University "Alexandru Ioan Cuza" of Iaşi
Faculty of Computer Science

TECHNICAL REPORT

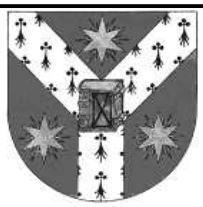
Textual Entailment

(Ph.D. Thesis)

Adrian Iftene

TR 09-02, October 2009

ISSN 1224-9327



Universitatea "Alexandru Ioan Cuza" Iaşi
Facultatea de Informatică
Str. Berthelot 16, 6600-Iaşi, Romania
Tel. +40-32-201090, email: bibl@infoiasi.ro

TEXTUAL ENTAILMENT

Adrian Iftene

Thesis submitted to the "Al. I. Cuza" University of Iaşi
for the degree of Doctor of Philosophy
in Computer Science

Department of Computer Science
"Al. I. Cuza" University of Iaşi
Iaşi, Romania
March, 2009

Adrian Iftene
Department of Computer Science
"Al. I. Cuza" University of Iaşi
Iaşi, Romania
E-mail: adiftene@info.uaic.ro

Prof. Dr. Dorel Lucanu, Chair ("Al. I. Cuza" University of Iaşi)
Prof. Dr. Dan Cristea, Supervisor ("Al. I. Cuza" University of Iaşi)
Prof. Dr. Dan Tufiş (Romanian Academy, Bucharest Branch)
Conf. Dr. Gabriela (Şerban) Czibula ("Babeş-Bolyai" University of Cluj-Napoca)
Prof. Dr. Bernardo Magnini (Trento University, Italy)

To my family

Abstract

Recognising Textual Entailment (RTE) was proposed in 2005 as a generic task, aimed at building systems capable of capturing the semantic variability of texts and performing natural language inferences. These systems can be integrated in NLP applications, improving their performance in fine-grained semantic analysis.

This thesis presents the relevant current work in the field and the author's contribution regarding the creation of a Textual Entailment system. Given two text fragments, the task of Textual Entailment answers the question whether the meaning of one text can be inferred (entailed) from another text.

The second chapter presents an overview of the main approaches used by systems participating in RTE competitions from 2005 to 2008, with their strengths and weaknesses. The methods used for identifying the entailment relation are varied, ranging from simple methods such word overlap, to complex methods that use semantic knowledge, logical provers, probabilistic models or learning models. Since the 2007 edition, the participating systems demonstrated their maturity and utility in others tracks such Question Answering and Answer Validation Exercise of CLEF2007.

The author's contribution is outlined in the third and fourth chapters, where a detailed presentation of the Textual Entailment developed system is given. The system's performance was high in RTE competitions and was ranked as one of the top Textual Entailment applications. In order to improve on the running time, the system architecture adopted was the peer-to-peer model, and the system behaviour is similar to that of a computational Grid.

Chapter five presents how the Textual Entailment system was used with success in the QA@CLEF competition, in order to improve the quality of the Question Answering system for the Main task and the Answer Validation Exercise.

Acknowledgements

This thesis would not have been possible without the help and support of many people: professors, students, colleagues, RTE-competitions organizers and participants.

First of all, I would like to thank Prof. Dr. Dan Cristea, my supervisor, for introducing me to the field of Computational Linguistic, and giving me the impetus to compete against others in order to better myself.

A big 'thank you' goes out to my colleagues Ionuț Pistol and Diana Trandabăț, with whom I have collaborated during my PhD years; they always offered very good comments, remarks and suggestion regarding my work.

A special thanks goes to my former student Alexandra Balahur-Dobrescu with whom I collaborated in 2007 and to members of the NLP group of Iasi with whom I collaborated since 2005. In alphabetical order they are Iustin Dornescu, Maria Husarciuc, Alex Moruz, and Marius Răschip.

This thesis would not be as it is without the useful comments and suggestions of several people who advised me at crucial points in my research. In alphabetical order they are Liviu Ciortuz, Cornelius Croitoru, Dorel Lucanu, and Dan Tufiş. Thank you very much.

An important contribution to my work comes from my continuously collaboration with my students during classes and projects. Their innovative and non-conventional solutions and ideas made me see a fresh perspective.

Important to my research was also the support received: a scholarship from SIEMENS VDO Iaşi and material support from projects: SIR-RESDEC (PN2 number D11007), GRAI (CEEX number 74), LT4eL (STREP 027391 in FP6-2004-IST-4), ROTEL (CEEX number 29).

Last and most importantly, I would like to mention my family. There are not enough words to thank them for their help, support and everything they did for me.

There certainly are persons that I should thank and that I have forgotten, and for that I apologise. Thank you!

Table of Contents

List of Figures

List of Tables

1. Introduction

1.1. *Motivation*

The recognition of textual entailment is one of the recent challenges of the Natural Language Processing (NLP) domain and one of the most demanding. The systems participating in this competition must do something more than the systems from other NLP competitions: to prove capabilities of understanding how language works. Indeed, as specified in (Bos and Marker, 2005) recognizing entailment bears similarities to Turing's famous test to assess whether machines can think, as access to different sources of knowledge and the ability to draw inferences seem to be among the primary ingredients for an intelligent system. Moreover, many NLP tasks have strong links to entailment: in Summarization (SUM), a summary should be entailed by the text; Paraphrases (PP) can be seen as mutual entailment between a text T and a hypothesis H; in Information Extraction (IE), the extracted information should also be entailed by the text; in Question Answering (QA) the answer obtained for one question after the Information Retrieval (IR) process must be entailed by the supporting snippet of text.

While the language variability problem is well known in Computational Linguistics, a general unifying framework has been proposed only recently in (Dagan and Glickman, 2004). In this approach, language variability is addressed by defining the notion of *entailment* as a relation that holds between two language expressions (i.e. a text *T* and a hypothesis *H*) if the meaning of *H*, as interpreted in the context of *T*, can be inferred from the meaning of *T*. The entailment relation is directional as the meaning of one expression can entail the meaning of the other, while the opposite may not. *Textual Entailment Recognition* (Dagan et al., 2005) has recently been proposed as an application independent task to capture such inferences.

A few samples of text-hypothesis pairs from the *Forth Recognising Textual Entailment (RTE-4)* Challenge in 2008 are presented below[1]:

ID	Text	Hypothesis	Task	Judgment
10	In the end, defeated, Antony committed suicide and so did Cleopatra, according to legend, by putting an asp to her breast.	Cleopatra committed suicide.	Information Retrieval	Entailment
15	Boris Becker has told a German court that he made financial mistakes 10 years ago but denied	Becker was a tennis	Information Retrieval	Unknown

[1] All examples in this paper are from the corpus released as part of the RTE challenge, keeping the original identifiers. Data from all editions can be accessed at the 2008 challenge site: http://www.nist.gov/tac/tracks/2008/rte/. I will only specify the edition from which the examples are taken.

ID	Text	Hypothesis	Task	Judgment
	deliberately cheating his taxes.	champion.		
435	Marshall spent 10 years on the national speed-skating team, competing at the 1992, 1994 and 1998 Olympics.	Marshall won the 1994 and 1998 Olympics.	Question Answering	Unknown
471	Barely six months after her marriage with the French President, supermodel Carla Bruni has admitted having problems with her "conservative" hubby Nicolas Sarkozy's "right-wing politics".	Carla Bruni is the French President.	Question Answering	Contradiction
624	A mower which cuts the grass by itself is being marketed by a Co. Meath entrepreneur, Patrick McCann.	A mower cuts the grass 100% automatically.	Summarization	Entailment
886	The Koreans oppose the WTO's aim to lower trade barriers for agricultural imports, saying such moves will flood the Korean market with cheap rice and bankrupt Korean farmers.	The Koreans approve of a WTO proposal.	Information Extraction	Contradiction
935	Two U.S. soldiers were also killed in Baghdad and two wounded when their patrol was attacked with a car bomb.	A car bomb attack occurred in Baghdad.	Information Extraction	Entailment

Table 1: Examples of text-hypothesis pairs, taken from the RTE3 test set

From the table above we can see how sets of (text, hypothesis) pairs from test data were obtained using four tasks: Information Retrieval (IR), Question Answering (QA), Summarization (SUM) and Information Extraction (IE). Also, we can see that the possible answers for these pairs can be "Entailment" (when hypothesis can be entailed from text), "Contradiction" (when hypothesis is contradicted by text) and "Unknown" (when the truth of the hypothesis cannot be deduced on a text basis).

1.2. *What is Textual Entailment?*

Within the textual entailment framework, a text *T* is said to entail a textual hypothesis *H* if the truth of *H* can be inferred from *T*. This means that *most people would agree that the meaning of T implies that of H*. Somewhat more formally, we say that *T entails H* when some representation of *H* can be "matched" (modulo some meaning-preserving transformations to be defined below) with some (or part of a) representation of *T*, at some level of granularity and abstraction.

Below, we have informal variants for textual entailment definition. Definition from (Dagan et al., 2005) is:

Variant 1 We say that a text T entails a hypothesis H if, typically, a human reading T would infer that H is most likely true.

A common definition of entailment in formal semantics (Chierchia. and McConnell-Ginet, 2001) is the following:

Variant 2 A text *T entails another text H* if *H* is true in every circumstance (possible world) in which *T* is true.

According to approach of participants in the RTE challenge, below we can see different definitions for textual entailment concept:

Variant 3 *T entails H* if we have a sequence of transformations applied to *T* such that we can obtain *H* with an overall cost below a certain threshold, empirically estimated on the training data (Kouylekov and Magnini, 2005).

Variant 4 *T entails H* if we succeed to extract a maximal subgraph of XDG_T that is in a subgraph isomorphism relation with XDG_H, through the definition of two functions f_C and f_D (Pazienza et al, 2005).

Variant 5 *T entails H* if the truth of H can be inferred from T within the context induced by T (Guidelines of RTE-4 challenge[2]).

1.3. *Recognising Textual Entailment*

The past *Recognising Textual Entailment* competitions (RTE-1 in 2005, RTE-2 in 2006 and RTE-3 in 2007) were organized by PASCAL [3] (Pattern Analysis, Statistical Modelling and Computational Learning) - the European Commission's IST-funded Network of Excellence for Multimodal Interfaces.

In 2008, at it's forth edition, the challenge was organized within the Text Analysis

[2] RTE-4 Guidelines: http://www.nist.gov/tac/tracks/2008/rte/rte.08.guidelines.html
[3] Pascal: http://www.pascal-network.org/

Conference[4]. The Text Analysis Conference (TAC) is a new series of evaluation workshops organized to encourage research in Natural Language Processing and related applications, by providing a large test collection, common evaluation procedures, and a forum for organizations to share their results.

Year-to-year, new features were added in every new competition. In the following we will sketch the main characteristics of every challenge as well as the new added.

1.3.1. First Challenge[5]

The *RTE* Challenge starts in 2005 (Dagan et al., 2005) and it is an attempt to promote an abstract generic task that captures major semantic inference needs across applications. The task requires recognising, given two text fragments, whether the meaning of one text can be inferred (entailed) from another text.

In textual entailment competition, participants in the evaluation exercise are provided with pairs of small text snippets (one or more sentences in English), which were called Text-Hypothesis (T-H) pairs. They must build a system that should say, for each pair, if there is entailment or not.

As a first step towards the goal for this challenge, a dataset of Text-Hypothesis (T-H) pairs of small text snippets was created, corresponding to the general news domain. Examples were manually labelled for entailment – whether T entails H or not – by human annotators, and were divided into Development and Test datasets. Participating systems were asked to decide for each T-H pair whether T indeed entails H or not, and results were compared to the manually created gold standard.

