut down1, cut down on1, make a reduction in1, reduce1, trim2, trim back1, trim down1, decrease2, lessen2, make smaller1, minify1)
trim2:	(trim2, bring down6, cut2, cut back-2, cut down1, cut down on1, make a reduction in1, trim back1, trim down1)

Vectors of semantically close lexemes can thus be assumed to overlap with a certain probability—although not always, as can be observed, e.g., for *construct1*: (construct1, build1, make17) and *create1*: (create1, produce2, make6), and for *differentiate1*: (differentiate1, distinguish1, separate3, secern1) and *distinguish3*: (distinguish3, mark3, differentiate2). However, the transitivity of semantic similarity will often allow us to establish a semantic link between them as well.[3]

Once the similarity of verbal relations is computed according to their WN synonym synsets, clustering algorithms are applied to group similar verbs.

[3]This is not to say that we will always succeed in grouping semantically similar relation labels calculating the similarity of their synset-vectors. For this, the word sense distinction in WN is too fine-grained and sometimes also too *ad hoc*.

For the clustering of verbal relations, we follow the standard clustering methodology, evaluating our test against gold standard solutions. During clustering experimentation, clustering algorithms are run as many iterations as it takes to reach a better result against the gold standard classes. During clustering iterations, several parameters need to be varied, since it is not clear which set-up results in the optimal clustering analysis. In our clustering experiments, we varied: the clustering paradigms (partitional, agglomerative and graph-partitioning-based clustering), different functions to optimize the solution and the number of desired verb classes.

Labeling groups of similar relations

As already stated, it is possible to apply cluster labeling techniques in order to label with a common concept groups of similar relations. As it is not clear which is the most appropriate cluster labeling technique for our purposes, it is necessary to test several labeling strategies.

A labeling approach that is solely based on the cluster content may have difficulties in providing discriminative and representative labels. Thus, it is necessary to also investigate the contribution of external resources for cluster labeling. This means that clusters are enriched with other terms retrieved from external resources such as a thesaurus.

Unfortunately, there is still no standard evaluation methodology for cluster labeling and there are no standard benchmarks to compare alternative labeling methods (Carmel et al., 2009). To test the effectiveness of our cluster labeling strategies, we plan to contrast a set of the automatically generated labels to gold standard labels. Moreover, the automatic and gold standard labels are judged by human evaluators.

Chapter 6

Approach

In this chapter, the methodology presented in Chapter 5 is applied to develop a verbal relation extraction and generalization system for specialized discourse and, more precisely, for patent claims in English. The aim of the system is to obtain empirical evidence that supports our basic assumptions, presented in Chapter 1. These assumptions state that in order to extract verbal relations and their arguments it is necessary to identify the valency structure of verbs and that to capture the valency structure of a predicate, a deep syntactic representation is needed.

As stated in Chapter 5, relations from patent claims can be obtained following a two-stage procedure consisting in:

(1) distillation of relation tuples via deep dependency parsing.

(2) generalization of the obtained relations in the previous stage via clustering and cluster labeling.

Taking into account this two-stage procedure, we designed a system architecture for relation extraction and their further generalization from patent claims, which is presented in Figure 6.1. Our system architecture is based on a pipeline model.[1] The tasks are modularized and the modules are run

[1] Our system architecture is considered a pipeline model in contrast to a parallel system architecture where all the modules operate simultaneously without waiting for the results of an earlier module.

on a patent claim one by one, in their entirety, with the accumulative results of the earlier modules serving as input to the latter modules.

The architecture is composed of two main modules, Relation Distillation and Relation Generalization. Each of these two modules is divided, in turn, into several sequential submodules. The aim of the Relation Distillation module is to provide deep syntactic structures via dependency parsing, from which relation tuples can be distilled. The goal of the Relation Generalization module is to generalize the obtained relations. In the following sections, each module of the architecture is described in detail.

6.1 Relation distillation

As indicated above, relation distillation is done using deep dependency parsing. In accordance with the discussion in Chapter 5, we divide the task of distilling relations via deep dependency parsing into two sub-stages:

1. Apply a preprocessing algorithm to reduce the complexity of patent claims.
2. Parse the simplified claims syntactically.

As illustrated in Figure 6.1, the sub-stages presented above are realized as two different sub-modules, Preprocessing and Syntactic parsing, respectively.

Preprocessing: Simplification of patent claims

The objective of the simplification, which is described in detail in (Bouayad-Agha et al., 2009a,b), is to improve parsing without any loss of relevant linguistic information. The simplification involves (cf. also Figure 6.2 for illustration):

1. Patent structuring: labeling of patent document structuring sections such as title, abstract, claims, etc., by XML labels.
2. Linguistic preprocessing: part-of-speech (POS) tagging and chunking analysis using Tree-Tagger (Schmid, 1994) with its off-the-shelf English parameter set.

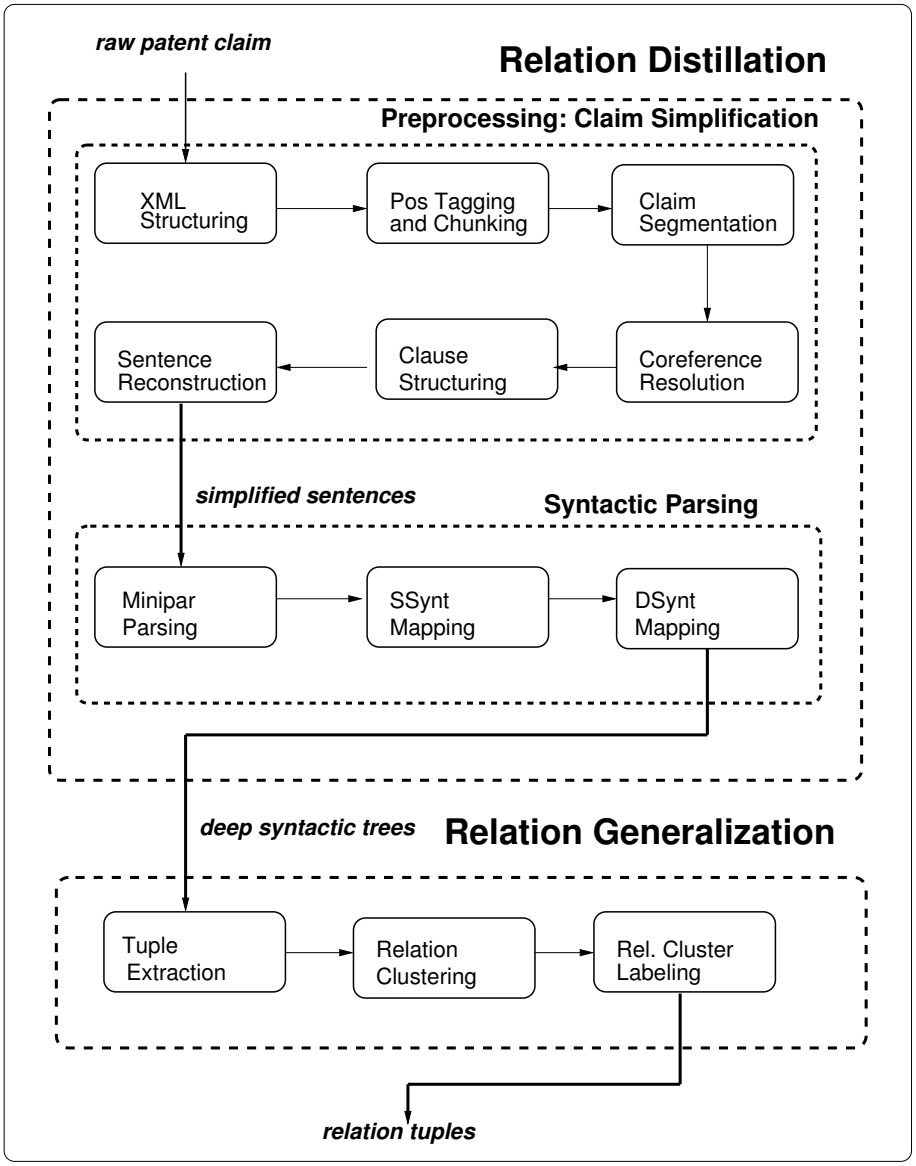

Figure 6.1: Relation extraction and generalization data flow.

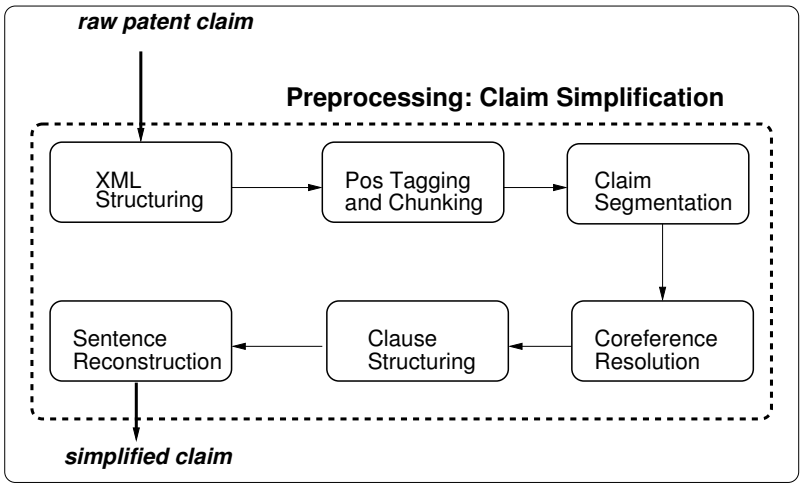

Figure 6.2: Sentence simplification data flow.

3. Claim Segmentation: segmentation of a claim sentence into minimal content units.

4. Coreference resolution: establishment of coreference links between NPs that denote the same object.

5. Claim tree derivation: building a claim tree drawing upon coreference links and other information such as tree structure and segment boundary keywords.

6. Sentence reconstruction: reconstruction of the individual segments in order to obtain grammatically correct independent sentences out of each of them.

Patent structuring

In order to process raw (unstructured) patent claim documents, first the patent document sections are identified, marking structural units such as the number of the patent, title, abstract, claims with their references to other claims, number of independent and dependent claims, number of independent claims with dependent and the maximum depth of the claim, in terms of XML labels.

Linguistic preprocessing

After patent structuring, the claims are processed with TreeTagger (Schmid, 1994) to obtain POS tagging and chunking information, which are standard analyses in corpus-oriented computational linguistics and do not require any further mention here.

Claim segmentation

Claim segmentation is carried out using a supervised machine-learning based segmentation algorithm. For this purpose, we manually built a segmentation gold standard composed of 5882 segments. As the size of the gold standard is small, the complete set is used for training a segmentation model.[2] In the training set, each token of a claim is represented in a feature vector with the segmentation feature set *true* if the token corresponds to a segment border, and *false* otherwise. Given a token, a feature vector is constructed using lexical, punctuation and POS-tagging information from the token itself and from three tokens to the right and three tokens to the left of the token. In Table 6.1, the features used to characterize segment borders are presented.

Feature	**Values**
Morpho-syntactic category	POS-Tagging tag set of TreeTagger
Type of constituent	Chunking tag set of TreeTagger (e.g., NC, CV, PC, etc.)
Keywords	*comprise*, *include*, *wherein*, *whereby*, *which*, *that*, *to*, *by*, *when*, *while*, *where*, *for*, *thereby*, *such*, *so*, *and*, *or*, *characterized*, *as*, *further*
Punctuation marks	comma, colon, semicolon
Segmentation	boolean

Table 6.1: Features used in the claim segmentation algorithm.

[2]The evaluation of the claim segmentation model is done using ten-fold cross validation. The evaluation of the segmentation algorithm is based on a segment alignment, see Subsection 7.1.

Figure 6.3: Segmentation result for claim (1).

1. An automatic focusing device comprising:
2. an objective lens
3. for focusing a light beam
4. emitted by a light source on a track of an information recording medium;
5. a beam splitter
6. for separating a reflected light beam
7. reflected by the information recording medium at a focal spot thereon
8. and through the objective lens from the light beam
9. emitted by the light source;
10. an astigmatic optical system
11. including an optical element
12. capable of causing the astigmatic aberration of the separated reflected light beam;
13. a light detector
14. having a light receiving surface
15. divided, except the central portion thereof, into a plurality of light receiving sections
16. which are arranged symmetrically with respect to a first axis
17. extending in parallel to the axial direction of the optical element
18. and to a second axis
19. extending perpendicularly to the first axis
20. and adapted to receive the reflected beam transmitted through the optical element
21. and to give a light reception output signal
22. corresponding to the shape of the spot of the reflected light beam
23. formed on the light receiving surface;
24. a focal position detecting circuit capable of giving an output signal
25. corresponding to the displacement of the objective lens from the focused position, on the basis of the output signal given by the light detector;
26. and a lens driving circuit
27. which drives the objective lens along the optical axis on the basis of the output signal given by the focal position detecting circuit.

With the training set at hand, we trained the Weka's J48 decision tree learner. For instance, when applied to the original claim (1), the segmentation algorithm divides the claim into 26 segments, as illustrated in Figure 6.3 (more details can be found in Ferraro (In Press)).

Coreference resolution

For coreference determination, a simple coreference resolution algorithm has been implemented that relies on the patent claims characteristic of repeating a nominal phrase (NP) instead of using a pronoun in order to indicate a coreference.[3] Roughly, we consider that two NPs corefer if they are identical noun phrases (without determiners). The pseudo code of the coreference resolution algorithm is presented in Figure 6.4.

Figure 6.4: Pseudo code of the claim coreference resolution algorithm.

```
Procedure ClaimCoreference
Given: C: Claims annotated with chunks
Returns: Claims annotated with coreferences
begin
References ← 0
NPList ← NPChunks(C)
for (chunk : NPList) do
  reference = createNewReference(chunk)
  references.ADD(reference)
end for
while baseNP : references do
  resolveReferences(baseNP)
end while
```

During the application of the coreference resolution algorithm, all the NPs from a claim are collected and coreferences are resolved. This means that for a whole patent, the coreference of each claim is resolved independently from the others.

For illustration, consider the coreference resolution results when the algorithm is applied to claim (1) in Figure 6.5. Each line in the table represents

[3]NP repetition (instead of, for instance, pronominalization) is one of the means used in patents to avoid ambiguity.

Figure 6.5: Coreference resolution result of claim (1).

corefid="coref1" start="6" end="8" tokens="an objective lens"
corefid="coref1" start="50" end="52" tokens="the objective lens"
corefid="coref1" start="194" end="196" tokens="the objective lens"
corefid="coref1" start="222" end="224" tokens="the objective lens"
corefid="coref4" start="88" end="91" tokens="a light receiving surface"
corefid="coref4" start="173" end="176" tokens="the light receiving surface"
corefid="coref5" start="182" end="182" tokens="circuit"
corefid="coref5" start="242" end="242" tokens="circuit"
corefid="coref6" start="23" end="26" tokens="an information recording medium"
corefid="coref6" start="39" end="42" tokens="the information recording medium"
corefid="coref9" start="84" end="86" tokens="a light detector"
corefid="coref9" start="211" end="213" tokens="the light detector"
corefid="coref11" start="33" end="36" tokens="a reflected light beam"
corefid="coref11" start="167" end="170" tokens="the reflected light beam"
corefid="coref15" start="148" end="150" tokens="the optical element"
corefid="coref15" start="68" end="70" tokens="an optical element"
corefid="coref15" start="125" end="127" tokens="the optical element"
corefid="coref17" start="186" end="188" tokens="an output signal"
corefid="coref17" start="206" end="208" tokens="the output signal"
corefid="coref17" start="233" end="235" tokens="the output signal"
corefid="coref19" start="203" end="204" tokens="the basis"
corefid="coref19" start="230" end="231" tokens="the basis"
corefid="coref23" start="16" end="18" tokens="a light source"
corefid="coref23" start="59" end="61" tokens="the light source"
corefid="coref25" start="11" end="13" tokens="a light beam"
corefid="coref25" start="54" end="56" tokens="the light beam"
corefid="coref25" start="81" end="82" tokens="light beam"
corefid="coref29" start="114" end="116" tokens="a first axis"
corefid="coref29" start="136" end="138" tokens="the first axis"

an NP that corefers with al least another NP; *corefid* stands for the coreference identifier, the *start* and *end* labels indicate the position of the NP in the claim, and *tokens* the literal NP. Note that when NPs corefer they have the same *corefid*.

Claim tree derivation

Given that each claim is a single sentence, the identification of the claim structure of this sentence relies on the identification of subordination and coordination relations between the minimal content units of the sentence. We assumed that the claim structure is a tree. To build a claim tree, the claim-structuring algorithm searches for the best claim structure in a space restricted by the application of two rules, according to a supervised machine learning algorithm.

The first rule is applied when the probability that two spans are related by a subordination relation is higher than a given confidence score. The second rule is applied when the probability that two spans are related by a coordination relation is higher than a given confidence score. The machine learning algorithm for the identification of subordination and coordination relations was trained using a claim structuring gold standard. The claim structuring gold standard was manually built by annotating subordination and coordination relations between the 5882 segments from the segmentation gold standard. The features used to characterize the relations are presented in Figure 6.6. The first column of the table stands for the feature description and the second column for their corresponding values.

During the claim structuring stage, a complete *n-ary* tree is constructed. For a given set of segments, the number of possible analyses can grow exponentially, thus rendering the problem of exploring all possible trees intractable when the number of segments is high enough. Due to the complexity of claim sentences, large segment sets are common; (see Figure 6.3). For this reason, a variation of a local beam search algorithm, namely Breadth-First Search (BFS), was used to search amongst the various possible clause trees, with the metrics calculated for each application of a rule serving as the objective function guiding the algorithm. The goal of this algorithm is to build a rooted tree. In order to obtain the optimal tree, the algorithm is allowed to backtrack once it has reached a goal and explore alternative trees, up to a fixed limit set to ensure computational efficiency.

The leaves of the tree are minimal content units and the nodes are intermediate groups of minimal content units or text spans. The pseudo-code of the BFS algorithm is given in Figure 6.7. BFS is a top-down algorithm; thus, the tree expansion begins at the root node. At the initial state, all the segments are put into a queue and the root node is the node to be expanded. A node can be expanded by applying an action. Each application of a rule is

Keyword Features	**Values**
First span token1, First span token2, First span token3, Second span token1, Second span token2, Second span token3	*comprise*, *include*, *wherein*, *whereby*, *which*, *that*, *to*, *by*, *when*, *while*, *where*, *for*, *thereby*, *such*, *so*, *and*, *or*, *characterize*, *characterise*, *as*, *further*, *in*, *OTHER*
POS-Tag Features	**Values**
First span token1, First span token2, Second span token1, Second span token2	CC, IN, J, N, V, W, DT, PRN, ADV, TO, OTHER, ','
Chunk Features	**Values**
Span1 chunk1 begin, Span1 chunk2 begin, Span1 chunk1 end, Span1 chunk2 end, Span2 chunk1 begin, Span2 chunk2 begin, Span2 chunk1 end, Span2 chunk2 end	NC, VC, O, PC, ADVC, ADJC, PRT, SBAR, LST, CONJC
Other Features	**Values**
Coreferences between spans	numeric
End comma first span	boolean
End comma second span	boolean
End semicolon first span	boolean
End semicolon second span	boolean
Colon first span	boolean
Final dot second span	boolean
Segments of first span	numeric
Segments of second span	numeric
Position of first span	beginning, middle, last
Position of second span	beginning, middle, last
Length first span	numeric
Length second span	numeric
Span relation	subordination, coordination

Figure 6.6: Features used in the clause structuring algorithm.

```
Procedure BFS:
enqueue segments
while nodesToExpand != empty do
  pruneTree(nodesToExpand)
  for nodesToExpand do
    newNode = expand(node)
    nodesToExpand.add(newNode)
  end for
end while
```

Figure 6.7: Pseudo code of the Breadth-First Search algorithm.

an action, with a confidence score of its application assigned by the machine learning algorithm. The actions are ordered according to their confidence score, and only the best actions are kept. A new state, thus a child node, is created for each of the best actions. All the child nodes obtained by expanding a node are added to the queue and the process is repeated until no new actions can be applied. After the first iteration, the algorithm prunes the tree and selects only the best nodes for expansion according to the confidence score of each rule. After exploring the best possible tree orders, the BFS algorithm returns several sequences of action application, thus several possible trees. The best tree is the one whose rules have the highest average confidence score.

