Grammar-based Automatic Extraction of Definitions. Applications for Romanian

Adrian Iftene
Faculty of Computer Science
"Al. I. Cuza" University of Iaşi
Romania
adiftene@info.uaic.ro

Diana Trandabăţ
Faculty of Computer Science
"Al. I. Cuza" University of Iaşi
Romania
dtrandabat@info.uaic.ro

Ionuț Pistol
Faculty of Computer Science
"Al. I. Cuza" University of Iaşi
Romania
ipistol@info.uaic.ro

Abstract

This paper presents part of our work in the LT4eL project [1] regarding the grammar developed by the Romanian team in order to extract definitions from texts. Some qualitative results come in order to evaluate our grammar rules. Among the applications of this kind of grammar we will discuss the possible inclusion of the grammar rules into a question answering system in order to extract answers for definition type questions. Another possible usage of those rules envisages the extraction of supplementary knowledge from linguistic resources like Wikipedia. The benefits of such an extra-knowledge resource are evident in textual entailment systems, where some resources like WordNet, Acronyms database or Dirt cannot cover all the requirements of the system.

Keywords

Definition Extraction, Question Answering, Textual Entailment.

1. Introduction

Under the framework of the FP6 European project LT4eL[1] (Language Technology for e-Learning), an environment for collecting and (semi)automatic exploiting language resources has been created, as the main objective of the project is to provide functionalities based on language technologies and to integrate semantic knowledge in Learning Management Systems. The first step was to create, for the 9 languages involved (Bulgarian, Czech, Dutch, English, German, Maltese, Polish, Portuguese and Romanian), a multilingual corpus, partially parallel, of almost 5.5 million words, annotated and uploaded on the project's portal[2] [2].

In order to improve the management, distribution and retrieval of the learning material by automatically attaching metadata (such as keywords and definitions) to any text, a necessary step was the observation of those metadata in the annotated corpus. Therefore, the corpus was manually annotated to keywords (the words or expressions that a user of the Learning Management System would use to retrieve documents referring that notion), definitions of various terms and semantic concepts. Using the manual annotated documents, a grammar was created for the automatic identification of definitions in texts. Apart for the use in this project, we present also two other envisaged applications.

After briefly describing the formats of the learning materials used on the LT4eL project and the annotation of the definitions, Section 3 will describe the Romanian grammar. Section 4 presents several possible applications of the grammar in order to improve the quality of complex systems like a Question Answering system and a Textual Entailment system, before drawing some conclusions and further directions.

2. The learning material

The linguistic resources - called learning objects (LO) - were selected according to their language, format (.doc, .pdf, plain text, .html or other), domain (broadly: the use of computers in education), or Intellectual Properties Rights. After their automatic conversion to XML [2], the objects were linguistically annotated (tokens, part-of-speech, lemma, chunks) and converted into a unitary XML format[3] with basic formatting information extracted from the .txt and/or .html versions of the document, called basic XML. This version of the document was used for the manually annotation of keywords and definitions. The corpus collected for the Romanian language contains 56 documents summing approx. 700.000 words.

For the annotation process, we understood by a definition a concise explanation, description of a concept's meaning or type. A definition has two parts: the defined term and the defining context. An example of definition extracted from the Romanian corpus is:

Ro: *[{Cetățenia Uniunii Europene}*$_{Def\_term}$*]*$_{DEF\_PART1}$, *prevăzută în tratatul de la Roma şi mai apoi în cel de la Maastricht [este caracterizată de drepturi, de obligații şi de implicarea în viața politică]* $_{DEF\_PART2}$.

[1] LT4eL – http://www.lt4el.eu/

[2] http://consilr.info.uaic.ro/uploads_lt4el/

[3] http://ufal.mff.cuni.cz/~spousta/lt4el/html2xml/LT4ELBase.dtd

En: *[The European Union citizenship]* $_{DEF\_PART1}$*, as considered by the Treaty of Rome and the Treaty of Maastricht [is characterized by rights, obligations and the involvement in the public life]* $_{DEF\_PART2}$.

where the defined term is *Cetăţenia Uniunii Europene (En: European Union citizenship)*, and the definition is marked between brackets [ ]. One can see that not the entire sentence was considered to be the definition, since an explanatory attributive sentence can be considered outside the definition scope. In order to obtain this split, the definition is marked as continuing, and the parts of the defining context are marked successively.

