# A Language Independent Approach for Recognizing Textual Entailment

Adrian Iftene and Alexandra Balahur-Dobrescu

"Al.I.Cuza" University of Iasi, Romania
adiftene@info.uaic.ro, alexyys13@yahoo.com

**Abstract.** Textual Entailment Recognition (RTE) was proposed as a generic task, aimed at building modules capable of capturing the semantic variability of texts and performing natural language inferences. These modules can be then included in any NLP system, improving its performance in fine-grained semantic differentiation. The first part of the article describes our approach aimed at building a generic, language-independent TE system that would eventually be used as a module within a QA system. We evaluated the accuracy of this system by building two instances of it - for English and Romanian and testing them on the data from the RTE3 competition. In the second part we show how we applied the steps described in [1] and adapted this system in order to include it as module in a QA system architecture. Lastly, we show the results obtained, which point out significant growth in precision.

## 1 Introduction

**Recognizing textual entailment** RTE[1] [3] is the task of deciding, given two text fragments, whether the meaning of one text is entailed (can be inferred) from the other text. The aim in defining this task was to create an application-independent framework for assessing means of capturing major semantic inferences needed in many NLP applications. Examples of such applications are: Information Retrieval (IR), Question Answering (QA), Information Extraction (IE), and Text Summarization (SUM).

Formally, textual entailment is defined in [1] as a directional relation between two text fragments, termed *T - the entailing text*, and *H - the entailed text*. It is then said that *T* entails *H* if, typically, a human reading *T* would infer that *H* is most likely true. This definition is based on (and assumes) common human understanding of language as well as common Background Knowledge.

TE systems compete each year in the RTE competition, organized by PASCAL[2] (Pattern Analysis, Statistical Modeling and Computational Learning) - the European Commission's IST-funded Network of Excellence for Multimodal Interfaces.

**Question Answering** (QA) Systems are one of the main research topics in the Natural Language Processing field. These systems not only employ various discourse

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

Received 17/10/07
Accepted 07/12/07
Final Version 16/01/08

analysis and processing tools, but also require theoretical studies and formalizations of many language issues, like questions structure and implied knowledge. QA systems receive natural language queries and not keywords and return precise answers and not documents as output. Finding an answer to a question relies heavily on two things: identification of the required information, and the quantity and quality of information available, therefore on the corpus from which the information can potentially be found.

One of the competitions for QA systems is organized within CLEF (Cross-Language Evaluation Forum). CLEF supports the development of applications for digital libraries by creating an infrastructure for the testing, tuning and evaluation of Information Retrieval systems operating in European languages, both in monolingual and cross-lingual contexts. Within the QA@CLEF evaluation track, we have been participating since 2006 with a Romanian-English multilingual system. Having been confronted with the problem of semantic variability, this year we decided to include an English TE module in the QA system.

The results obtained by using the English TE system within the English QA system have proven to be encouraging, by producing significant growth in precision. Therefore, we decided to build a TE system for Romanian which to use within the Romanian QA system. In order to do that, we tried to replace each of the components of the English system with ones that work on Romanian. The original idea consisted in adapting all components previously used for the English TE system for Romanian. However, this approach generated many problems, since Romanian is not a widely used language and thus the linguistic resources available for it are rather sparse. After repeated trials aimed at adapting modules and resources, we decided to build a generic system using the general tools (such as GATE which is available in 9 languages) and resources that exist for many languages (as WordNet and Wikipedia). Out aim is to build a competitive baseline system, which can be improved for any language with additional specific rules. What follows is a description of the components of the generic system. The evaluation is performed on two languages: English and Romanian. Naturally, we expect a difference in the accuracy score of the system between the two languages, since the English WordNet has 117.659 synsets,[3] and Romanian only approximately 45.000 synsets. Also, the English and Romanian Wikipedia are incomparable in information volume[4] (the English Wikipedia has over 2 million articles and the Romanian Wikipedia only 94 thousands). Subsequently, for the system test we used all datasets[5] from the RTE3 competition and in order to test the system behavior on Romanian we translated the English pairs to Romanian.

## 2 The Generic Textual Entailment System

The main idea is to transform the hypothesis making use of extensive semantic knowledge from sources like WordNet, Wikipedia, and a database of acronyms.