The dataset was collected with regard to different text processing applications, like Information Retrieval (IR), Comparable Documents (CD), Reading Comprehension (RC), Question Answering (QA), Information Extraction (IE), Machine Translation (MT), and Paraphrase Acquisition (PP). Each portion of the dataset was intended to include typical T-H examples that correspond to success and failure cases of the actual applications. The collected examples represent a range of different levels of entailment reasoning, based on lexical, syntactic, logical and world knowledge, at different levels of difficulty. The distribution of examples in this challenge has been somewhat biased to choosing non-trivial pairs, and also imposed a balance of

[4] TAC Conference: http://www.nist.gov/tac/
[5] RTE: http://pascallin.ecs.soton.ac.uk/Challenges/RTE/

True and False examples. For this reason, the systems' performances in applicative settings might be different than the figures for the challenge data, due to different distribution of examples in particular applications. Yet, the data does challenge systems to properly handle a broad range of entailment phenomena. In the end, 567 examples were in the development set and 800 were in the test set, and evenly split to True/False examples.

Finally, the task definition and evaluation methodologies were at their beginning and clearly not mature yet.

The first four results from RTE-1 are presented in the table below:

Group	**Accuracy**	**Methods**
Department of Computer Science Universidad Autonóma de Madrid, Spain (Pérez and Alfonseca, 2005)	0.700	Application of BLEU algorithm
Department of Language Sciences ITC-IRST University Ca' Foscari Venice, Italy (Delmonte et al., 2005)	0.606	Text Understanding Grammatical Relations Semantic Roles
The MITRE Corporation, USA (Bayer et al., 2005)	0.586	Linguistic analysis and inference Machine Translations Alignment
Computer Science Department, Bar Ilan University, Israel (Glickman et al., 2005)	0.586	Web-based Estimation of Lexical Entailment Probabilities

Table 2: Best results in RTE-1

In this competition most of the groups focused on words overlap between T and H, and for that they used stemming, lemmatization, part-of-speech tagging, and applied statistical measures. Advanced methods tried to find a relation between words using semantic knowledge from WordNet, semantic roles or statistical information from Web data. Also, the modelling and use of world knowledge was at the beginning. The systems using logical prover, probabilistic models or supervised learning models were also at the beginning and had difficulties in modelling and running.

1.3.2. Second Challenge[6]

Following the success and impact of RTE-1, the main goal of the second challenge, held in 2006

[6] RTE-2: http://pascallin.ecs.soton.ac.uk/Challenges/RTE2/

(Bar-Haim et al., 2006), was to support the continuity of research on textual entailment. Their main focus in creating the RTE-2 dataset[7] was to provide more "realistic" text-hypothesis examples, based mostly on outputs of actual systems. As in the previous challenge, the main task is judging whether a hypothesis H is entailed by a text T. Again, the examples represent different levels of entailment reasoning, such as lexical, syntactic, morphological and logical. Data collection and annotation processes were improved this year; including cross-annotation of the examples across the organizing sites (most of the pairs were triply annotated). The data collection and annotation guidelines were revised and expanded.

The RTE-2 dataset consists of 1600 text-hypothesis pairs, divided into a development set and a test set, each containing 800 pairs. The organizers focused on four out of the seven applications that were present in RTE-1: Information Retrieval (IR), Information Extraction (IE), Question Answering (QA), and multi-document summarization (SUM). Within each application setting, the annotators selected positive entailment examples, as well as negative examples (50%-50% split, as in RTE-1). In total, 200 pairs were collected for each application in each dataset and each pair was annotated with its related task (IE/IR/QA/SUM) and entailment judgment (YES/NO, released only in the development set). In order to make the challenge data more accessible, the organizer also provided some pre-processing for the examples, including sentence splitting and dependency parsing.

The main task in the RTE-2 challenge was classification – entailment judgment for each pair in the test set. The evaluation criterion for this task was accuracy – the percentage of pairs correctly judged. A secondary task was ranking the pairs, according to their entailment confidence. In this ranking, the first pair is the one for which the entailment is most certain, and the last pair is the one for which the entailment is least likely (i.e. the one for which the judgment as "NO" is the most certain). A perfect ranking would place all the positive pairs (for which the entailment holds) before all the negative pairs. This task was evaluated using the Average precision measure, which is a common evaluation measure for ranking (e.g. in information retrieval) (Voorhees and Harman, 1999).

First four results are presented in next table:

Group	**Accuracy**	**Methods**
Language Computer Corporation,	0.7538	Classification-based approach

[7] RTE-2 data sets: http://pascallin.ecs.soton.ac.uk/Challenges/RTE2/Datasets/

Group	Accuracy	Methods
Richardson, Texas, USA (Hickl et al., 2006)		Lexico-semantic information derived from text processing applications with a large collection of paraphrases
Language Computer Corporation Richardson, Texas, USA (Tatu et al., 2006)	0.7375	Knowledge Representation Logic form, axioms Temporal representation
DISCo, University of Milano-Bicocca and DISP, University of Rome "Tor Vergata", Italy (Zanzotto et al., 2006)	0.6388	Learning algorithm
Human Language Technology Research Institute University of Texas at Dallas Richardson, Texas (Adams, 2006)	0.6262	Lexical overlap Feature extraction (unmapped negations, lexical edit distance)

Table 3: Best results in RTE-2

The better systems in this edition were more oriented to a deeper analysis of the text-hypothesis pairs and use of additional semantic information from various types of resources. The results and tests show that the systems using only the lexical overlap cannot pass over the limit of 60 %.

The big difference between the first two results and the next results comes from the fact that the first system used additional information from a very large corpus automatically collected from the web (Hickl et al., 2006) and the second one use the extra information that comes from a big resource with logical axioms and world knowledge representation (Tatu et al., 2006).

Interesting and encouraging in this edition was the fact that a diversity of methods and heuristics appeared and were used with success, and seemed to be very promising for the next editions.

1.3.3. Third Challenge[8]

RTE-3 followed the same basic structure of the previous campaigns (Giampiccolo et al., 2007). Like in previous editions, something new was introduced i.e. a limited number of longer texts (up

[8] RTE3: http://pascallin.ecs.soton.ac.uk/Challenges/RTE3/

to a paragraph in length) were incorporated in order to have more realistic scenarios. However, the majority of examples remained similar to those in previous challenges, providing pairs with relatively short texts.

As in the previous challenges, the RTE-3 dataset consisted of 1600 text-hypothesis pairs, equally divided into a development set and a test set. The same four applications from RTE-2 – namely IE, IR, QA and SUM – were considered as settings or contexts for the pair's generation. 200 pairs were selected for each application in each dataset. Each pair was annotated with its related task (IE/IR/QA/SUM) and entailment judgment (YES/NO, obviously released only in the development set).

Another innovation was represented by a resource pool[9], where contributors had the possibility to share the resources they used. In fact, one of the key conclusions at the second RTE Challenge Workshop was that entailment modelling requires vast knowledge resources that correspond to different types of entailment reasoning. This resource pool was built in response to competitor's demands, and may serve as a portal and forum for publicizing and tracking resources, and reporting their use.

In addition, an optional pilot task, called "*Extending the Evaluation of Inferences from Texts*" was set up by the US National Institute of Standards and Technology (NIST), in order to explore two other sub-tasks closely related to textual entailment: differentiating unknown entailments from identified contradictions and providing justifications for system decisions. In the first sub-task, the idea was to drive systems to make more precise informational distinctions, taking a three-way decision between "YES", "NO" and "UNKNOWN", so that a hypothesis being unknown on the basis of a text would be distinguished from a hypothesis being shown false/contradicted by a text. As for the other subtask, the goal for providing justifications for decisions was to explore how eventual users of tools incorporating entailment can be made to understand the way decisions were reached by a system, as users are unlikely to trust a system that gives no explanation for its decisions. The pilot task exploited the existing RTE-3 Challenge infrastructure and evaluation process by using the same test set, while utilizing human assessments for the new sub-tasks.

The first four results are in Table 4:

[9] TE Ressource Pool: http://aclweb.org/aclwiki/index.php?title=Textual_Entailment_Resource_Pool

Group	Accuracy	Methods
Language Computer Corporation, Richardson, Texas, USA (Hickl and Bensley, 2007)	0.8000	Discourse Commitments Lexical Alignment Knowledge Extraction
Language Computer Corporation Richardson, Texas, USA (Tatu and Moldovan, 2007)	0.7225	Logic Representation of Events Coreference Resolution NLP Axioms
"Al. I. Cuza" University, Romania (Iftene and Balahur-Dobrescu, 2007a)	**0.6913**	**Tree edit distance algorithm** **Semantic variability** **Grammar rules for rephrasing**
Human Language Technology Research Institute University of Texas at Dallas Richardson, Texas (Adams et al., 2007)	0.6700	Lexical overlap Dependency graphs Supervised learning algorithm

Table 4: Best results in RTE-3

Regarding the pilot with 3-way classification, the first four results are:

Group	Accuracy
Language Computer Corporation, Richardson, Texas, USA (Hickl and Bensley, 2007)	0.73
Language Computer Corporation Richardson, Texas, USA (Tatu and Moldovan, 2007)	0.71
Computer Science Department Stanford University (Chambers et al., 2007)	0.59
"Al. I. Cuza" University, Romania (Iftene and Balahur-Dobrescu, 2007a)	**0.57**

Table 5: Best results in Pilot Task from RTE-3

In this edition the maturity of the systems was shown by better results achieved in comparison with previous editions. From this year, the utility of textual entailment systems was shown in Question Answering (QA) track and in Answer Validation Exercise (AVE) from CLEF2007, where it was used by QA systems and AVE systems in answer validation and ranking.

1.3.4. Fourth Challenge[10]

In 2008, the competition was organized by NIST [11] (National Institute of Standards and Technology) in TAC[12] (Text Analysis Conference). TAC is a new series of evaluation workshops organized to encourage research in Natural Language Processing and related applications, by providing a large test collection, common evaluation procedures, and a forum for organizations to share their results. TAC 2008 had three tracks: Question Answering, Recognising Textual Entailment, and Summarization. Each track is a continuation of an evaluation series previously organized under different frameworks, though specific tasks in a track may differ from previous years.

The 2008 RTE Track included the three-way classification task piloted in RTE-3 as main task. The goal of making a three-way decision of "YES", "NO" and "UNKNOWN" is the same as in the previous edition i.e. to drive systems to make more precise informational distinctions; a hypothesis being unknown on the basis of a text should be distinguished from a hypothesis being shown false/contradicted by a text. The classic two-way RTE task was also offered, in which the pairs where T entailed H were marked as ENTAILMENT, and those where the entailment did not hold were marked as NO ENTAILMENT. The descriptions of the tasks are presented below:

The **three-way RTE task** is to decide whether:

- ***T entails H*** - in which case the pair will be marked as ENTAILMENT;
- ***T contradicts H*** - in which case the pair will be marked as CONTRADICTION;
- The ***truth of H cannot be determined on the basis of T*** - in which case the pair will be marked as UNKNOWN.

The **two-way RTE task** is to decide whether:

- ***T entails H*** - in which case the pair will be marked as ENTAILMENT;
- ***T does not entail H*** - in which case the pair will be marked as NO ENTAILMENT.

The RTE-4 dataset was made of 1000 pairs (300 each for IE and IR, 200 each for SUM and QA). The number of pairs was increased to 300 in two of the application settings, namely Information Extraction and Information Retrieval, as they have proven to be more difficult than the analysis of the results in previous challenges. In this competition there was no development data set, and the participants used only the past RTE data for training.

