NLP-based Extraction of Modificatory Provisions Semantics

Alessandro Mazzei
Dipartimento di Informatica,
Università di Torino
Corso Svizzera 185
10149, Torino, ITALY
mazzei@di.unito.it

Daniele P. Radicioni
Dipartimento di Informatica,
Università di Torino
Corso Svizzera 185
10149, Torino, ITALY
radicion@di.unito.it

Raffaella Brighi
CIRSFID,
Università di Bologna
Via Galliera, 3
40121, Bologna, Italy
raffaella.brighi@unibo.it

ABSTRACT

In this paper we illustrare a research based on NLP techniques aimed at automatically annotate modificatory provisions. We propose an approach which pairs deep syntactic parsing with rule-based shallow semantic analysis relying on a fine-grained taxonomy of modificatory provisions. The implemented system is evaluated on a large dataset hand-crafted by legal experts; the results are discussed and future directions of the research outlined.

Categories and Subject Descriptors

I.7 [**Document and Text Processing**]: Document and text editing

General Terms

Documentation, Experimentation, Languages

Keywords

Legal documents semantics, natural language processing, automatic extraction, syntactic approach.

1. INTRODUCTION

Intelligent indexing, querying, searching, filtering, retrieving and annotating the ever increasing amount of text documents is a major challenge in the field of Artificial Intelligence. Information extraction, i.e. the field that investigates the automatic extraction of salient information from texts, devised a set of tools and techniques to face this challenge [2]. However, one major obstacle to the automatic managing of legal documents is the natural "natural language barrier", i.e. the translation of a sentence into some form of semantic interpretation [13]. Natural Language Processing (NLP) approaches language complexity by considering several layers of analysis. The most critical and complex layers are concerned with the syntactic and semantic analyses of a sentence. Syntactic and semantic analyses of a sentence consist of individuating syntactic and semantic relations: such relations are often grouped into a single structure, usually a tree for the syntactic analysis. How much *accurate* do need to be these analyses? From a theoretical perspective only "deep" analyses are able to capture the full meaning conveyed by a sentence. From a practical perspective, the answer depends on the specific problem that we need to solve, and "shallow" approaches can be well suited.

In the legal domain, some approaches have been proposed to build systems for automatically identifying and classifying structural portions of legal documents and their intra- and inter-references [6, 14]. Other researches are being carried out to produce semantic analysis [16, 17, 13]. Such interest is witnessed by various initiatives at the national and international levels, that have established XML standards for describing legal sources and schemas to identify legal documents [12]. Since the annotation process is expensive and error-prone, such efforts will not be really useful unless are available tools to extract in automatic (or semi-automatic) fashion the *structural* and *semantic* data from legal texts. For instance, on the basis the Italian standard NormeInRete (NIR)[1] allows using a text editor to mark up in an automated or semi-supervised fashion structural partition and normative references. In contrast, the task of automatic annotation of semantic data is still an open issue.

In this paper we describe a NLP system that combines deep syntactic analysis and shallow semantic interpretation of natural language in order to enhance the NIR annotation with semantic meta-data.[2] The system uses the Turin University Parser (TUP, [11, 10]) to build the deep syntactic structure of the sentences, and a rule-based semantic interpreter to fill a frame representing the shallow semantic content of the sentence. From a theoretical point of view, this research shows that the combination of deep syntax and shallow semantics is well suited in specialized domains, such as the legal texts containing *modificatory provisions* (see below), in which the language is more controlled. This allows using a specialized legal and linguistic background knowledge, that has been formalized as an *ad-hoc* taxonomy. From a practical perspective, this research provides human annotators with a tool that can greatly speed-up the annotation of semantic meta-data in normative documents.

In this paper we single out a subset of all possible semantic metadata annotations, and namely the annotation of *modificatory provisions*. A modificatory provision is a change made to one or more clauses within a text (its articles, paragraphs, etc.), or to the entire text along with its annexes (repeal of an entire law), or to the relations that hold among the constituent provisions of a legal system (as when a decree-law is made into law). Modificatory provisions

ICAIL-2009 Barcelona, Spain

[1] `http://www.normeinrete.it/`

[2] This contribution is the result of the cooperation between the Modelling Legal Informatics Resource Group of CIRSFID (University of Bologna) and the NLP Group, Department of Computer Science (University of Turin).

are particularly relevant, since they affect the whole normative system. It should be considered, in this regard, that a lavish use of normative modifications tends to undermine the certainty of the law, so that the changes are sometimes fragmentary and incoherent, making it even more difficult to clearly understand what is the law, or which one of several versions of a provision counts as law.

The work is structured as follows. We introduce the representation adopted (Section 2) and a classification of the various sorts of modifications (Section 2.2). We then illustrate the overall approach (Section 3): after describing the preprocessing step, in which the relevant portions of the provisions are identified (Section 3.1) and the syntactic parsing step (Section 3.2), we describe the process of normative modification extraction (Section 3.3). We then report the result of the experimentation, and discuss them, subsequently outlining both the limitations and the strengths of the implemented system (Section 4). After surveying on the related works (Section 5), we conclude and point out future directions for the present research (Section 6).

2. DOMAIN REPRESENTATION

This work relies on the *NormeInRete* standard (or NIR)[3] for Italian Legal Text. NIR standard defines some structural elements that are used to mark up the main partitions of a text of law, as well as its atomic parts (such as articles, paragraphs, subparagraphs, and lettered and numbered items) and any non-structured text fragment.

2.1 NIR Modificatory Provisions

A provision can be qualified through a specially defined space called <*meta*>, in which a *URN* connects the element expressing a qualification with the textual element referred to (be this an atomic element, or a text string). Modificatory provisions are presently investigated.