An example of annotated learning material, in basic XML format, is presented in figure 1, for the definition discussed above.

```
<definingText comment="" id="def37" status=""
continue="y" def="dt35" part="1">
  <markedTerm id="dt35" comment="" dt="y"
kw="n" status="">
    <tok rend=" /b, /p, p" base="cetăţenie"
ctag="Ncfsry" id="t960"> Cetăţenia </tok>
    <markedTerm id="k36" comment="" dt="n"
kw="y" status="">
      <tok rend="" base="Uniunii_Europene"
ctag="Ed" id="t961">Uniunii_Europene</tok>
    </markedTerm>
  </markedTerm>
</definingText>
<tok rend="" base="," ctag="COMMA" id="t962">,
</tok>
<tok rend="" base="prevedea" ctag="Vmp--sf"
id="t963">prevăzută</tok>
<tok rend="" base="în" ctag="Spsa"
id="t964">în</tok>
<tok rend="" base="tratat" ctag="Ncmsry"
id="t965">Tratatul</tok>
<tok rend="" base="de_la" ctag="Spca"
id="t966">de_la</tok>
<tok rend="" base="Roma" ctag="Np"
id="t967">Roma</tok>
<tok rend="" base="(0.67)ş" ctag="Vmis1s"
id="t968">şi</tok>
<tok rend="" base="mai" ctag="Rp"
id="t969">mai</tok>
<tok rend="" base="apoi" ctag="Rgp"
id="t970">apoi</tok>
<tok rend="" base="în" ctag="Spsa"
id="t971">în</tok>
<tok rend="" base="acela" ctag="Pd3msr"
id="t972">cel</tok>
<tok rend="" base="de_la" ctag="Spca"
id="t973">de_la</tok>
<tok rend="" base="Maastricht" ctag="Np"
id="t974">Maastricht</tok>
<definingText comment="" id="def38" status=""
continue="y" def="dt35" part="2">
  <tok rend="" base="fi" ctag="Vaip3s"
id="t975">este </tok>
  <tok rend="" base="caracteriza" ctag="Vmp--
sf" id="t976">caracterizată</tok>
  <tok rend="" base="de" ctag="Spsa"
id="t977">de</tok>
  <tok rend="" base="drept" ctag="Ncfp-n"
id="t978">drepturi</tok>
  <tok rend="" base="," ctag="COMMA"
id="t979">,</tok>
  <tok rend="" base="de" ctag="Spsa"
id="t980">de</tok>
  <tok rend="" base="obligaţie" ctag="Ncfp-n"
id="t981">obligaţii</tok>
  <tok rend="" base="(0.62)ş" ctag="Ncmpry"
id="t982">şi</tok>
  <tok rend="" base="de" ctag="Spsa"
id="t983">de</tok>
  <tok rend="" base="implicare" ctag="Ncfsrn"
id="t984">implicare</tok>
  <tok rend="" base="în" ctag="Spsa"
id="t985">în</tok>
  <tok rend="" base="viaţă" ctag="Ncfsry"
id="t986">viaţa</tok>
  <tok rend="" base="politic" ctag="Afpfsrn"
id="t987">politică</tok>
</definingText>
```

Figure 1. Example of manually annotated definition

3. Romanian grammar

For the automatic annotation of the definitions found in the learning objects, the approach throughout the LT4eL consortium was to develop local grammars for the 9 represented languages (English, Dutch, German, Polish, Bulgarian, Maltese, Czech, Romanian, and Portuguese) to extract definition patterns. The main difficulties addressed were due to the different manner of expressing the definitions, especially if the lexicalization of the introducing words (like the verbs “is”, “represents” etc.) were to be kept minimal. Other problems were raised by interrupted definitions or by the ending point of a definition, if the definition ends before the sentence punctuation marks.

The linguistic information from the manually annotated definitions is used as starting point in identifying possible grammar patterns that could form a definition. Previous work within this area shows that the use of local grammars which match syntactic structures of defining contexts are really useful when deep syntactic and semantic analysis is not present [3, 4].