[10] RTE-4: http://www.nist.gov/tac/tracks/2008/rte/index.html
[11] NIST: http://www.nist.gov/
[12] TAC: http://www.nist.gov/tac/

The first four results from 2-way and 3-way tasks (Giampiccolo et al., 2008):

Group	Accuracy	Methods
Language Computer Corporation, Richardson, Texas, USA (Bensley and Hickl, 2008)	0.746	Discourse Commitments Lexical Alignment Knowledge Extraction
"Al. I. Cuza" University, Romania (Iftene, 2008c)	**0.721**	**Tree edit distance algorithm** **Semantic variability** **Grammar rules for rephrasing** **Contradiction identification**
Saarland University and DFKI GmbH, Germany (Wang and Neumann, 2008b)	0.706	Dependency trees algorithm Named entities module Tree Skeleton extraction algorithm
LITS, Tsinghua University, Beijing, China (Li F. et al., 2008)	0.659	True entailment recognition: word match, NE match, syntactic and semantic match False entailment recognition: named entities mismatch, quantifier and negation mismatch

Table 6: Best results in 2-way task from RTE-4

Regarding the 3-way task classification, the first four results are:

Group	Accuracy	Methods
"Al. I. Cuza" University, Romania (Iftene, 2008c)	**0.685**	**Tree edit distance algorithm** **Semantic variability** **Grammar rules for rephrasing** **Contradiction identification**
Concordia University, Montreal, Quebec, Canada (Siblini and Kosseim, 2008)	0.616	Ontology alignment Machine learning algorithms
Saarland University and DFKI GmbH, Germany (Wang and Neumann, 2008)	0.614	Dependency trees algorithm Named entities module Tree Skeleton extraction algorithm
LITS, Tsinghua University, Beijing, China	0.588	True entailment recognition

Group	Accuracy	Methods
(Li et al., 2008)		False entailment recognition

Table 7: Best results in 3-way task from RTE-4

In this edition the competitors were oriented on identification of no entailment cases (Iftene, 2008c), (Li et al., 2008). A new direction is related to the on acquiring and aligning ontologies to recognize textual entailment (Siblini and Kosseim, 2008).

We can notice, regarding the results, how the best results and the average values in the 3-way task and in 2-way task are lower than those achieved in last year's competition, even though a comparison is not really possible as the datasets were actually different.

1.3.5. Overall Results

The results obtained until now in the 2-way task of the RTE competitions are presented below, and a year-by-year improvement can be easily observed, even though a comparison is not really possible as the datasets were actually different from edition to edition:

	# of groups	Average Precision	Best Precision	UAIC[13]
2005	16	55.12 %	70.00 %	-
2006	23	58.62 %	75.38 %	-
2007	26	61.14 %	80.00 %	68.13 %
2008	26	57.30 %	74.60 %	72.10 %

Table 8: Overview of the 2-way task in RTE Competition

The results obtained until now by UAIC in the 3-way task of the RTE competitions are presented below, and a year-by-year improvement can be easily observed:

	# of groups	Average Precision	Best Precision	UAIC
2007	10	47.1 %	73.1 %	56.9 %
2008	13	51.0 %	68.5 %	68.5 %

Table 9: Overview of the 3-way task in RTE Competition

[13] UAIC is the term of the system with which I participated in RTE-3 and in RTE-4 competitions. The presentation of this system makes the content of this thesis.

1.4. *Conclusions*

This chapter presents the RTE competitions from 2005 to 2008. We showed the new characteristics of every edition and how the quality of training and test data was improved by year to year.

The number of participating groups increases from year to year showing the positive feedback which RTE has received from the NLP community. Until now a set of interesting approaches have been proposed, but there seems to be still room for improvement, as the average performances of the systems showed.

The next chapter present the most effective methods used until now, with positive and negative aspects.

2. Trends in Textual Entailment

In an overview of the systems participating in the First Textual Entailment challenge, of 2005, we saw that the approaches used are based on word overlap (Herrera, 2005), statistical lexical relations (Bayer et al., 2005), WordNet similarities (Herrera, 2005), syntactic matching (Delmonte et al., 2005), world knowledge (Bayer et al., 2005), logical inference (Akhmatova, 2005), inversion transduction grammars (Wu, 2005), (Jijkoun and de Rijke, 2005), edit distance between parsing trees (Kouylekov and Magnini, 2005), among others (Dagan et al, 2005). The majority of the systems experiment with different threshold and parameter settings to estimate the best performance. The parameter adjustment process is related to the carrying out of numerous experiments and still the settings selected after these experiments may lead to incorrect reasoning.

In the second edition, of 2006, the main directions were generally the same, only algorithms were more sophisticatedly and also the results were better (average precision grew up from 55.12 % in 2005 to 58.62 % in 2006). New directions are related to semantic role labelling (Hickl et al., 2006), Machine Learning classification (Inkpen et al., 2006 and Kozareva, 2006), using of background knowledge (Tatu et al., 2006), acquisition of entailment corpora (Hickl et al., 2006). Some groups tried to detect non entailment, by looking for various kinds of mismatch between the text and the hypothesis. This approach is related to an observation from (Vanderwende et al., 2005), which suggested that it is easier to detect false entailment.

In the third edition, of 2007, we can notice a move toward deep approaches. The groups were oriented on the approaches based on the syntactic structure of Text and Hypothesis, on semantic understanding of the texts and also on verification of the content and new situations and contexts that meet in the test data. A special attention was given to the named entities, where (Tatu and Moldovan, 2007) had special rules for Person names, and where (Iftene and Balahur-Dobrescu, 2007a) had special rules for all named entities. Some form of relation extraction has been introduced: patterns built manually (Chambers et al., 2007), information extracted automatically by a system (Hickl and Bensley, 2007). Also, in comparison to previous editions, now the longer texts need anaphora resolution: (Delmonte, 2007), (Bar-Haim et al., 2007), (Iftene and Balahur-Dobrescu, 2007a).

In the following subsections I make an inventory of the characteristics of the main directions used in the RTE challenges, from 2005 until now, putting in evidence positive and negative aspects.

2.1. *General Algorithms*

2.1.1. Application of the BLEU Algorithm

The first important direction from RTE-1 was an application of the BLEU algorithm for recognising textual entailments. We will see what BLEU algorithm is and how it can be used in recognition of entailments.

2.1.1.1. The BLEU Algorithm

The BLEU[14] algorithm was created by (Papineni et al., 2001) as a procedure to rank systems according to how well they translate texts from one language to another. Basically, the algorithm looks for *n-gram* coincidences between a candidate text (the automatically produced translation) and a set of reference texts (the human-made translations).

The pseudo code of BLEU is as follows:

- For several values of *n* (typically from 1 to 4), calculate the percentage of *n-grams* from the candidate translation that appears in any of the human translations. The frequency of each *n-gram* is limited to the maximum frequency with which it appears in any reference.
- Combine the values obtained for each value of *n*, as a weighted linear average.
- Apply a brevity factor to penalize short candidate texts (which may have *n-grams* in common with the references, but may be incomplete). If the candidate is shorter than the references, this factor is calculated as the ratio between the length of the candidate text and the length of the reference which has the most similar length.

It can be seen from this pseudo code that BLEU is not only a keyword matching method between pairs of texts. It takes into account several other factors that make it more robust:

- It calculates the length of the text in comparison with the length of reference texts. If the candidate text is shorter than the reference texts, this is considered to be an indicative of a poor quality translation and thus, BLEU penalizes it.
- The measure of similarity can be considered as a precision value that calculates how many of the *n-grams* from the candidate appear in the reference texts.

[14] Bilingual Evaluation Understudy

Important is that an *n-gram* that is repeated very often in the candidate text does not increment the score if it only appears a few times in the references.

- The final score is the result of the weighted sum of the logarithms of the different values of the precision, for *n* varying from 1 to 4. It is not interesting to try higher values of *n* since coincidences longer than four-grams are very unusual.

BLEU's output is always a number between 0 and 1. This value indicates how similar the candidate and reference texts are. In fact, the closer the value is to 1, the more similar they are. (Papineni et al., 2001) report a correlation above 96% when comparing BLEU's scores with the human-made scores. A variant of this algorithm has also been applied to evaluate text summarization systems (Lin and Hovy, 2003) and to help in the assessment of open-ended questions (Alfonseca and Pérez, 2004).

2.1.1.2. BLEU Algorithm in Recognising Textual Entailments

The approach in (Pérez and Alfonseca, 2005) consists in using the BLEU algorithm that works at the lexical level, to compare the entailing text (T) and the hypothesis (H). Next, the entailment is judged as true or false according to BLEU's output.

Once the algorithm is applied, they had seen that the results confirm the use of BLEU as baseline for the automatic recognition of textual entailments. Furthermore, they showed that a shallow technique can reach around 50% of accuracy.

In order to recognize entailments using BLEU, the first decision is to choose whether the candidate text should be considered as part of the entailment (*T*) or as the hypothesis (*H*). In order to make this choice, they did a first experiment in which they considered the *T* part as the reference and the *H* as the candidate. This setting has the advantage that the *T* part is usually longer than the *H* part and thus the reference would contain more information that the candidate. It could help the BLEU's comparison process since the quality of the references is crucial and in this case, their number has been dramatically reduced to only one (while in the rest of the applications of BLEU the number of references is always higher).

The output of their algorithm which uses BLEU was taken as the confidence score and it was also used to give the final answer to each entailment pair. They performed an optimization procedure for the development set that chose the best threshold according to the percentage of success of correctly recognized entailments. The value obtained was 0.157. Thus, if the BLEU's

output is higher than 0.157 the entailment is marked as "YES", otherwise as "NO".

The result of 0.7 obtained by the system in RTE-1 shows that the method can be used with success in textual entailment. The main limitations of the system come from the fact that it does not use any syntactical or semantic information, and for a text and a hypothesis with many words in common the answer will be "YES", although the meaning is different.

In the following competitions the *n-gram word similarity* method was very popular among the participating systems (6 systems in 2006, 11 systems in 2007 and over 14 in 2008).

2.1.2. Textual Entailment as Syntactic Graph Distance

Graph distance/similarity measures are widely recognized to be powerful tools for matching problems in computer vision and pattern recognition applications (Bunke and Shearer, 1998) and it was used with success in RTE of 2005 by Pazienza, Pennacchiotti and Zanzotto. Objects to be matched (two images, patterns, text and hypothesis in RTE task, etc.) are represented as graphs, turning the recognition problem into a graph matching task.

Thus, following (Dagan and Glickman, 2004), since the hypothesis *H* and text *T* may be represented by two syntactic graphs, the textual entailment recognition problem can be reduced to graph similarity measure estimation, although textual entailment has particular properties (Pazienza et al., 2005):

a) Unlike the classical graph problems, it is not symmetric;

b) Node similarity can not be reduced to the *label level* (e.g. token similarity);

c) Similarity should be estimated also considering linguistically motivated *graph transformations* (e.g., nominalization and passivization).

The authors (Pazienza et al., 2005) see the textual entailment as a transitive oriented relation holding in one of the following cases:

1. T *semantically subsumes* H (e.g., in *H:* [The cat eats the mouse] and *T:* [the cat devours the mouse], *eat* generalizes *devour*).
2. T *syntactically subsumes* H (e.g., in *H:* [The cat eats the mouse] and *T:* [the cat eats the mouse in the garden], T contains a specializing prepositional phrase).
3. T *directly implies* H (e.g., *H:* [The cat killed the mouse], *T:* [the cat devours the mouse], "*cat devours the mouse*" assume that this mouse was *killed* before).