We report here the modificatory provision model and definitions, presented in [15], on which our research is based. In a *modificatory clause* we can individuate:

- *ActiveNorm*: provision that states the normative modification;
- *PassiveNorm*: provision that is affected by the modification;
- *Action*: action produced by the active provision on the passive norm;
- *Times*: interval of enter in force of the modificatory provision and the interval of efficacy;
- *Content*: the part of the speech that models the old text to replace or repeal in the modified provision, as well as the new text is inserted in the destination;
- *Purview*: a part used to describe a modification, as by specifying any exception, extensions, or authorized interpretations;
- *Space*: a function used to specify a geographical area to which the modification applies;
- *Conditions*: where a modification is an effect dependent on an event, a geographic space, or a class (or domain) of application.

The NormeInRete (or NIR) standard includes in its Document Type Definitions [1] a part dedicated to modifications to implement this model in XML. Figure 1 illustrates how a non-qualified provision can be enriched with semantic metadata (bold formatted) by marking it up in XML through NormeInRete.

Semantic metadata are linked to structural elements by an URN to assert the kind of modification (action), the active and passive norms, and other sub-elements describing the action. Several classes are used to qualify the behavior of modificatory provisions: these classes are identified by the namespace *dsp:*. Every class of modificatory provisions is modeled as well by a number of sub-elements that further specify it.

In the example in Figure 1, the tag *dsp:substitution*, linked to the text of modificatory provision specifies the Action; the tag *dsp:norma*, linked to the structural element *rif* (normative reference) specifies the Passive Norm; the tag *novella*, linked to the structural element *virgolette* (quoted text) specifies that the Quoted Text should be added in the Active Norm; the tag *novellando*, linked to structural element *virgolette* specifies that the quoted text should be deleted in the Active Norm.

2.2 A Taxonomy for Modifications

The detailed taxonomy taken as reference for the NLP analysis, is presented with full details in [7]. We report here the basic five categories definition:

1. A change made to the provision text or form (an *integration*, *replacement*, *deletion*, *relocation*) or to the provision meaning (an *interpretation* or *variation of meaning* or a *modification of clauses*);
2. A change made to the range of a provision (an *extension* of its subject matter or range of application or a provision stating a *derogation* to the same);
3. A change made to the *temporal parameters* of the norm (the time of its entry into force, and the time when it becomes effective);
4. A change made to the *legal status* of the norm within the legal system (a decree-law that is made into law, an international treaty that is transposed into domestic law);
5. A change made to the *powers* conferred under a norm within the legal system (examples being a *legge delega*, used by Parliaments to entrust the government with issuing a legislative decree under which certain public laws may be passed; or a legislative decree entrusting a ministry to deregulate a certain subject matter within its competence; or again a EU directive transposed into domestic law).

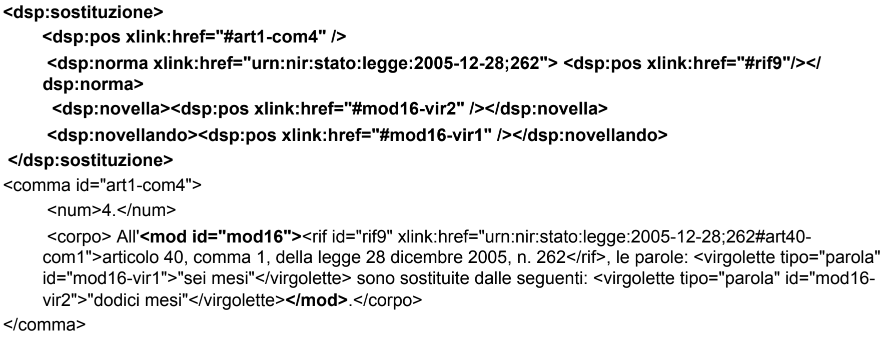
Modificatory Provision Attributes

Furthermore, thanks to the regularity of the language used in active modificatory provisions, we have individuated and encoded some frequent verbs. Such verbs are often paired to other recurrent elements that specify the relevant modificatory action: *Date*, *Quoted text*, *Position*, *Condition*.

Date. A date can express either the time a modification is applied (effective immediately or at some time in the future), or it can modify a term, or signal the end/beginning of a temporal modification.

Quoted text (Quotation marks). Text enclosed within quotation marks can be used to define a concept. Moreover, it can be used in a modificatory clause as text to be inserted into the passive or target document to be modified (the intervening string is called *novella*).

[3] http://www.normeinrete.it/



```
<dsp:sostituzione>
     <dsp:pos xlink:href="#art1-com4" />
      <dsp:norma xlink:href="urn:nir:stato:legge:2005-12-28;262"> <dsp:pos xlink:href="#rif9"/></
     dsp:norma>
       <dsp:novella><dsp:pos xlink:href="#mod16-vir2" /></dsp:novella>
      <dsp:novellando><dsp:pos xlink:href="#mod16-vir1" /></dsp:novellando>
 </dsp:sostituzione>
<comma id="art1-com4">
      <num>4.</num>
      <corpo> All'<mod id="mod16"><rif id="rif9" xlink:href="urn:nir:stato:legge:2005-12-28;262#art40-
     com1">articolo 40, comma 1, della legge 28 dicembre 2005, n. 262</rif>, le parole: <virgolette tipo="parola"
     id="mod16-vir1">"sei mesi"</virgolette> sono sostituite dalle seguenti: <virgolette tipo="parola" id="mod16-
     vir2">"dodici mesi"</virgolette></mod>.</corpo>
</comma>
```

Figure 1: A substitution provision with structural and semantic markup for the Italian phrase *All'articolo 40, comma 1, della legge 28 dicembre 2005, n. 262, le parole "sei mesi" sono sostituite dalle seguenti "dodici mesi" (At the article 40, comma 1 of the law December 28, 2005 number 262, the words "six monts" are to be substituted by the following "twelve monts").*

Finally, it can be used to individuate the text being replaced or deleted: in this case, the string will be said to be a *novellando*).