For the set of segments from claim (1) shown in Figure 6.3, we show the derived claim tree in Figure 6.8.

Sentence reconstruction

Once the claim structuring is finished, the reconstruction of each segment takes place. This process consists of transforming each segment into a complete independent sentence. The sentence reconstruction is done using a set of five rules, which are the following ones:

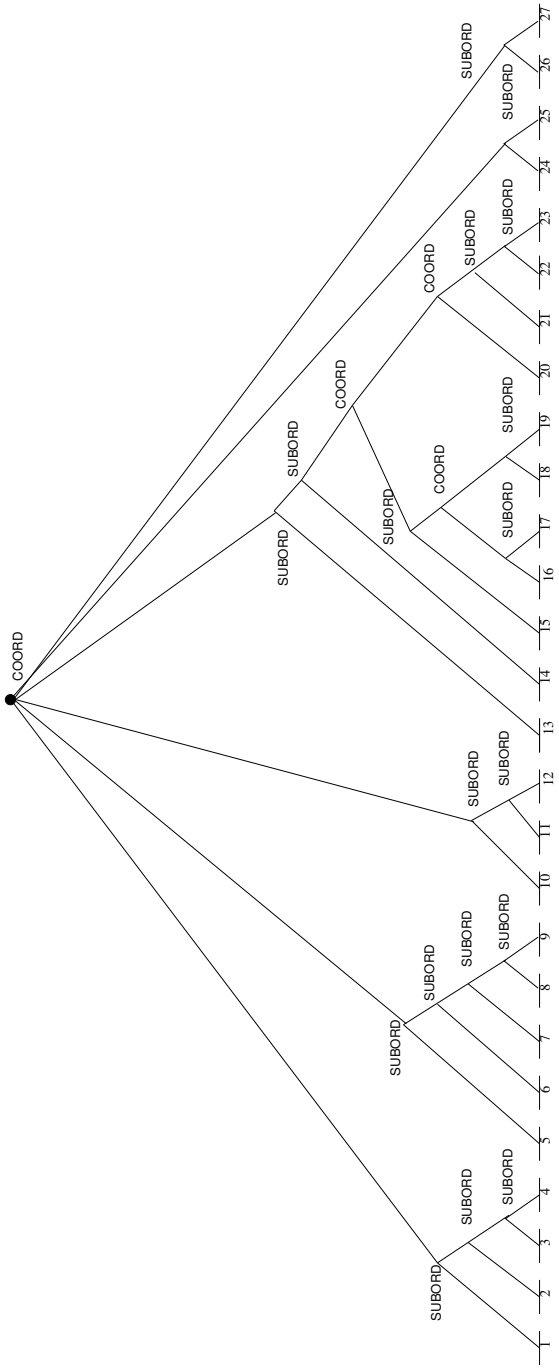

Figure 6.8: Claim structuring tree for claim (1).

Sentence Reconstruction Rule (1)

IF the segment starts with a subordinating conjunction OR
a coordinating conjunction OR an adverb
REMOVE that element
FI

Example (a):

wherein the second sealing surface comprises a selector valve body.
The second sealing surface comprises a selector valve body.

Example (b):

and the cross-sectional area of said neck is gradually reduced.
The cross-sectional area of said neck is gradually reduced.

Sentence Reconstruction Rule (2)

IF the segment starts with *by*
SET the segment to passive voice
FI

Example:

by applying a cutting edge.
A cutting edge is applied.

Sentence Reconstruction Rule (3)

IF the segment is a single NP
ADD '*there is*' at the beginning
FI

Example:

A diamond sintered body tool.
There is a diamond sintered body tool.

Sentence Reconstruction Rule (4)

IF the main verb is not in present tense
SET the main verb into present
FI

Example:

Said segments comprising a tapered diameter.
Said segments comprise a tapered diameter.

Sentence Reconstruction Rule (5)

IF a segment does not start with an NP
GET the first NP of the previous segment
ADD the NP as subject
FI

Example:

[. . .] and said bore defined by said segments comprising a tapered diameter.
Said segments comprising a tapered diameter.

When applied to the original claim (1), the sequence of the procedures of simplification provides the following result (2).

The most obvious differences between (1) and (2) are that in (2): (i) nearly each minimal content unit forms a sentence, and (ii) the arguments of the predicative lexemes are reordered to be closer to their heads.

(2) An automatic focusing device comprises: an objective lens; a beam splitter; an astigmatic optical system; a light detector; a focal position detecting circuit capable of giving an output signal and a lens driving circuit. The objective lens focusses light beam. The light source emits a light beam on a track of an information recording medium. The beam splitter separates the reflected light beam. The information recording medium reflects the reflected light beam at a focal spot thereon and through the objective lens from the light beam. The light source emits the light beam. The astigmatic optical system includes an optical element. The optical element is capable of causing the astigmatic aberration of the separated reflected light beam. The light detector has a light receiving surface. The light receiving surface is divided, except the central portion thereof, into a plurality of light receiving sections. The light receiving sections are arranged symmetrically with respect to a first axis and to a second axis. The first axis extends in parallel to the axial direction of the optical element. The second axis extends perpendicularly to the first axis. The second axis is adapted to receive the reflected beam. The reflected beam is transmitted through the optical element.

The second axis is adapted to give a light reception output signal corresponding to the shape of the spot of the reflected light beam. The reflected light beam is formed on the light receiving surface. The focal position detecting circuit capable of giving an output signal corresponding to the displacement of the objective lens from the focused position, on the basis of the output signal given by the light detector. The lens driving circuit drives the objective lens along the optical axis on the basis of the output signal given by the focal position detecting circuit.

Parsing simplified sentences

Once claim simplification is performed, syntactic parsing takes place. As illustrated in Figure 6.9, the syntactic parsing module comprises three sub-modules.

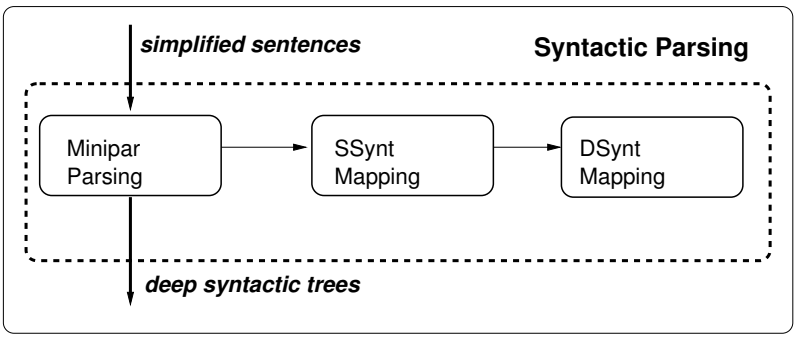

Figure 6.9: Syntactic Parsing module.

The simplified sentences are parsed using the off-the-shelf dependency parser (Klein and Manning, 2003a) by the Minipar Parsing sub-module. We chose Minipar because it produces syntactic structures that can be mapped onto the Deep-Syntactic Structures (DSyntSs) of the MTT model, which serves us a basis for the relation extraction, (see Chapter 3) and because it is robust and accurate enough for our task. Consider a sample structure obtained as output from MiniPar on the left side of Figure 6.10.

On the basis of the information provided by MiniPar, we obtain DSyntSs in two steps (Mille and Wanner, 2008a). First, the MiniPar structures are mapped onto MTT-Surface-Syntactic Structures (SSyntSs). Second, the SSyntSs are mapped onto DSyntSs. Despite their similarity, Minipar output structures and SSynts show some crucial differences, which are due

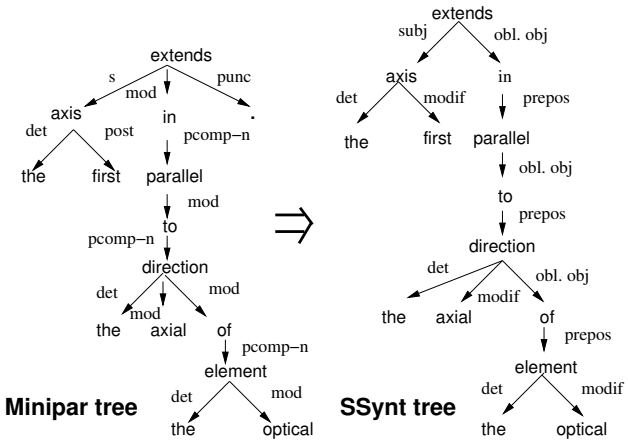

Figure 6.10: Example of a MiniPar syntactic dependency structure (left) and its SSyntS correspondence (right).

to the theoretical divergences of MTT and the linguistic model underlying Minipar. This makes the Minipar-to-SSyntS projection less than straightforward. The current version of the Minipar-SSyntS grammar contains 137 rules (Mille and Wanner, 2008b). Its evaluation of 1324 sentences has shown that 99% of well-formed Minipar-structures are correctly mapped onto SSyntSs.

Consider the structures of Figure 6.10 for an illustration of the Minipar-SSyntS mapping. During the subsequent SSynt-DSynt transition stage, the following main actions are performed:

1. Verbal tense auxiliary forms are mapped onto attribute-value pairs;
2. Determiners are removed using the same strategy, i.e., they appear in DSynt as attribute-value pairs "definiteness = DEF / INDEF" associated with the governing noun;
3. Governed prepositions, such as, e.g., the preposition *by* when it introduces the agent in a passive construction, are removed from the structure;

Cf. Figure 6.11 for the mapping of the SSyntS, which corresponds to the MiniPar structure presented above, to the DSyntS.

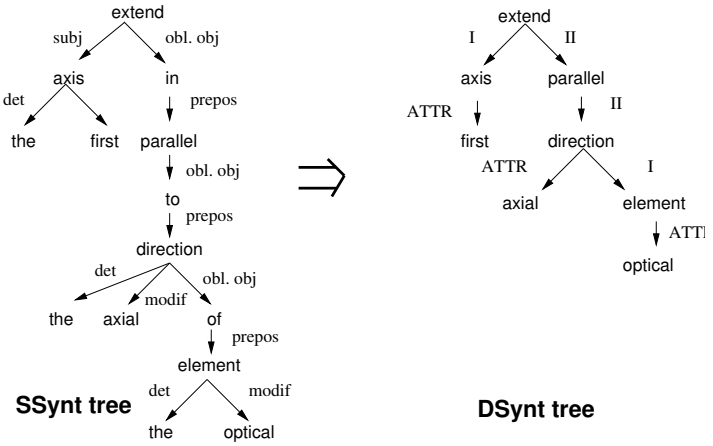

Figure 6.11: Example of a mapping from a SSyntS to a DSyntS.

6.2 Relation generalization

Following the system architecture presented in Figure 6.12, the Relation Generalization module comprises three sub-modules: Tuple Extraction, Relation Clustering and Relation Cluster labeling.

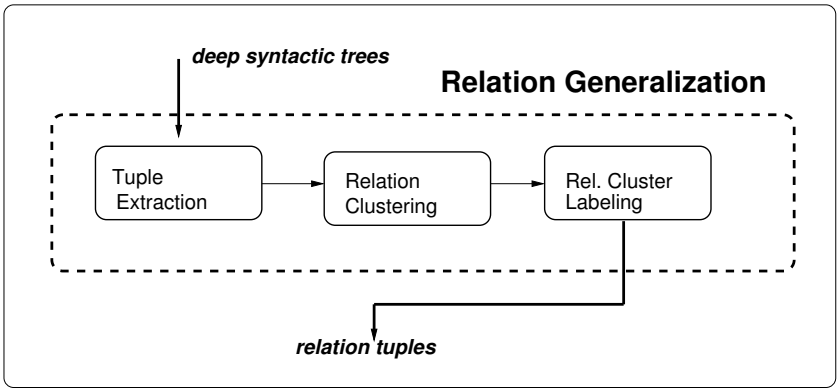

Figure 6.12: Relation Generalization module.

The tasks of each sub-module can be summarized as follows:

- Tuple Extraction: it consists in obtaining relation tuples;
- Relation Clustering: it consists in clustering the obtained relations;

- Relation Clusters Labeling: it consists in labeling the relation clusters.

Obtaining relation tuples

Once the DSyntSs are obtained, we first extract the dependency relations between the individual lexemes, and convert them into content relation tuples. During the conversion of these dependency relations, the following actions are performed:

- the argument tags are eliminated,
- overlapping pairs of relations are aggregated into one relation.

Thus, for the DSyntS in Figure 6.11, we obtain the dependency relations presented on the left of Table 6.2 and their corresponding tuples after conversion (on the right of Table 6.2.)

Table 6.2: Example of relation tuples conversion.

Dependecy relations	**Relation tuples**
extend -I→ axis	axis <extend>
extend -II→ parallel	<extend> parallel
parallel -I→ axis	axis <parallel>
parallel -II→ direction	<parallel> direction
direction -I→ element	element <direction>

Then we apply simple aggregation rules whose goal is to merge overlapping tuples when possible. As explained below, two types of tuples can be merged, actantial and attributive ones.

Actantial tuples merging rule:

IF (I(V_i, A_1) & II(V_i, A_2))
SET (V_i(A_1, A_2))
FI

Actantial tuples of the same transitive verb are aggregated such that the verb becomes the relation label; for example, given the tuples: emit<I>source and emit<II>laser beam, we obtain a single tuple, source<emit>laser beam;

Attributive tuples merging rule:
IF (ATTR(A_1, A_2) & II(A_2, A_3))
SET (A_2(A_1, A_3))
FI

Attributive tuples of a prepositional relation are merged such that the preposition appears as the relation label; for example, given the tuples apparatus<ATTR>for and for<II>use, we obtain the single tuple apparatus<for>use.

Relation clustering

Once the relation tuples have been distilled, their names are clustered in accordance with their semantic similarity, applying the approach presented in Chapter 5. This is done in order to generalize the relations as much as possible (in order to find the most adequate labels in the next stage). The clustering procedure consists of two steps. In the first step, the similarity between the collected relation names is calculated. In the second step, those names are clustered according to their similarity.

In a generic setting, we must know in which sense a word is used in order to be able to assess its similarity to another word. Thus, to compare [*to*] *separate* with [*to*] *divide* as they appear in patent claims, we must know that it is the *separate* meaning 'to bring physically apart' rather than 'to part company', which is of relevance. In principle, we would need to apply word sense disambiguation, which is a rather complex task. However, extending the *one-sense-per-discourse* assumption (Gale et al., 1992) to the *one-sense-per-patent-domain*, assumption we may hypothesize that we can largely dispense with word sense disambiguation. That is, we can assume that in a specific patent domain, a given word is mainly used in the same sense. For instance, [*to*] *separate* in patents of the domain of Machine Tools will mean, in the majority of cases, 'action of bringing parts physically apart', and [*to*] *record* in Optical Recording Devices patents will mean 'action of registering information on a digital medium'. Therefore, if there is a sense of the verb V that is similar to a sense of the verb W, we will detect this similarity by measuring the similarity of all senses of V with all senses of W (see Chapter 7 for a verification of this assumption). For

this purpose, we construct a synonymy vector for each sense v_i of V and for each sense w_j of W captured in WordNet (WN): $v_i = (v_{i_1}, v_{i_2}, \ldots, v_{i_n})$, with n as the number of synonyms for v_i and $w_j = (w_{j_1}, v_{j_2}, \ldots, w_{j_m})$, with m as the number of synonyms for w_j. The similarity between V and W is then assumed to be the maximum similarity between any sense v_i of V and any sense w_j of W, calculated as the maximum vector cosine: $Sim(V,W) = max \cos(\vec{v_i}, \vec{w_j})$, with $i = 1, \ldots, k$ (k as the number of senses of V) and $j = 1, \ldots, p$ (p as the number of senses of W). To obtain the cosine between the sense v_i and the sense w_j, we use the standard cosine formula;[4] cf. equation 6.1:

$$\cos(\vec{v_i}, \vec{w_j}) = \frac{\vec{v_i} \bullet \vec{w_j}}{|\vec{v_i}| \bullet |\vec{w_j}|} = \frac{\sum_{s=1}^{N} v_{i,s} w_{j,s}}{\sqrt{\sum_{s=1}^{N} v_{i,s}^2} \sqrt{\sum_{s=1}^{N} w_{j,s}^2}} \tag{6.1}$$

(with N as the normalized length of the synonymy vectors).

The obtained similarities between the relation names are most conveniently represented in terms of a similarity matrix, as shown in Table 6.3. Each row i/column j of this matrix represents a relation label expressed by a verb, and the value in the cell *(i, j)* is the similarity between label i and label j. With this similarity matrix at hand, we can cluster the relations based on semantic similarity grounds. For this purpose, we apply the Cluto clustering algorithms (Karypis, 2003).[5]

We experimented with three different clustering algorithms, each of them implementing a different clustering paradigm: (i) partitional clustering, (ii) agglomerative clustering, and (iii) graph-partitioning-based clustering, combining different criterion functions to optimize the solution and testing a varying number of clusters.

A variant of graph-partitioning-based clustering, namely the *optimized repeated bisections* (ORB) algorithm (Zhao and Karypis, 2002), performed best. The ORB algorithm divides a given collection of entities into k clusters by performing a sequence of $k-1$ bisections. First, the whole collection

[4] Cosine has been chosen because it is the most common measure for calculating similarity of decomposed semantic descriptions of lexical units: it is rather simple and reliable.

[5] As most high-dimensional datasets, the resulting matrix is a sparsely populated matrix. Cluto takes into account this sparsity and requires memory that is roughly linear in relation to the input size.

Table 6.3: Similarity matrix.

	encode	encrypt	inscribe	format	record	file	⋯	write
encode	0.0	0.41	0.29	0.0	0.0	0.0	⋯	0.0
encrypt	0.41	0.0	0.03	0.0	0.0	0.0	⋯	0.09
inscribe	0.29	0.03	0.0	0.0	0.07	0.0	⋯	0.15
format	0.0	0.0	0.0	0.0	0.02	0.0	⋯	0.02
record	0.0	0.0	0.07	0.02	0.0	0.72	⋯	0.36
file	0.0	0.0	0.0	0.0	0.72	0.0	⋯	0.0
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋱	⋮
write	0.0	0.09	0.15	0.02	0.36	0.0	⋯	0.0

is divided into two clusters; then, one of these clusters is selected and bisected, and so on, until the desired number of clusters (= k) is reached. Each bisection and the overall solution are optimized using the $\mathcal{H}_1$ criterion function:

$$\mathcal{H}_1 = maximize\,\frac{\mathcal{I}_1}{\mathcal{E}_1} = \frac{\sum_{r=1}^{k} ||D_r||^2/n_r}{\sum_{r=1}^{k} n_r D_r^t D/||D||} \tag{6.2}$$

where D_r is the composite vector of the cluster number r, n_r the size of r, k the number of clusters, D_r^t is the composition of the cluster r, and D is the composite vector of the entire relation label collection.