For the syntactic representation, they rely on the extended dependency graph (*XDG*) (Basili and Zanzotto, 2002). An $XDG = (C, D)$ is basically a dependency graph whose nodes *C* are *constituents* and whose edges *D* are the *grammatical relations* among the constituents. Constituents are lexicalized syntactic trees with explicit *syntactic heads* and *potential semantic governors* (gov). Dependencies in *D* represent typed and ambiguous relations among a constituent, the head, and one of its modifiers. Ambiguity is represented using *plausibility* (between 0 and 1).

Taking this into account, they consider the representations of T and H with corresponding graphs XDG_T and XDG_H, and define a measure $E(XDG_T, XDG_H)$ for the entailment relation. They work under two simplifying assumptions: *H* is supposed to be a sentence completely describing a fact in an assertive or negative way and *H* should be a simple *S-V-O* sentence (subject, verb, object order). Their measure has to satisfy the following properties:

(a) having a range between 0 and 1, assigning higher values to couples that are more likely in entailment relation, and a specific orientation, $E(XDG_T, XDG_H) \neq E(XDG_H, XDG_T)$;

(b) The overlap between XDG_T and XDG_H has to show whether a subgraph of XDG_T implies the graph XDG_H. Linguistic transformations (such as nominalization, passivization, and argument movement), as well as negation, must also be considered, since they can play a very important role.

The problem is to extract the maximal subgraph of XDG_T that is in a subgraph isomorphism relation with XDG_H, through the definition of two functions f_C (over nodes) and f_D (over edges) (Pazienza et al, 2005).

This is possible if the selection process of the subsets of the graphs nodes guarantees the possibility of defining the function f_C. This procedure should try to map each constituent of XDG_H to its most similar constituent in XDG_T. If this is done, the bijective function f_C is derived by construction. The mapping process is based on the notion of *anchors*. The set of anchors *A* for an entailment pair contains an anchor for each of the hypothesis constituents having correspondences in the text *T*. For example in the entailment pair shown below, propositions:

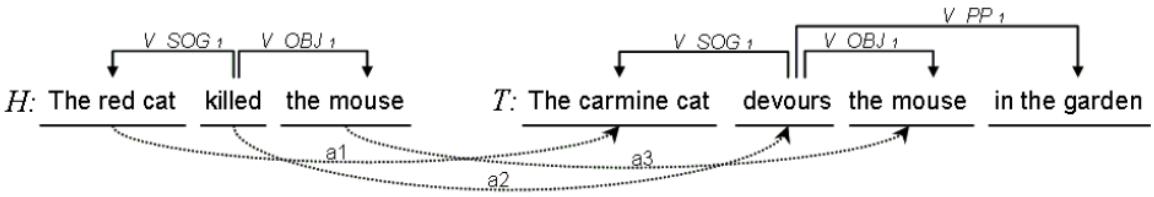


Figure 1: Example of entailment from (Pazienza et al., 2005)

f_C produces the mapping pairs *[The red cat - The carmine cat], [killed - devours], [the mouse - the mouse]*.

Syntactic similarity, defined by f_D, will capture how much similar the syntactic structure accompanied is to the two constituents (i.e., the edges of the graphs), by considering both their syntactic properties (i.e., the common dependencies) and the semantic properties of the constituents to which they are linked (i.e., the similarity of the anchor of the linked constituents).

Both semantic and syntactic similarity (derived respectively from f_C and f_D) must be taken into consideration to evaluate the overall graph similarity measure, as the former captures the notion of node subsumption, and the latter the notion of edge subsumption.

The method is more sophisticated in comparison with the BLUE approach, and it considers both syntactic and semantic levels. Initial, in RTE-1, this method was completed with different alignment methods and matching algorithms and used by many groups in the following challenges, those of 2006 (Katrenko and Adriaans, 2006 and Zanzotto et al. 2006), 2007 (Burchardt et al., 2007 and Ferrés and Rodríguez, 2007) and 2008 (Padó et al., 2008).

2.1.3. Tree Edit Distance Algorithms

The core of this approach (Kouylekov and Magnini, 2005) is a tree edit distance algorithm applied on the dependency trees of both the text and the hypothesis. If the distance (i.e. the cost of the editing operations) among the two trees is below a certain threshold, empirically estimated on the training data, then the authors assign an entailment relation between the two texts.

The authors designed a system based on the intuition that the probability of an entailment relation between *T* and *H* is related to the ability to show that the whole content of *H* can be mapped into the content of *T*. The more straightforward the mapping establishment can be, the more probable the entailment relation is. Since a mapping can be described as a sequence of editing operations needed to transform *T* into *H*, where each edit operation has a cost associated with it, they assign an entailment relation if the overall cost of the transformation is below a

certain threshold, empirically estimated on the training data.

According to their approach, *T entails H* if we have a sequence of transformations applied to *T* such that we can obtain *H* with an overall cost below a certain threshold. The underlying assumption is that pairs between which an entailment relation holds have a low cost of transformation. The transformations types (i.e. deletion, insertion and substitution) are determined by a set of predefined entailment rules, which also determine a cost for each editing operation. The authors have implemented the tree edit distance algorithm described in (Zhang and Shasha, 1990) and applied it to the dependency trees derived from *T* and *H*. Edit operations are defined at the single nodes level of the dependency tree (i.e. transformations on subtrees are not allowed in the current implementation). Since the (Zhang and Shasha, 1990) algorithm does not consider labels on edges, while dependency trees provide them, each dependency relation *R* from a node *A* to a node *B* has been re-written as a complex label *B-R* concatenating the name of the destination node and the name of the relation.

All nodes except for the root of the tree are relabelled in such way. The algorithm is directional: the aim is to find the best sequence (i.e. less costly) of edit operations that transforms *T* (the source) into *H* (the target).

According to the constraints described above, the following transformations are allowed:

- **Insertion**: insert a node from the dependency tree of *H* into the dependency tree of *T*. When a node is inserted, it is attached to the dependency relation of the source label.
- **Deletion**: delete a node *N* from the dependency tree of *T*. When *N* is deleted, all its children are attached to the parent of *N*. It is not required to explicitly delete the children of *N* as they are going to be either deleted or substituted on a following step.
- **Substitution**: change the label of a node N_1 in the source tree into a label of a node N_2 of the target tree. Substitution is allowed only if the two nodes share the same part-of-speech. In case of substitution, the relation attached to the substituted node is changed with the relation of the new node.

The initial approach used by (Kouylekov and Magnini, 2005) determined, based on the distance between trees, the final answer for the current pair. It should be noticed that this system does not use external resources like WordNet, paraphrases collection, or resources with named entities or acronyms. The next systems were more complex and combined the initial approach with machine learning algorithms (Kozareva, Montoyo, 2006) or used the probabilistic transformations of the trees (Harmeling, 2007). Other systems used the representation of texts

with dependency trees using MINIPAR[15] for that (Katrenko, Adriaans, 2006), (Iftene and Balahur-Dobrescu, 2007a), (Bar-Haim et al., 2008).

2.1.4. Logical Inference

In 2005, Bos and Markert used several shallow surface features to model the text, hypothesis and their relation to each other. They expected some dependency between the surface string similarity of text and hypothesis and the existence of entailment. This string similarity measure uses only a form of extended word overlap between text and hypothesis, taking into account identity of words, as well as synonymy and morphological derivations revealed by WordNet (Fellbaum, 1998).

To introduce an element of robustness into their approach, they used model builders to measure the "distance" from an entailment. The intuition behind this approach is as follows: If *H is entailed by T*, the model for *T+H* is not more informative compared to the one for *T*, and hence does not introduce new entities. Put differently, the domain size for *T+H* would equal the domain size of *T*. In contrast, if *T does not entail H*, *H* normally introduces some new information (except when it contains negated information), and this will be reflected in the domain size of *T+H*, which becomes larger than the domain size of *T*. It turns out that this difference between domain sizes is a useful way of measuring the likelihood of entailment. Large differences are mostly not entailments, small differences usually are.

They use a robust wide-coverage CCG-parser (Bos et al., 2004) to generate fine-grained semantic representations for each *T/H*-pair. The semantic representation language is a first-order fragment of the *DRS*[16] *language* used in *Discourse Representation Theory* (Kamp and Reyle, 1993), conveying argument structure with a neo-Davidsonian analysis and including the recursive *DRS* structure to cover negation, disjunction, and implication.

Given a *T/H* pair, a theorem prover can be used to find answers to the following conjectures:

1. *T* implies *H* (shows entailment)
2. *T+H* are inconsistent (shows no entailment)

[15] MINIPAR: http://ai.stanford.edu/~rion/parsing/minipar_viz.html
[16] Discourse Representation Theory

Let's assume that the function *DRS* denotes the *DRS* corresponding to *T* or *H*, and FOL[17] the function that translates a *DRS* into first-order logic. Then, if the theorem prover manages to find a proof for

$$FOL(DRS(T)) \rightarrow FOL(DRS(H)) \qquad \text{(A)},$$

(which is an equivalent form for 1.) we will know that we are dealing with a true entailment. In addition, to use a theorem prover to detect inconsistencies in a *T/H* pair, we give an equivalent form for 2.:

$$\neg FOL(DRS(T); DRS(H)) \qquad \text{(B)}$$

If the theorem prover returns a proof for (B), we know that *T* and *H* are inconsistent and *T* definitely does not entail *H* (assuming that *T* and *H* are themselves consistent).

Their work uses background knowledge (*BK*) for a better accuracy. Thus, instead of just giving $FOL(DRS(T); DRS(H))$ to the theorem prover, they supply it with BK influence: $(BK \wedge FOL(DRS(T); DRS(H)))$. They generate background knowledge using three kinds of sources: **generic knowledge, lexical knowledge,** and **geographical knowledge**.

In the first edition of 2005, five groups used logical provers and offered deep semantic analysis. One system (Raina et al., 2005) transformed the text and hypothesis into logical formula (like in Harabagiu et al., 2000) and it calculated the "cost" of proving hypothesis from text. In 2006 only two systems used logical inferences and one of the systems achieved the second result of the edition (Tatu et al., 2006). In 2007 the number of systems using logical inferences grew up to seven and the first two results used the logical inferences (Hickl 2007 and Tatu 2007). In RTE-4 nine groups used logical inferences in order to identify the entailment relation, and two of them were oriented to that (Clark and Harrison, 2008) and (Bergmair, 2008).

2.1.5. Probabilistic Considerations

2.1.5.1. Modelling Textual Entailment in Probabilistic Terms

The uncertain nature of textual entailment calls for its explicit modelling in probabilistic terms. In (Dagan and Glickman, 2004) a general generative probabilistic setting for textual entailment was proposed. This probabilistic framework may be considered analogous to (though different than)

[17] First Order Logic

the probabilistic setting defined for other phenomena, like language modelling and statistical machine translation.

The definition of entailment in formal semantics (Chierchia. and McConnell-Ginet, 2001) specifies that a text *T entails another text H* if *H* is true in every circumstance (possible world) in which *T* is true. Let's take the example from (Glickman et al., 2006): for a hypothesis H_1 = "*Marry Speaks French*" and a candidate text T_1 that includes the sentence "*Marry is a Fluent French Speaker*", it is clear that T_1 strictly entails H_1, and humans are likely to have high agreement regarding this decision. In many other cases, though, entailment inference is uncertain and has a probabilistic nature. For example, a text T_2 that includes the sentence "*Marry was born in France.*" does not strictly entail the above H_1 (i.e. there are circumstances in which someone was born in France but yet doesn't speak French). Yet, it is clear that T_2 does add substantial information about the correctness of H_1. In other words, the probability that H_1 is indeed true given the text T_2 ought to be significantly higher than the prior probability of H_1 being true. Thus, in this example, the text does substantially increase the likelihood of the correctness of the hypothesis, which naturally extends the classical notion of certain entailment. Given the text, we expect the probability that the hypothesis is indeed true to be significantly higher than its probability of being true without reading the text.