Position. The position is expressed by function words such as 'before', 'after', 'between', 'from', and 'to', followed by a quoted string or atomic document partition (paragraph, line, index, title): the position denotes the point where a modification occurs in the passive or target text.

Condition. The condition expresses the constraints to the modification: conditions are usually related to an event, to a legal form, or to a place.

Such information is exploited in the analysis process described in Section 3.

3. ANNOTATING MODIFICATORY PROVISIONS

To annotate modifications with meta-information we have devised a three-step process. In the first step we retrieve the possible location of a modificatory provision within the document, and we simplify the input sentences, so to maintain only text portions that convey relevant information. In the second step we perform the syntactic analysis of the retrieved sentences; in the third step we semantically annotate the retrieved provisions by using its syntactic structure and the modificatory provisions taxonomy introduced above. In the following we describe these three processes.

3.1 Retrieving Modificatory Provisions

Although legal documents can be very large, often only small fragments contain semantically meaningful information with respect to the information extraction task. The localization of the information in the document is not an easy task that deeply affects the performance of the information extraction system [2]. We automaticaly pre-process XML documents that are compliant with the NIR DTD,[4] and we use such XML structure to prune irrelevant information. In this work we assume that all possible modificatory provisions are enclosed among the tags[5] ***<corpo>**CORPO-TEXT**</corpo>***[1]. Let us consider, e.g., the modificatory provision contained in: *All'articolo 40, comma 1, della legge 28 dicembre 2005, n. 262, le parole "sei mesi" sono sostituite dalle seguenti "dodici mesi"* (rough translation: *At the article 40, comma 1 of law December 28th, 2005 number 262, the words "six months" are substituted by the following "twelve months"*, Figure 2).

```
<corpo>
  All'
  <rif id="rif9" xlink:href="urn:nir:stato:legge:2005-12-28;262art40-com1">
    articolo 40, comma 1, della legge 28 dicembre 2005, n. 262
  </rif>
  , le parole
  <virgolette tipo="parola" id="mod16-vir1">
    "sei mesi"
  </virgolette>
  sono sostituite dalle seguenti
  <virgolette tipo="parola" id="mod16-vir2">
    "dodici mesi"
  </virgolette>
  .
</corpo>
```

Figure 2: Fragment of a XML NIR document tagged as a *<corpo>*.

In order to simplify the work of the syntactic parser, we perform two additional rewritings on the *CORPO-TEXT*, that make use of the NIR XML annotation of the document. We search for the tags *<rif id="id-rif">RIF-TEXT</rif>*[6] that enclose a text fragment denoting a reference to a particular fragment of a legal document, and we replace *RIF-TEXT* with the corresponding *id-rif*. Similarly, we search for the tags *<virgolette id="id-vir">VIR-TEXT</virgolette>*[7] that enclose a text fragment reporting sequence of words of a legal document (to a text), and we replace *VIR-TEXT* with the corresponding *id-vir*. These two transformations shorten the text with no loss of information, since the *id-rif* and the *id-vir* exactly refer to the text fragments denoted by *RIF-TEXT* and *VIR-TEXT*. By applying the pre-processing step to the text of Figure 2, we obtain the sentence *All'RIF9, le parole VIR1 sono sostituite dalle seguenti VIR2.* (*At RIF9, the words VIR1 are substituted by the following VIR2*).

Furthermore, the structure of the considered <comma> can be rather complex: let us consider the following set of modifications. *A decorrere dalla data di entrata in vigore del presente regolamento sono abrogati:a) la RIF135; b) il RIF137; c) RIF138* (*From the date of entry in force of current regulation, a) the RIF135; b) the RIF137; c) RIF138 are repealed*). The XML encoding along with

[4] http://www.normeinrete.it/sito_area3-ap_stan_rappresentazione_xml.htm.

[5] *Corpo* is the Italian word for *body*

[6] *Rif* stands for *riferimento*, tat is Italian word for *reference*.

[7] *Virgolette* is the Italian word for *quotes*.

```
<comma id="art46-com1">
    <num>1.</num>
    <alinea> A decorrere dalla data di entrata in vigore del presente regolamento sono abrogati:</alinea>
    <el id="art46-com1-let1">
        <num>a)</num>
        <corpo> la <rif id="rif135" xlink:href="urn:nir:stato:legge:1939-02-02;374">legge 2 febbraio 1939, n.
374</rif>;</corpo>
    </el>
    <el id="art46-com1-let2">
        <num>b)</num>
        <corpo> il regolamento di cui al <rif id="rif137"
xlink:href="urn:nir:stato:regio.decreto:1940-12-12;2052:regolamento">regio decreto 12 dicembre 1940, n.
2052</rif>;</corpo>
    </el>
    <el id="art46-com1-let3">
        <num>c)</num>
        <corpo> l'<rif id="rif138"
xlink:href="urn:nir:stato:decreto.legislativo.luogotenenziale:1945-03-01;82#art23">articolo 23 del decreto
legislativo luogotenenziale 1&deg; marzo 1945, n. 82</rif>.</corpo>
    </el>
</comma>
```

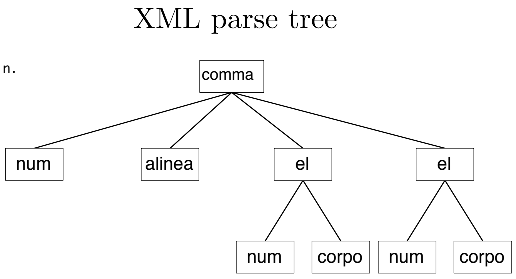


Figure 3: In the structure of a *comma* various nesting levels can be present.

the XML tree are presented in Figure 3. The corpo can be a *nominal phrase* that lacks of the verb, that could instead be placed in the alinea. In this case the same verb can be the *head* of multiple corpi. Therefore, regardless the XML structure, that can include further levels such as el (*element letters*), en (*element numbers*) and *numbers* recursively nested, we keep track of the text contained in the tag alinea that plays the role of *title* for the normative modifications contained in the same comma.