The creation of the Romanian grammar started with some simple rules and their application over the manual annotated files. Observing repeatedly the cases left aside, we improved the grammar with more complex rules and different lexical items. The drawback of this approach is that the results are corpus dependent.

3.1 Categorization of Definitions

Definitions have been categorized in six types in order to reduce the search space and the complexity of rules. The types of definitions observed in Romanian texts have been classified as follows:

1. **“is_def”** – Definitions containing the verb “este” (En: is):

Example: “*Prescurtare pentru Hyper Text Mark Up Language, HTML* ***este*** *tot un protocol folosit de World*

Wide Web.” (En: Abbreviation for Hyper Text Mark Up Language, HTML **is** also a protocol used by World Wide Web).

2. **“verb_def”** – Definitions containing specific verbs, different by “este” (En: is). The verbs identified for Romanian are “indica” (En: denote), “arăta” (En: show), “preciza” (En: state), “reprezenta” (En: represent), “defini” (En: define), “specifica” (En: specify), “consta” (En: consist), “fixa” (En: name), “permite” (En: permit).

Example: “*Poşta electronică* ***reprezintă*** *transmisia mesajelor prin intermediul unor rețele electronice*.” (En: Electronic mail **represents** sending messages through electronic networks).

3. **“punct_def”** – Definitions which use punctuation signs like the dash “-”, brackets “()”, comma “,” etc.

Example: “*Bit* **–** *prescurtarea pentru binary digit*” (En: Bit **–** shortcut for binary digit)

4. **“layout_def”** – Definitions that can be deduced by the layout: they can be included in tables when the defined term and the definition are in separate cells or when the defining term is a heading and the definition is the next sentence.

Ro:

Organizarea datelor	*Cel mai simplu mod de organizare este cel secvențial.*

En:

Data organizing	The simplest method is the sequential one.

5. **“pron_def”** – Anaphoric definitions, when the defining term is expressed in a precedent sentence and it is only referred in the definition, usually pronoun references.

Example: *“...definirii conceptului de baze de date.* ***Acesta*** *descrie metode de modelare ale problemelor reale în scopul definirii unor structuri care să elimine redundanțele în stocarea datelor.*” (En: …defining the database concept. **It** describes methods of modeling real problems in order to define structures which eliminate redundancy in data collecting.)

6. **“other_def”** – Other definitions, which cannot be included in any of the previous categories. In this category are constructions which do not use verbs as the introducing term, but a specific construction, such as “*i.e.*”.

Example: “*triunghi echilateral, adică cu toate laturile egale*” (En: equilateral triangle i.e. having all sides equal).

The distribution of the definition types in Romania corpus is presented in table 1:

Type	**Manual**	**%**	**Automatic**	**%**
is_def	70	33.8	204	32.8
verb_def	116	56.0	272	43.8
punct_def	15	7.2	124	20.0
layout_def	2	1.0	21	3.4
pron_def	4	2.0	0	0.0
Total	**207**		**621**	

Table 1: Distribution of the definitions into types

The above table states that 33% of the total of definitions (both for manual and automatic definitions) are introduced by the verb *a fi* (“to be”). An interesting observation is that the definitions introduces by something else than a verb sums approximately 10% of the manually and around 23% of the automatic detected definitions. The big difference indicates the fact that more manual markups are needed in our corpus.

3.2 The grammar

The XML transducer lxtransduce developed in [5] is used to match a grammar against several files in XML format. Lxtransduce supplies a format for the development of grammars either in pure text or in XML documents. The grammars files are XML documents built according a specific DTD. In our grammar, we created rules for each type of defining context and a ”main” rule used to call the rules one-by-one at different runs.

All these rules were observed through the observation of the manually extracted definitions.

3.3 Grammar rules

The grammar for extracting Romanian definitions starts with several simple rules which identify different part of speech.