Let's see the notations used in (Glickman et al., 2006): *T* denotes a space of possible texts, and *t* in *T* a specific text. Meanings are captured in the model by hypotheses and their truth values. Let *H* denote the set of all possible *hypotheses*. A hypothesis *h* in *H* is a propositional statement which can be assigned a truth value. A *semantic state of affairs* is captured by a mapping from *H* to {*0 = false*, *1 = true*}, denoted by $w: H \rightarrow \{0, 1\}$ (entitled here *possible world*, following common terminology). A possible world *w* represents a concrete set of truth value assignments for all possible propositions. Accordingly, *W* denotes the set of all possible worlds.

The probability distribution of the source, over all possible texts and truth assignments $T \times W$, is assumed to reflect only inferences that are based on the generated texts. In particular, the probability for generating a true hypothesis *h* that is not related at all to the corresponding text is determined by some prior probability *P*(*h*). For example, *h* = "*Paris is the capital of France*" might have a prior smaller than *1* and might well be false when the generated text is not related at all to Paris or France.

In (Dagan and Glickman, 2004) two types of events were defined over the probability space for $T \times W$:

1. For a hypothesis h, where Tr_h is denoted the random variable whose value is the truth value assigned to h in the world of the generated text. Correspondingly, $Tr_h = 1$ is the event of h being assigned a truth value of 1 (True).
2. For a text t, we will use t to also denote the event that the generated text is t (as usual, it is clear from the context whether t denotes the text or the corresponding event).

Thus t probabilistically entails h (denoted as $t \Rightarrow h$) if t increases the likelihood of h being true, i.e. $P(Tr_h = 1 \mid t) > P(Tr_h = 1)$, or equivalently if the point wise mutual information, $I(Tr_h = 1, t)$, is greater than 1. Knowing that $t \Rightarrow h$, $P(Tr_h = 1 \mid t)$ serves as a probabilistic confidence value for h being true given t.

2152. The Web Approach

In 2005, Glickman and Dagan came with a new approach that used the web in order to calculate the probabilities. They performed unsupervised empirical estimation of the lexical entailment probabilities, $P(Tr_u = 1 \mid T_v)$, based on word co-occurrence frequencies as revealed by the web. Following the above probabilistic model, they assumed that the web is a sample generated by a language source. Each document represents a generated text and a (hidden) possible world. Given that the possible world of the text is not observed, they do not know the truth assignments of the hypotheses for the observed texts. Therefore, they further make the simplest assumption that all hypotheses stated verbatim in a document are true and all others are false and hence $P(Tr_u = 1 \mid T_v) = P(T_u \mid T_v)$.

The text and hypotheses of all pairs in the development set and the test set were tokenized by the following simple heuristic – split at white space and remove any preceding or trailing characters belonging to the following group: ([{)]}"'`.,;:-!?. A stop list was applied to remove frequent tokens. Counts were obtained using the *AltaVista* search engine, which supplies an estimate for the number of results (web-pages) for a given one or two token query.

In 2005 (Glickman and Dagan, 2005) participated with a system based on Web probabilistic approach. The system presented in (Bayer et al., 2005) used the EPILOG event-oriented probabilistic inference engine presented in (Schubert & Hwang, 2000).

The GROUNDHOG System used in RTE-2 by (Hickl et al., 2006) combined linguistically-naive probabilistic approaches with richer forms of lexico-semantic information and the system built by (Burchardt and Frank, 2006) used two probabilistic systems for frame and

role annotation, *Fred and Rosy* (Erk and Pado, 2006).

In RTE-3 the system (Hamerling, 2007) investigated the extension of transformation-based approaches toward probabilistic settings. Also, we can notice an improvement version of the system used in RTE-2 (Buchardt et al., 2007).

In RTE-4 the Monte Carlo technique (Bergmair, 2008) estimates, in a probabilistic sense, the validity for a formula associated to entailment relation. In (Varma et al., 2008) authors used the Probabilistic Hyperspace Analogue to Language (pHAL) (Jagadeesh et al., 2005) as the language modelling mechanism.

2.1.6. Atomic Propositions

In this section, an approach is presented based on the hypothesis that an entailment relation in the sentence pairs can be discovered by comparing the atomic propositions contained in the text and hypothesis sentences (Akhmatova, 2005). The comparison of atomic propositions (Akhmatova, 2005) is performed via an automated deduction system OTTER[18]. The propositions are extracted from the output of the Link Parser (Sleater and Temperley, 1993), and semantic knowledge is taken from the WordNet database. On its current stage, the system is capable to recognize basic entailments: semantically and syntactically and is potentially capable to use more external and internal knowledge to deal with more complex entailments.

An *atomic proposition* is a minimal declarative statement (or a small idea) that is either *true* or *false* and whose truth or falsity does not depend on the truth or falsity of any other proposition. (From example offered by (Akhmatova, 2005): *Coffee boosts energy and provides health benefits.* – two atomic propositions can be extracted: *Coffee boosts energy.* and *Coffee provides health benefits*.) To break a sentence into its atomic propositions the author uses the algorithm presented in (Jurafsky and Martin, 2000), and the author believes that a deep semantic and syntactical analysis is vital to solve the problem. In spite of the poor result aside, the system seems to have a high potential.

Another approach (Hickl and Bensley, 2007) extracts a set of publicly-held beliefs (known as *discourse commitments*) from text and hypothesis. The heuristics used to extract *discourse commitments* include: sentence segmentation, syntactic decomposition, supplemental expressions, relation extraction, and coreference resolution. The authors showed that once a set of discourse commitments had been extracted from a text-hypothesis pair, the task of recognising

[18] OTTER: www-unix.mcs.anl.gov/AR/otter

textual entailment could be reduced to the identification of one (or more) commitments from the text which are most likely to support the inference of each commitment extracted from the hypothesis. The results obtained by this system, which correctly classifies more than 80 % of the test data, show that this is the strongest of the methods employed.

2.1.7. Machine Learning

Starting with the first edition, the number of systems that used machine learning algorithms to determine the result of the entailment relation was considerable. The aim was to use results offered by these algorithms for answer classification instead of using thresholds established by human experts on training data. The features used by these systems include lexical, semantic, grammatical attributes of verbs, nouns and adjectives, named entities, and were calculated using the WordNet taxonomy (Miller, 1995), the VerbOcean semantic network (Chlonsky and Pantel, 2004), a Latent Semantic Indexing technique (Deerwester et al., 1990), or the ROUGE metrics (Lin, Hovy, 2003). Other features like negation were identified by inspecting the semantic representation of text with DRS[19] (Kamp and Reyle, 1993) for the presence of negation operators. These parameters were evaluated by machine learning algorithms such as SVM (Joachims, 2002) or such as C5.0 (Quinlan, 2000), or used binary classifications like Bayesian Logistic Regression (BBR)[20] and TiMBL (Daelemans et al., 1998).

Starting with RTE-2, the interest for using machine learning grew constantly. Thus, the number of systems that used machine learning for classification was increased from seven in 2005 to fifteen in 2006 and sixteen in 2007 and 2008. The approaches are various and their results depend on the capability of authors to identify relevant features. In (Inkpen et al., 2006), matching features are represented by lexical matches (including synonyms and related words), part-of-speech matching and matching of grammatical dependency relations. Mismatch features include negation and numeric mismatches. The MLEnt system (Kozareva, 2006) models lexical and semantic information in the form of attributes and, based on them, proposed 17 features. In (Ferrés and Rodríguez, 2007), the authors computed a set of semantic based distances between sentences. The system of (Montejo-Ráez et al., 2007) used semantic distance between stems, subsequences of consecutive stems and trigrams matching. The features identified by (Li et al., 2007) include lexical semantic similarity, named entities, dependent content word pairs, average

[19] Discourse Representation Theory
[20] BBR: http://www.stat.rutgers.edu/~madigan/BBR/

distance, negation, and text length.

As we can see, the interest to incorporate machine learning algorithms in textual entailment was very high and thus, the results enhanced year by year.

2.2. *Lexical Resources*

2.2.1. Using WordNet in Textual Entailment

When a word from the hypothesis does not match the same word in the text, many groups try to perform the matching using synonyms from WordNet (Miller, 1995) for all possible senses of the words. This type of approach is based on the assumption that the lexical unit *T* entails the lexical unit *H* if they can be *synonyms,* according to *WordNet,* or if there is a relation of *similarity* between them. Examples from RTE-1 corpus[21] include pairs of synonyms like *discover* and *reveal*, *obtain* and *receive*, *lift* and *rise*, *allow* and *grant*, etc.

Another, useful relation for entailment indicated by WordNet is the *hyponymy* relation (the relationship between a *specific word* and a *general word*, when the former is included within the latter). The use of these relations taken from WordNet improved the quality of identification of entailment cases.

One of the first systems that used WordNet is the system built in (Herrera et al, 2005) that used a very simple matching algorithm based on synonymy and hyponymy, focused on searching all the branches starting at any leaf from hypothesis tree and showing a match with any branch from text's tree.

From edition to edition more and more groups used either WordNet or eXtended WordNet (Harabagiu et al., 1999). Thus, in 2005 seven systems used WordNet in order to find a match between words from hypothesis and words from text. In RTE-2 ten groups reported using WordNet and one group (Tatu et al., 2006) used the eXtended WordNet, and in RTE-3, 21 groups used the WordNet resource and 2 groups used the eXtended WordNet (Tatu and Moldovan, 2007) and (Iftene and Balahur-Dobrescu, 2007a). In edition from 2008, 24 groups used WordNet resource and 2 groups used eXtended WordNet (Iftene, 2008) and (Clark and Harrison, 2008).

2.2.2. Using World Knowledge in Textual Entailment

From the beginning, the textual entailment definition assumes common human understanding of

[21] RTE-1 data sets: http://www.pascal-network.org/Challenges/RTE/Datasets/

the language as well as common world knowledge. The first systems using world knowledge (Bayer et al., 2005), (Bos and Markert, 2005) and (Fowler at al., 2005) exploited the semantic knowledge, the hypernymy relation from WordNet or geographical knowledge from external resources.

In 2006, only four systems used background knowledge, but in comparison to the previous edition, the background knowledge included inference rules and paraphrase templates and acquisition (automatic and manual) of additional entailment corpora. The participants' reports point out one reason for the shortcoming of current systems: the lack of linguistic and background knowledge. It seems that the best performing systems were those which better dealt with these issues. (Hickl et al., 2006) used a very large entailment corpus, automatically collected from the web, following (Burger and Ferro, 2005). Their GROUNDHOG system is able to derive a variety of lexico-semantic information from the text, including information about named entities, coreference, and syntactic and semantic dependencies. The authors adapted techniques used successfully in paraphrase recognition, for textual entailment recognition. In order to train the GROUNDHOG system, they extracted positive and negative examples of textual entailment from large newswire corpora (around 220,000 additional entailment pairs). Interesting in this approach is the way that the authors obtained 101,000 positive examples using (Burger and Ferro, 2005) and around 120,000 negative examples. These corpora contributed with approximate 10 % to the overall accuracy they achieved.