3.2 Syntactic Analysis

The TUP is a rule-based parser that returns the syntactic structure of sentences in the dependency format. Constituency formalisms represent syntactic relations by grouping contiguous words in nested structures, i.e. the constituents (or phrases). In contrast, dependency formalisms represent syntactic relations by connecting a dominant word the *head* (e.g., the verb) and a dominated non-necessary contiguous word, the *the dependent* (e.g., the noun). The connection between these two words is usually represented by using labeled directed edges (e.g., *subject-relation*): the collection of all dependency relations of a sentence form a tree, rooted in the main verb. A dependent is called *argument* of the head if it is obligatory for the correctness of the sentence, or alternatively is called *modifier* if it is not obligatory. Constituents have been customarily used in languages with strict word order language, as English, whilst dependents are more common in freer word order languages, as Italian [11].

Dependency format can encode shallow as well as deep syntactic analysis, depending on how fine grained are the labels marking the head-dependents edges. For instance, a shallow dependency format will have just few labels neglecting complex syntactic phenomena as coordination: the TUP is based on 27 principal label.

Before starting the real syntactic analysis, the TUP performs two preliminary steps, i.e. the *morphological analysis* and the *part of speech tagging*, that are necessary to recover the *lemma* (the normal form) and the part of speech TAG (e.g. verb, noun, adjective, etc.) of the words. At this point the TUP processes the sentence in three phases: chunking, coordination and verbal subcategorization.

Chunking

The chunks produced by the TUP are complex (in the sense that nested chunks are allowed) and are built by a procedure that applies chunking rules to larger and larger chunks, using a predefined sequence of chunking levels. For instance, adverbial chunks (e.g. *[very often]$_{ADV}$*) are built before adjectival chunks (e.g. *[[very often]$_{ADV}$ useful]$_{ADJ}$*), which, in turn, are built before nominal chunks (e.g. *[[[very often]$_{ADV}$ useful]$_{ADJ}$ solutions]$_{NOUN}$*), and so on. Currently, the parser includes 326 chunking rules, which are grouped according to the syntactic category of the head.

Coordination

Coordination notoriously is one of the hardest problems in NL. And coordination abounds in Italian legal texts. We report an example below[8]:

> *È approvata la proposta formulata dalla Regione Campania, in merito alle domande presentate per il bando del* 2003 ***e (coord)*** *riferite alla predetta Regione per le attività estrattive**, (coord)** manifatturiere**, (coord)** di servizi**, (coord)** delle costruzioni e (**coord**) di produzione **e (coord)** distribuzione di energia elettrica**, (coord)** di vapore **e (coord)** acqua calda.*[9]

As it can be seen, some coordination is expressed by commas and they can be nested. In order to cope with these complexities, coordination is handled by means of a set of procedurally implemented heuristics that, when a conjunction is encountered, perform the following steps:

1 *look for the best second conjunct*. In our example (in case of composed conjuncts, out of the brackets, the head word): **and**$_1$ → referred; **and**$_2$ → of [production]; **and**$_3$ → distribution; **and**$_4$ → [hot] water;

2 *Look for possible first conjuncts* (many of them; see below for the final choice);

3 *Choose the best pair.* In our example: **and**$_1$ → presented; **and**$_2$ → of [construction]; **and**$_3$ → production;**and**$_4$ → steam.

4 *Move back to ascertain if any comma can act as a previous conjunction of a sequence.*. This enables the system to recognized as conjunctions all commas except the last one (the one

[8] The English pseudo-translation aims to keep the ordering of the Italian words. In this translation, the conjunctions are labelled with subscripts for reference purposes in the following description.

[9] *It is approved the proposal made by the Regione Campania, concerning the applications presented for the advertisement of 2003* ***and***$_1$ *(**coord**) referred to the aforementioned Region for the activities extractive**, (coord)** manufacturing**, (coord)** of services**, (coord)** of construction* ***and***$_2$ *(**coord**) of production* ***and***$_3$ *(**coord**) distribution of electricity**, (coord)** of steam* ***and***$_4$ *(**coord**) hot water.*

before "of steam"). This is due to the presence of the preposition ("of" steam) that prevents the system from accepting "electricity, steam and hot water" as a sequence.

Verbal Subcategorization

After the previous steps, the partially built parsing structure includes a set of (possibly very large) chunks, including prepositional modifiers and conjunctions, and some verbs[10]. The various chunks are the verbal dependents: they are now attached to the verbs (via rules that take into account distance from chunks and verbs, intervening subordinating conjunctions, relative pronouns, and so on). After this, each verb is associated with a set of unlabelled dependents. The final task, in order to provide the semantic interpreter with all required information, is to determine the labels of the arcs.