For instance, the rule presented in figure 2 identifies adverbs by looking in the ctag attribute at the first letter. If this first letter is “r” then we have detected an adverb:

```
<rule name="Adv">
 <query match="tok[@ctag[starts-with(.,'r')]"/>
</rule>
```

Figure 2: Simple grammar rule

Those rules can be combined in order to obtain more complex rules. Figure 3 presents an example of entities that are actually combination of simple entities:

```
<rule name="Nominal">
  <seq>
   <ref name="undef" mult="?" />
   <ref name="AdjP" mult="?" />
   <ref name="Noun" />
   <ref name="AdjP" mult="?" />
  </seq>
 </rule>
```

Figure 3: Composed grammar rule

After identifying the different structures, the general rules are created. Figure 4 presents the grammar rules for the “is_def” definitions. The lemma for the verb must be “fi” (En: be) and its part of speech label (contained in the *ctag* tag) must be “vmip3” (verb main indicative present third person). Another condition is that we have an entity “DefNominal” or “UndefNominal” (definite or indefinite noun), entity defined through a complex rule as the one presented in figure 3.

```
<rule name="may_be_term">
  <seq>
     <query match="tok[@base='fi' and
           substring(@ctag,1,5)='vmip3']"/>
     <first>
       <ref name="UndefNominal" />
       <ref name="DefNominal" />
     </first>
</seq>
```

Figure 4: “is_def” grammar rule

Another type of rule is the one that identifies the end of the sentence, thus considered the end of the definition (Figure 5):

```
<rule name=”main” wrap=”definingText”
attrs="def_type1='punct_def'">
  <seq>
   <ref name="NP" wrap="markedTerm"
attrs="dt='y'"/>
   <ref name="may_be_term" />
   <repeat-until name="anything">
         <query match="tok[(@base='.' and
@ctag='period') or (@base=';' and
@ctag='scolon')]" />
   </repeat-until>
   <query match="tok[(@base='.' and
@ctag='period') or (@base=';' and
@ctag='scolon')]" />
  </seq>
 </rule>
```

Figure 5: Sentence boundary rule

3.4 Grammar evaluation

The XML transducer, lxtransduce, is used to match our grammar, conforming to an XML format specified within the tool by using XPath. When a match is found, a rule is applied and a definition is marked in the file. The tool was run on every type of definitions and the results are presented in table 2 (P = precision, R= recall and F2 = F measure):

Definition Type	Result
is_def	**Sentence-level matching:** P: 0.5366, R: 1.0, F2: 0.7765 **Token-level matching:** P: 0.0648, R: 0.3328, F2: 0.14
verb_def	**Sentence-level matching** P: 0.7561, R: 1.0, F2: 0.9029 **Token-level matching** P: 0.0471, R: 0.1422, F2: 0.085
punct_def	**Sentence-level matching** P: 0.1463, R: 1.0, F2: 0.3396 **Token-level matching** P: 0.0025, R: 0.1163, F2: 0.0072
layout_def	**Sentence-level matching** P: 0.0488, R: 1.0, F2: 0.1333 **Token-level matching** P: 0.0007, R: 0.1020, F2: 0.0022

Table 2: Romanian grammar evaluation

For each definition type, the precision and recall were calculated at two levels: at the token level and at the sentence level [5]. At ***token level***, precision is understood as the number of tokens simultaneously belonging to a manual definition and an automatically found definition, divided by the number of tokens in automatically found definitions. Correspondingly, recall is the ratio of the number of tokens simultaneously in both definition types to the number of tokens in manual definitions. At the ***sentence level***, a sentence is taken as a manual or automatic definition sentence if and only if it contains a (part of a), respectively, manual or automatic definition. Therefore, precision and recall are calculated analogous to the values used in token level.

The best results are obtained for the definitions introduced by verbs (the most common cases). Among those, the definitions introduced by the verb “is” are the most difficult to identify, since the verb appears very frequently in the language and many cases of non-defining contexts are wrongly considered. An example of a wrong annotation is:
<definition>o asemenea practica este recomandata in cadrul documentelor complexe . </definition> (En: Such an praxis is recommended within complex documents.)

For the pron_def and other_def, although there are several manually annotated definitions, the rules considered aren’t accurate enough and require improvement.

4. Definition Extraction Applications

4.1 Definitions extraction in a Question Answering system

Question Answering (QA) can be defined as the task which takes a question in natural language and produces one or more ranked answers from a collection of documents. The QA research area has emerged as a result of a monolingual English QA track being introduced at TREC[4].

[4] Text Retrieval and Evaluation Conference - http://trec.nist.gov/

QA systems normally adhere to the pipeline architecture composed of three main modules: question analysis, paragraph retrieval and answer extraction [5].