Since RTE-3, the number of systems using background knowledge grows up to eight. We can notice that the systems ranked the first three used world knowledge in different forms and also used different methods in order to obtain this information. (Tatu and Moldovan, 2007) used a large number of world knowledge axioms from eXtended WordNet, event and temporal information provided by the TARSQI[22] toolkit (Verhagen et al., 2005), logic form representations of events, negation, coreference and context, and new improvements of lexical chain axiom generation. (Clark et al., 2007) presented an analysis of some of the knowledge requirements of RTE3, and commented on some available sources of that knowledge. Three of them were presented in detail: WordNet (Fellbaum, 1998), DIRT paraphrase database (Lin and Pantel, 2001), FrameNet (Baker et al, 1998). They also noticed that the analysis serves to highlight the depth and variety of knowledge demanded by RTE3, and roughly contributes to organizing these

[22] Temporal Awareness and Reasoning Systems for Question Interpretation

requirements.

In RTE-4 the number of systems that used background knowledge grows to ten. In (Krestel et al., 2008) the authors designed a Fuzzy Believer to build knowledge bases from large amounts of complete newspaper articles. Two systems (Iftene, 2008) and (Clark and Harrison, 2008) used rules in order to encapsulate the world knowledge. In (Bar-Haim et al., 2008) the authors defined a proof system, based on application of entailment rules, which provides a principled and uniform mechanism for incorporating a wide variety of inference knowledge.

2.3. Conclusions

From edition to edition, the diversity of methods grows up and revealed new approaches for identification of entailments between texts. Starting from simple lexical methods that check the words overlap, the complexity of methods was increased when for texts representing were used graphs, trees or logical formulas. In all these cases, the systems calculate using different methods the distance between text and hypothesis representations, and on basis of thresholds or machine learning algorithms classify the systems answers in “Entailment” and “No entailment” classes.

The results show, how systems that rely on deep analysis such as syntactic matching and logical inference can considerably outperform lexical systems. The success of best performing systems suggests that perhaps the most important factors for deep entailment systems are the amount of linguistic and background knowledge, and the size of training corpora, rather than the exact method for modelling *T* and *H* and the exact inference mechanism.

3. The System

The main idea of the system built for the competition in 2007 (Iftene and Balahur-Dobrescu, 2007a) and improved for the competition in 2008 (Iftene, 2008c) is to map every node from the hypothesis to a node from the text using extensive semantic knowledge from sources like DIRT, WordNet, Wikipedia, VerbOcean and acronyms database. After the mapping process, we associate a local fitness value to every word from the hypothesis, which is used to calculate a global fitness value for current fragments of text. The global fitness value is decreased in cases in which a word from the hypothesis cannot be mapped to one word from the text or when we have different forms of negations for mapped verbs. In the end, using thresholds identified in the training step for global fitness values, we decide, for every pair from test data, whenever the entailment holds.

The system with which we participated in RTE-4 represents an enhanced version of the system used in RTE-3 (Iftene and Balahur-Dobrescu, 2007a). Additionally, we added new rules and used new semantic resources with the aim to better identify the contradiction cases. Figure 2 shows the knowledge basis and processing chains of the system (with grey are the new added components):

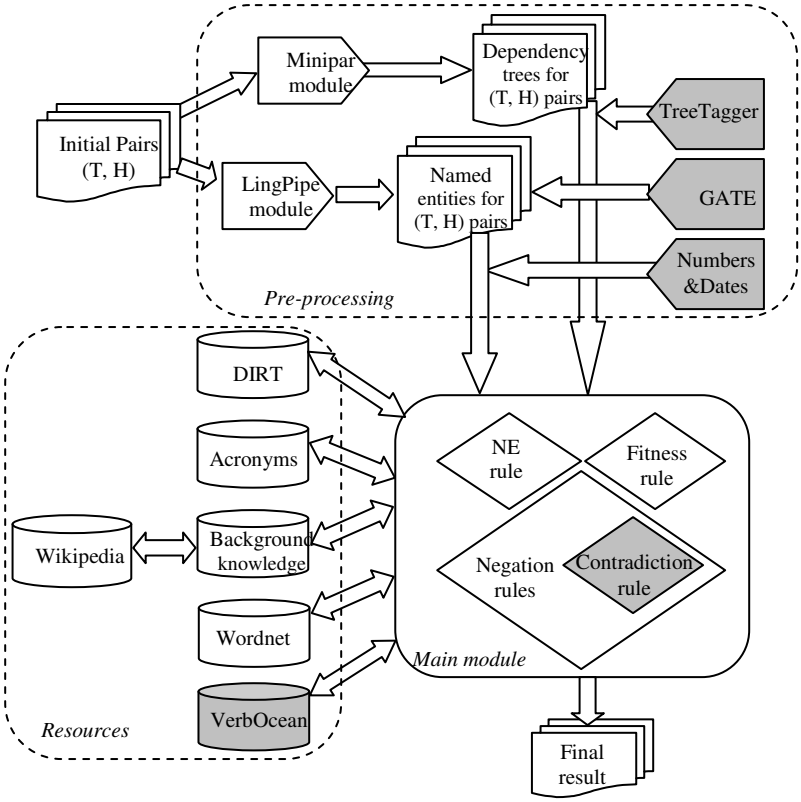


Figure 2: System architecture

Figure 2 displays the pre-processing parallel chains which make use of MINIPAR, LingPipe, GATE, and TreeTagger modules, providing the input for the core module. The latter uses five databases: DIRT, Acronyms, Background knowledge, WordNet and VerbOcean, computing scores for each pair. Employing the structure of the training data, a threshold between the "Yes" and "No" entailment pairs was established, and another threshold between the "Unknown" and "Contradiction" cases, which are then used to classify the pairs within the test set.

In the first pilot task from RTE-3 (this task corresponds to the 3-way task from RTE-4), while studying the training data, we noticed that out of all answers classified with "No", the majority, i.e. approximately 247 out of 400, were classified with "Unknown". Thus, the approach in re-classifying the answers of the system consisted of transforming the "No" category into two sub-categories, according to the score given by the system. Furthermore, the system considered that the pairs with a score of 0, resulted from not finding a Named Entity from the hypothesis in the text, are cases of "No" entailment and that the other cases in which the system had classified the pair as having "No" entailment in the two-way scheme can now be considered as "Unknown".

A second approach in re-classifying the answers of the system consisted in introducing two thresholds – one separating the pairs classified with "No" from those classified with "Unknown" and those classified with "Unknown" and those classified with "Yes", respectively. The values of these two thresholds were obtained on the training data, in its three way answer scheme. The two runs of the system submitted correspond to those two methods presented.

In the second pilot task from RTE-3, for the answers given by the system in the RTE task, the technical motivation for the score given was printed in a file, according to the tools that were used and the problems that appeared.

In the *competition of 2008*, all our efforts were oriented to the improvement of the system for the 3-way task. A special attention was directed to the "Contradiction" cases, and the new tools used, the resources and rules added are related to this part.

Next sections of this chapter present the main components of the UAIC system. In the last sections, I describe the obtained results in the competition, the limit of the system and future work.

3.1. *Pre-processing*

3.1.1. Preparation

In order to improve the quality of tools output, before using them on the initial test file[23] some additional steps are performed. Thus, we replace in all test data the expression "hasn't" with "has not", "isn't" with "is not", "couldn't" with "could not", "aren't" with "are not", "didn't" with "did not", "doesn't" with "does not", "won't" with "will not", "don't" with "do not". The meaning of the text remains the same after transformation, but the MINIPAR output is better for this new text.

For the hypothesis "*Ingrid Betancourt hasn't seen her children for 6 years.*" belonging to the pair 489 of the RTE-4 test set, before the transformation, the MINIPAR output was that in Table 10. The MINIPAR transforms a text into dependency tree (more details about MINIPAR output are in section 3.1.3.1).

```
(
E0      (()     fin C   *       )
1       (Ingrid ~ U     2       lex-mod         (gov Ingrid Betancourt))
2       (Betancourt     Ingrid Betancourt N     5       s       (gov see))
3       (hasn   have have       5       have    (gov see))
4       ('t     ~ U     3       lex-mod         (gov have))
5       (seen   see V   E0      i       (gov fin))
E2      (()     Ingrid Betancourt N     5       subj    (gov see)       (antecedent 2))
6       (her    ~ N     7       gen     (gov child))
7       (children       child N5        obj     (gov see))
8       (for    ~ Prep  7       mod     (gov child))
9       (6      ~ N     10      amount-value    (gov years))
10      (years  ~ N     8       pcomp-n         (gov for))
11      (.      ~ U     *       punc)
)
```

Table 10: MINIPAR output before transformation

After transformation for the new form "*Ingrid Betancourt has not seen her children for 6 years.*" the MINIPAR output is:

```
(
E0      (()     fin C   *       )
1       (Ingrid ~ U     2       lex-mod         (gov Ingrid Betancourt))
2       (Betancourt     Ingrid Betancourt N     5       s       (gov see))
3       (has    have have       5       have    (gov see))
```

[23] RTE4 test set: http://www.nist.gov/tac/protected/2008/RTE4_TEST-SET.xml.gz

4	**(not**	**~ A**	**5**	**amod**	**(gov see))**			
5	**(seen**	**see V**	**E0**	**i**	**(gov fin))**			
E2	(()	Ingrid	Betancourt	N	5	subj	(gov see)	(antecedent 2))
6	(her	~ N	7	gen	(gov child))			
7	(children		child	N	5	obj	(gov see))	
8	(for	~ Prep	7	mod	(gov child))			
9	(6	~ N	10	amount-value	(gov years))			
10	(years	~ N	8	pcomp-n	(gov for))			
11	(.	~ U	*	punc)				
)								

Table 11: MINIPAR output after transformation

We can see that before transformation in Table 10 how the negation was linked to the verb "have" and after the transformation in Table 11 how it is linked to the verb "see".

Also, before sending the text to the LingPipe, we replace in the initial test texts some punctuation signs like quotation marks "", brackets (), [], {}, commas, etc. with the initial sign between spaces. Again, the meaning of the text is the same, but the LingPipe output is better processed further after this transformation. LingPipe identifies in our texts named entities (more details about it are presented in next section 3.1.2.1).

For the hypothesis 616 from the RTE-4 test set *Transwide officially launches "MarCo"*, before transformation, the LingPipe output was with "*"*" inside of named entity like below:

<ENAMEX TYPE="LOCATION">**MarCo"**.</ENAMEX>

After transformation, the new hypothesis with spaces before and after quotation marks *Transwide officially launches " MarCo "* the LingPipe output is corrected:

<ENAMEX TYPE="LOCATION">**MarCo**</ENAMEX>

The next step splits the obtained test file into 1000 pairs of files, each containing a text and a hypothesis.

3.1.2. Named Entities Identification

3.1.2.1. LingPipe

All files obtained after the preparation steps are then sent to the LingPipe[24] module in order to find the named entities. LingPipe is a suite of Java libraries for the linguistic analysis of human language. The major tools included in this suit are:

- **Sentence**. LingPipe extracts sentences heuristically, by identifying tokens that end

[24] LingPipe: http://alias-i.com/lingpipe/

sentences.

- **Parts of Speech**. Part-of-speech tagging assigns classical grammatical categories to tokens.
- **Named Entities**. Named entity recognition finds mentions of named entities in text.
- **Coreference** determines when two entity mentions in a text refer to the same entity in the world.

From this suite the system has used only the Named Entities recognition tool with the scope to eliminate the pairs that have a named entity only in the hypothesis, concluding thus for "Unknown" cases.