[3] http://wordnet.princeton.edu/man/wnstats.7WN
[4] http://meta.wikimedia.org/wiki/List_of_Wikipedias
[5] http://www.pascal-network.org/Challenges/RTE3/Datasets/

Additionally, we built a system to acquire the extra Background Knowledge needed and applied complex grammar rules for rephrasing in English. These grammar rules can be translated into any language, and we will see how this is done for Romanian.

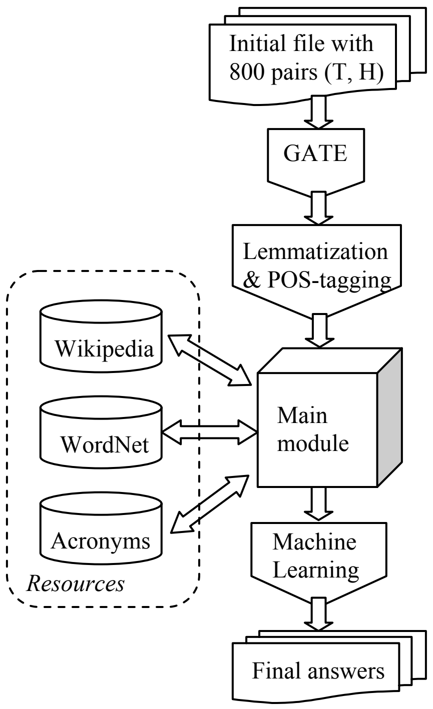

**Fig. 1.** The process requires an initial pre-processing, followed by the execution of a main module. This uses the output of the first phase and obtains in the end an output that is used as input for two machine learning algorithms. The machine learning algorithms eventually classify all pairs

In the first phase, the initial file is split into 800 pairs of Text and Hypothesis. All these files are then processed with GATE and the contained named entities are identified. The following step consists in applying a module that identifies the lemma and the part of speech for every word. The main module uses all this information and expands the list of words from H and T using Wikipedia, WordNet and the Acronyms resources. Eventually, the main module applies the semantic rules and calculates the fitness for every pair (T, H). The final answer regarding the truth value of the entailment relation between T and H is decided by applying two machine learning algorithms, C4.5 [5] and SMO [6] for Support Vector Machine (SVM). We will further observe the use of each of these components.

## 3.1 Pre-processing

**GATE**

GATE (General Architecture for Text Engineering [2]) has the role of identifying named entities in the text and hypothesis. GATE can identify the following types of named entities: person names, countries, locations, organizations, dates, numbers, etc. The rule involving the identification of proper names, numbers and dates was very important to our English system architecture (around 16% of the total system accuracy was gained using this rule). This was an important motivation for its adaptation. The reason for which we used GATE was the number of plugins available for processing in 10 languages: English, French, German, Spanish, Italian, Chinese, Arabic, Romanian, Hindi and Cebuano.

What we used in the generic TE system was a combination between the NEs of the current language of the system used and the English lists of NEs, since the English database is more complex and many NEs are names of persons that have the same form in all languages.

**Lemmatization**

In order to perform the lemmatization, we apply a Perl module that uses a database containing on separate lines all possible forms for every lemma. This phase also includes the identification of the part-of-speech for every word. As it will further be seen, POS-tagging is important when applying the negation rules, since only verbs can be influenced by negative or positive contexts.

## 3.2 Main Module

The main module receives separate files for each text-hypothesis pair from the initial data (test or development). For each of the text snippets, the corresponding file contains the lemmas of the words and the marked named entities. We further proceed to identifying and eliminating the stop words using a language dependent list with stop words (this list must be created for every language, and usually contains words with high frequency like articles, prepositions and so on). The remaining resources used are WordNet, the Acronyms Database and the Background Knowledge. They are used to expand each remaining word from the hypothesis to a list of similar or related terms and thus increase the probability to find the initial word or any of its equivalents in the list of words from the text.

**WordNet**: Nouns, verbs, adjectives and adverbs are grouped into sets of cognitive synonyms (synsets), each expressing a distinct concept [4]. Synsets are interlinked by means of conceptual-semantic and lexical relations,[6] such as synonymy, antonymy, hypernymy, etc. WordNet is presently available in 15 languages.

The **acronyms' database** helps our program to find relations between the acronym and its meaning, for example "US - United States". For English, we use a database

[6] http://wordnet.princeton.edu/

with acronyms.[7] For Romanian, we automatically extracted a list of acronyms from a collection of Romanian newspaper articles on topics related to economics and politics. A list of acronyms for Romanian, including the domains of economics, politics etc. can be found also on the web.[8] We think such lists exist for all languages and if not, they can be relatively easily built using a large collection of newspaper articles in the interest language and a set of specific extraction patterns.