As shown in equation 6.2, $\mathcal{H}_1$ is an hybrid criterion function, which combines the internal criterion function $\mathcal{I}_1$ and the external criterion function $\mathcal{E}_1$: while $\mathcal{I}_1$ produces a clustering solution that optimizes a function defined over the elements that are part of a cluster (without taking into account the elements assigned to different clusters), $\mathcal{E}_1$ optimizes a function that takes into account the difference between the different clusters, trying to separate the elements of each cluster from the entire collection.

We experimented with different values for k, with the goal of finding a balance between cluster purity and the number of singleton clusters. Purity is reciprocally proportional to the number of singleton clusters. In particular, purity is 1 if each element is assigned to its own cluster - which, obviously, is not our goal. Figure 6.13 shows the results of the trials with k from 35 to 80. According to these trials, the best k is around 60. This means that, given 321 verbal relation labels in our experimental collection, each cluster

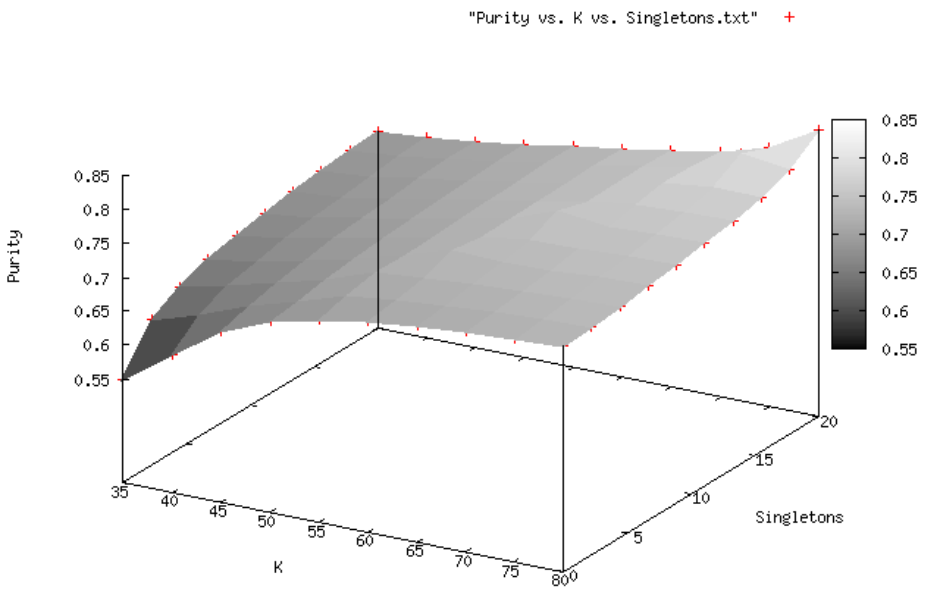

Figure 6.13: Clustering experiments with different ks, purity and number of singleton clusters.

will ideally contain about five labels. We think that small clusters reflect the structure of the data in a conservative approach to our task because, in order to keep the expressiveness of the relation labels, we do not want them to be too abstract (which would be the consequence of large clusters). Some sample clusters are given in Table 6.4.

The most homogeneous cluster among the clusters in Table 6.4 is {displace, impress, move, travel}. Only one member of this cluster is not well grouped: impress. The reason is that in WN one of the senses of move stands for "emotional or cognitive impact". The complete result of our clustering approach is presented in Annex A.

Relational cluster labeling

In the previous Section, we described how to distill verbal relation tuples from a patent claim corpus and how to cluster them in accordance with

Table 6.4: Examples of obtained clusters.

1.	{construct, create, effect, make, produce}
2.	{curve, cut, reduce, trim}
3.	{decrease, descend, fall}
4.	{display, exhibit, expose}
5.	{displace, impress, move, travel}
6.	{form, shape}
7.	{beam, radiate, send, transmit}
8.	{comprise, contain, block, stop}
9.	{become, come, release, turn}
10.	{associate, connect, join, link, relate}

their semantic similarity, with the aim of unifying similar relations. In order to use the generalized relations as relation labels in a conceptual-map like representation, we identified a suitable name for each cluster of relations via cluster labeling techniques.

As already mentioned in Chapter 4 , two main strategies of cluster labeling can be identified in the literature: (i) internal cluster labeling and (ii) differential cluster labeling. In the first, the label for a given cluster is chosen drawing solely on the cluster itself. In the second, such a label is chosen by contrasting this cluster with the other available clusters.

We have selected a total of seven cluster labeling strategies, which are applied to the clustering solution mentioned above. Three strategies correspond to internal cluster labeling techniques and the remaining four to differential cluster labeling. Some of our strategies, no matter whether internal or differential, propose as labels lexemes retrieved from external lexical-knowledge resources such as WordNet (WN) and the Open-Office Thesaurus, so they may be considered also as indirect labeling. In what follows, each strategy is described in detail.

Internal cluster labeling strategies

As already mentioned, cluster-internal labeling selects labels that only depend on the contents of the cluster of interest. No comparison is made with the other clusters. We explored three different strategies of internal cluster labeling, which are, (i) Frequency-oriented labeling; (ii) Verb hyperonym-oriented labeling, and (iii) Noun hyperonym-oriented labeling.

Frequency-oriented labeling (Freq)

The Frequency-oriented labeling strategy chooses as cluster label the member of the cluster with the highest frequency in the corpus. Thus, for the cluster:

C_i = {bound:63, limit:74, restrain:21, inhibit:101, fasten:49, fix:53, secure:13, lock:28}

inhibit would be suggested as cluster label by this strategy.[6] This strategy is motivated by classical cluster labeling techniques that choose one of the members of the cluster as its label. It has the advantage of being simple. However, intuitively, in the context of finding a common label for semantically similar but still different lexemes, an abstract label that represents all the members of the cluster seems more suitable.

Verb hyperonym-oriented labeling (VHyper)

The Verb hyperonym-oriented labeling strategy chooses as cluster label the most frequent hyperonym of the cluster according to the WN verb hierarchy. Only hyperonyms of verb senses with non-zero similarity (Section 6.2) are taken into account. First, for each member of the cluster, all its WN hyperonyms are retrieved and the most frequent hyperonym set is selected. From this set, the most frequent lexeme is chosen as the cluster label. Cf. for illustration Table 6.5. The first column shows the verb senses with non-zero similarity of a cluster; the second column shows the hyperonym synset corresponding to each member of the cluster. The most frequent hyperonym set in our example is:

> bound3, check4, confine1, limit1, restrain2, restrict3, throttle1, trammel2, decide1, decide upon1, determine4, make-up one's-mind1

[6] The suffix ':X' denotes the frequency of the corresponding member in our patent corpus.

Table 6.5: Hyperonyms of the different senses of the members of a sample cluster used in the VHyper cluster labeling strategy.

Cluster members	WN Hyperonyms of the cluster members
bound1	bound1, jump1, leap1, spring1, change position1, move3
bound2	border2, bound2, form the border of1, be made up of1, enclose2
bound3	bounce1, bound3, rebound1, recoil2, ricochet1, spring3, ...
bound4	bound4, check4, confine1, limit1, restrain2, restrict3, throttle1, ...
limit1	bound3, check4, confine1, limit1, restrain2, restrict3, throttle1, ...
limit2	limit2, put restrictions on1, restrict2, ...
limit3	circumscribe2, confine, limit3
restrain1	hold back2, keep14, keep back1, restrain1, prevent2
restrain2	confine1, hold3, restrain2
restrain3	constrain1, cumber1, encumber1, restrain3, ...
restrain4	bound4, check4, confine1, limit1, restrain4, restrict3, throttle1, ...
restrain5	check4, contain3, control2, curb1, hold in2, moderate3, restrain5, ...
restrain6	intimidate2, restrain6, discourage2
inhibit1	conquer1, curb2, inhibit1, stamp down1, subdue2, suppress1, ...
inhibit2	inhibit2, restrict3, restrain2, trammel2, limit1, bound3, confine1, throttle1
fasten1	fasten1, fix2, fix firmly1, secure2, ...
fasten2	become fast1, become fixed1, fasten2, ...
fasten3	fasten3, connect4, connect together1, link1, put together1, tie3, ...
fasten4	fasten4, tighten1, alter1, change1
fix1	bushel1, doctor3, fix1, furbish up1, amend2, repair1, restore4, ...
fix2	cook2, fix2, make38, prepare2, ready1, create1, make2
fix3	decide upon1, determine4, fix3, set3, specify2, decide1, ...
fix4	fasten1, fix4, fix firmly1, secure2, connect3, ...
fix5	fix5, get8, pay back2, pay off6, penalize2, punish4
fix6	fix,6 prepare for microscopic study1, set up15, set7
fix7	deposit3, fix8, pose8, posit1, situate2, displace2, ...
fix8	fix9, fixate1, make fixed1, stable or stationary1, ...
fix9	desex1, desexualize2, fix10, make infertile1, sterilize2, ...
secure1	obtain1, procure1, secure1, acquire1, get hold of1 ...
secure2	fasten1, fix2, fix firmly1, secure2, connect3, ...

Continued on next page

Table 6.5 – *continued from previous page*

Cluster members	WN Hyperonyms of the cluster members
secure3	secure3, guarantee1, vouch2
secure4	plug1, secure4, stop up1, close14, fill10, fill up3
secure5	batten1, batten down1, secure6, fortify1, strengthen1
lock1	fasten with a lock1, lock1, fasten1, fix2, secure2, ...
lock2	engage10, lock2, mesh1, operate6, displace4, make move1, move2
lock3	become immoveable1, become rigid1, lock3, engage6
lock4	interlace2, interlock2, lock4, hold2, take hold2
lock5	lock5, embrace2, hug1, bosom2, squeeze6
lock6	lock6, lock away1, lock in2, lock up7, put away1, shut away2, ...

The most frequent lexeme in our corpus from this hyperonym set is *limit*. *Limit* is thus chosen as cluster label. This strategy is motivated by the fact that the cluster label should be more abstract to ensure that all members of the cluster are reflected by it.

Noun hyperonym-oriented labeling (NHyper)

The Noun hyperonym-oriented labeling strategy chooses as cluster label the most frequent hyperonym among the members of a cluster, drawing upon the WN noun hierarchy. This is possible because some verbs are derived from nouns or vice versa such that the WN noun hierarchy can be used to explore the relations between verbs (Yang and Powers, 2005). The motivation behind this strategy is that, while for verbs in WN only relatively flat hyperonym hierarchies are available, nominal hyperonym hierarchies tend to be richer and deeper. However, after experimenting with this strategy, we decided to discard it because it requires substantial morphological preprocessing (at least stemming and morphological derivation analysis) to achieve good quality.

Thesaurus-Frequency-oriented labeling (ThesFreq)

The Thesaurus-Frequency-oriented labeling strategy chooses as cluster label the most frequent lexeme found in a cluster populated by lexemes from

the Open-Office Thesaurus. The cluster population by lexemes from the thesaurus is realized as follows. Given a cluster, for each of its members, the verbal lexemes associated with them are retrieved from the thesaurus and assigned to the cluster. For an illustration of the cluster population with lexemes from the thesaurus, see Table 6.6. The first column shows the cluster members; the second column shows the lexemes corresponding to each member of the cluster in the thesaurus.

Table 6.6: Cluster members and its associated lexemes in the thesaurus.

Cluster members	Thesaurus matches
bound	jump, leap, spring, move, border, enclose, hold in, confine, restrict, restrain, trammel, limit, throttle, control, hold in, contain, check, curb, moderate, bounce, resile, take a hop, spring, rebound, recoil, reverberate, ricochet, jump, leap, spring, adhere, hold fast, bond, bind, stick, stick to, set, attach, tie, bond, relate, bandage, fasten, **fix**, secure, tie down, tie up, truss, hold, oblige, obligate, relate, adhere, hold fast, cover, tie, constipate, indispose
limit	restrict, restrain, trammel, bound, confine, throttle, control, hold in, hold, contain, check, curb, moderate, circumscribe, decrease, lessen, minify, specify, set, determine, **fix**, choose, take, select, pick out, suppress, keep, keep back, hold back, prevent, limit, check, moderate, disable, disenable, incapacitate, encumber, cumber, constrain, intimidate, discourage
inhibit	suppress, stamp down, subdue, conquer, curb, control, hold in, hold, contain, check, moderate, restrict, restrain, trammel, limit, bound, confine, throttle
fasten	**fix**, secure, attach, tighten, change, alter, modify
fix	fasten, secure, attach, specify, set, determine, limit, choose, take, select, pick out, cook, ready, make, prepare, create from raw material, create from raw stuff, pay back, pay off, get, get even, get back, establish, found, plant, constitute, institute, prepare, set up, ready, gear up, fixate, sterilize, sterilise, desex, unsex, desexualize, operate on, operate, situate, posit, deposit, put, place, pose, position, lay, prepare, ready, gear up, change, alter, modify

Continued on next page

Table 6.6 – *continued from previous page*

Cluster members	Thesaurus's matches
secure	procure, obtain, fasten, **fix**, attach, guarantee, vouch, ensure, insure, assure, plug, stop up, close, fill up, batten, batten down, strengthen, beef up, fortify
lock	fasten, **fix**, secure, lock up, lock up, engage, mesh, operate, move, displace, engage, interlock, interlace, hold, take hold, interlock, embrace, hug, bosom, squeeze, overwhelm, overpower, sweep over, whelm, overcome, overtake, lock in, lock away, put away, shut up, shut away, lock up, confine, pass, go through, go across, construct, build, make

The most frequent lexeme in the cluster populated by synonyms from the thesaurus is *fix* (highlighted in boldface in Table 6.6). *Fix* is thus chosen as the label. This strategy is motivated by the fact that thesaurus group lexemes according to similarity of meaning or synonymy. Our intuition is that by enriching clusters with synonyms retrieved from the thesaurus, the possibility of finding a common abstract label increases. This intuition is based on two observations: (i) we can look for the most frequent common thesaurus synonym among the clusters, avoiding the restriction of assigning as a label a lexeme from the cluster itself, (ii) we can further measures such as Mutual Information, which are best suited for clusters that contain overlapping terms.

Differential cluster labeling strategies

Differential cluster labeling strategies label a cluster by comparing the members in one cluster with the members occurring in other clusters. The techniques used for feature selection in NLP, such as Mutual Information (MI) and χ^2 feature selection, can also be applied for purposes of differential cluster labeling (Manning et al., 2008). We have implemented four differential labeling strategies, using MI and χ^2 as measures. In order to facilitate comprehension, we introduce the definitions of MI and χ^2 measures.

Mutual Information (MI)

The MI of two random variables expresses their mutual dependence or the amount of information they have in common.[7] In other words, it measures how much knowing one of these variables reduces the uncertainty about the other. For instance, for the variables X and Y, MI is defined as:

$$I(X,Y) = \sum_{x \epsilon X} \sum_{y \epsilon Y} p(x,y)\, log2\, \frac{p(x,y)}{p_1(x)p_2(y)} \tag{6.3}$$

where $p(x,y)$ is the joint probability distribution of the two variables, $p_1(x)$ is the probability distribution of X, and $p_2(y)$ is the probability distribution of Y. The variable X corresponds to the notion of membership in a cluster, and the variable Y corresponds to the notion of the presence of a term (verbal relations in our approach). As both variables can have values 0 or 1, the equation can be rewritten as follows:

$$I(C,T) = \sum_{c \epsilon 0,1} \sum_{t \epsilon 0,1} p(C=c, T=t)\, log2\, \frac{p(C=c, T=t)}{p(C=c)p(T=t)} \tag{6.4}$$

In this case, $p(C = 1)$ represents the probability that a randomly-selected verb is a member of a particular cluster, and $p(C = 0)$ represents the probability that it is not. Similarly, $p(T = 1)$ represents the probability that a randomly-selected verb contains a given term, and $p(T = 0)$ represents the probability that it does not. The joint probability distribution function $p(C, T)$ represents the probability that two events occur simultaneously. For example, $p(0, 0)$ is the probability that a verb is not a member of cluster c and does not contain term t; and so on.

A cluster label can be created by calculating the MI of each term in the cluster and selecting the term with the highest MI value.

Chi-Square test

The Pearson χ^2 test can be used to calculate the probability that the occurrence of an event matches the initial expectations.[8] In particular, it

[7] The application of MI in cluster labeling is based on (Manning et al., 2008, 363).

[8] The application of χ^2 selection in cluster labeling is based on Manning et al. (2008).

can be used to determine whether two events, A and B, are statistically independent. The χ^2 is defined as follows:

$$\chi^2 = \sum_{a \epsilon A} \sum_{b \epsilon B} \frac{(O_{a,b} - E_{a,b})^2}{E_{a,b}} \tag{6.5}$$

where $O_{a,b}$ is the observed frequency of a and b co-occurring, and $E_{a,b}$ is the expected frequency of co-occurrence. In the case of cluster labeling, the variable A is associated with the membership in a cluster, and the variable B is associated with the presence of a term. Both variables can have values of 0 or 1, so the equation can be rewritten as follows:

$$\chi^2 = \sum_{a \epsilon 0,1} \sum_{b \epsilon 0,1} \frac{(O_{a,b} - E_{a,b})^2}{E_{a,b}} \tag{6.6}$$

$O_{0,1}$ is the observed number of documents that are in a particular cluster but do not contain a certain term, and $E_{1,0}$ is the expected number of documents that are in a particular cluster but do not contain a certain term. Our initial assumption is that the two events are independent, so the expected probabilities of co-occurrence can be calculated by multiplying individual probabilities:

$$E(1,0) = N(P(C = 1) * P(T = 0) \tag{6.7}$$

where N is the total number of verbs in the collection.

A cluster label can be created by calculating the χ^2 of each term in the cluster and selecting the term with the highest χ^2 value.

VHyper-MI-oriented labeling (VHyper-MI)

The VHyper-MI-oriented labeling strategy chooses the hyperonym with the highest MI value as the cluster label. First, each cluster member is populated with its WN hyperonyms as in the VHyper-oriented labeling strategy. Then, for a given cluster, we calculate the MI of each hyperonym in the cluster and select as label the hyperonym with the highest MI value.

The following example shows the MI between the cluster C_i (see Table 6.5) and the term *throttle*. First, the individual probabilities and the joint probabilities are calculated (where N = Total number of verbs in the collection).

Individual probabilities
P(C = 1) = # of documents in the cluster / N = 5/184
P(C = 0) = # of documents not in the cluster / N = 179/184
P(T = 1) = # of documents containing the term / N = 6/184
P(T = 0) = # of documents not containing the term / N = 178/184
Joint probabilities
P(C = 1, T = 1) = 4 / 184
P(C = 1, T = 0) = 1 / 184
P(C = 0, T = 1) = 2 / 184
P(C = 0, T = 0) = 177 / 184

Then, these probabilities are introduced into equation6.4 to obtain the MI value. The MI between the cluster C_i and the term *throttle* is 1394.56. We create a label for the cluster C_i by calculating the MI of each term in the cluster and select the term with the highest MI value. See, for example, Table 6.7, where a sample of the results is given. The term with the highest MI value turns out to be *moderate*. Therefore, it is chosen as the cluster label.