The *Named entity recognizers* in LingPipe are trained from a corpus of data. It extract mentions of people, locations or organizations in English news texts, and mentions of genes and other biological entities of interest in biomedical research literature.

For instance, simple named entity recognizer for English might find the *person* mention *John J. Smith* and the *location* mention *Washington* in the text "*John J. Smith lives in Washington*". The output file is in XML format and we can see in Table 12 the result for this case:

```
<output>
<s i="0">
<ENAMEX TYPE="PERSON">John J. Smith</ENAMEX>
 lives in
<ENAMEX TYPE="LOCATION">Washington</ENAMEX>
.
</s>
</output>
```

Table 12: Example of a LingPipe output

3.1.2.2. GATE

In the case of Named Entities of type PERSON, we additionally used gazetteer from GATE (Cunningham et al., 2001), which contains finer-grained classes of entities. In a manner that is similar to the approach presented in (Tatu and Moldovan, 2007) we distinguish between the cases in which the *family name* or the *first name* found in the hypothesis are missing from the text.

GATE (General Architecture for Text Engineering) is a framework which enables users to build and deploy language engineering components and resources in a robust way. The GATE architecture allows the development of easy applications for various language processing tasks,

and also to build and annotate corpora and carry out evaluations on the applications generated. The framework can be used to develop applications and resources in multiple languages.

The GATE components are one of three types:

- **Language Resources** represent entities such as lexicons, corpora or ontology;
- **Processing Resources** represent entities that are primarily algorithmic, such as parsers, generators or n-gram modellers;
- **Visual Resources** represent visualization and editing components that participate in GUIs.

We have used **English gazetteer**, a component of the *Processing Resources*. The English gazetteer consists of lists such as person names, regions, cities, organizations, days of the week, etc. It does not only consist of entities, but also of names of useful indicators, such as typical company designators (e.g. 'Ltd.'), titles, etc.

3.1.2.3. Numbers Identification

In addition to using LingPipe and GATE, we build a set of patterns with the aim to identify numbers and calendar dates. In the RTE-3 system we missed some NE types, but in RTE-4 we have added specific patterns for identifying *percentages*, *measure values*, *time periods*, *order numerals*, etc. Table 13 gives examples of identified numbers:

Numbers type	Examples from RTE4 test set
number	31; 129; 2.23; 200,000 one; two; hundred thousand; sixteen 1.5 million
percentages	40%, 1.1% 7 percent; 48.3 percent
measure values	1.7 kg; one-tone; 20-minute 483 millimetres; 35 kilometres; 15,800 square kilometres 508m; 1,667ft $20,000; £8m
years	1999; 2008 1970s; '80s; 2008's
dates	1 October 2006; July 25, 2000 May 2001, June 20; 30 B.C.

Numbers type	Examples from RTE4 test set
hours	8:30 am; 10 p.m.; 13:00 PST; 20:00 UTC
intervals	2000-01; March 12-14, 2008 10.30-11.15am; 9-12; 1 to 1.25
order numerals	first; second; fourth 1st; 101st; 3rd; 29th
other values	98-88; 0-0; 30-14; 747-438; 40-member

Table 13: Example of numerical expressions identified by patterns in the RTE-4 test set

3.1.3. Dependency Trees Building

3.1.3.1. MINIPAR

Along with the identification of named entities, we transform both the text and the hypothesis into dependency trees with MINIPAR[25] (Lin, 1998). MINIPAR is a broad-coverage parser for the English language. An evaluation with the SUSANNE[26] corpus shows that MINIPAR achieves about 88% precision and 80% recall with respect to dependency relationships.

MINIPAR represents the grammar as a network, where the nodes represent grammatical categories and the links represent types of syntactic (dependency) relationships. The grammar network consists of 35 nodes and 59 links. Additional nodes and links are dynamically created to represent subcategories of verbs. The lexicon in MINIPAR is derived from WordNet. With additional proper names, the lexicon contains about 130K entries (in base forms).

The output produced by MINIPAR is a graph in which the nodes are the words for the parsed sentence labelled with their grammatical categories and the edges are relations between the words labelled with grammatical relationships.

The output for the sentence "*Le Beau Serge was directed by Chabrol*." is as given in Table 14.

```
(
E0(()       fin           C    *)
1 (Le       ~             U    3  lex-mod (gov Le Beau Serge))
2 (Beau     ~             U    3  lex-mod (gov Le Beau Serge))
3 (Serge    Le Beau Serge N    5  s       (gov direct))
4 (was      be            be   5  be      (gov direct))
5 (directed direct        V    E0 i       (gov fin))
E2(()       Le Beau Serge N    5  obj     (gov direct) (antecedent 3))
```

[25] MINIPAR: http://www.cs.ualberta.ca/~lindek/minipar.htm
[26] The SUSANNE corpus is a subset of the Brown Corpus of American English

```
6 (by       ~             Prep 5  by-subj (gov direct))
7 (Chabrol  ~             N    6  pcomp-n (gov by))
8 (.        ~             U    *  punc)
)
```

Table 14: MINIPAR output – plain text

Here, the first column contains the *label* of the *word* from second column, and next columns are in order: the *root* of the word, the grammatical *category*, *parent label* from dependency tree, grammatical *relation*, *parent root* of the word and *antecedent label*. More details about MINIPAR output are in Appendixes 1 and 2. The visual graph of the sentence is displayed in Figure 3.

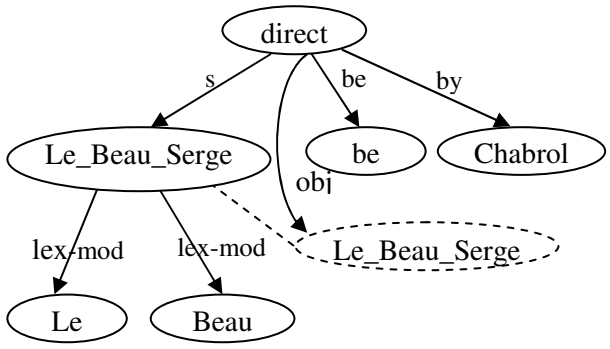


Figure 3: MINIPAR output – visual graph

Main MINIPAR relations are presented in table below (Martínez et al., 2002):

No	Relation	Direct	Indirect	Description
1	By-subj	X		Subj with passives
2	compl	X		Complement (PP, inf/fin clause) of noun
3	Mod	X		Modifier
4	Obj	X		Object
5	pcomp-c	X		Clause of pp
6	Pcomp-n	X		Nominal head of pp
7	Pred	X		Predicative (can be A or N)
8	Subj	X		Subject

Table 15: Main MINIPAR relations

For every node of the MINIPAR output (which represents a simple word belonging to a sentence), a stamp called **entity** was considered with three main features: *the node lemma*, *the*

father lemma and *the edge label* (like in Figure 4).

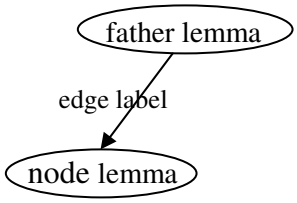


Figure 4: Entity components

Using this stamp, we can easily distinguish between nodes of the trees, even if these have the same name and the same father. In the example from Figure 3, for the "son" nodes we have two entities (*Le_Beau_Serge, direct, s*) and (*Le_Beau_Serge, direct, obj*).

3.1.3.2. TreeTagger

Because in some cases the part-of-speech (POS) and lemma identified by MINIPAR are wrong, we use the TreeTagger[27] tool that correctly identifies the words POS and replaces the wrong POS identified by MINIPAR. This step is very important, especially for verbs, because our algorithm starts from verb mapping and all the next steps depend on it. This step is applied only when the MINIPAR doesn't identify any verb in the hypothesis.

The TreeTagger is a tool for annotating text with part-of-speech and lemma information which has been developed within the TC project[28] at the Institute for Computational Linguistics of the University of Stuttgart. The TreeTagger has been successfully used on 12 languages (including English) and is easily adaptable to other languages if a lexicon and a manually tagged training corpus are available. For the sentence "*Le Beau Serge was directed by Chabrol*" the output is presented below:

Word	**POS**	**Lemma**
Le	NP	Le
Beau	NP	Beau
Serge	NP	Serge
was	VBD	be
directed	VVN	direct

[27] TreeTagger: http://www.cele.nottingham.ac.uk/~ccztk/treetagger.php. The TreeTagger precision is 96.36 %, and the MINIPAR precision is under 80 %.
[28] TC Project: http://www.ims.uni-stuttgart.de/projekte/tc

Word	POS	Lemma
by	IN	by
Chabrol	NP	Chabrol
.	SENT	.

Table 16: TreeTagger output

The tagset used by the TreeTagger is a refinement of Penn-Treebank[29] tagset (Santorini, 1990). In the tagset used by TreeTagger, the second letter of the verb part-of-speech tags is used to distinguish between forms of the verb "to be" (B), the verb "to have" (H), and all the other verbs (V). "VHD" is the POS tag for the past tense form of the verb "to have", i.e. for the word "had".

The meanings of the tags in Table 16 are presented below (all Penn-Treebank tagset is presented in Appendix 3):

Acronym	Description
NP	Proper noun, singular
VBD	Verb, "be" verb, past tense
VVN	Verb, other verbs, past participle
IN	Preposition or subordinating conjunction

Table 17: The meaning of tags in Table 16

As an example in which TreeTagger helps our system in the correct identification of POS for words, we can take the pair 355 with the hypothesis "*The University of Exeter and Cornwall Wildlife Trust investigates suspicious cetacean deaths in Cornwall*". For this sentence MINIPAR identifies the POS for the word *investigates* equal with "U" and lemma equal with "*investigates*"; these value are wrong. For the same sentence the TreeTagger correctly identifies the POS for *investigates* equal with "VVZ" and a correct lemma equal with "investigate".

3.2. *The Hypothesis Tree Transformation*

From now on, the main goal is to map every entity in the dependency tree associated with the hypothesis (called the *hypothesis tree*) to an entity in the dependency tree associated with the text

[29] The Penn TreeBank Project: http://www.cis.upenn.edu/~treebank/

(called the *text tree*) (Iftene and Balahur-Dobrescu, 2007a).

The mapping between entities can be negotiated in two ways: *directly* (when entities from hypothesis tree exist in the text tree) or *indirectly* (when entities from text tree or hypothesis tree cannot be mapped directly and need transformations using external resources). Using this type of mapping between an entity from hypothesis tree and an entity from text tree, we calculate a *local fitness value* which indicates the appropriateness between entities. Based on local fitness of node and on local fitness of its father, we build for node an *extended local fitness* and in the end, using all partial values, we calculate a normalized value that represents the *global fitness*.

When an entity belonging to the hypothesis tree can be mapped directly to more entities from the text tree, we select the mapping which increases the global fitness with the highest value. When it is not possible to map an entity of the hypothesis tree to another entity of the text tree, we use external resources for that: DIRT, VerbOcean and WordNet (for verbs), Acronyms database and Background Knowledge (for named entities), eXtended WordNet and WordNet (for nouns and adjectives). In the following, we will revise the main characteristics of these resources.

3.2.1. The Resource DIRT

DIRT[30] (Discovery of Inference Rules from Text) is both an algorithm and a resulting knowledge collection created by Lin and Pantel at the University of Alberta (Lin and Pantel, 2001). The algorithm automatically learns paraphrase expressions from text using the Distributional Hypothesis[31] (Harris, 1954) over paths in dependency trees. A path, extracted from a parse tree, is an expression that represents a binary relationship between two nouns. In short, if two paths tend to link the same sets of words, DIRT hypothesizes that the meanings of the corresponding patterns are similar.