The first module is the question analyzer. The input to this module is a natural language question and the output is one or more question representations which will be used at subsequent stages. At this stage most systems identify the semantic type of the entity sought by the question, determine additional constraints on the answer and the question type and extract keywords to be employed by the next module.

The paragraph retrieval module is typically based on a conventional information retrieval search engine in order to select a set of relevant candidate paragraphs/sentences from the document collection.

At the last phase, answer extraction and ranking, the representation of the question and the candidate answer-bearing snippets are compared and a set of candidate answers are produced and then ranked using likelihood measures.

Accordingly to the answer type, we have the following type of questions [7]:

- ***Factoid*** – The question refers to a single answer, as for instance: "*Who discovered the oxygen?*" or "*When did Hawaii become a state?*" or "*What football team won the World Coup in 1992?*".
- ***List*** – The answer to a list question is an enumeration: "*What countries export oil?*" or "*What are the regions preferred by the Americans for holidays?*".
- ***Definition*** – These questions require a complex processing of the texts and the final answer consist of a text snippet or is obtain after summarization of more documents: "*What is quasar?*" or "*What is a question-answering system?*"

We present below the system developed this year for the QA@CLEF2007[5] competition and the way we dealt with definition questions. This system is based on the cross-lingual system built last year for the English-Romanian task [8].

In the case of DEFINITION questions, the candidate paragraphs extracted in the information retrieval phase are matched against a set of rules of our Romanian grammar. The rules from the Romanian grammar were translated from lxtransduce format to Perl patterns. The reason of this rule transformation is that the QA system uses an annotation of the corpus (lemma part of speech, name entity, etc.) different form the file format considered by lxtransduce, and the size of the corpus didn't allow for any format changes.

Thus, each possible definition having as defined term the focus of the question is extracted and added to a set of candidate answers, together with a score revealing the reliability of the pattern it matched.

The set of NPs in the snippet is also investigated to detect those NPs containing the defined term surrounded by other functional words (this operation is motivated by cases like the *Atlantis space shuttle*, where the correct definition for *Atlantis* is *space shuttle*). The selected NPs are added to the set of candidate answers with a lower score.

The set of candidate answers is then ordered according to the score attached to each answer and to the number of other candidate answers it subsumes. The highest ranked candidate answers are presented as final answers. Figure 6 presents the architecture of the Romanian Question Answering System that competed to the QA@CLEF competition this year.