Label candidates	MI value
moderate	1391.80
restrict	1394.56
throttle	1394.56
restrain	1389.33
put restrictions on	1388.25
check	1387.05

Table 6.7: Example of the VHyper-MI results.

VHyper χ^2-oriented labeling (VHyperχ^2)

The VHyper-χ^2-oriented labeling strategy chooses the hyperonym with the highest χ^2 value as the cluster label. The clusters are populated by the WN hyperonyms of the VHyper-MI strategy. Using the same data as in the MI example, we calculated the expected probabilities and the observed frequencies and introduced them into the equation6.6 to calculate the χ^2 test. Consider the following example that calculates the χ^2 between the cluster C_i and the term *throttle*.

Expected probabilities
E (0,0) = 173.16
E (0,1) = 5.83
E (1,0) = 4.83
E (1,1) = 0.16
Observed frequencies
O (0,0) = # verbs not in C_i that do not contain *throttle* = 173
O (0,1) = # verbs containing *throttle* = 6
O (1,0) = # verbs in C_i = 5
O (1,1) = # verbs in C_i that contain *throttle* = 4

The χ^2 between the cluster C_i and the term *throttle* is 90.30. We created a label for the cluster C_i by calculating the χ^2 of each term in the cluster and selected the term with the highest χ^2 value. See, for example, Table 6.8, where a sample of the results is given. For the C_i cluster the term with the highest χ^2 value turns out to be *throttle*.

Label candidate	**χ^2 value**
throttle	90.30
confine	82.29
hold-in	76.31
restrain	65.83
check	57.68
fasten with a lock	34.82

Table 6.8: Example of the VHyper χ^2 results.

Thesaurus-MI-oriented labeling (ThesMI)

The Thesaurus-MI-oriented labeling strategy chooses the thesaurus lexeme with the highest MI value as the cluster label. The clusters are populated with their thesaurus matches as in the ThesFreq strategy. An example of the results is shown in Table 6.9. For the C_i cluster, the term with the highest MI value is *restrict*.

Thesaurus-χ^2-oriented labeling (Thesχ^2)

The Thesaurus-χ^2-oriented labeling strategy chooses the thesaurus lexeme with the highest χ^2 value as the cluster label. Again, the clusters are popu-

Label candidates	MI value
restrict	1392.68
interlock	1388.25
stick	1384.86
tie	1377.48
fix	1374.42
lessen	1374.34

Table 6.9: Example of the Thesaurus-MI results.

Label candidates	χ^2 value
restrict	76.91
trammel	76.90
curb	76.31
hold in	65.83
control	34.48
fasten	20.70

Table 6.10: Example of the Thesaurus-χ^2 oriented labeling results.

lated with their thesaurus matches as in the ThesFreq strategy. An example of the results is shown in Table 6.10. For the C_i cluster the term with the highest χ^2 value is *restrict*.

Examples of the application of the internal and differential cluster labeling strategies

The examples in this section show some results of the application of the cluster labeling strategies previously presented to a number of clusters composed manually (so-called *gold standard clusters*. We present two tables of examples corresponding to the internal and differential cluster labeling strategies, respectively. In the tables, 'GS' stands for "gold standard labels", i.e., labels assigned to the corresponding clusters manually by a human annotator.

Table 6.11 displays the results of the application of the internal cluster labeling strategies ('Freq', 'VHyper' and 'ThesFreq' denote the labeling strategies we worked with).

Table 6.11: Examples of the performance of the internal cluster labeling strategies with manually compiled clusters.

Gold Standard Clusters	GS	Freq	VHyper	ThesFreq
{comprise, contain, have, include}	contain	comprise	comprise	get
{bound, limit, restrain, inhibit}	limit	inhibit	determine	bind
{tighten, fasten, fix, secure, deposit}	fix	fix	fix	attach
{compress, trim, reduce, minimize}	reduce	reduce	cut	lessen
{extract, pull-out}	extract	extract	remove	take-out
{remove, cut, delete, erase, exclude}	remove	remove	remove	take-out
{enter, insert, interpose, introduce, enclose}	insert	insert	connect	introduce
{apply, feed, provide, give, use, supply, render}	produce	provide	provide	give
{hold , maintain, retain, support, prevent}	keep	support	maintain	hold
{accord, allow, let, permit}	let	accord	have	permit

We can observe that the human annotator (GS) chosen for all seven clusters a member from a cluster as its label—which suggests that Freq may be the most adequate strategy.

In two out of ten cases, the labels chosen by Freq and VHyper coincide with GS (cf. fix and remove) and Freq and VHyper coincide in one more label: comprise for the cluster {comprise, contain, have, include}. The only questionable suggestion made by Freq is inhibit for the cluster {bound, limit, restrain, inhibit}, while VHyper can be considered as failing in: determine for {bound, limit, restrain, inhibit}. Furthermore, it assigns the same label (remove) to two different clusters—which is certainly undesirable. Only one label chosen by ThesFreq coincides with GS: give. ThesFreq can be considered as failing in three suggestions: get for the cluster {comprise, contain, have, include}, bind for the cluster {bound, limit, restrain, inhibit}, and attach for the cluster {tighten, fasten, fix, secure, deposit}.

Table 6.12 presents some examples of the performance of the application of the differential cluster labeling strategies ('VHyper-MI', 'VHyper χ^2', 'ThesMI' and 'Thes χ^2' denote the labeling strategies we worked with).

Table 6.12: Examples of the performance of the differential cluster labeling strategies with manually compiled clusters.

Gold Standard Clusters	**GS**	**VHyper-MI**	**VHyperχ^2**	**ThesMI**	**Thesχ^2**
{comprise, contain, have, include}	contain	comprise	incorporate	incorporate	incorporate
{bound, limit, restrain, inhibit}	limit	restrict	restrict	throttle	restrict
{tighten, fasten, fix, secure, deposit}	fix	put	lay	find out	localise
{compress, trim, reduce, minimize}	reduce	trim down	thin-out	find-out	minify
{extract, pull-out}	extract	move forcibly	pull-up	pull-up	press-out
{remove, cut, delete, erase, exclude}	remove	erase	kill	cancel	take-out
{enter, insert, interpose, introduce, enclose}	insert	shut-it	enclose	pull-in	pull-in
{apply, feed, provide, give, use, supply, render}	produce	administer	furnish	furnish	furnish
{hold , maintain, retain, support, prevent}	keep	hold on	hold on to	defend	defend
{accord, allow, let, permit}	let	grant	grant	consent	consent

Label candidate	MI	Corpus frequency
come near	1390.48	2
fill	1390.48	87
get together	1390.48	0

Table 6.13: Example of the fallback strategy based on frequency.

As shown in table 6.12, none of the strategy labels matches the GS labels. We can observe that most of the labels chosen by the strategies are different from the cluster members, as for {apply, feed, provide, give, use, supply, render} cluster, where the assigned labels: administer and furnish can be considered as correct. Some labels are questionable as put and lay for the cluster {tighten, fasten, fix, secure, deposit}.

A quantitative evaluation of the cluster labeling strategies is given in Chaper 7.

Cluster labeling fallback strategies

Sometimes, differential labeling strategies assigned the highest probability value to a number of label candidates. In such a situation, we need to further apply a strategy to choose a label from the list of candidates, what we call a *fallback strategy*. A *fallback strategy* can be seen as a mechanism for carrying forth cluster labeling despite the incapacity of a labeling strategy to propose a single lexeme as a cluster label.

In this work, we have implemented two fallback strategies for cluster labeling, which have been chosen because of their simplicity. In what follows, we present them.

Frequency labeling (FreqF)

From a list of label candidates with the same probability, this strategy chooses the candidate with the highest frequency in the corpus as a label. Table 6.13 illustrates the application of the fallback strategy based on frequency when applied to the result of ThesMI given C_i.

$$\mathsf{C}_i = \{\mathsf{converge},\ \mathsf{meet},\ \mathsf{satisfy}\}$$

As shown in the table, three label candidates have the same highest MI value, thus the fallback strategy based on frequency chose the candidate *fill* as a label, as it has the highest frequency in our corpus.

Random fallback (RandF)

From a list of label candidates with the same probability, this strategy chooses the label randomly. Table 6.14 shows some examples of the random fallback strategy when applied to the result of the ThesMI strategy.

Cluster	Random label
converge, meet, satisfy	get together
record, file	impeach
accord, allow, let, permit	grant

Table 6.14: Example of the random fallback strategy.

In Table 6.14, at least one label can be considered as incorrect, e.g., *impeach* for the cluster {record, file}.

6.3 Illustration of the approach

Using the content relation retrieving procedure described in this chapter, we obtain from the simplified patent claim (4), the content relations displayed in Table 6.15. Each row in the table consists of a relation tuple, with the verbs as the labels (highlighted in boldface) and the arguments of the relation between brackets.

For the simplified patent (4) the system is able to retrieve twenty five verbal relations. Note that our approach is not restricted to the extraction of binary relations. Thus, for the verb *reflect*, the system identifies four arguments: *information recording medium*, *reflected light beam*, *focal spot* and *objective lens*. This is possible because we use deep syntactic trees to represent the syntactic structure of the sentences. It is important to notice that the arguments of the verb are ordered. This means that the arguments of the tuples are consistent with the valency structure of the verb. Thus, *the information recording medium reflects the light beam* is not the same as *the light beam reflects the information recording medium*.

Table 6.15: Relations extracted from claim (4).

comprise (automatic focusing device, objective lens)
comprise (automatic focusing device, beam splitter)
comprise (automatic focusing device, astigmatic optical system)
comprise (automatic focusing device, light detector)
comprise (automatic focusing device, focal position detecting circuit)
comprise (automatic focusing device, lens driving circuit)
focus(objective lens, light beam)
emit (light source, light beam, track of an information recording medium)
separate(beam splitter, reflected light beam)
reflect(information recording medium, reflected light beam, focal spot, objective lens)
emit(light source, light beam)
include(astigmatic optical system, optical element)
be-capable(optical element, **cause**(optical element, astigmatic **aberration**(separated reflected light beam)))
have(light detector, a light receiving surface)
divide (- , light receiving surface, plurality of light receiving sections)
arrange(-, light receiving sections, **symmetrically**, first axis and to a second axis))
extend(first axis, **parallel**(axial **direction**)
extend(second axis, **perpendicularly**(first axis)
be-adapted(second axis, **receive**(second axis, **transmit**(optical element, reflected beam)))
be-adapted(second axis, **give**(second axis, light reception output signal))
correspond(light reception output signal, **shape**(reflected light beam, spot)
form(- , reflected light beam, light receiving surface)
capable(focal position detecting circuit, **give**(focal position detecting circuit, output signal))
drive(lens driving circuit, objective lens)

Some tuples can be embedded into other ones, e.g.: *extend(first axis, parallel(axial direction)* where the adjective *parallel* is a relation by itself. Our verbal extraction procedure takes into account this fact, instantiating verbal relation arguments with other relations when needed. Consider, for instance, the tuple *correspond(light reception output signal, shape(reflected light beam, spot)*, where the second argument of *correspond* is the predicate noun *shape*, which takes *reflected light beam* and *spot* as arguments.

Note that when an argument of a relation is not instantiated in the text, the position of the argument is empty, as in *form(- , reflected light beam, light receiving surface)*.

Table 6.16: Generalized relations from claim (2).

comprise (automatic focusing device, objective lens)
comprise (automatic focusing device, objective lens)
comprise (automatic focusing device, beam splitter)
comprise (automatic focusing device, astigmatic optical system)
comprise (a light detector, light detector)
comprise (automatic focusing device, focal position detecting circuit)
comprise (automatic focusing device, lens driving circuit)
point(objective lens, light beam)
emit (light source, light beam, track of an information recording medium)
separate(beam splitter, reflected light beam)
emit(information recording medium, reflected light beam, focal spot, objective lens)
emit(light source, light beam)
comprise(astigmatic optical system, optical element)
be-capable(optical element, **cause**(optical element, astigmatic aberration))
comprise(light detector, a light receiving surface)
divide (- , light receiving surface, plurality of light receiving sections)
set(-, light receiving sections, symmetrically, first axis and to a second axis))
extend(first axis, parallel, axial direction)
extend(second axis, perpendicularly, first axis)
be-adapted(second axis, **receive**(second axis, **transmit**(optical element, reflected beam)))
be-adapted(second axis, **give**(second axis, light reception output signal))
correspond(light reception output signal, spot shape
perform(- , reflected light beam, light receiving surface)
capable(focal position detecting circuit, **give**(focal position detecting circuit, output signal))
drive(lens driving circuit, objective lens)

Some of the extracted relations presented in Table 6.15 have been generalized, as shown in Table 6.16. Recall that the aim of the generalization process is to unify similar relations in order to reduce the number of relations in a conceptual representation. From twenty five retrieved relations, six have been generalized, e.g.; the relations *include* and *have* have been generalized by the lexeme *comprise*. In Table 6.16, at least one of the generalizations is not appropriate as it changes the meaning of the sentence, as in the tuple: *point(objective lens, light beam)*, which corresponds to the simplified sentence: *The objective lens focuses the light beam.*

A detailed evaluation of the approach presented in this chapter is given and discussed in Chapter 7.

Chapter 7

Evaluation results

In this Chapter, we present and discuss a data driven evaluation of our approach to verbal relation extraction and generalization presented in Chapter 6. As most NLP systems, our application consists of a series of modules performing several NLP tasks in an stratified model; thus modules are executed sequentially.

The performance of each module is evaluated in isolation. The results are verified (or contrasted) against baselines or gold standard solutions, depending on the task. In the next sections, we present the evaluation results and a discussion of the most prominent stages of our system: a detailed evaluation of the simplification module; an evaluation of our relation extraction approach in contrast with other approaches; an evaluation of our approach on clustering of similar relations and finally, an evaluation of the implemented cluster labeling strategies.

We refrain from evaluating the Minipar parser since it has already been evaluated on the SUSANNE corpus where it has shown a coverage of 0.78 of the dependency relationships with 0.89 precision (Lin, 1998).

7.1 Evaluation of the simplification module

We present the evaluation of the most relevant stages of our simplification module, i.e., segmentation (as it is the basis for claim structuring and sentence reconstruction), coreference (as it is used by the claim structuring

algorithm) and claim structuring (as it is the output of the simplification module).

Evaluation of the segmentation

For the evaluation of the segmentation, we use a strict evaluation that counts 1:1 alignments between gold standard segments (comprising 5882 segments) and automatically determined segments. Our baseline addresses minimal content unit segmentation with a simple script that identifies every piece of text between punctuation marks as segments. The pseudo code of the segmentation baseline script is shown in Figure 7.1.

```
Procedure BASELINE SEGMENTATION
Given: C: Corpus of claims
Returns: the claims corpus segmented into minimal units
begin
punctuationMarks ← [“,”,“;”, “:”]
tokenList ← tokens(C)
while tokenListHasNext do
  if token = puntuationMark then
    token.ADD(break line)
  end if
end while
```

Figure 7.1: Pseudo code of the segmentation baseline script.

The evaluation of the segmentation algorithm is presented in Table 7.1. We achieve an F-score of 0.73, more than 20 points above the baseline.

	# automatic segments	# 1:1 aligments	F-score
Baseline	4078	3282	0.52
Our system	5342	4327	0.73

Table 7.1: Evaluation results for segmentation (# = number of segments).

In order to identify the most common errors produced by the segmentation algorithm we performed a qualitative evaluation. We found that the most common errors on segmentation are due to three reasons:

- Ambiguous commas;
- Consecutive segmentation keywords;
- Failure to consider multi-word expressions as keywords.

Segmentation errors cause by **ambiguous commas** are due to the fact that, despite commas being one of the features we use to characterize segments borders, not all commas are segment delimiters. Cf., for illustration, the following automatically segmented fragment that contains three commas (a fragment is a string between square brackets, with an integer sub-script indicating the segment number).

[A method of recording an information signal,]$_1$ [in particular an EFM-modulated signal,]$_2$ [on a record carrier,]$_3$

In the example above, all the commas are considered segments delimiters resulting in a wrong segmentation. The correct analysis would have been to maintain the fragment as a single segment as we are not interested in considering adjectival and prepositional phrases as individual segments.

Segmentation errors due to **consecutive segmentation keywords** result in undesirable segments that contain a single unit. This happens because the algorithm segments every time a segmentation keyword is encountered, even when the keywords are consecutive. Cf., for instance, the following example contains two consecutive keywords, a verb in past participle (*intended*) and a purpose marker (*for*) followed by a gerund (*recording*). Example (a) shows a wrong segmentation, while example (b) shows its correct segmentation.

(a) [. . . being recorded in the servo track and the servo-track]$_1$ [portion]$_2$ [intended]$_3$ [for recording being provided with a periodic track modulation]$_4$

(b) [. . . being recorded in the servo track and the servo-track portion]$_1$ [intended for recording being provided with a periodic track . . .]$_2$

The segmentation algorithm fails to consider *multi-word expressions as keywords* even though expressions as *so as*, *so that*, *such that* are indeed segments frontiers in our segmentation gold standard, cf., for illustration, an

example where the algorithm does not recognize the *so as* keyword as a segmentation boundary feature.

(a) [transcribing the mask pattern from the mask plate onto the substrate so as to form guide tracks and address tracks thereon;]$_1$

(b)[transcribing the mask pattern from the mask plate onto the substrate]$_1$ [so as to form guide tracks and address tracks thereon;]$_2$

We think that the global performance of the segmentation can be improved, for example, given richer information such as multi-word keywords. Regarding errors caused by consecutive segmentation keywords, they may be resolved by applying a set of correction rules after the segmentation algorithm, as done in (Tjong and Sang, 2001). To solve the problem of ambiguous commas, a possible solution is to constrain their application as keywords, for example, by combining commas with other context features.

Evaluation of the coreference resolution

We evaluated our coreference resolution approach on a set of 30 claims annotated manually with coreferences. The manual annotation resulted in a corpus of 199 coreferences. The results presented in Table 7.2 show an F-score of 0.81, which reinforces the observation of NP repetition in patent material. We performed an error analysis and found the following main types of errors:

- Chunking errors: the chunker failed to mark a chunk as an NP, resulting in a false negative.
- NP modifier: identical NPs used as modifiers in a more complex NP were marked as coreferring when they were not, for example, *inner surface of the substrate* and *inner surface of the boot.*
- Abstract NPs: abstract nouns like *information* or *direction* are often repeated within a claim but do not corefer.
- Partial match: some co-referring NPs such as *rotary valve assembly* and *valve assembly* are not an exact match and are thus not marked as coreferring,

# manual coref.	# automatic coref.	# automatic correct	F-score
199	190	159	81%

Table 7.2: Evaluation results for coreference (# = number of corefering NPs).

Although the coreference resolution algorithm is robust and performs reasonable well in patent claims, the algorithm may be improved by applying a more flexible approach on the coreference determination. For example, a string similarity measure could be used to calculate if two NPs corefered instead of the perfect match approach that has been used so far.

Evaluation of claim structuring

We performed the evaluation of claim structuring using as input the raw claims and the claims manually segmented and coreferenced (i.e, perfect input). Our baseline does claim structuring based on right branching given the number of segments in the gold standard. For the evaluation, we count the number of identical spans between automatic and manual structuring, excluding the top span as this would always be counted as correct. In order to be able to compare spans, we automatically mapped each segment of the raw input to its corresponding gold standard segment. The results of the evaluation are shown in Table 7.3.

	# automatic spans	# correct spans	F-score
Perfect input	201	114	%56
Raw	143	75	%42
Baseline	227	79	%35

Table 7.3: Performance of claim structuring on the test corpus (# = number of spans).