**The Background Knowledge** was used in order to expand the named entities and the numbers. It was built semi-automatically, and it used a module in which language could be set according to the current system working language. Thus, the corresponding Wikipedia[9] could be selected. For every named entity or number in the hypothesis, the module extracted from Wikipedia a file with snippets with information related to them.

It is possible that for languages other than English the file with snippets should be empty. In these cases we used the English Wikipedia, which is larger. Examples of possible relations are presented below:

**Table 1.** Background Knowledge

| Initial NE | Relation | Wikipedia NE |
|---|---|---|
| American | [in] | America |
| America | [is] | United States of America |
| 2 | [is] | February |
| Chinese | [in] | China |

Subsequently, we use this file with snippets and some previously set patterns with the aim of identifying relations between NEs. For all languages envisaged, we can apply the same format of patterns used for English and, additionally, we must add rules taking into account the specifics of the language. The patterns were initially built from the rules used to extract definitions in a Question Answering system. In order to reduce the search space and the complexity of rules, the latter have been categorized in six types:

1. **"Is" Definitions** containing the verb "is":
   Example: "*Abbreviation for Hyper Text Mark Up Language, HTML* **is** *also a protocol used by World Wide Web*".
2. **Specific Verbs Definitions** containing verbs, different from "is". The verbs are "denote", "show", "state", "represent", "define", "specify", "consist", "name", and "permit".
   Example: "*Electronic mail* ***represents*** *sending messages through electronic networks*".
3. **Punctuation Definitions** which use punctuation signs like the dash "-" or brackets "()".
   Example: "*Bit – shortcut for binary digit*".

[7] http://www.acronym-guide.com
[8] http://www.abbreviations.com/acronyms/ROMANIAN
[9] http://en.wikipedia.org/wiki/Main_Page

4. **Layout 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.
   Example:

| | |
|---|---|
| Data organizing | The simplest method is the sequential one. |

5. **Anaphoric definitions**, when the defining term is expressed in a precedent sentence and it is only referred in the definition, usually pronoun references.
   Example: "*...defining the database concept. **It** describes methods of modeling real problems in order to define structures which eliminate redundancy in data collecting*".
6. **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: "*equilateral triangle **i.e.** having all sides equal*".

If such a relation is found, it is saved to an output file. Usually, not all relations are correct, but those that are will help the system at the next run.

Our patterns identify two kinds of relations between words:

- "is", when the module extracts information using the grammar rules presented above;
- "in" when information extracted with rules contains also specific words which characterize the inclusion: ***in, part of, included, region of***, etc.

In this case, the current node does not influence the fitness of the pair for the [is]-type relations, and the fitness receives some penalties for the [in]-type relation.

For Romanian, similar to the English approach, we also built six types of rules in order to identify definition contexts.

**Semantic Variability Rules**: negations and context terms

For every verb from the text and hypothesis we consider a Boolean value which indicates whether the verb has a negation or not, or, equivalently, if it is related to a verb or adverb **diminishing** its sense or not. For that, we use the POS-tags and a list of words we consider as introducing a negation: "*no*", "*don't*", "*not*", "*never*", "*may*", "*might*", "*cannot*", "*should*", "*could*", etc. For each of these words we successively negate the initial truth-value of the verb, which by default is "*false*". The final value depends on the number of such words.

Since the mapping is done for all verbs in the text and hypothesis, regardless of their original form in the snippet, we also focused on studying the impact of the original form of the verb on its overall meaning within the text. Infinitives can be identified when preceded by the particle "*to*". Observing this behavior, one complex rule for negation was built for the particle "*to*" when it precedes a verb. In this case, the sense of the infinitive is strongly influenced by the active verb, adverb or noun before the particle "*to*", as follows: if it is being preceded by a verb like "*allow*", "*impose*", "*galvanize*" or their synonyms, or adjective like "*necessary*", "*compulsory*", "*free*" or their synonyms or noun like "*attempt*", "*trial*" and their synonyms, the meaning of the verb in infinitive form is stressed upon and becomes **certain**. For all other cases, the particle "*to*" diminishes the certainty of the action expressed in the infinitive-form verb. Based on the synonyms database with the

English thesaurus,[10] we built two separate lists – one of **certainty stressing** (preserving) – "*positive*" and one of **certainty diminishing** – "*negative*" words. Some examples of these words are "*probably*", "*likely*" – from the list of **negative** words and "*certainly*", "*absolutely*" – from the list of **positive** words.