This is made by exploiting knowledge about the subcategorization frames of verbs. Each verb is assigned to one or more *verbal subcategories*; each of them, in turn is associated with a *verbal frame*. The final goal of this step is to find, for each verb in the sentence, the best match between its dependents (found in the previous steps) and the verbal frames associated to its possible subcategories. The task is made more complex by the existence of *transformations*, that affect the possible surface realizations of a verbal frame. For instance, the verb "autorizzare" (authorize) has just one possible subcategory, i.e. *trans-a* (transitive verbs admitting a theme governed –in Italian– by the "a" preposition[11]). Although the base description of *trans-a* involves a *subject*, and *object* and a *theme*, as in

> *[Il terzo comma]*$_{subj}$ *autorizza [il Consiglio]*$_{obj}$ *[a inviare aiuti alimentari]*$_{theme}$
> *([The third paragraph]*$_{subj}$ *authorizes [the Council]*$_{obj}$ *[to send food aids]*$_{theme}$*)*

a sentence as the following must be accepted

> *[Il Consiglio]*$_{obj}$ *è autorizzato [a inviare aiuti alimentari]*$_{theme}$
> *([The Council]*$_{obj}$ *is authorized [to send food aids]*$_{theme}$*)*

This is in fact obtained by means of two transformations: *passivization* and *null-agt-compl*. The first of them produces the passive form; the second one enables for the deletion of the agent complement which, in other cases, may appear. In case of deletion, the parser introduces a trace, that records the presence of a covert argument. Note that "to send food aids" is a clause that undergoes the same kind of processing. Consequently, "food aids" is recognized the object of "send". The subject is deleted, but in this case it is recovered from the main clause: when its verb is "authorize", then the implicit subject of the dependent clause is equal to the object of "authorize" (the Council): this rule is called *obj-equi*. Note also that in the passive form, the deep function of "The council" (i.e. "object") must be (and in fact is) recovered in order to allow *obj-equi* to work properly also in this case. Finally, in the passive example, the indirect object of "send" is known to exist, but it is unspecified; the actual final representation is:

> *([The Council]*$_{obj,1}$ *is authorized [*t_u*]*$_{subj}$
> *[to send [*t_1*]*$_{subj}$ *[*t_u*]*$_{indobj}$ *[food aids]*$_{obj}$*]*$_{theme}$*)*

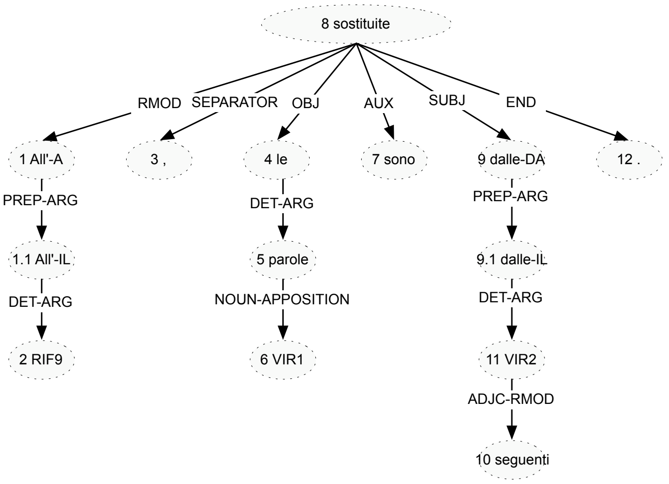


Figure 4: Syntactic analysis of the sentence: *All'articolo 40, comma 1, della legge 28 dicembre 2005, n. 262, le parole "sei mesi" sono sostituite dalle seguenti: "dodici mesi" [At the article 40, comma 1 of the law December 28, 2005 number 262, the words "six monts" are substituted by the following: "twelve monts"].*

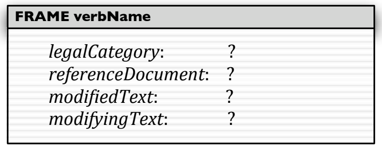
That is, "some unknown entity authorizes the Council that the Council sends to some other unknown entity food aids".

Such a representation is the basis for the final step of semantic interpretation, described in the next section.

3.3 Semantic Interpretation

The semantic interpreter is a *rule-based* system. Rules rely on two different sorts of information. The first one consists of the tree produced in output by the syntactic parser; the second one comes from the legal provisions taxonomy (Section 2.2).

A modification is represented in terms of a *semantic frame* composed by *slots* [2], such as *legalCategory*, *referenceDocument*, *modifyingText* and *modifiedText*.


FRAME verbName

legalCategory:	?
referenceDocument:	?
modifiedText:	?
modifyingText:	?

Each rule maps the input sentence onto a *syntactic pattern* derived from the taxonomy in order to fill the slot of a semantic frame.

One general rule is charged to test whether the root node of the syntactic tree is a verb, and if it belongs to the modificatory provisions taxonomy. In this case we have a fundamental cue that the sentence being analyzed contains a modificatory provision and –based on the taxonomy–, we are informed of the *legalCategory* of the modification at hand. Each other rule is composed of a set of IF-THEN tests on the content of the verb arguments and the verb modifiers to fill the slots of current frame.

For example, the syntactic tree corresponding to the *substitution* modification in Figure 4 has as root node the verb "sostituire" (*substitute*). As a consequence, the semantic frame associated to the *legalCategory: substitution* is instantiated, and a set of tests are executed on the verb dependents –i.e., the children nodes– to fill the appropriate slots. The slots are proper of the legal category *substitution*, so that verbs such as *substitute*, *change*, *modify*, etc. have

[10] Actually, some inputs can be without verbs. In this case, either the analysis now includes a single chunk (and the task is completed) or one of the chunks is chosen as the head and the others are attached to it. This is still accomplished via heuristics.

[11] Some class names reflect the first implementation (for Italian). Actually *trans-a* covers also English verbs for which the preposition of the theme is "to" (as "I authorize him to leave").

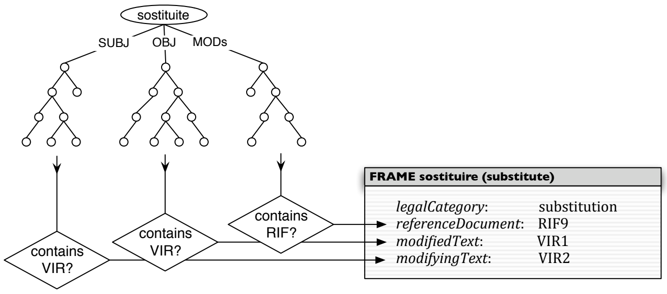


Figure 5: The *positionAndBothArguments__rule* performs three tests on te parse tree and then fills the appropriate slots, accordingly.