The results of the claim structuring are below our expectations. An error analysis showed us that the claim structuring algorithm is not reliable when the number of segments is more than thirty, approximately. Coordinated segments are more difficult to group than subordinated ones. As a consequence, text spans that are related by a coordination of coordinations are also problematic. In general, the structure of the first and second level in the claim tree are correct but the algorithm performs more poorly when

trying to group sub-trees at higher levels. We think that the claim structuring stage can be improved by applying a simpler approach. A possibility would be to re-use the claim structuring gold standard to train a binary classifier that learns when two segments are related, disregarding the type of relation.

7.2 Evaluation of verb extraction module

The goal of the evaluation of this stage is to assess how well our model is able to distil relations of the type <arg_1> VERB <arg_2> from patent claims. For the purpose of a comparative evaluation, we applied our verbal relation extraction procedure and then the relation extraction procedure proposed by (Blohm and Cimiano, 2007) to the claims of four European patents (EP0007902A1, EP0548937A1, EP0067095A2, and EP0080884A2). We have chosen Blohm and Cimiano (2007) as a reference because they aim to extract all the verbal relations available in a text, as we do. From the existing 94 verbal relations annotated as gold standard, our procedure correctly extracted 67 relations (which amounts to 71%), while Blohm and Cimiano's procedure extracts 51 relations (54%).

The main reason of failure in our procedure of verbal relation extraction was an erroneous syntactic output structure, which means that no valid DSynt-structure could be derived. See, for example, Figure 7.2 for an erroneous Minipar structure, which corresponds to the sentence *A groove receives said any risen portion of improper thickness of said photo-resist film.* In the example, there are at least two errors that prevent a correct mapping to a DSynt representation. First, the structure is composed of two sub-trees, which have been related by an empty node and linked to the sub-trees by empty relations. This means that Minipar was not able to deliver a complete syntactic structure. These errors prevent the identification of the second argument of the verb *receive*, which is *portion*. Second, there is a right attachment error. The prepositional phrase *of said photo-resist film* is wrongly attached to *thickness*. The correct analysis is that the prepositional phrase is attached to *portion*, given that *photo-resist film*, the NP included, is its first actant.

The failures of (Blohm and Cimiano, 2007) chunk-based procedure are the following:

- Patterns are too restrictive;

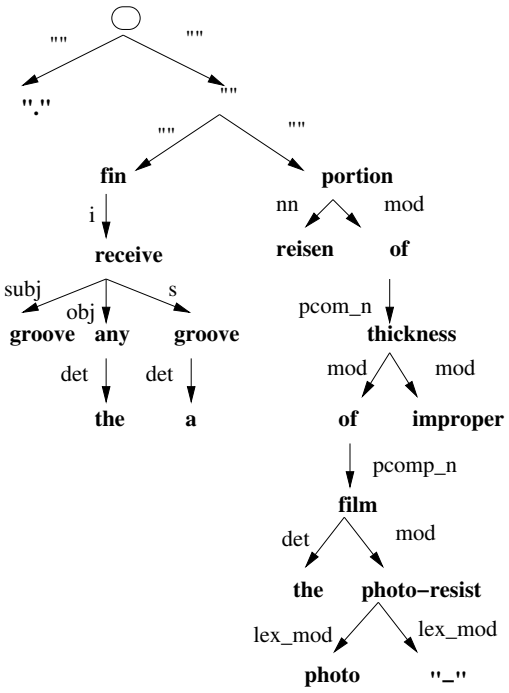

Figure 7.2: Example of an erroneous Minipar output structure.

- Limitations regarding the arguments of a relation;

- Long distance relation dependencies.

Patterns are too restrictive

The two patterns used by Blohm and Cimiano (2007) do not capture all verbal relations (recall that they use <NP1, V, NP2> and <NP1, V, PP, NP2> to search for the argumentative structure of a verbal relation V). See, for instance, Example (i), which demonstrates that the patterns used are too restrictive given that they do not take into account the possibility of a prepositional phrase before the verbal phrase. In Example (ii) there is a relation expressed by an <NP1, V, ADJC> pattern, which again, is not considered by Blohm and Cimiano.

Example (i)
<NC> the reflectivity </NC>
<PC> of
<NC> the layer </NC> </PC>
<VC> is decreased </VC>
<PC> by
<NC> irradiation </NC> </PC>
Example (ii)
<NC> the support member </NC>
<VC> is </VC>
<ADJC> substantially rigid </ADJC>
Example (iii)
<NC> which </NC>
<VC> comprises </VC>
<NC> a semiconductor ray source </NC>
Example (iv)
<NC> the phase detector </NC>
<VC> to coincide </VC>
<NC> the focal point </NC>

Limitations regarding the arguments of a relation

NP1, NP2, and PP are not necessarily arguments of V, as shown in Example (iii) and (iv). In these cases the patterns are correct but their instantiation does not reflect the arguments of the verb.

Long distance relation dependencies

The relevant NP1, NP2 or PP are situated at a long distance from V and are thus not captured by the patterns. See, for illustration, Example (v) where NP1 is not the first argument of the verb *vary*. Instead, the first argument of *vary* is *focusing means*, which is at a long distance from the verb itself in the claim fragment: [...] *said focusing means, which is parallel to the control means, varies the position of said ray spot to the optical axial direction* [...].

Even then syntactic dependency structures are more reliable for relation extraction, chunking performs quite well disregarding the complexity of patent

Example (v)
<NC> the control means </NC>
<VC> varies </VC>
<NC> the position </NC>

claim sentences. But, as expected, the chunking patterns used by Blohm and Cimiano (2007) are too restricted, specially when applied to patent claims texts that have a complex syntactic structure that is not captured by the chunking analysis.

7.3 Evaluation of clustering

We performed two evaluations of the clustering module. One evaluation is devoted to assess the quality of our clustering approach against random clustering as a baseline, which is a standard procedure in the clustering literature. The other evaluation is devoted to assess the accuracy of the clustering results against gold standard clusters.

For these evaluations, we use a corpus of 2076 English patent claims. From this corpus, a list of the 193 most frequently used verbs is compiled (verbs with less than three occurrences are discarded).

The verbs have been first clustered manually to obtain the gold standard (reference) clusters (54 in total) by two annotators not involved in the work otherwise. The clustering procedure consisted of two rounds. In the first round, which already showed a high mutual agreement between the annotators, the annotators acted independently; the second round was a consensus round (the complete gold standard is presented in Appendix A).

Before computing clustering evaluation measures, it is necessary to define clustering evaluation requirements from the point of view of our goals.

Inspired by Schulte im Walde (2006), we define requirements for the task of clustering verbs into classes. First, general requirements on a clustering evaluation are defined and second, requirements on the verb-specific clustering evaluation are given. Finally, based on the clustering evaluation demands, the clustering evaluation measures used to asses the quality of our clustering approach are presented.

General demands on clustering evaluation

- The evaluation should be defined without a bias towards a specific number and size of clusters.
- The evaluation measures should distinguish the quality of (i) the whole clustering partition C and (ii) the specific clusters C_i.

Linguistic demands on clustering evaluation

- The clustering result should not be a single cluster, $\mid C \mid = 1$, as a single cluster does not represent an appropriate model for verb classes.
- The clustering result should not be a clustering partition with singletons only, as it does not represent a verb classification.

Taking into account the mentioned demands, we define a set of evaluation measures to assess the performance of our clustering approach. These evaluation measures are presented in the next subsection.

Clustering evaluation measures

In this subsection, we present the evaluation measures chosen to evaluate the performance of our clustering approach. These measures are: Purity, Adjusted Rand Index and Pair wise evaluation. These measures have been chosen for two main reasons:

(i) they satisfy the evaluation demands defined in the previous section,

(ii) they have been used in previous works on verb clustering, which facilitates the comparison with other approaches.

Before presenting the evaluation measures for clustering, we need to introduce the notion of contingency tables, as they are used in a number of evaluation measures presented in this subsection.

Contingency tables are a typical means for describing and defining the association between two partitions that can be defined as,

A C x M contingency table is a C x M matrix with rows C_i, $1 \leq i \leq k$ and columns M_j,$1 \leq j \leq l$. t_ij denotes the number of objects that are common to the set C_i in partition C (the clustering result) and the set m_j in partition M (the gold standard classification). Summing over the row or column values gives the marginal values t_i and t_j, referring to the number of objects in classes C_i and M_j, respectively. Summing over the marginal values results in the total number of n objects in the clustering task (Schulte im Walde, 2006).

The number of pairs with reference to a specific matrix value x is calculated as $\binom{x}{2}$. For illustration purposes, a C x M contingency table is defined in example (a).

(a)

$$M = \{ \mathrm{M}_1 = \{\mathrm{a,b,c}\}; \mathrm{M}_2 = \{\mathrm{d,e,f}\} \}$$

$$C = \{ \mathrm{C}_1 = \{\mathrm{a,b}\}; \mathrm{C}_2 = \{\mathrm{c,d,e}\}; \mathrm{C}_3 = \{\mathrm{f}\} \}$$

M corresponds to a manual or gold standard clustering solution. The manual clustering solution is composed by two partitions. The first partition comprises three instances, namely, *a*, *b* and *c*. The second partition comprises three instances *d*, *e* and *f*. C corresponds to an automatic clustering solution, composed by three partitions. The first partition comprises the instances *a* and *b*; the second partition comprises the instances *c*, *d* and *e* and, finally, the third partition comprises the instance *f*.

For the example presented above, the corresponding contingency table is presented in Table 7.4.

	M_1	M_2	
C_1	$t_{11} = 2$	$t_{12} = 0$	$t_{1.} = 2$
C_2	$t_{21} = 1$	$t_{22} = 2$	$t_{2.} = 3$
C_3	$t_{31} = 0$	$t_{32} = 1$	$t_{3.} = 1$
	$t_{.1} = 3$	$t_{.2} = 3$	$n = 6$

Table 7.4: Example of a C x M contingency table example.

The difference between the two clustering solutions can be assessed by a variety of statistical tests. As already mentioned, we use Purity, Pair wise

evaluation and Adjusted Rand Index to compare the clustering result with the gold standard solution.

Cluster purity evaluation

Purity is a simple and transparent evaluation measure of clustering quality (Manning et al., 2008):

$$purity(C, \Omega) = \frac{1}{N} \sum_{i=1}^{k} \max_{j} |w_i \cap c_j| \tag{7.1}$$

where $\Omega = \{w_1, w_2, \ldots, w_i, \ldots, w_k\}$ is the set of automatically derived clusters, $C = \{c_1, c_2, \ldots, c_j, \ldots, c_m\}$ is the set of gold standard clusters, and N is the total number of elements in the whole collection.

To compute *purity*, each cluster w_i is assigned a gold standard cluster c_j with which it has the maximal overlap, such that the *purity* is reflected by the number of elements of w_i correctly assigned to c_j, normalized by N. Perfect clustering has a *purity* of 1, while bad clustering has *purity* values closer to 0.

Given the manual and automatic classification in example (a), Purity is (1/8) x (2+2) = 0.5

Purity is reciprocally proportional to the number of singleton clusters. Thus, we cannot use *purity* to evaluate the quality of clustering against the number of clusters (Manning et al., 2008, 328). To take this into account we also applied other clustering evaluation measures.

Adjusted Rand Index evaluation

The Adjusted Rand Index or Index Adjusted by Chance (Hubert and Arabie, 1985) aims to stablish an overall comparison between de manual and the automatic clustering. The Adjusted Rand Index (ARI) is based on the Rand Index (Rand, 1971), which measures the percentage of decisions that are correct.

Given a set of n elements $S = \{O_1,\ldots,O_n\}$ and two partitions of S to compare $X = \{x_1,\ldots,x_r\}$ and $Y = \{y_1,\ldots,y_s\}$, the following is defined:

- a, the number of pairs of elements in S that are in the same cluster in X and in the same cluster in Y.
- b, the number of pairs of elements in S that are in different clusters in X and in different clusters in Y.
- c, the number of pairs of elements in S that are in the same set in X and in different sets in Y.
- d, the number of pairs of elements in S that are in different clusters in X and in the same cluster in Y.

So that the Rand Index R, can be expressed, as:

$$\text{Rand Index} = \frac{a+b}{a+b+c+d} = \frac{a+b}{\binom{n}{2}} \tag{7.2}$$

where, $a + b$ are considered as the number of agreements between X and Y and $c + d$ as the number of disagreements between X and Y.

For the example in (a):

$a = 2\{(a,b),(d,e)\}$

$b = 4\{(a,c),(b,c),(d,f),(e,f)\}$

$c = 6\{(a,c),(b,c),(c,d),(c,e),(d,f),(e,f)\}$

$d = 9\{(a,c),(c,d),(c,d),(a,e),(a,f),(b,c),(b,d),(b,e),(b,f),(c,f)\}$

$$\text{Rand Index} = \frac{2+9}{2+4+6+9} = \frac{11}{21} = 0.52 \tag{7.3}$$

The general form of an index corrected by chance is:

$$\text{ARI} = \frac{Index - ExpectedIndex}{MaximunIndex - ExpectedIndex} \tag{7.4}$$

The *Index* refers to the observed number of object pairs in which the partitions coincide. The *Expected index* is the number of object pairs with

class agreements that are attributable to a particular cell in the contingency table, defined by the number of pairs in the column divided by the total number of pairs. The *Maximum index* is the number of objects pairs given by the average number of possible pairs in the clustering and the manual classification. According to Hubert and Arabie (1985), the ARI can be written as:

$$\mathrm{ARI} = \frac{\sum_{ij}\binom{ni}{2} - \frac{\sum_i \binom{n_i}{2}\sum_j \binom{n_j}{2}}{\binom{n}{2}}}{\frac{\sum_i \binom{n_i}{2} + \sum_j \binom{nj}{2}}{2} - \frac{\sum_i \binom{n_i}{2}\sum_j \binom{n_j}{2}}{\binom{n}{2}}} \tag{7.5}$$

Thus, for example (a) and its contingency Table 7.4, the computation of the ARI is as follows:

$$\begin{aligned} \mathrm{ARI} &= \frac{4 - \frac{6*5}{\binom{6}{2}}}{\frac{6+5}{2} - \frac{6*5}{\binom{6}{2}}} \\ &= \frac{4 - 0.2}{5.5 - 0.2} \\ &= \frac{3.8}{5.3} = 0.71 \end{aligned} \tag{7.6}$$

Pair wise evaluation

The Pair wise evaluation by (Hatzivassiloglou and Mckeown, 1993) consists of assessing the common clustering membership of object pairs in a gold standard cluster (GC) and in an automatic cluster (AC). On the basis of common cluster membership, precision and recall values are calculated in the standard way:

- true positives (TP): the number of common pairs in the GC and the AC.

- false Positives (FP): the number of pairs in the AC, but not in the GC.
- false Negatives (FN): the number of pairs in GC, but not in AC.

Pair-wise Precision is the number of true positives divided by the number of pairs in the automatic classification:

$$\text{Precision} = \frac{TP}{TP + FP} \tag{7.7}$$

Pair-wise Recall is the number of true positives divided by the number of pairs in the gold standard classification:

$$\text{Recall} = \frac{TP}{TP + FN} \tag{7.8}$$

Pair-wise F-score is the harmonic mean of precision and recall:

$$\text{F-score} = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{7.9}$$

Thus, for example (a):

$$\begin{aligned} \text{Precision} &= \frac{2}{2+4} = 0.3 \\ \text{Recall} &= \frac{2}{2+6} = 0.25 \\ \text{F-score} &= 2 * \frac{0.3 * 0.25}{0.3 + 0.25} = 0.27 \end{aligned} \tag{7.10}$$

Note that Rand Index can also be written as:

$$\text{ARI} = \frac{TP + TN}{TF + TF + TF + TF} \tag{7.11}$$

Clustering evaluation with the Pair wise method has the advantage that it uses measures such as recall, precision and F-score, which are already familiar within the research community.

Our clustering vs. baselines

We have defined two clustering baselines based on random clustering that are evaluated by the evaluation measures defined above. Results of random clustering will help to check the success of the learning scenario. Thus, clustering close to the random baseline indicates either a failure of the clustering algorithm or that the verbs types cannot be grouped consistently based on our criterion.

- Baseline (a): Verbs are randomly assigned to a cluster (with a cluster number between 1 and the number of manual classes), repeating this process 50 times. The result baseline value is the average value of the 50 repetitions.
- Baseline (b): The total of manual clusters and its category sizes are preserved and verbs are randomly assigned to them. Again, the result baseline value is the average value of the 50 repetitions.

Clustering evaluation results

The verbs have been clustered using the clustering strategy that best performed in our trials: the optimized repeated bisections strategy. As already pointed out above, the best results were achieved for 60 clusters (i.e., k = 60). The clustering evaluation results against gold standard clusters are presented in Table 7.5.

As shown in Table 7.5, our approach performs clearly above the two baselines for all the evaluation measure calculated, achieving a cluster *purity* of 0.73. Our clustering strategy achieves a similar performance as Korhonen et al. (2006), which is one of the very few works that addresses the problem of clustering of verbs taken from a specialized domain. They achieved a

Eval. measures	Baseline (a)	Baseline (b)	Our clustering
Purity	0.33	0.34	0.73
ARI	0.02	0.01	0.34
Pair Wise Recall	0.01	0.03	0.32
Pair Wise Precision	0.02	0.03	0.33
Pair Wise F-score	0.02	0.03	0.32

Table 7.5: Clustering evaluation results.

similar *purity* (namely 0.72) when clustering 192 verbs from the biomedical domain with the Information Bottleneck (Tishby et al., 1999) and Information Distortion (Dimitrov et al., 2001) methods into 53 clusters (against a gold standard divided into 50 classes).

A re-implementation of Korhonen et al. (2006) algorithm with the purpose of applying it to our data for a direct comparison seems infeasible, as this would be based on the previous extensive work of the first author on the acquisition and clustering of sub-categorization frames (Korhonen, 2002; Korhonen et al., 2003). However, the similarity of the settings of both works suggests that such a re-implementation is not even necessary. On the other hand, as already argued by Korhonen et al. (2006), domain-specific and general discourse lexical classifications cannot be easily compared due to major differences between the goals of the two types of tasks.

Clustering error analysis

In addition to the evaluation figures given by the evaluation measures, we also performed an error analysis on the automatically generated clusters to better understand the quality of our clustering approach.

From the 60 clusters obtained automatically, our clustering approach is able to produce 22 clusters that align 1:1 to the gold standard clusters, which means that 36.66% of the data has been perfectly grouped. Nine of the clusters obtained automatically correspond to clusters with singletons (clusters composed only by one verb). As already mentioned, clusters with singletons are undesirable as they do not represent a verb classification. The verbs that belong to singletons are the following: *discriminate*, *repeat*, *inhibit*, *exclude*, *compress*, *focus*, *interpose*, *process* and *denote*.

The rest of the clusters have a *purity* below 1 which means that they were not aligned 1:1 to the gold standard clusters. *Purity* is below 1 when: (i) a partition (or cluster) is incomplete; (ii) a partition contains more units than its more similar gold standard cluster; (iii) a partition contains outliers. For example, the automatically generated cluster: *enter*, *insert*, *introduce* has a *purity* of 0.66. During evaluation, the cluster is partially aligned with the most similar gold standard cluster, which is *enter*, *insert*, *introduce*, *interpose*, *enclose*. We can observe that the automatic partition is incomplete as two verbs, *interpose* and *enclose*, are missing. In the automatic solution the verb *interpose* is a singleton and the verb *enclose* belongs to another partition.