For the Romanian version of the TE system, we identified negation rules and context influencing words and introduced similar rules. For negation, we consider one of the following words (pure form of negation or a modal verb in indicative or conditional form): "*nu*", "*poate*". Subjunctives are identified by the fact that they are preceded by the particle "*să*". In this case, if the subjunctive is preceded by a word like "*permite*", "*impune*", "*indica*", "*propune*" or their synonyms, adjectives like "*necesar*", "*obligatoriu*", "*liber*" or their synonyms, or nouns like "*încercare*", "*posibilitate*", "*opțiune*" and their synonyms, the meaning becomes positive. For the context influencing words, we built, as for the English system, two lists, one containing words like "*sigur*", "*absolut*", "*categoric*", "*cert*", "*precis*", "*inevitabil*", "*infailibil*" for **context stressing words** and "*probabil*", "*posibil*", "*fezabil*", "*realizabil*", "*practicabil*" – for **context diminishing words**.

**Rule for Named Entities** from hypothesis without correspondence in text
Additionally, we have a separate rule for named entities from the hypothesis without correspondence in the text. If the word is marked as named entity by GATE, we try to use the acronyms' database or obtain information related to it from the background knowledge. In the event that even after these operations we cannot map the word from the hypothesis to one word from the text, we increase a value that counts the problems regarding the named entities in the current pair. We then proceed to calculating a fitness score measuring the syntactic similarity between the hypothesis and the text, further used as one of the features that the two classifiers used are trained on.

## 3.2 Fitness Calculation

The main idea is to see in which position we find the expanded word from the hypothesis in the text. Below there is an illustration of the manner in which the fitness is calculated for pair 2 of the RTE3 test set:

```
<pair id="2" entailment="NO" task="IE" length="short" >
        <T>The sale was made to pay Yukos' US$ 27.5 billion tax bill,
        Yuganskneftegaz was originally sold for US$9.4 billion to a little known
        company Baikalfinansgroup which was later bought by the Russian
        state-owned oil company Rosneft .</T>
        <H> Yuganskneftegaz costed US$ 27.5 billion.</H>
</pair>
```

In this case, the hypothesis is transformed as follows:

1. After eliminating the stop words, we obtain the following list of keywords, which contains the lemmas of the words in the hypothesis:

[10] http://thesaurus.reference.com/

{*Yuganskneftegaz, cost, US$, 27.5, billion*}

2. This list is then expanded using the English WordNet, resulting in the following list:
{*Yuganskneftegaz,* {*cost, price, toll*}*, US$, 27.5,* {*billion, one thousand million, one million million, trillion, jillion, zillion*}}
3. In what follows, the expanded list is enlarged by using the Background Knowledge (BK). In the BK, we find *Yuganskneftegaz [in] Russia* and we replace *Yuganskneftegaz* with the list {*Yuganskneftegaz, Russia*}.
4. Lastly, using the acronyms collection, we further expand the list of terms for US with *United States.*

Eventually, the complete list is: {{*Yuganskneftegaz, Rusia*}, {*cost, price, toll*}, {*US$, United States Dollar*}, *27.5*, {*billion, one thousand million, one million million, trillion, jillion, zillion*}}.

Using this list, we build a matrix containing the appearances of words from the hypothesis in the text without stop words:

**Table 2.** Mapping of the Hypothesis to the Text

| Word Number | Word Lexical Family | Positions in Text |
|---|---|---|
| 1 | Yuganskneftegaz, Rusia | 12 |
| 2 | cost, price, toll | 3, 13, 33 |
| 3 | US$, United States Dollar | 7 |
| 4 | 27.5 | 8 |
| 5 | billion, one thousand million, one million million, trillion, jillion, zillion | 9,17 |

The formula for calculating the **fitness** is the following:

$$Fitness = \frac{\sum_i \max \frac{1}{abs(PositionInText_i - PositionInText_{i-1})}}{NumberOfWords - 1} \quad (1)$$

Where *i* represents the "Word Number" from the above table, and takes values between 2 and the maximum value of "Word Number" (in this case 5). For the example considered, using formula (1), the result is:

$$Fitness = \frac{1 + \frac{1}{4} + 1 + 1}{4} = \frac{3.25}{4} = 0.8125$$

In order to classify the pairs, we train two classifiers – C4.5 and SMO, using as characteristics the fitness score, the number of direct matches, number of indirect matches and number of problems regarding the matching between the Named Entities in the hypothesis and the text.