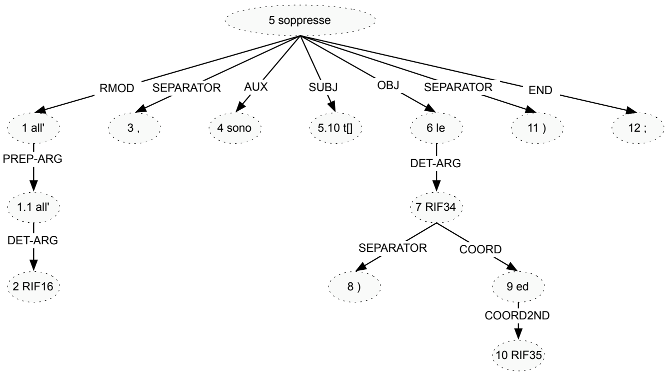


Figure 6: Parse tree of coordinated repeals.

all the same slots. In this way we can add further verbs to the legal categories by taking advantage of their shared semantic frames. In particular, the rules are charged to discover whether in the syntactic arguments like *subject*, *object* or in syntactic modification like *modification* any constant such as *RIF* or *VIR* is present.

We briefly describe the procedure followed to annotate the modification contained in the sentence *All'RIF9, le parole VIR1 sono sostituite dalle seguenti VIR2* [*At the RIF9, the words VIR1 are substituted by the following VIR2*], whose parse tree is presented in Figure 4. The set of rules built to handle substitution are tested, and the *positionAndBothArguments__rule* is executed. The *IF-THEN* rule maps a syntactic pattern onto a set of semantic slots (Figure 5):

IF
· the subtree attached to the verb by a *RMOD* label (that is *modifier*) contains a RIF_1 constant; **AND**

· the subtree attached to the verb by a *SUBJ* label (that is *subject*) contains a VIR_1 constant; **AND**

· the subtree attached to the verb by a *OBJ* label (that is, *object*) contains a VIR_2 constant

THEN
· fill *referenceDocument* with he RIF_1 **AND**

· fill *modifiedText* with the VIR_2 **AND**

· fill *modifyingText* with the VIR_1

Some further rules are designed to account for complex linguistic constructions, such as the case of coordination. Let us consider the following sentence: *All'articolo 1, comma 2, sono soppresse le lettere d) ed f)*; [*At the article 1, comma 2, the letters d) and f) are deleted;*]. It is conveniently converted into: *All'RIF16, sono soppresse le RIF34) ed RIF35)*, [*At RIF16, the RIF34) and RIF35) are deleted*]. The TUP recognizes coordination and marks it with *COORD* relation, as it is shown in the bottom-right corner of Figure 6. The semantic frame corresponding to the deletion of *RIF34* is filled similarly to the case of the substitution (the main difference being that the slot *modifyingText* is not present); additionally the semantic interpreter recognizes the presence of a coordinate. In facts, the reference *RIF34* has a descendant node *RIF35* in a subtree that is connected by the *COORD* labeled edge.

The case involving 2 coordinated *Objects*, such as *RIF34* and *RIF35*, is rather straightforward because we are handling homogeneous objects (both are *RIF*s), and because there is no ambiguity about the fact that the *objects* are coordinated. To extend the coverage, here are some further sorts of coordination types: 1. Al *RIF22* le parole: *VIR4* e *VIR7* sostituiscono le parole: *VIR8* e *VIR11*, rispettivamente; (rough translation: At the *RIF22* the words: *VIR4* and *VIR7* substitute the words: *VIR8* and *VIR11*, respectively;). 2. Al *RIF1* le parole *VIR1* sostituiscono *VIR4* e *VIR5* e al *RIF2* le *VIR12* sostituiscono *VIR13*, *VIR14* e *VIR15* (rough translation: At *RIF1*, the words *VIR1* replace the *VIR4* and *VIR5*, and at *RIF2* the *VIR12* replaces the *VIR13*, *VIR14* and *VIR15*).

As a general strategy, to handle such cases we bound the combinatorics menacing the computation by assuming that coordinates only can be homogeneous (that is, a *VIR* cannot be coordinated to a *RIF*).

4. EXPERIMENTATION AND EVALUATION

From a practical viewpoint, our research is intended to assist human annotators in individuating legal provisions and qualifying them with as many details as possible. To assess our work, we started from a set of 1540 files NIR *DTD* 2.0 *valid*, containing $4,635$ modifications. In this dataset the three legal categories (or types) *integration*, *substitution* and *deletion* are by far the most common types, as it is shown in Table 1; namely, $3,285$ modificatory provisions fall into one of such classes, thereby covering 70.87% of the grantotal.

At the current stage of development, we deal with *articolati* (that is, documents composed of articles) and we consider only modificatory provisions of either *integration*, *substitution* or *deletion* type. The intersection between the restriction on the file type and on the modificatory provisions type resulted in considering 181 files, for a total amount of $11,646$ XML *corpo* elements (see Section 3.1) containing $2,148$ modificatory provisions (namely, 744 integrations, 842 substitutions and 562 deletions).[12] Such files contain 1994 modifications hand-annotated by the CIRSFID legal experts [7], which were considered for the experimentation.