The automatically generated cluster *accommodate*, *adapt*, *adjust*, *transcribe* has a *purity* value of 0.75. This example corresponds to partitions that contain more units than its most similar gold standard cluster. In this case, it is aligned with the gold standard cluster *accommodate*, *adapt*, *transcribe*, except for the verb *adjust*, which belongs to the cluster: *adjust*, *arrange*, *place*, *position*, *set*, *base* in the gold standard solution.

Outliers are considered verbs that are far from any other verb and therefore do not fit well into any cluster. If there are verbs that are very different from other verbs and not similar to each other, they can have a big impact during cluster formation. The consequence is that the number of clusters increases or that the clusters are less homogeneous. The clustering method we use takes into account this fact and creates an *rag bag* or miscellaneous cluster that contains all verbs that do not fit with the rest of the data. In our clustering solution the cluster with outliers has a *purity* of 0.23. The verbs that belong to this cluster are: *accord*, *desire*, *disengage*, *embed*, *emit*, *enclose*, *face*, *file*, *include*, *irradiate*, *minimize*, *mirror*, *open*, *prevent*, *reflect*, *reproduce*, *weld*. However, as pointed out by Amigó et al. (2009), most of the clustering evaluation measures, including the ones presented here, do not considered the possibility to assign a different weight to the items in the rag bag cluster. Consequently, when an item is introduce in a *rag bag* cluster, the precision of the overall distribution decreases.

The ARI and the Pair wise F-score evaluation are based on pair counting. These measures are more suitable for measuring the agreement between two partitions with different number of clusters. A drawback of these measures is that changes in big clusters have an excessive negative impact (Amigó et al., 2009). In the ARI and Pair wise evaluation, FP and FN instances are equally weighted, which maybe not suitable for certain applications. In

our application, grouping pairs of dissimilar verbs is worse than separating similar verbs. Thus, we are more interested in improving precision. As shown in Table 7.5, the precision we achieve is slightly higher than recall.

WSD-based similarity vs. automatic similarity evaluation

The evaluation measures used to evaluate the performance of the clustering strategy can also be used to verify our one-sense-per-patent-domain assumption and to cross-check the similarity of all WN-senses of a verb V_i with all WN-senses of the verb V_j (see section 6.2). This allows us to dispense with the costly task of word sense disambiguation (WSD). For this purpose, we selected 100 verbs from 28 gold standard clusters of our verbal relation set and disambiguated them manually. The disambiguated verbs were clustered, again manually, according to the similarity of their senses in WN, as illustrated in Table 7.6.

As shown in Table 7.8, the *purity* of the obtained clusters against the original 27 clusters ranged at 0.64. In a second round, the 100 verbs were again clustered, but this time automatically by our strategy described in Chapter 6 without prior disambiguation, as illustrated in Figure 7.7. The cluster *purity* was 0.71. The results of the rest of the evaluation measures shown in Table 7.8 are also similar in both approaches. From this outcome, we can infer from our working assumption that we can dispense with a WSD stage without major harm because the results of clustering remain valid. Korhonen et al. (2006) came to an analogous conclusion concerning the uniformity of verb senses in the biomedical domain when examining the *purity* of the clusters there.

7.4 Evaluation of cluster labeling

The evaluation of cluster labeling was done resorting to human judges. For the evaluation, we used 54 verb clusters (see Section 7.3) as the list of clusters to name. The 54 clusters were presented to three judges, together with the labels assigned to each of the clusters by our system and by a human collaborator (the gold standard labels, see Annex B), such that the judges did not know the source of a label.

For each cluster, the judges were asked to qualify both labels as 'correct', 'partially correct' or 'incorrect'. The evaluation results of the internal clus-

1. {record, file}
2. {compress, trim, reduce, minimize, cut}
3. {delete, erase, exclude, remove}
4. {desire, require}
5. {come, approach, face}
6. {select, take}
7. {repeat, reproduce, disperse, propagate}
8. {differentiate, distinguish, discriminate, separate}
9. {adhere, bond}
10. {encode, encrypt, inscribe, write, compose}
11. {define, delimit, specify, characterize}
12. {associate, connect, weld, join, link}
13. {relate, couple, calculate, process}
14. {conduct, direct, guide, lead}
15. {carry, execute, perform}
16. {tighten, fasten, fix, secure}
17. {deposit, locate, situate}
18. {bound, limit, restrain, inhibit}
19. {address, cover, handle, coat}
20. {adjust, arrange}
21. {place, position, set, base}
22. {block, stop}
23. {cause, construct, create, do, effect, make, produce, constitute, embed, establish, shape, form}
24. {engage, lock}
25. {operate, control}
26. {curve, wind}
27. {decrease, descend, fall, taper}
28. {display, exhibit, expose, show}

Table 7.6: 28 manually disambiguated clusters.

1. {select}
2. {compose}
3. {base}
4. {weld}
5. {locate}
6. {coat}
7. {reduce}
8. {construct}
9. {adhere, bond}
10. {place, position}
11. {deposit, situate}
12. {do, make}
13. {delete, erase}
14. {associate, relate}
15. {fasten, tighten}
16. {form, shape}
17. {execute, perform}
18. {decrease, fall}
19. {control, operate}
20. {cut, trim}
21. {bound, limit, restrain}
22. {encode, encrypt, inscribe}
23. {differentiate, distinguish, separate}
24. {display, exhibit, expose}
25. {conduct, direct, guide, lead}
26. {connect, join, link}
27. {address, adjust}
28. {approach, arrange, block, calculate, carry, cause, characterize, come, compress, constitute, couple, cover, create, curve, define, delimit, descend, desire, discriminate, disperse, effect, embed, engage, establish, exclude, face, file, fix, handle, inhibit, lock, minimize, process, produce, propagate, record, remove, repeat, reproduce, require, secure, set, show, specify, stop, take, taper, wind, write}

Table 7.7: Result of the disambiguated set automatically clustered.

Eval. measures	WSD-Verbs	WSD-OurApproach
Purity	0.64	0.71
ARI	0.2	0.3
Pair Wise Recall	0.41	0.42
Pair Wise Precision	0.06	0.1
Pair Wise F-score	0.11	0.16

Table 7.8: WSD-similarity calculus vs. our similarity calculus.

ter labeling strategies are shown in Table 7.9 and the results of differential cluster labeling strategies are shown in Table 7.10.

Table 7.9: Internal clustering labeling strategies evaluation.

	% Correct	% Partially correct	% Incorrect
Gold standard	77%	17%	7%
Freq	78%	20%	2%
VHyper	43%	25%	32%
ThesFreq	58%	26%	16%

Table 7.9 suggests that the Freq strategy, which chooses as the label of a cluster its most frequent member, shows better results, achieving a 78% correctness. The VHyper strategy shows significantly worse results than Freq, achieving a correctness of only 43%. This is, as has already been seen in Chapter 6, largely because the hyperonyms in WN tend to be too abstract to serve as a label of their hyponyms—as is, e.g., also the case with *move* for the cluster {disperse, propagate} (just to add another example). The ThesFreq strategy shows acceptable results, achieving a correctness of 58% and a partial correctness of 26%. The weakness of the ThesFreq strategy is that although the meaning of the labels are related to the verbs in the clusters they are not considered appropriate as labels, e.g., *travel* for the cluster {follow, trace}. A baseline strategy that arbitrarily chooses a member of a given cluster as the label of this cluster reaches a 31% match with the gold standard labels.

Table 7.10 shows the results of the differential cluster labeling strategies. As in internal labeling strategies, the results that used a thesaurus are better than the ones that used verb hyperonyms from WN. The best differential strategy is ThesMI, achieving a correctness of 70%. The Thesχ^2 strategy

Table 7.10: Differential clustering labeling strategies evaluation.

	% Correct	% Partially correct	% Incorrect
Gold standard	77%	17%	7%
VHyper-MI	50%	45%	5%
VHyper χ^2	60%	27%	13%
ThesMI	70%	25%	5%
Thes χ^2	67%	22%	11%

has a slightly lower score, achieving correctness of 67%.

According to the qualitative evaluation, the performance of one of the internal cluster labeling strategies, 'Freq', is the most similar to that of a human, while the strategies that are based on WN hyperonyms, which are most commonly used for lexical labeling, perform significantly more poorly. This is partly due to the fact that most of the work on lexical labeling target the labeling of nominal rather than verbal clusters and WordNet, which is used as reference resource, has, in general, very flat verbal hierarchies. Differential strategies that use the thesaurus as an external resource show competitive results, as they are close to the human's judgements. A weakness of differential labeling is that sometimes labels are a bit bizarre as they correspond to low frequency terms. For example, in the ThesMI strategy the cluster {become, convert, turn} is labelled with the term: *metamorphose*, which is judged as 'partially correct'. Although the label reflects the meaning of the cluster it is considered inappropriate to be used as a term in the technical domains of our patent claim corpus (optical recording devices and mechanical tools).

The evaluation of fallback clustering labeling strategies have also been done by human judges. According to the evaluators, when the *random fallback strategy* is applied, 37.09% of labels are judged as correct and, when the *frequency fall back strategy* is applied, 50% of the labels are considered as correct.

7.5 Conclusions

In this Chapter, we have presented a data driven evaluation of the most important modules of our system's architecture to understand their weak

points in isolation. Some important modules like claim structuring and cluster labeling have also been evaluated from perfect inputs so that no errors from earlier modules are encountered. In this thesis, we do not present a measure of the global performance of the system to identify the earliest module in the chain that prevents the system from finding the mentioned relations and its arguments. The main reason is that a global evaluation required the development of a test set of the extracted relations, which is a very costly task that we cannot address.

Due to the characteristics of the patent claim texts, we have developed a simplification module that avoids any loss of information by the analysis of the claim's structure and coreference relations. During simplification, the performance of the segmentation and claim structuring are crucial steps. Evaluation results on these two tasks are promising. So far, not many works target the problem of patent claim simplification and patent claim structure derivation and, we are the first to present supervised machine learning solutions to these problems in the patent processing domain.

During the first step of simplification, when the patent claim document structure takes place, which is an important feature when doing patent processing that enables, for example, the organization of patent material in databases for further retrieving. But, as expected, errors are not rare and must be dealt with. For instance, a claim referring back to itself directly or indirectly, or a claim referring back to a claim that does not exist. The output of the simplification module should also be considered as a valid paraphrasing system, in particular in our application since it obviously facilitates the comprehension of the content of patent claims. Moreover, simplification material can be easily parsed and processed further such that deep syntactic parsing or shallow semantic structures can be obtained, which should be considered also as an advantage to our approach.

The utility of the simplification module has been also confirmed by the results obtained from the evaluation of the relation extraction stage. We demonstrate, with a comparative evaluation, that relation extraction techniques are more reliable when they operate with deep syntactic structures rather than surface syntactic analysis like chunking. But, as already mentioned, patent claims have a complex linguistic style and the state-of-the-art syntactic parsers cannot be applied to claim texts with sufficiently quality and a previous preprocessing stage, as in our simplification approach.

In this thesis we focus on relations expressed by verbs as they are the most productive predicate form. We use deep syntactic structures rather than

surface syntactic structures to identify the verbs an their arguments, with reasonable accuracy. In order to generalize over the obtained relations, we further apply clustering, an unsupervised learning technique. Evaluation on clustering demonstrates that our relation similarity approach achieves similar results than other works, but our solution is simpler and faster. We calculate the similarity of relations by comparing their synonym synsets, in contrast to traditional approaches of relational or verb similarity that use the arguments of the relations, sub-categorization frames and other syntactic information as features in the relational similarity calculus, which are obviously hard to obtain, prone to errors and not independent of the corpus. The results obtained in clustering labeling show that cluster labeling internal methods are efficient, but they fail to distinguish terms that are frequent in the collection as a whole from those that are frequent in only one cluster. One of the consequences of this weakness is that the same label could be assigned to more than one cluster. With respect to differential labeling, we need to take into account that very low frequency terms should be omitted as label candidates as they would not be the best in representing a whole cluster. However, in our experiments, no frequency filter has been applied and all the terms are considered as label candidates.

Chapter 8

Final conclusions

In this Chapter, we present the final conclusions of this thesis. First, the contributions of the thesis are specified, then the limitations of the approach proposed in the thesis are discussed and possible applications are described. Finally, potential directions for future work are examined.

8.1 Contributions

This section summarizes the main contributions of the thesis, which are:

- the first linguistic study on the patent claim style,
- a framework for diverse, *n-ary* and unlexicalized RE,
- a framework for unsupervised relation generalization,
- a claim paraphrasing application.

Certain aspects of these contributions have been published in collaboration with other members of the TALN Group, UPF:

- G. Ferraro and L. Wanner. In Press. Labeling semantically motivated clusters of verbal relations. *Procesamiento de Lenguaje Natural 49.*

- G. Ferraro In Press. Algoritmo supervisado para la detección de unidades discursivas: el caso de las reinvindicaciones de patentes. In *Actas del XXIX Congreso Internacional de AESLA: Empirismo y herramientas analíticas para la Lingüística Aplicada del Siglo XXI.*

- G. Ferraro and L. Wanner 2011. Towards the derivation of verbal content relations from patent claims using deep syntactic structures. *Knowledge-Based Systems.* 24:1233-1244.

- L. Wanner, S. Bott, N. Bouayad-Agha, G. Casamayor, G. Ferraro, J. Gran, A. Joan, F. Lareau, S. Mille, V. Vidal. 2011. Paraphrasing and multilingual summarization of patent claims. *PATExpert: A Next Generation Patent Processing Service.*

- N. Bouayad-Agha, G. Casamayor, G. Ferraro, S. Mille, V. Vidal, L. Wanner. 2009. Improving the comprehension of legal documentation: the case of patent claims. In *ICAIL.*

- N. Bouayad-Agha, G. Casamayor, G. Ferraro, L. Wanner. 2009. Simplification of patent claim sentences for their paraphrasing and summarization. In *FLAIRS.*

The first linguistic study about the style of patent claims

The study carried out in this thesis provides new insights on the linguistic nature of patents claims. We know from experience in other NLP-fields, such as automatic summarization, machine translation and text simplification, among others, that NLP-applications perform better if their approaches are linguistically motivated and if they operate with linguistic information. Knowing the linguistic peculiarities of patent claims helps to answer a series of questions:

- is it possible to use the state-of-the-art NLP technologies such as chunking and syntactic parsing directly on patent claims?
- how should standard tools be adapted to the patent domain?
- which aspects/regularities encountered in the claims can be exploited to develop or improve NLP tools?
- are some sections of the patent more relevant to specific application tasks than others?

To the best of our knowledge, our study is the first study to be carried out on the linguistic characteristics of patent claims. This study aims to contribute with answers to the mentioned questions. In short, to provide insight on the linguistic nature of patent claims with the aim to facilitate the development of NLP applications in the context of automatic claim processing. This study is furthermore useful for patent users such as patent writers and patent analysts, as it provides an easy to understand linguistic description of the claims, accompanied by examples.

A framework for diverse, *n-ary* and unlexicalized RE

The RE approach proposed in this thesis is new in the sense that it operates with deep syntactic (quasi-semantic) dependency structures, in contrast to similar RE works that use surface syntactic dependency structures. Because we operate directly over deep dependency tree structures, the order of the predicate arguments are correctly identified. This ensures that, given a relation, each of its arguments is precisely instantiated in its corresponding slot in the valency structure. As relation tuples are derived from abstract structures (deep syntactic trees), the relations are not limited with respect to their type or number. This is particularly useful for open relation extraction applications, where the types of relations of interest are not known in advance. The fact that relation extraction is done directly over deep tree structures guaranties that no restriction on the arity must be imposed. This is a fundamental aspect that must be taken into account for the identification of the valency structure of any predicate unit.

Another advantage is that we focus on a holistic framework for the description of natural language across the thesis, namely the MTT. Within this framework, we elaborate a description of the codification of relations in natural language. This is helpful because the description on relations in natural language is theoretically sustained, meanwhile, most of the state-of-the-art RE approaches do no follow any linguistic theory.

The linguistic constructions in patent claims are extremely complex. The fact that our strategy is able to cope with them with a rather good performance lets us assume that it can be also ported to other genres.

A framework for unsupervised relation generalization

In this thesis, we presented a framework for unsupervised relation generalization, via clustering and cluster labeling techniques. To group similar relations we used synonymy, extracted from Wordnet as features. The performance of our relation generalization strategy is similar to the performance of state-of-the-art techniques, but our approach is simpler and independent of the corpus.

Once similar relations are grouped together, those groups are assigned with labels that represent them. This means that we are able to deliver a relation classification that is not restricted to a predefined typology. Thus, the number of possible relation classes and the number of relations is open.

A claim paraphrasing application

The preprocessing module of the architecture presented in Chapter 6, namely the claim simplification module, is also a claim paraphrasing application. Thus, the output of the simplification module contains the same information as the original claims, but in a more readable way. The originality of our simplification/paraphrasing module is that it relies on surface linguistic analysis to break down complex sentences as the patent claims are. Besides, two components of the simplification module are novel. We are the first to propose a supervised machine learning based claim segmentation algorithm, and we are the first also to propose a supervised machine learning approach to build a claim tree structure. For details on this application and its evaluation, see (Bouayad-Agha et al., 2009a,b)

8.2 Limitations

Despite the contributions made by the thesis to the field of NLP-processing of specialized discourse, our proposal has several limitations that must be taken into account:

- the claims vs. other patent sections,
- some claim types are not covered in this thesis,
- our linguistic study of patent claims is limited to English,

- our approach is English dependent.

The claims vs. other patent sections

Regardless of the fact that the claims are the most important section of the patents, this thesis does not capture their whole complexity nor the complexity of patents. To provide an accurate overview of the claims it is necessary to also work with the description section of patents, which is a task that has not been addressed. This is because a claim, must be interpreted in the light of the definitions provided in the description section, in which explanations about how to make and use the invention are given. This can be tracked, for example, aligning some important noun phrases from the claims, e.g., the main components of the invention, with segments in the description that elaborate on those components.

In the same vein, an illustration of the claim contents may help patent users in several tasks such as search of similar technologies and inspiration for new inventions, among others. However, to truly understand the content of the claims, the users should also consider reading the entire claims as well as other patent sections like the description of the invention, the description of the drawings, the background of the invention, etc.

Some claim types are not covered

The approach presented in this thesis does not cover all the possible types of claims, specifically, it does not cover chemistry patents: patents in chemistry, which are described using Markush and Swiss claims, have indeed a very different style from the rest of the claims. Cf., for example, a claim that describes the composition of a formula: *A compound of claim 1 which is 2,3,4,4a,6,7,11b,12,13,13a-decahydro-1H-dibenzo[a,f]quinolizin-13-ol and pharmaceutically acceptable salts thereof.* In consequence, to deal with chemistry patents, it is necessary to apply specific NLP technologies, as e.g., BioNLP techniques.

Our claim linguistic study is limited to English

In this dissertation, we have presented a linguistic study on the claim style that is limited to patents in English. In addition, it would be interesting

to also analyse patents written in other languages such as Chinese, German, Japanese and Korean, as they are the most used languages in patent applications around the world.