### 3.2 Results

In order to evaluate the quality of the generic system, we used two sets of data from the development and testing parts of the RTE3 competition. Every set consists of 800 texts - hypothesis pairs. The output of main module serves as input for two machine learning algorithms: SVM and C4.5. The results are close: SVM with precision equal with 0.634 and C4.5 with precision equal with 0.619. For SVM we can see below, the results of the predicted classification into Yes and No entailment against the gold standard.

**Table 3.** Detailed results using SVM classification

| Class | Precision | Recall | F-Measure |
|---|---|---|---|
| YES | 0.620 | 0.739 | 0.674 |
| NO | 0.656 | 0.523 | 0.582 |
| YES+NO | **0.634** | **0.631** | **0.628** |

In order to observe the system behavior, we trained and classified the pairs using SVM on two languages: English and Romanian. The results shown for the development data are given for 66% of the set used for development and the rest used for testing. For the Romanian system, we translated both the development and test sets of pairs into Romanian. The data in the table below show that the results of the system running on Romanian are lower than the ones obtained for English. The reasons for this difference are the volume and quality of the resources available in both languages: WordNet and Wikipedia.

**Table 4.** Evaluation results on English and on Romanian using SVM

| Language | Development Data | Test Data |
|---|---|---|
| English | 0.648 | 0.634 |
| Romanian | 0.567 | 0.561 |

It is important to point out the fact that when compared to the results of the systems which participated in the RTE3 competition this year, our system is among the 10 best from 26 groups, over the average of all competitors results.

## 4 Using the TE System in the QA Competition

Information within a corpus can be expressed in a large variety of ways. QA systems must resolve this semantic variability problem, as they must identify texts from which the expected answer can be inferred. A good solution to this problem could be using a TE system and pursuing the steps described as follows:
Being given the question [1]:

***Q:*** *"Who is John Lennon's widow?"*
It can be transformed in a statement with a variable PERSON:
***Statement:*** *PERSON is John Lennon's widow.*

Among the answers containing the key expression John Lennon, the following could be found:

***Answer (Text):*** *"Yoko Ono unveiled a bronze statue of her late husband, John Lennon, to complete the official renaming of England's Liverpool Airport as Liverpool John Lennon Airport."* From this text, a candidate for the variable PERSON can be extracted – *Yoko Ono*. The hypothesis can be built by replacing the variable PERSON with the found candidate.

***Hypothesis:*** *"Yoko Ono is John Lennon's widow".*

The proving of whether the candidate term is correct and the answer to the question is right is done by evaluating the entailment relation between the Text and the Hypothesis.

## 4.1 Why Use a TE Module in a Question Answering System?

The aim in using the TE system as a module in the general architecture of a QA system is improving the ranking between possible answers for questions in which the answer type is PERSON, LOCATION, DATE and ORGANIZATION.

The idea is to select all relevant named entities from the extracted snippets for one question and replace with them the variables from the patterns associated to the question. In this manner, we will obtain more hypotheses for one text (represented by the snippet). For every hypothesis, we calculate the fitness score and eventually select the named entity for which the highest value is obtained. Consequently, we compare the highest value obtained for each snippet and select the global best value.

In order to see the behavior of our generic Textual Entailment system, we performed some tests using the Romanian data from the CLEF competition. For question 1: "*Ce faimos romancier, nuvelist şi realizator american de povestiri a trăit între anii 1899 şi 1961?*" (What famous American novelist, and short story writer lived between 1899 and 1961?), the module that was responsible with information retrieval and extraction, returned two snippets:

**S1:** "*Petru Popescu este un romancier, scenarist şi realizator de filme american de origine română. A emigrat în Statele Unite ale Americii în anii 1980, unde s-a impus drept romancier şi autor de scenarii ale unor filme de la Hollywood.*" ("*Petru Popescu is an American novelist, script writer and movie maker of Romanian origin. He emigrated to the United States around the 80s, where he imposed as novelist and Hollywood movies script writer.*")

**S2:** "*Americanul Ernest Hemingway (1899-1961), autor de povestiri, nuvelist şi romancier, şi romancierul rus Yuri Olesha (1899-1976) s-au născut la aceeaşi dată.*" ("*The American Ernest Hemingway (1899-1961), tales and short stories writer, and novelist and the Russian novelist Yuri Olesha(1899-1976), were born on the same date.*")