We tested our system with two accuracy measures:

- **Measure A**. The percentage of modificatory provisions correctly computed as the pair $\langle$*type, position*$\rangle$, where *type* (*legal category* in previous Section) is one in $\{$*integration, substitution, and deletion*$\}$, and *position* is the constant identifying the file and the position into the file where the modification occurs;

[12] The dataset is available for download at the URL: `http://www.di.unito.it/~radicion/icail2009/`

Table 1: Number of modifications for each type, showing that most modifications are of *integration*, *substitution* and *deletion* type.

modif. type	♯ of modif.	modif. type	♯ of modif.
sostituzione	1,368	ratifica	31
integrazione	1,186	attuadelega	28
abrogazione	731	sospensione	21
variazioni	405	retroattivita	9
deroga	303	inapplicazione	8
estensione	136	ricollocazione	7
modtermini	133	recepisce	5
vigenza	61	ultrattivita	1
attua	59	annullamento	0
posticipo	43	proroga	0
attuadelegifica	35	reitera	0
inautentiche	33	reviviscenza	0
converte	32	**Total**	4,635

- **Measure B**. The accuracy computed as the percentage of modificatory provisions correctly computed as the tuple ⟨*type, position, novella, novellando*⟩, where *type* and *position* are the same as in the Measure A, and *novella* and *novellando* are both excerpts of quoted text. The *novella* (*modifyingText* in the above notation) is the portion of text being added, whilst *novellando* (*modifiedText* in the above notation) is the portion of text being modified. Both the *novella* nor the *novellando* may be absent (e.g., *deletions* usually lack of the *novella*).

Our system obtained 93.6% *precision* and 76.9% *recall*, computed with the Measure A; and 82.2% *precision* and 67.5% *recall* computed according to the Measure B. More details on results are provided in Table 2, where the *recall* for both measures on the specific types of modificatory provisions is provided.

Table 2: The *recall* obtained on each type of modificatory provisions.

	integration	*substitution*	*deletion*
Measure A	87.8%	86.6%	48.0%
Measure B	75.8%	75.5%	44.5%

It is instructive to inspect the errors, both to complete the assessment of the implemented system and to point out future improvements. The overall result gives an estimation of the robustness and of the goodness of the main approach, and of implemented system as well. However, since we are testing a rather complex architecture that involves modules *i*) to extract the meaningful parts of the documents (Section 3.1), *ii*) to parse them (Sec. 3.2), and *iii*) to individuate and annotate legal modifications (Sec. 3.3), it is interesting to disaggregate the results of such components, and to especially focus on the results provided in the semantic annotation process alone. In this case, one could compare the modifications computed by our system with the modifications that it would have been actually possible to find (instead of comparing the results with all the modifications present, as we did before).

The rationale is that since both in the phases *i*) and *ii*) it is possible that some excerpts of legal texts are skipped, thereby undermining the performance of the semantic annotation phase. The system skipped a number of *corpo* elements for various reasons. Most errors occurred in the parsing process, and were consequence of various reasons, e.g., too complex syntactic structures, or unknown words (such as the verb *anteporre*, *to place (sth.) before*). Further errors were due to errors in the documents, such as misspelled words or in the hand annotated XML files. We then recorded the number of ⟨*corpo*⟩ elements that were processed. In the dataset there were 11,646 ⟨*corpo*⟩ elements, 9,831 of which were correctly analyzed, and 1,815 were skipped. This first datum is interesting *per se*. The fact that some 15.6% of all possible texts containing modifications are lost before the semantic annotation phase suggests that an improvement on the input handling phase and –meantime– a rigorous check of the annotated files are necessary. In order to shed light on the performances of the semantic annotation, we defined the *virtual recall* measure, as the percentage of correctly identified modificatory provisions w.r.t. the number of modificatory provisions that are actually analyzed by the parser. If we consider the *virtual recall* computed with the *Measure B*, our performances raise from about 10% to 14%: the exact figures are reported in Table 3.

Table 3: The *virtual recall* obtained on each type of modificatory provisions.

	integration	*substitution*	*deletion*
Measure A	97.8%	97.5%	62.4%
Measure B	84.4%	85.0%	57.7%

Clearly, at a glance we obtain poor results in the annotation of *deletions*. Based on a preliminary inspection of deletions that were not correctly analyzed, we stipulate that one major reason of failure is due to the pre-processing stage. Our hypothesis on the handling the *corpo* and *alinea* elements was to some extent oversimplifying, and needs further refinements. Moreover we believe that a more sophisticated pre-processing phase would be beneficial to the overall performances of the system.

Even tough we are far from full automatic semantic annotation of provisions, the experimental results ($82.2 - 67.5\%$ of precision-recall, measure B) show that provision annotators can profitably use our system in order to speed-up the recognition and annotation of modificatory provisions.

5. RELATED WORKS

Our approach has some similarities with a number of previous works. The work in [3] encodes the meaning of modificatory provisions with semantic frames, but it is related to the automatic generation of modificatory provisions rather than with their analysis, so that this system can be hardly compared with ours.

A project that has many similarities with our research is SALEM [4, 5, 17, 8]. Similar to our approach, SALEM automatically annotates the modificatory provisions of NIR documents by using syntactic parsing and a rule-based strategy to fill the semantic frames. However, there are three differences between our project and SALEM:

1. In contrast with the original version of SALEM, we have a pre-processing step in which the fragments of the text that contain explicit reference are substituted with the reference `id` by using the NIR XML structure. This step allows reducing the work of the syntactic parser. However, the very last version of SALEM follows this approach too [8].

2. In our project we use the TUP, i.e. a deep syntactic parser. In contrast, in the SALEM project only a shallow syntactic analysis is produced by a chunk parser. By using a deep parser, we can cover the analyses of a wider range of syntactic phenomena that cannot be accounted for in the pure chunk approach, as coordination or relative clauses.