The approach is English dependent

To some extent, the RE approach proposed in this thesis may be considered as language independent because we operate with deep syntactic relations, which are language independent. However, other components used along the proposed framework make the complete proposal English dependent. For instance, in order to apply our RE approach to patent claim data, a claim preprocessing module was implemented. This module operates with hard-coded strings written in English, e.g., subordinate and coordinate conjunctions and clause boundary keywords, among others. Furthermore, the preprocessing also relies on supervised machine learning methods which use manually created training data derived from claims written in English. Similarly, other tools and resources used are for English like TreeTagger, Minipar and Wordnet. For instance, Wordnet, which is the basis of our relation classification component, is a lexical database of English. Other Wordnets have also been developed, at least partially, for other languages, which are important in the domain of patents like Finnish, French, German, Japanese and Korean. Despite the fact that these initiatives are promising, it is known that these Wordnets are less developed than the English version. In addition, the mapping of the English Wordnet to other languages has been criticised (Cristea, 2004), basically because of the differences underlying the lexicon of each language. For example, sometimes a synset member in English cannot be translated to another language because the lexeme does not exist in the target language.

8.3 Possible applications

The scope of the application of relation extraction in both patent processing and general language applications is very wide. The output of our system represents the content of a text in terms of tuples, which express the objects mentioned in the text and the relation between those objects. The tuples can serve as input to, for example, the following applications:

- Content visualization

- Ontology learning
- Search of similar documents
- Question answering

Content visualization

Due to the complexity of the patent domain and the huge amount of data, advanced visual techniques are needed to support the analysis of large patent collections and portfolios. In this vein, a drawing or an image that illustrates the composition and function of an invention described in the claims is highly desirable. The RE and generalization approach of this thesis attempts to contribute especially to this task.

The relation generalization task aims to ensure that similar relations are assigned a unique and meaningful relation label. The reduction of the number and types of relations to a limited set is especially useful for visualization because drawings are easier to understand when the set of relations to plot is small (usually, around six different types of relations are recommended by visualization experts). Similarly, the coreference resolution algorithm implemented in this thesis facilitates the reduction of redundancy or duplicates nodes in a drawing; for instance, the noun phrases which are corefered are explicitly marked. Therefore, the nodes with the same coreference identifier can be merged, resulting in a simpler content presentation.

Moreover, the patent structuring algorithm, which marks the document sections such as title, abstract, the dependency between independent claims and dependent claims, etc., provides an output that is especially useful for visualization purposes. For example, the dependency between independent and dependent claims can be used to make flexible presentation set-ups. Thus, based on the dependency claim structure, it would be possible to only visualize independent claims or only independent claims and their direct dependants, among other presentation set-ups.

Ontology learning

An ontology models concepts and relations that are relevant in a particular domain. The process of (semi-)automatically enriching an existing ontology is called *ontology learning*. Ontology learning can also be defined as a

knowledge acquisition task, which usually acquires knowledge from text. In the context of ontology learning from text, the RE extraction component proposed in this thesis may be used for discovering new related pairs of lexical items in a corpus. After that, the lexical items from the new tuples can be mapped to concepts in the ontology and the relations from those tuples can be in instantiated in the ontology.

However, perhaps the most interesting aspect of this thesis related to ontology learning is the labeling of similar relations that go beyond IS-A. An ontology learning component may detect several relations between two concepts. For example, between 'Producer' and 'Product', the learner component may detect relations like *produce*, *manufacture*, *consume*, *advertise*, *sell*, *assemble*, etc. To instantiate those relations in an ontology, it is necessary or at least desirable, to group similar relation tuples and to label the obtained group with an abstract concept. For example, the relations *produce*, *manufacture*, *assemble*, which are semantically similar, can be grouped together and labelled with a common concept that subsumes them. A possible concept for this purpose could be *make*. A common relation label for similar tuples is desirable, on the one hand, because its allows for subsuming similar relations under the same abstract concept and, on the other hand, because abstract terms are more appropriate in an ontology.

The relation generalization approach proposed in this thesis to address these problems is very well suited. Note that the problem of grouping similar relation tuples is similar to the problem of grouping similar verbs, and the problem of labeling new relations in an ontology is similar to the problem of labeling groups of similar relations to be used as arc labels in a conceptual map representation.

Search of similar documents

It is possible to compare the content of documents in terms of their tuple representations (Hachey, 2009). Finding similar documents is at the core of IR systems, as it is assumed that if a document matches a user query, documents that are similar to it are also relevant to the user.

In the patent domain, the search for similar documents is useful in many situations. For example, it may help to avoid possible infringements; to assist an inventor to recognize patentable aspects of his designs, to learn lessons from prior art and to look for inspiration, among others.

Moreover, finding similar patents is useful for patent valuing techniques. Patent valuing is a complex task that aims to predict the economic benefits of a patent. A popular patent valuing methodology is based on the extraction of relevant indicators (age of the patent, length of the description, number of claims, etc.) from a set of patents which are similar to the one that is being valued. In consequence, the quality of the collected similar patents is crucial as the further steps rely upon it.

Question answering

The goal of question answering (QA) is to find answers for natural language questions in a large document collection, e.g., a corpus or the web. Most QA systems consist of three modules: (i) question processing, which determines the answer type and collects keywords, (ii) information retrieval, which retrieves a set of relevant text passages using the keywords, and (iii) answer extraction, which consists in the analysis of the retrieved passages from which the answer is extracted.

Depending on the application task, QA systems can be classified as *closed-domain* and *open-domain*. *Closed-domain* QA systems deal with questions from a specific domain and may also be restricted to a limited set of questions predefined beforehand. RE is at the core of *closed-domain* QA systems because they rely on relation patterns for the extraction of a particular relation from a corpus. Think, for instance, of a QA system that aims to answer frequent questions in a tourism office such as locations of museums. In this context, relation extraction patterns that instantiate a particular relation tuple in a corpus are searched beforehand and questions are answered using the collected data. For this purpose, relation extraction patterns can be formulated in terms of syntactic dependency patterns. The advantage of syntactic dependency patterns in contrast to surface lexico-syntactic patterns or surface strings has already been mentioned in Section 4.1.

Open-domain QA systems deal with questions about any topic and relies on large text collections. In these systems, RE is a fundamental component of the answer extraction module. The aim of this module is to extract answers from the text passages provided by the information retrieval module. The answer extraction is constrained by the relations encountered in the question. Thus, to select a text passage as an answer, the relation or relations extracted from the question should be matched with the text passages. For this task, relation extraction based on dependency parsing is very popular.

Above all, because it captures relations between words regardless of their order and distance in the text.

8.4 Future work

The research presented in this dissertation can be taken further in several directions. In this section, we summarize some possible improvements and extensions of our approach.

Improvement of the current approach

To improve the performance and reduce the complexity of our approach, one option is to use a more powerful syntactic parser in an attempt to dispense with the claim simplification module. Currently, we carry out experiments with (Bohnet, 2009)'s stochastic parser, which demonstrated a very good performance and robustness in the CoNLL'09 competition, and we are confident that it will also perform very well in our application. To be able to parse long claim sentences, we are experimenting with a version of Bohnet's parser, which is based on transitional parsing. Transitional parsing typically has linear complexity while graph-based parsing has a quadratic complexity. This suggests that a transitional parser is more adequate to parse patent claims as it requires less effort to parse long sentences.

Extension of the current approach

As relation tuples are defined on the basis of predicate units and their valency structure, the RE approach can be extended to extract relations denoted by predicative nouns or adjectives. Note that while relations denoted by predicative nouns and adjectives can be extracted from deep syntactic dependency structures using the same strategy we used in the thesis, the classification of similar relations should be rethought.

We carried out some preliminary studies on the semantic classification of adjectives using a Wordnet version linked to SUMO.[1] With a set of adjectives retrieved from the corpus, we built a vector for each adjective using as

[1]SUMO or Suggested Upper Merged Ontology, is an upper ontology, which means that it is limited to generic concepts.

Table 8.1: Examples of clustering and cluster labeling of similar adjectives.

Adjective cluster	Cluster label
parallel, perpendicular, opposite, peripheral, central, lateral, above, concentric, horizontal, vertical, adjacent, axial, coaxial	spatial position
circular, semicircular, cylindrical, hexagonal, orthogonal, triangular, square, cubic, elliptic, hemispherical, polygonal, conical	shape
green, grey, white, blue	color
compact, obtuse, slow, thick, tight	Subjective Assessment Attribute
main, primary, principal	Subjective Assessment Attribute
only, unique, unitary, whole	Subjective Assessment Attribute

features its parent and children nodes in SUMO. Then, we ran a clustering algorithm to assess whether these features were enough to discriminate between semantically different adjectives. Other than that, we evaluated to what extent the name of a common parent node between those adjectives in the same cluster can serve as a cluster label. Consider Table 8.1 for an illustration of the output in this study. In the first three clusters, adjectives are correctly grouped and the labels proposed are valid, while for the rest of the clusters, the grouping is not very accurate and the labels assigned are not adequate. The labels are not adequate, on the one hand, because the labels are the same for three different clusters and, on the other hand, because the label *Subjective Assessment Attribute*, which is a node in the SUMO hierarchy, is meaningless. In SUMO, the *Subjective Assessment Attribute* is the class for attributes (or adjectives) which cannot be assigned to a meaningful class because of the lack of an objective criterion for its classification. Another disadvantage of SUMO is that its hierarchy is relatively flat; in consequence, many attributes are assigned to abstract classes such as *Normative Attribute* and *Psychological Attribute*, which seems not appropriate to label groups of similar adjectives.

Taking the presented study into consideration, we conclude that adjective classification should be considered from another angle. For instance, it would be interesting to investigate whether other lexical resources such as *Cyc Knowledge Server* are more suitable for the cluster labeling task.

For future work, it would also be interesting to apply a state-of-the-art word disambiguation component prior to the relation classification. This is because relation classification without a sense disambiguation component is risky, overall when the relation classification deals with texts from a

general discourse corpus, where ambiguity and denominative variation are more common. Finally, it would be interesting to apply our approach to a large-scale classification of verbs.

Bibliography

E. Agichtein and L. Gravano. Snowball: Extracting relations from large plain-text collections. In *Proceedings of the 5th ACM International Conference on Digital Libraries*, pages 85–94, 2000. 38

B. A. Amernick. *Patent law for the nonlawyer: A guide for the engineer technologist and manager*. Van Nostrand Reinhold, New York, 1991. 10, 14

E. Amigó, J. Gonzalo, J. Artiles, and F. Verdejo. A comparison of extrinsic clustering evaluation metrics based on formal constraints. *Inf. Retr.*, 12(4):461–486, August 2009. ISSN 1386-4564. doi: 10.1007/s10791-008-9066-8. URL http://dx.doi.org/10.1007/s10791-008-9066-8. 118

J. Apresjan, I. Boguslavsky, B. Iomdin, L. Iomdin, A. Sannikov, and V. Sizov. A syntactically and semantically tagged corpus of Russian state of the art and prospects 1. In *Proceedings of LREC. Genova, Italy*, 2006. 32

R. Barzilay and L. Lee. Learning to paraphrase: an unsupervised approach using multiple-sequence alignment. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, NAACL '03, pages 16–23, Stroudsburg, PA, USA, 2003. Association for Computational Linguistics. 47

B. Beamer, S. Bhat, B. Chee, A. Fister, A. Rozovskaya, and R. Girju. Uiuc: A knowledge-rich approach to identifying semantic relations between nominals. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, pages 386–389, Prague, Czech Republic, June 2007. Association for Computational Linguistics. URL http://acl.ldc.upenn.edu/W/W07/W07-2085.pdf. 40

M. Berland and E. Charniak. Finding parts in very large corpora. In

Proceedings of the 37th annual meeting of the Association on Computational Linguistics, pages 57–64, Morristown, NJ, USA, 1999. Association for Computational Linguistics. ISBN 1-55860-609-3. doi: http://dx.doi.org/10.3115/1034678.1034697. 38

S. Blohm and P. Cimiano. Harvesting relations from the web - quantifying the impact of filtering functions. In *Proceedings of the 22nd Conference on Artificial Intelligence (AAAI'07)*, pages 1316–1323, 2007. 38, 106, 107, 109

B. Bohnet. *Textgenerierung durch Transduktion linguistischer Strukturen*. DISKI 298, AKA Berlin, 2006. 34

B. Bohnet. Efficient parsing of syntactic and semantic dependency structures. In *Proceedings of the Conference on Natural Language Learning (CoNLL)*, pages 67–72, Boulder, Colorado, 2009. Association for Computational Linguistics. 136

N. Bouayad-Agha, G. Casamayor, G. Ferraro, S. Mille, V. Vidal, and Leo Wanner. Improving the comprehension of legal documentation: the case of patent claims. In *Proceedings of the 12th International Conference on Artificial Intelligence and Law*, ICAIL '09, pages 78–87, New York, NY, USA, 2009a. ACM. ISBN 978-1-60558-597-0. doi: 10.1145/1568234.1568244. URL http://doi.acm.org/10.1145/1568234.1568244. 62, 130

N. Bouayad-Agha, G. Casamayor, G. Ferraro, and L. Wanner. Simplification of patent claim sentences for their paraphrasing and summarization. In *FLAIRS Conference*, 2009b. 62, 130

C. Brew and S. Schulte im Walde. Spectral clustering for German verbs. In *Empirical Methods in NLP*, pages 117–124, 2002. 47

S. Brin. Extracting patterns and relations from the world wide web. In *WebDB*, pages 172–183, 1998. 38

S. Brügmann and L. Wanner. Advanced document processing techniques overview of the state of the art. *Deliverable 8.1, PATExpert FP6-028166*, 2006. 3, 46

R. C. Bunescu and R. J. Mooney. A shortest path dependency kernel for relation extraction. In *HLT '05: Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 724–731, Morristown, NJ, USA, 2005. Association for Computational Linguistics. doi: http://dx.doi.org/10.3115/1220575.1220666. 38

D. Carmel, H. Roitman, and N. Zwerdling. Enhancing cluster labeling

using wikipedia. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '09, pages 139–146, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-483-6. 50, 60

G. Cascini and P. Rissone. Pat-analyzer: a tool to speed-up patent analysis with triz perspective. *Proceedings of the ETRIA World Conference: TRIZ Future*, 2003. 45, 46

G. Cascini and D. Russo. Computer-aided analysis of patents and search for triz contradictions. *International Journal of Product Development*, 4 (1):52–67, January 2007. URL http://www.metapress.com/content/4txbexhlhba83hww. 55

M. Ciaramita, A. Gangemi, E. Ratsch, J. Saric, and I. Rojas. Unsupervised learning of semantic relations between concepts of a molecular biology ontology. In *Proceedings of International Joint Conference on Artificial Intelligence (IJCAI2005)*, pages 659–664. Edinburgh, UK, 2005. 38

P. Cimiano, M. Hartung, and E. Ratsch. Finding the appropriate generalization level for binary ontological relations extracted from the genia corpus. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC)*, pages 161–169, 2006. 39

M. A. Covington. A fundamental algorithm for dependency parsing. In *Proceedings of the 39th Annual ACM Southeast Conference*, pages 95–102, 2001. 27

Mihăilă C. Forăscu C. Trandabăt D. Husarciuc M. Haja G. Postolache O. Cristea, D. Mapping Princeton Wordnet synsets onto Romanian Wordnet synsets. In *Romanian Journal on Information Science and Technol ogy*, volume 7, 2004. 132

A. Culotta and J. Sorensen. Dependency tree kernels for relation extraction. In *ACL '04: Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, page 423, Morristown, NJ, USA, 2004. Association for Computational Linguistics. doi: http://dx.doi.org/10.3115/1218955.1219009. 39

D. Davidov and A. Rappoport. Unsupervised discovery of generic relationships using pattern clusters and its evaluation by automatically generated sat analogy questions. In *Meeting of the Association for Computational Linguistics*, pages 692–700, 2008. 48

G. Dias, S. Pais, F. Cunha, H. Costa, D. Machado, T. Barbosa, and B. Martins. Hierarchical soft clustering and automatic text summarization for accessing the web on mobile devices for visually impaired people. In

Proceedings of the FLAIRS Conference, pages 231–236, 2009. 50

A. G. Dimitrov, E. G. Dimitrov, and J. P Miller. Neural coding and decoding: Communication channels and quantization. In *Network: Computation in Neural Systems*, pages 441–472, 2001. 117

G. Doddington, A. Mitchell, M. Przybocki, L. Ramshaw, S. Strassel, and R. Weischedel. The automatic content extraction (ace) program - tasks, data, and evaluation. *Proceedings of LREC 2004*, 2004. 39

O. Etzioni, M. Banko, S. Soderland, and D. S. Weld. Open information extraction from the web. *Communication of the ACM*, 51(12):68–74, 2008. ISSN 0001-0782. doi: http://doi.acm.org/10.1145/1409360.1409378. URL http://portal.acm.org/citation.cfm?id=1409378. 38

O. Etzioni, A. Fader, J. Christensen, S. Soderland, and Mausam. Open information extraction: The second generation. In *IJCAI 2011, Proceedings of the 22nd International Joint Conference on Artificial Intelligence, Barcelona, Catalonia, Spain, July 16-22, 2011*, pages 3–10, 2011. 38

C. Fellbaum. *Wordnet: An Electronic Lexical Database*. Bradford Books, 1998. 38, 57

G. Ferraro. Algoritmo supervisado para la detección de unidades discursivas: el caso de las reinvindicaciones de patentes. In *Actas del XXIX Congreso Internacional de AESLA: Empirismo y herramientas analíticas para la Lingüística Aplicada del Siglo XXI*, Universidad de Salamanca, Spain., In Press. 67

W.A. Gale, K.W. Church, and D. Yarowsky. One sense per discourse. In *Proceedings of the Workshop on Speech and Natural Language at the Human Language Technology Conference*, pages 233–237. Association for Computational Linguistics, 1992. 79

R. Girju, A. Badulescu, and D. Moldovan. Automatic discovery of part-whole relations. *Computational Linguistics*, 32(1):83–135, 2006. ISSN 0891-2017. doi: http://dx.doi.org/10.1162/coli.2006.32.1.83. 38

R. Girju, S. Szpakowicz, P. Nakov, P. Turney, V. Nastase, and D. Yuret. Semeval-2007 task 04: Classification of semantic relations between nominals, 2007. 39

C. Giuliano, A. Lavelli, and L. Romano. Relation extraction and the influence of automatic named-entity recognition. *ACM Trans. Speech Lang. Process.*, 5(1):1–26, 2007. ISSN 1550-4875. doi: http://doi.acm.org/10.1145/1322391.1322393. 38, 40

E. J. Glover, K. Tsioutsiouliklis, S. Lawrence, D. M. Pennock, and G. W. Flake. Using web structure for classifying and describing web pages. In

Proceedings of the 11th international conference on World Wide Web, WWW '02, pages 562–569, New York, NY, USA, 2002. ACM. ISBN 1-58113-449-5. 50

B. Hachey. Multi-document summarisation using generic relation extraction. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1 - Volume 1*, EMNLP '09, pages 420–429, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics. ISBN 978-1-932432-59-6. URL http://dl.acm.org/citation.cfm?id=1699510.1699565. 134

T. Hasegawa, S. Sekine, and R. Grishman. Discovering relations among named entities from large corpora. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, ACL '04, Stroudsburg, PA, USA, 2004. Association for Computational Linguistics. 48

V. Hatzivassiloglou and K. R. Mckeown. Towards the automatic identification of adjectival scales: Clustering adjectives according to meaning. In *Proceedings of the 31st Annual Meeting of the ACL*, pages 172–182, 1993. 114