For the first snippet, S1, we have only one possible answer, which is *Petru Popescu.* Our hypothesis will be: *Petru Popescu, faimos romancier, nuvelist, realizator de povestiri American, a trăit între anii 1899 şi 1961.* Since the hypothesis contains the numbers 1899 and 1961 which don't appear in snippet S1, we use the named entity rule, obtaining a fitness score of 0.52 and a number of named entities with problems of 2. Together with the discussed features, the test pair is introduced in the classifiers as test case. After running the system, the pair is classified with NO entailment.

In the second snippet we have two named entities of type PERSON: *Ernest Hemingway* and *Yuri Olesha*. We obtain two hypotheses:

**H2_1:** *Ernest Hemingway, faimos romancier, nuvelist, realizator de povestiri American, a trăit între anii 1899 şi 1961.*

**H2_2:** *Yuri Olesha, faimos romancier, nuvelist, realizator de povestiri American, a trăit între anii 1899 şi 1961.*

The fitness for pair (*H2_1, S2*) is 0.75, and for pair (*H2_2, S2*) is 0.647. Together with the discussed features, the two test pairs are introduced in the classifiers as test cases, and both pairs are classified with YES. Since the fitness for the first NE is greater than the fitness of the second NE, we conclude that it is possible for both NEs to represent the correct answer, but the probability that *Ernest Hemingway* should be the correct answer is greater than the probability that *Yuri Olesha* should be the correct answer.

For the types specified, we built specific patterns according to the answer type:

**Table 5.** Patterns built for Location, Date and Organization answers types

| | | |
|---|---|---|
| LOCATION | Where was she born? | She was born **in** LOCATION. |
| DATE | When was the reorganized edition of the poem published? | The reorganized edition of the poem was published **at** DATE. |
| ORGANIZA TION | What computer software company headquartered in San Jose was founded in 1982? | ORGANIZATION, **a** computer software company headquartered in San Jose was founded in 1982. |

### 4.2 Results

Adding the TE module in the QA system improves the capability of the QA system to choose with a higher probability the right answer in the case of complex statements, which express the same idea, but with different actors and contexts. However, in the case of fragments which do not represent coherent statements, the TE module within the QA system is useless.

Including a TE module in a QA system results in clear improvements in the quality of the latter. The tests performed on Romanian, in the case of questions with answers of type PERSON and LOCATION, show an increase in accuracy of up to 5%.

## 5 Conclusions

Our work offers a solution for implementing a generic TE system. The quality of the implemented system depends on the quality of resources such as WordNet and

Wikipedia for a specific language. Additionally, we use lists of stop words, acronyms, words that create positive contexts and negative contexts. Our aim in this approach was to offer a baseline for any language for which a TE system can be implemented.

We also showed how this generic system was used as module within a Romanian QA system, resulting in improved ranking between the possible answers for questions of the type PERSON and LOCATION.

In the future, we plan to further develop the system in order to also be able to process questions with answers of type DATE and ORGANIZATION. Furthermore, we will try to test the performance of the TE system for Spanish.

## References

1. Bar-Haim, R., Dagan, I., Dolan, B., Ferro, L., Giampiccolo, D., Magnini B., Szpektor, I.: The Second PASCAL Recognising Textual Entailment Challenge. In Proceedings of the Second PASCAL Challenges Workshop on Recognizing Textual Entailment. Venice. Italy. (2006)
2. Cunningham, H., Maynard, D., Bontcheva, K., Tablan, V.: GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications. Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics (ACL'02). Philadelphia, July (2002)
3. Dagan I., Glickman O., Magnini, B.: The PASCAL Recognising Textual Entailment Challenge. In Quiñonero-Candela et al., editors, MLCW 2005, LNAI Volume 3944. Springer-Verlag (2006) 177-190
4. Fellbaum, C.: WordNet: An Electronic Lexical Database. MIT Press, Cambridge, Mass. (1998)
5. Platt, J.: Fast Training of Support Vector Machines using Sequential Minimal Optimization. Advances in Kernel Methods - Support Vector Learning, B. Schoelkopf, C. Burges, and A. Smola, eds., MIT Press (1998)
6. Quinlan, J. R.: C4.5: Programs for machine learning. San Francisco: Morgan Kaufmann (1993)