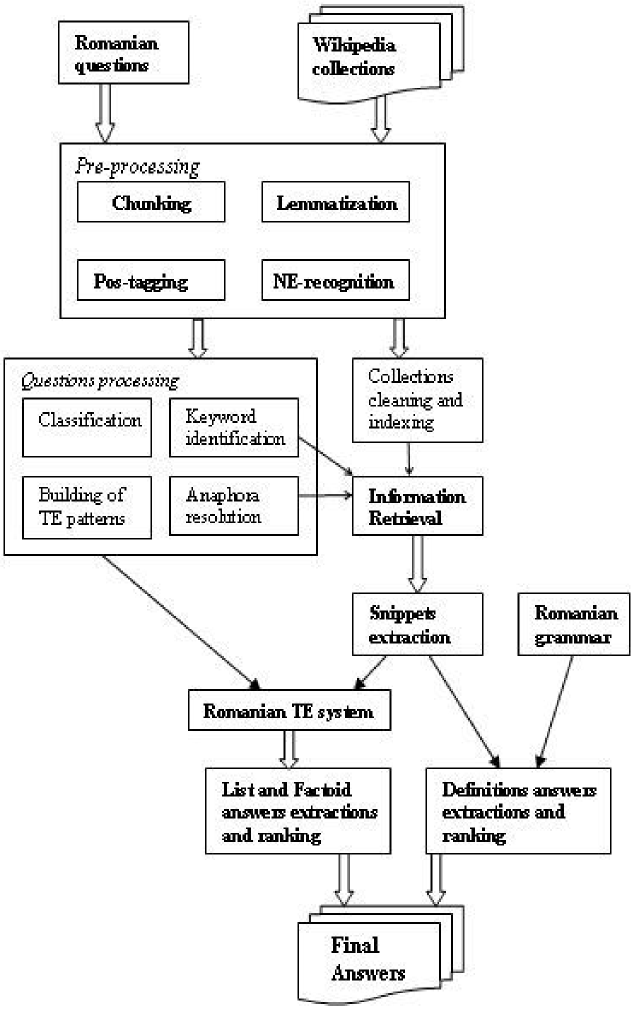

Figure 6: Architecture of the Romanian Question Answering System

[5] Cross Language Evaluation Forum - http://www.clef-campaign.org/2007.html

4.2 Building a Background Knowledge database for Textual Entailment

Within the Textual Entailment competition[6] [9], participants in the evaluation exercise are provided with pairs of small text snippets (one or more sentences in English), which are named Text-Hypothesis (T-H) pairs. The participants must build a system that, for each pair, should say if there is entailment or no (if the text entails the hypothesis). The complexity of this task comes from the complexity of the applications needed in order to assess a semantic relationship between text segments: Information Retrieval (IR), Question Answering (QA), Information Extraction (IE), and Text Summarization (SUM).

Our system architecture is based on a peer-to-peer network design, in which neighboring computers collaborate in order to obtain the global fitness for every text-hypothesis pair [10]. The main idea is to transform the hypothesis making use of extensive semantic knowledge from sources like DIRT, WordNet, Wikipedia, acronyms database, etc. Additionally, we built a system to acquire the required extra background knowledge and applied complex grammar rules for rephrasing. We calculated then the distance between the dependency trees associated to the initial text and to the new hypothesis. Eventually, based on the computed score, we decide for which pairs we have entailment.

There is information that cannot be deduced from the databases and thus we require additional means of gathering extra information such as the one presented in table 3.

Argentine [is] Argentina
Netherlands [is] Holland
2 [is] two
Los Angeles [in] California
Chinese [in] China

Table 3: Background knowledge

The background knowledge was built semi-automatically, for the named entities and for numbers from the hypothesis without correspondence in the text. For these named entities, we used a module to extract from Wikipedia[7] snippets with information related to them. In the snippets extracted from Wikipedia we try to identify the defining texts. For each such context:

a) we identify the "core" of the definition (which is either the verb "to be" or another definition introducing verb or a punctuation mark).

b) we extract from the left hand part of the "core": all the name entities (left NEs)

c) we extract from the right hand side of the "core": all name entities (right NEs)

d) we compute the Cartesian product between left NEs and right NEs and add the resulting pairs to the existing background knowledge base.

Subsequently, we use this file with snippets (*Argentina* for instance in Figure 8) and the several patterns in order to identify the relations between the entities in The goal in this endeavor is to identify a known relation between two named entities.

```
ar |calling_code = 54 |footnotes = Argentina also has a territorial dispute
Argentina', , Nacion Argentina (Argentine Nation) for many legal purposes), is
in the world. Argentina occupies a continental surface area of
Argentina national football team
```

Figure 8: Snippets extracted for Argentina

If such a relation is found, we make the association and save it to an output file. For our case only line "Argentina [is] Argentine" is added to the background knowledge. Another example of extracted knowledge for the NE "Netherlands" is presented in table 4:

Netherlands [is] Dutch
Netherlands [is] Nederlandse
Netherlands [is] Antillen
Netherlands [in] Europe
Netherlands [is] Holland
Antilles [in] Netherlands

Table 4: Results for Netherlands

All these relations are added to the background knowledge database and will be used at the next run. Not all relations are correct, but the relation "*Netherlands [is] Holland*" will help us at the next run.

Our patterns identify two kinds of relations between words:

- "is", when the module extracts information in the form: *'Argentine Republic' (Spanish: 'Republica Argentina', IPA)'* or when explanations about the word are between brackets, or when the extracted information contains one verb used to define something, like "is", "define", "represent": *'2' ('two') is a number.*

- "in" when information is of the form: *'Chinese' refers to anything pertaining to China* or in the form *Los Angeles County, California*, etc.

In the case of the 'is' relation, we use the same rules as those in the grammar and for the 'in' relation we add more specific rules. Usually, 'in' relation appear between specific named entities like LOCATION or COUNTRY or ORGANIZATION, and in these cases it is determined by specific words like: "in", "located", "from", etc.

In order to be able to see each component's relevance, the system was run in turn with each component removed.