3. Similar to our approach, SALEM uses a provision taxonomy to build the semantic frame. However, SALEM produces classification and analysis of general provisions (with very high experimental results), whilst our project is concerned on modificatory provision, and as a consequence our taxonomy is more fine grained on modifications.

The work described in [13] uses a deep syntactic parser. Similar to our approach, in this work is used a deep syntactic parser (Collins' parser [9]) to build a full syntactic description of the legal sentences. Anyway, Collins' parser produces constituency structure rather than dependency structure, as the TUP parser. Moreover, in contrast to our approach, McCarty uses a logic language, i.e. a deep semantic structure, to represent the entire semantics of the sentences, rather than focusing on semantic frames. Indeed the aim of the McCarty's work is to extract enough information to build a question answering system on judicial opinions, while the final aim of our project is the automatic annotation of the meta information regarding modificatory provisions.

6. CONCLUSIONS AND FUTURE WORKS

We have introduced the relevant problem of the automatic annotation of legal provisions, involving several –challenging– open issues in the Law and Artificial Intelligence community. In particular, current research has a twofold interest: on theoretical accounts, we showed that coupling a hybrid deep syntactic parsing and a shallow semantic interpreter leads to promising results. On the other hand, on a practical bases, we delivered a NLP robust system that can speed-up the process of semantic metadata annotation of legal documents containing modificatory provisions. Extensions to account for further types of modificatory provisions will be addressed in future work.

We have recalled the NIR representation adopted for the both legal texts and modificatory provisions; we have introduced the taxonomy of legal modifications and the corresponding metadata format that is used throughout the process of annotation of modificatory provisions.

We then have illustrated the extraction of modificatory provisions as a three-steps process, where we first retrieve the relevant excerpts of text, parse them, and map the resulting parse trees onto the appropriate semantic frame.

Finally, we have described a hand-crafted dataset used as a gold-standard for experimenting with the implemented system; we have then discussed the results obtained and reviewed some interesting types of errors, pointing out possible improvements for the future.

7. REFERENCES

[1] AIPA. Formato per la rappresentazione elettronica dei provvedimenti normativi tramite il linguaggio di marcatura XML. Circolare n. AIPA/CR/40, 22 aprile 2002.

[2] D.E. Appelt and D.J. Israel. Introduction to information extraction technology. Tutorial at IJCAI-99, 1999.

[3] Timothy Arnold-Moore. Automatic generation of amendment legislation. In *Proceedings of the International Conference on Artificial Intelligence and Law (ICAIL)*, pages 56–62, 1997.

[4] R. Bartolini, A. Lenci, S. Montemagni, V. Pirrelli, and C. Soria. Semantic mark-up of italian legal texts through nlp-based techniques. In *Proceedings of LREC 2004*, pages 795–798, 2004.

[5] C. Biagioli, E. Francesconi, A. Passerini, S. Montemagni, and C. Soria. Automatic semantics extraction in law documents. In *ICAIL '05: Proceedings of the 10th international conference on Artificial intelligence and law*, pages 133–140, New York, NY, USA, 2005. ACM.

[6] C. Biagioli, E. Francesconi, P. Spinosa, and M. Taddei. The NIR project: Standards and tools for legislative drafting and legal document web publication. In *Proceedings of ICAIL Workshop on e-Government:Modelling Norms and Concepts as Key Issues*, pages 69–78, 2003.

[7] R. Brighi and M. Palmirani. Legal Text Analysis of the Modification Provisions: a Pattern Oriented Approach. In *Proceedings of the International Conference on Artificial Intelligence and Law (ICAIL)*, 2009.

[8] M. Cherubini, G. Giardiello, S. Marchi, S. Montemagni, P. Spinosa, and G. Venturi. NLP-based metadata annotation of textual amendments. In *Proc. of WORKSHOP ON LEGISLATIVE XML 2008, JURIX 2008*, 2008.

[9] M. Collins. Three generative, lexicalised models for statistical parsing. In *In Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics*, pages 16–23, 1997.

[10] L. Lesmo. The rule-based parser of the nlp group of the university of torino. *Intelligenza Artificiale*, 2(4):46–47, June 2007.

[11] L. Lesmo and V. Lombardo. Transformed Subcategorization Frames in Chunk Parsing. In *Proc. 3rd Int. Conf. on Language Resources and Evaluation (LREC 2002)*, pages 512–519, Las Palmas, 2002.

[12] C. Lupo, F. Vitali, E. Francesconi, M. Palmirani, R. Winkels, E. de Maat, A. Boer, and P. Mascellani. General XML format(s) for legal Sources - ESTRELLA European Project. Deliverable 3.1, Faculty of Law, University of Amsterdam, Amsterdam, The Netherlands, 2007.

[13] L. T. McCarty. Deep semantic interpretations of legal texts. In *ICAIL '07: Proceedings of the 11th international conference on Artificial intelligence and law*, pages 217–224, New York, NY, USA, 2007. ACM.

[14] M. Palmirani and R. Brighi. An XML Editor for Legal Information Management. In R. Traunmüller, editor, *Electronic Government*, volume 2739 of *LNCS*, pages 421–429, Berlin / Heidelberg, 2003. Springer-Verlag.

[15] M. Palmirani and R. Brighi. Time model for managing the dynamic of normative system. *Electronic Government*, pages 207–218, 2006.

[16] J. Saias and P. Quaresma. A Methodology to Create Legal Ontologies in a Logic Programming Based Web Information Retrieval System. *Artificial Intelligence and Law*, 12(4):397–417, 2004.

[17] C. Soria, R. Bartolini, A. Lenci, S. Montemagni, and V. Pirrelli. Automatic Extraction of Semantics in Law Documents. In C. Biagioli, E. Francesconi, and G. Sartor, editors, *Proceedings of the V Legislative XML Workshop*, pages 253–266. European Press Academic Publishing, 2007.