Hays. *Basic Principles and Thechnical Variations in Sentence Structure Determination*. Santa Monica: RAND Corporation (Mathematical Division. P 1934), 1960. 26

Hays. *Dependence Theory: A Formalism and Some Observations*. Language 40 (511-525), 1964. 26

M. A. Hearst. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the 14th International Conference on Computational Linguistics*, pages 539–545, 1992. 38

L. Hubert and P. Arabie. Comparing partitions. *Journal of Classification*, 2(1):193–218–218, 1985. ISSN 0176-4268. 112, 114

L Iordanskaja and I. Mel'čuk. Establishing an Inventory of Surface-Syntactic Relations: Valence-Controlled Surface-Syntactic Dependents of the Verb in French. In Alain Polguère and Igor Mel'čuk, editors, *Dependency in Linguistic Description*, Studies in Language Companion, pages 151–234. John Benjamins Publishing Company, 2009. 26, 32

S. Kahane. Des grammaires formelles pour définir une correspondance. In *TALN'00*, 2000. 34

S. Kahane. *An International Handbook of Contemporary Research - The Meaning-Text Theory*, volume vol. 1. De Gruyter, 2003. 28, 29, 32

S. Kahane and A. Polguére. Formal foundations of lexical functions. In *Workshop on Collocation, ACL*, 2001. 30

G. Karypis. *CLUTO A Clustering Toolkit*. University of Minnesota, Department of Computer Science, 2003. 80

D. Klein and C. Manning. Accurate unlexicalized parsing. *Proceedings of the 41st meeting of the Association for Computational Linguistics*, 2003a. 45, 75

D. Klein and C. D. Manning. Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting of the Assocition of Computational Linguistics*, pages 423–430, 2003b. 45

A. Korhonen. *Subcategorization Acquisition*. PhD thesis, University of Cambridge, 2002. 117

A. Korhonen, Y. Krymolowski, and Z. Marx. Clustering polysemic subcategorization frame distributions semantically. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 64–71, 2003. 117

A. Korhonen, Y. Krymolowski, and N. Collier. Automatic classification of verbs in biomedical texts. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, ACL-44, pages 345–352, Stroudsburg, PA, USA, 2006. Association for Computational Linguistics. 47, 116, 117, 119

L. Lee. Measures of distributional similarity. In *Meeting of the Association for Computational Linguistics*, 1999. 48

B. Levin. *English Verb Classes and Alternations: A Preliminary Investigation*. Chicago University Press, Chicago, 1993. 47

D. Lin. Dependency-based evaluation of minipar. In *Proc. Workshop on the Evaluation of Parsing Systems*, Granada, pages 317–329, 1998. 101

D. Lin and P. Pantel. Dirt: Discovery of inference rules from text. In *Knowledge Discovery and Data Mining*, pages 323–328, 2001. 48

C.D. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Retrieval*. Cambridge University Press, Cambridge, 2008. 50, 53, 88, 89, 112

R. McDonald, F. Pereira, S. Kulick, S. Winters, Y. Jin, and P. White. Simple algorithms for complex relation extraction with applications to biomedical ie. In *Proceedings of the 43nd Annual Meeting of the Association for Computational Linguistics (ACL-05*, pages 491–498, 2005. 40, 42

I. A. Mel'čuk. *Dependency Syntax: Theory and Practice*. State University of New York Press, Albany, 1988. 26, 28, 31, 56

I. A. Mel'čuk and N. V. Pertsov. *Surface syntax of English: A formal model within the Meaning-Text framework*. Amsterdam: John Benjamins, 1987. 32

K. Mika. Findex: Search Result Categories Help Users when Document Ranking Fails. In *CHI '05: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 131–140, New York, NY, USA, 2005. ACM. ISBN 1-58113-998-5. doi: 10.1145/1054972.1054991. URL http://dx.doi.org/10.1145/1054972.1054991. 51, 52

J. Milićević. A short guide to the Meaning-Text Linguistic Theory. *Journal of Koralex*, Vol. 8:187–233, 2006. 30

S. Mille and L. Wanner. Making text resources accessible to the reader: The case of patent claims. In *Proceedings of LREC*, 2008a. 75

S. Mille and L. Wanner. Multilingual summarization in practice: the case of patent claims. In *12th Annual Conference of the European Association for Machine Translation*, pages 120–129, 2008b. 76

S. Mille, A. Burga, V. Vidal, and L. Wanner. Creating an MTT treebank of Spanish. In *Proceedings of MTT Conference 2009, Montreal, Canada.*, 2009. 32

S. Mille, A. Burga, and L. Wanner. Looking behind the scenes of syntactic dependency corpus annotation: Towards a motivated annotation schema of surface-syntax in Spanish. In *Proceedings of DepLing 2011, Barcelona, Spain.*, 2011. 32

J. Nivre. Dependency grammar and dependency parsing. Technical report, Växjö University: School of Mathematics and Systems Engineering, 2005. 27

N. Ogata and N. Collier. Ontology express: Statistical and non-monotonic learning of domain ontologies from text. In *Proceedings of the Workshop on Ontology Learning and Population held at the 16th European Conference on Artificial Intelligence (ECAI'2004) Valencia, Spain*, pages 43–48, August 22–24 2004. 38

S. Osinski and D. Weiss. A concept-driven algorithm for clustering search results. *IEEE Intelligent Systems*, 20:48–54, May 2005. ISSN 1541-1672. 51

P. Pantel and D. Ravichandran. Automatically labeling semantic classes. In *HLT - NAACL*, pages 321–328, 2004. 50, 52

P. Parapatics and M. Dittenbach. Patent claim decomposition for improved information extraction. In *Proceeding of the 2nd international workshop on Patent information retrieval*, PaIR '09, pages 33–36, New York, NY,

USA, 2009. ACM. ISBN 978-1-60558-809-4. 44, 46, 47

E. Pianta. Basic content distillery and meta-information acquisition techniques. *Deliverable 3.1, PATExpert FP6-028166*, 2007. 45, 46

D. Pressman. *Patent It Yourself.* Nolo, Berkeley, CA., 2006. ISBN 978-1-4133-0516-6. 10, 12

R. Quirk, S. Greenbaum, G. Leech, and J. Svartvik. *A comprehensive grammar of the English language.* Longman, 1985. 17

D. V. Radack. Reading and undestanding patent claims. In *JOM*, page 69, 1995. 24

W. M. Rand. Objective Criteria for the Evaluation of Clustering Methods. *Journal of the American Statistical Association*, 66(336):846–850, 1971. ISSN 01621459. 112

F. Reichartz, H. Korte, and G. Paass. Composite kernels for relation extraction. In *ACL-IJCNLP '09: Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, pages 365–368, Morristown, NJ, USA, 2009. Association for Computational Linguistics. 39, 40

J. Robinson. A dependency based transformational grammar. *Actes du X Congrs international des linguistes*, pages 807–813, 1970. 26

Frank Rosemblatt. A probabilistic model for information storage and organization in the brain. *Psychological. Rev., 68:386-407*, 1958. 40

B. Rosenfeld and R. Feldman. Ures: an unsupervised web relation extraction system. In *Proceedings of the COLING/ACL on Main conference poster sessions*, pages 667–674, Morristown, NJ, USA, 2006. Association for Computational Linguistics. 38

G. Salton. Abstracts chosen from recent issues of journals in the retrieval area. *SIGIR Forum*, 23(3-4):123–138, 1989. 48

G. Salton and C. Buckley. On the use of spreading activation methods in automatic information retrieval. In *SIGIR*, pages 147–160, 1988. 48

G. Salton and M. J. McGill. *Introduction to Modern Information Retrieval.* McGraw-Hill, Inc., New York, NY, USA, 1986. ISBN 0070544840. 48

H. Schmid. Probabilistic part-of-speech tagging using decisions trees. *Proceedings of the Conference on New Methods in Language Processing*, pages 44–49, 1994. 62, 65

S. Schulte im Walde. Experiments on the automatic induction of German semantic verb classes. *Computational Linguistics*, 32(2):159–194, 2006. 47, 109, 111

S. Sekine. Automatic paraphrase discovery based on context and keywords

between NE pairs. In *International Workshop on Paraphrase*, 2005. 41, 47, 48

J. G. Sheldon. *How to Write a Patent Application*. Practising Law Institute, 1992. 11

P. Siddharth and R. Ellen. Learning domain-specific information extraction patterns from the web. In *IEBeyondDoc '06: Proceedings of the Workshop on Information Extraction Beyond The Document*, pages 66–73, Morristown, NJ, USA, 2006. Association for Computational Linguistics. ISBN 1-932432-74-4. 38

R. Snow, D. Jurafsky, and A. Y. Ng. Learning syntactic patterns for automatic hypernym discovery. In Lawrence K. Saul, Yair Weiss, and Léon Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 1297–1304. MIT Press, Cambridge, MA, 2005. 38

L. Sun and A. Korhonen. Improving verb clustering with automatically acquired selectional preferences. In *Empirical Methods in NLP*, pages 638–647, 2009. 47

L. Sun, A. Korhonen, and Y. Krymolowski. Verb class discovery from-rich syntactic data. In *9th International Conference on Intelligent Text Processing and Computational Linguistics*, 2008. 47

M. Surdeanu and M. Ciaramita. Robust information extraction with perceptrons. In *Proceedings of the NIST 2007 Automatic Content Extraction Workshop (ACE07)*, 2007. URL `http://www.surdeanu.name/mihai/papers/ace07a.pdf`. 39, 40

L. Tesnière. *Eléments de syntaxe structurale*. Klincksieck, Paris, 1966. 26

N. Tishby, F.C. Pereira, and W. Bialek. The information bottleneck method. In *Proceedings of the 37th Annual Allerton Conference on Communication, Control and Computing*, pages 368–377, 1999. 117

E. F. Tjong and Kim Sang. Memory-based clause identification. In *Proceedings of the 2001 workshop on Computational Natural Language Learning - Volume 7*, ConLL '01, Stroudsburg, PA, USA, 2001. Association for Computational Linguistics. 104

P. Treeratpituk and J. Callan. Automatically labeling hierarchical clusters. In *Proceedings of the 2006 international conference on Digital government research*, dg.o '06, pages 167–176, New York, NY, USA, 2006. ACM. 50, 51

P. D. Turney and M. L. Littman. Corpus-based learning of analogies and semantic relations. *Machine Learning*, 60(1-3):251–278, 2005. 48

USPTO. Transitional phrases. Technical report, USPTO: United States

Patent and Trademark Office, August 2001. 11

A. Vlachos, A. Korhonen, and Z Ghahramani. Unsupervised and constrained dirichlet process mixture models for verb clustering. In *Workshop on Geometrical Models of Natural Language Semantics*, pages 74–82, 2009. 47

T. Yamaguchi. Acquiring conceptual relationships from domain-specific texts. In *Proceedings of the Second Workshop on Ontology Learning OL'2001*, pages 0–2, 2001. 39

D. Yang and D. Powers. Measuring semantic similarity in the taxonomy of wordnet. In *ACSC '05: Proceedings of the Twenty-eighth Australasian conference on Computer Science*, pages 315–322, Darlinghurst, Australia, Australia, 2005. Australian Computer Society, Inc. ISBN 1-920-68220-1. 86

D. Yang and D. Powers. Verb Similarity on the Taxonomy of Wordnet. In *Proceedings of the 3rd International Wordnet Conference (GWC-06), Jeju Island, Korea*, pages 121–128, 2006. 48, 57

Shih-Yao Yang, Szu-Yin Lin, Shih-Neng Lin, Chi-Feng Lee, Shian-Luen Cheng, and Von-Wun Soo. Automatic extraction of semantic relations from patent claims. *International Journal of Electronic Business Managemen*, 6(1):45–54, 2007. 42, 46, 47

A. Yates and O. Etzioni. Unsupervised methods for determining object and relation synonyms on the web. *Journal of Artificial Intelligence Research*, 2009. 48

O. Zamir and O. Etzioni. Grouper: A dynamic clustering interface to web search results. pages 1361–1374, 1999. 51

Y. Zhao and G. Karypis. *Criterion Functions for Document Clustering: Experiments and Analysis*. 2002. 80

Appendix

Appendix A

Examples of the verbal clustering gold standard

The verbal clustering gold standard is composed by 193 verbs grouped into 54 classes as shown in Table A.1. Verbs that are semantically similar such as *remove* and *erase* are grouped in the same cluster.

Table A.1: Gold standard clusters.

Cluster number	Cluster members
1	record , file
2	remove, delete, erase, exclude
3	compress, trim, reduce, minimize, cut
4	desire, require
5	come, approach, face
6	select, take
7	repeat, reproduce, disperse, propagate
8	differentiate, distinguish, discriminate, separate
9	characterize, define, delimit, specify
10	adhere, bond
11	encode, encrypt, inscribe, write, compose
12	associate, connect, weld, join, link, relate, couple
13	calculate, process
14	conduct, direct, guide, lead

Continued on next page

Table A.1 – *continued from previous page*

Cluster number	Cluster members
15	carry, execute, perform
16	tighten, fasten, secure, fix
17	deposit, locate, situate
18	bound, limit , restrain, inhibit
19	address, cover, handle, coat
20	adjust, arrange , place, position, set, base
21	block, stop
22	cause, construct, create, do, effect, make, produce, constitute, embed, establish, shape, form, generate
23	engage, lock
24	operate, control
25	curve, wind
26	decrease, descend, fall, tapper
27	display, exhibit, expose, show
28	obtain, receive
29	accord, allow, let, permit
30	amplify, expand, open
31	displace, move, travel, lift, rise, surface
32	impress, draw
33	converge, meet, satisfy
34	cross, intersect
35	track, trail
36	follow, trace
37	send, transmit, pass
38	fit, match
39	apply, feed, give, use, supply, render, provide
40	disengage, release
41	become, convert, turn
42	add, append
43	hold, maintain, retain, support, prevent
44	beam, irradiate, radiate, mirror , reflect, emit
45	enter, insert, interpose, introduce, enclose
46	accommodate, adapt, transcribe
47	comprise, contain, have, include
48	extract, pull-out
49	focus, indicate, orient, point, signal
50	denote, designate, denominate

Continued on next page

Table A.1 – *continued from previous page*

Cluster number	Cluster members
51	read, scan
52	bias, predetermine
53	switch, change, vary
54	quantify, measure

Appendix B

Examples of the verbal cluster labeling gold standard

The verbal cluster labeling gold standard is composed by the 54 classes from the clustering gold standard. For each of the 54 classes, a gold standard label that better describes the content of a given cluster has been assigned, as shown in Table B.1.

Table B.1: Automatic clusters.

Cluster number	Cluster members	Label
1	record , file	register
2	remove, delete, erase, exclude	remove
3	compress, trim, reduce, minimize, cut	reduce
4	desire, require	desire
5	come, approach, face	approach
6	select, take	select
7	repeat, reproduce, disperse, propagate	disperse
8	differentiate, distinguish, discriminate, separate	separate
9	characterize, define, delimit, specify	define
10	adhere, bond	relate
11	encode, encrypt, inscribe, write, compose	draw upon
12	associate, connect, weld, join, link, relate, couple	connect
13	calculate, process	process
14	conduct, direct, guide, lead	conduct
15	carry, execute, perform	perform

Continued on next page

Table B.1 – *continued from previous page*

Cluster number	Cluster members	Label
16	tighten, fasten, secure, fix	fix
17	deposit, locate, situate	locate
18	bound, limit , restrain, inhibit	limit
19	address, cover, handle, coat	handle
20	adjust, arrange , place, position, set, base	set
21	block, stop	stop
22	cause, construct, create, do, effect, make, produce, constitute, embed, establish, shape, form, generate	cause
23	engage, lock	lock
24	operate, control	control
25	curve, wind	wind
26	decrease, descend, fall, tapper	decrease
27	display, exhibit, expose, show	display
28	obtain, receive	get
29	accord, allow, let, permit	let
30	amplify, expand, open	expand
31	displace, move, travel, lift, rise, surface	move
32	impress, draw	write
33	converge, meet, satisfy	meet
34	cross, intersect	cross
35	track, trail	follow
36	follow, trace	trace
37	send, transmit, pass	transmit
38	fit, match	correspond
39	apply, feed, give, use, supply, render, provide	produce
40	disengage, release	free
41	become, convert, turn	change
42	add, append	provide
43	hold, maintain, retain, support, prevent	keep
44	beam, irradiate, radiate, mirror , reflect, emit	emit
45	enter, insert, interpose, introduce, enclose	insert
46	accommodate, adapt, transcribe	adapt
47	comprise, contain, have, include	contain
48	extract, pull-out	extract
49	focus, indicate, orient, point, signal	point
50	denote, designate, denominate	determine
51	read, scan	interpret
52	bias, predetermine	determine
53	switch, change, vary	change
54	quantify, measure	quantify

Appendix C

Examples of the automatic clusters

The set of 193 verbs from the clustering gold standard has been automatically clustered based on our approach presented in Chapter 6. The results are presented in Table C.1.

Table C.1: Automatic clusters.

Cluster number	Cluster members
1	discriminate
2	repeat
3	inhibit
4	exclude
5	compress
6	focus
7	interpose
8	process
9	disperse, propagate
10	apply, use
11	adhere, bond
12	differentiate, distinguish, separate
13	address, cover, handle
14	encode, encrypt, inscribe
15	extract, pull-out
16	form, shape

Continued on next page

Table C.1 – *continued from previous page*

Cluster number	Cluster members
17	indicate, signal
18	compose, write
19	amplify, expand
20	define, delimit, specify
21	deposit, fix, situate
22	measure, quantify
23	enter, insert, introduce
24	allow, let, permit
25	engage, lock, operate
26	coat, surface
27	bias, predetermine
28	delete, erase
29	associate, connect, join, link, relate
30	couple, fit, match
31	curve, cut, reduce, trim
32	bound, control, limit, restrain
33	base, constitute, establish
34	lift, rise, wind
35	decrease, descend, fall
36	locate, place, position, set
37	feed, generate, give, render
38	beam, radiate, send, transmit
39	draw, follow, trace
40	orient, point, taper
41	fasten, secure, tighten
42	be, characterize, exist
43	arrange, do, execute, perform
44	add, append, provide, supply
45	read, record, scan, show
46	calculate, conduct, direct, guide, lead, pass
47	cause, have, obtain, receive
48	remove, require, select, take
49	approach, converge, meet, satisfy
50	block, comprise, contain, stop
51	cross, intersect, track, trail
52	display, exhibit, expose
53	displace, impress, move, travel
54	change, convert, switch, vary
55	become, come, release, turn
56	accommodate, adapt, adjust, transcribe
57	carry, hold, maintain, retain, support
58	construct, create, effect, make, produce

Continued on next page

Table C.1 – *continued from previous page*

Cluster number	Cluster members
59	denominate, denote, designate
60	accord, desire, disengage, embed, emit, enclose, face, file, include, irradiate, minimize, mirror, open, prevent, reflect, reproduce, weld

This thesis also includes two appendices on CD-ROM with the following content:

Appendix D: Gold standard annotations

D.1. Claim segmentation gold standard files

D.2. Claim structuring gold standard files

D.3. Coreference gold standard files

Appendix E: Step by step processing examples

E.0. a patent claim in raw text

E.1. output of the patent structuring module

E.2. output of the linguistic preprocessing module

E.3. output of the claim segmentation module

E.4. output of the coreference resolution module

E.5. output of the claim tree structure module

E.6. output of the Minipar parser

E.7. output of the surface dependency parsing

E.8. output of the deep dependency parsing

E.9. output of the tuple extraction module