[6] http://www.pascal-network.org/Challenges/RTE/

[7] http://en.wikipedia.org/wiki/Main_Page

System Description	**Precision**	**Relevance**
Full system	0.6913	**-**
Without DIRT	0.6876	0.54 %
Without WordNet	0.6800	1.63 %
Without Acronyms	0.6838	1.08 %
Without BK	**0.6775**	**2.00 %**
Without Negations	0.6763	2.17 %
Without NEs	0.5758	16.71 %

Table 5: Components relevance

We can notice that the Background Knowledge resource is very important, and represent 2 % from total precision of the system.

The system presented here participated this year for first time in the RTE3[8] competition. From 26 competing groups, we obtained the third place with a precision of 69.13%.

5. Conclusions and Future work

This paper presented the Romanian grammar used in the European LT4eL project to automatically extract definitions from texts. The definitions were devised in 6 types, and the results of the system for each definition type were presented. The automatic discovery of definitions using a rule-based method can significantly improve a question answering system (for definition type questions) or the background acquisition useful for a textual entailment system.

A necessary further step in the improvement of the Romanian grammar is applying it to a new corpus, in order to verify that all the definitions extracted are real defining contexts.

6. Acknowledgements

This paper present the work of the Romanian team in the European project Language Technologies for e-Learning, STREP 027391 in FP6-2004-IST-4, and within the CEEX Rotel project number 29.

Special thanks goes to the other members of the Romanian team in the LT4eL project, Dan Cristea and Corina Forăscu, also involved in developing the Romanian grammar.

We also acknowledge the help provided by Claudia Borg and her analysis program in improving the grammar performances.

7. References

[1] P. Monachesi, L. Lemnitzer, and K. Simov. Language Technology for eLearning Poster presentation at First European Conference on Technology Enhanced Learning, 1-4 October, Crete, Greece. http://www.ectel06.org/index.html.

[2] I. Pistol, D. Trandabăţ, A. Iftene, D. Cristea, C. Forăscu. Processing Romanian linguistic Resources in the LT4eL project (in Romanian). In Proceedings of the Wokshop Linguistic Resources and Tools forProcessing Romanian Language, C. Forăscu, D. Tufiş, D. Cristea (eds.). Iasi, Romania, November 2006. University “Al.I. Cuza” Publishing House. 2006.

[3] S. Mureşan and J. Klavans. A Method for Automatically Building and Evaluating Dictionary Resources. Proceedings of LREC 2002.

[4] B. Liu, C. W. Chin, and H. T. Ng. Mining Topic-Specific Concepts and Definitions on the Web. Proceedings of the Twelfth International World Wide Web Conference (WWW'03). 2003.

[5] J. Carletta. Assessing agreement on classification tasks: The kappa statistic. Computational Linguistics, 22:249–254. 1996.

[6] R. Tobin. Lxtransduce A replacement for fsgmatch. http://www.ltg.ed.ac.uk/~richard/ltxml2/lxtransduce-manual.html. 2005.

[7] S. Harabagiu and D. Moldovan. Question answering. In Ruslan Mitkov, editor, Oxford Handbook of Computational Linguistics, chapter 31, pages 560 – 582. Oxford University Press, 2003.

[8] J. Lin. Evaluation of Resources for Question Answering Evaluation. In Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2005), Salvador, Brazil. 2005.

[9] G. Puşcaşu, A. Iftene, I. Pistol, D. Trandabăţ, D. Tufiş, A. Ceauşu, D. Ştefănescu, R. Ion, C. Orasan, I. Dornescu, A. Moruz, D. Cristea. Developing a Question Answering System for the Romanian-English Track at CLEF 2006. In Proceedings of the CLEF 2006 Workshop. 22-24 September. Alicante, Spain. 2006.

[10] I. Dagan, O. Glickman and B. Magnini. The PASCAL Recognising Textual Entailment Challenge. In Quiñonero-Candela et al., editors, MLCW 2005, LNAI Volume 3944, pages 177-190. Springer-Verlag. 2006.

[11] A. Iftene and A. Balahur-Dobrescu. Hypothesis Transformation and Semantic Variability Rules Used in Recognizing Textual Entailment. In Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing. Pp. 125-130. 28-29 June, Prague, Czech Republic. 2007.

[8] http://www.pascal-network.org/Challenges/RTE3/