The algorithm is based on an extended version of Harris' Distributional Hypothesis (Harris, 1985). Instead of using this hypothesis on words, Lin and Pantel apply it to paths in the dependency trees of a parsed corpus.

In the dependency trees generated by MINIPAR, each link between two words in a dependency tree represents a direct semantic relationship. A path allows representing indirect semantic relationships between two content words. A path is represented by concatenating dependency relationships and words along the path, excluding the words at the two ends.

[30] DIRT: http://aclweb.org/aclwiki/index.php?title=DIRT_Paraphrase_Collection

[31] The Distributional Hypothesis in Linguistics is that words that occur in the same contexts tend to have similar meanings

For the sentence in Figure 5, the path between *John* and *problem* is noted: **N:subj:V<find>V:obj:N>solution>N:to:N** (meaning "*X finds solution to Y*"). The root of the path is "*find*".

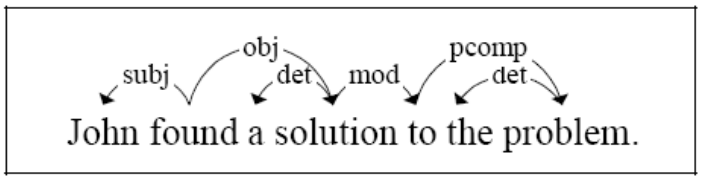


Figure 5: Example dependency tree (from Lin and Pantel, 2001)

A path begins and ends with two dependency relations. The open ends of the path are slots called: *SlotX* on the left-hand side and *SlotY* on the right-hand side. The words connected by the path are the fillers of the slots. For example, "*John*" fills the *SlotX* and "*problem*" fills the *SlotY* in the above example.

Lin and Pantel extract the fillers and frequency counts of all the slots of all the paths in a parsed corpus. The underlying assumption of the algorithm is that when the meanings of paths are similar, their corresponding sets of fillers share a large number of common words.

Richardson (Richardson, 1997) extracted semantic relationships (e.g., hypernym, location, material and purpose) from dictionary definitions using a parser and constructed a semantic network. He then described an algorithm that uses paths in the semantic network to compute the similarity between words. In a sense, the Lin and Pantel algorithm is a dual of Richardson's approach. While Richardson used paths as features to compute the similarity between words, they use words as features to compute the similarity of paths.

Lin (Lin, 2001) used the notation $|p, SlotX, w|$ to denote the frequency count of word w filling in the *SlotX* of a path p, and $|p, SlotX, *|$ to denote $\sum_w |p, SlotX, w|$, and $|*,*,*|$ to denote $\sum_{p,s,w} |p, s, w|$.

Following (Lin, 1998), the mutual information between a path slot and its filler can be computed by the formula:

$$mi(p, Slot, w) = \log\left(\frac{|p, Slot, w| \times |*, Slot, *|}{|p, Slot, *| \times |*, Slot, w|}\right)$$

The similarity between a pair of slots: $slot_1 = (p_1, s)$ and $slot_2 = (p_2, s)$, is defined as:

$$sim(slot_1, slot_2) = \frac{\sum_{w \in T(p_1,s) \cap T(p_2,s)} mi(p_1, s, w) + mi(p_2, s, w)}{\sum_{w \in T(p_1,s)} mi(p_1, s, w) + \sum_{w \in T(p_2,s)} mi(p_2, s, w)}$$

where p_1 and p_2 are paths, s is a slot, $T(p_i, s)$ is the set of words that fill in the s slot of path p_i.

The similarity between a pair of paths p_1 and p_2 is defined as the geometric average of the similarities of their *SlotX* and *SlotY* slots:

$$S(p_1, p_2) = \sqrt{sim(SlotX_1, SlotX_2) \times sim(SlotY_1, SlotY_2)}$$

The DIRT knowledge collection is the output of the DIRT algorithm for an over 1GB set of newspaper text (San Jose Mercury, Wall Street Journal and AP Newswire from the TREC-9 collection). It extracted 7 million paths from the parse trees (231,000 unique) from which paraphrases were generated. For example, here are the Top-20 paraphrases "*X solves Y*" generated by DIRT:

Y is solved by X	*X seeks a solution to Y*	*X overcomes Y*
X resolves Y	*X does something about Y*	*X eases Y*
X finds a solution to Y	*X solution to Y*	*X tackles Y*
X tries to solve Y	*Y is resolved in X*	*X alleviates Y*
X deals with Y	*Y is solved through X*	*X corrects Y*
Y is resolved by X	*X rectifies Y*	*X is a solution to Y*
X addresses Y	*X copes with Y*	

Table 18: Top-20 paraphrases for "X solves Y" generated by DIRT

For the verbs in the MINIPAR output, templates with DIRT like format are extracted (Iftene and Balahur-Dobrescu, 2007a). For the sample output in Figure 3, where a single verb "direct" exists, the following list of "full" templates is obtained: **N:s:V<direct>V:by:N** and **N:obj:V<direct>V:by:N**. To this list, another list of "partial" templates is added: **N:s:V<direct>V:**, **:V<direct>V:by:N**, and **N:obj:V<direct>V:**. The list of "partial" templates is obtained by eliminating, from each "full" template, the other part of speech except the verb, situated either before or after the verb.

In the same way, a list of templates for the verbs in the text tree was built. With these two lists of templates, the system performs a search in the DIRT database, and extracts the "best" fitting between lists, using template type (full or partial) and the DIRT score. The search will be

successful when there exists in DIRT a similarity relation between the verb from hypothesis and at least one verb from text.

It is possible for the result of the search process to be empty. This situation corresponds to the cases in which there aren't any similarity relations in DIRT between the verb from hypothesis and any verbs from text. In this case, it uses the corresponding noun form of the verbs and attempts to make the association between verbs and nouns.

In the successful cases, according to the search results, we have the following situations:

a) Left – left relations similarity

This case is described by the situation in which we have the following template for the hypothesis:

***relation*$_1$** *HypothesisVerb* *relation*$_2$

And for the text, the template:

***relation*$_1$** *TextVerb* *relation*$_3$

This is a classic case that often appears, in which a verb is replaced by one of its synonyms or equivalent expressions.

In this case, the hypothesis tree is transformed by the following two steps:

1. Replace the *relation*$_2$ with *relation*$_3$,
2. Replace the verb from the hypothesis with the corresponding verb from the text. (As it can be observed in Figure 6).

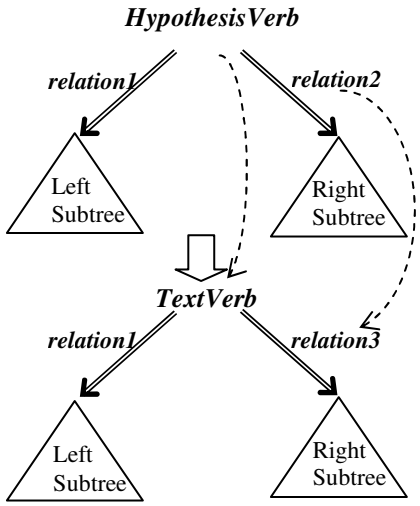


Figure 6: Left-left relation similarity

For example, in the RTE-3 test set we have pair 37 with the verb "start":

T: She was transferred again to Navy when the American Civil War began, 1861.

H: The American Civil War started in 1861.

In this case, for the text we have the template *N:s:V<begin>V:obj:N*, and for the hypothesis, the template *N:s:V<start>V:subj:N*. Using DIRT, hypothesis *H* is transformed into:

H': The American Civil War began in 1861.

Under this new form, *H* is easier to compare to *T*.

b) **Right – right relations similarity**: The same idea as the previous case

c) Left – right relations similarity

This case can be described by the situation in which we have the following template for the hypothesis:

***relation*$_1$** *HypothesisVerb relation*$_2$

And for text we have the template:

relation$_3$ *TextVerb* ***relation*$_1$**

In this case, we transform the hypothesis tree using the following three steps:

1. Replace the *relation*$_2$ with *relation*$_3$,
2. Replace the verb from the hypothesis with the corresponding verb from the text.
3. Rotate the subtrees accordingly: left subtree will be right subtree and vice-versa right subtree will become left-subtree (as it can be observed in Figure 7).

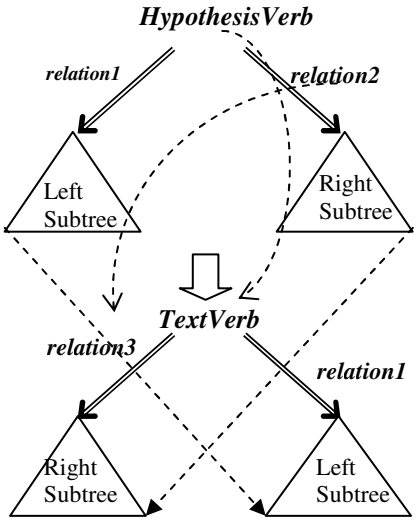


Figure 7: Left-right relation similarity

For example, in the RTE-3 test set we have pair 161 with the verb "attack":

T: The demonstrators, convoked by the solidarity with Latin America committee, verbally attacked Salvadoran President Alfredo Cristiani.

H: President Alfredo Cristiani was attacked by demonstrators.

In this case, for the text we have the template *N:subj:V<attack>V:obj:N*, and for the hypothesis, the template *N:obj:V<attack>V:by:N*. Using DIRT, hypothesis *H* is transformed into:

H': Demonstrators attacked President Alfredo Cristiani.

Under this new form, *H* is easier to compare to *T*.

d) **Right – left relations similarity**: The same idea as in the previous case

For every node transformed with DIRT, its local fitness is considered as being the similarity value indicated by DIRT.

322. VerbOcean

VerbOcean[32] is a broad-coverage semantic network of verbs created by Timothy Chklovski and Patrick Pantel at the University of Southern California's Information Sciences Institute (Chklovski and Pantel, 2004). VerbOcean is a graph of semantic relations between verbs, with 3,477 verbs (nodes) and 22,306 relations (edges). The authors detect *similarity*, *strength*, *antonymy*, *enablement*, and temporal *happens-before* relations between pairs of strongly

[32] VerbOcean: http://demo.patrickpantel.com/Content/verbocean/

associated verbs using lexico-syntactic patterns, querying the Web with Google.

In the table below we have the released relations between verbs, with some examples of data:

Semantic Relation	Example	Transitive	Symmetric	Number in VerbOcean
Similarity	produce::create	Y	Y	11,515
Strength	wound :: kill	Y	N	4,220
Antonyms	open :: close	N	Y	1,973
Enablement	fight :: win	N	N	393
happens-before	buy :: own; marry :: divorce	Y	N	4,205

Table 19: Possible relations between verbs from VerbOcean

Out of the semantic relations contained in VerbOcean we use the antonymy relation. Also known as *semantic opposition*, *antonymy* between verbs has several distinct subtypes. The antonymy relation can be a result of:

- Switching thematic roles associated with the verb (*buy :: sell, lend :: borrow*). Antonymy between static verbs (*live :: die, differ :: equal*) and antonymy between sibling verbs which share a parent (*walk :: run*) or an entailed verb (*fail :: succeed* both entail *try* (Fellbaum, 1998).
- From happens-before relation in the case of restitutive opposition (Cruse, 1986). Examples for this subtype can be *damage :: repair*, *wrap :: unwrap*.

Examples of antonymy relations from VerbOcean include: *assemble :: dismantle; ban :: allow; regard :: condemn, roast :: fry*.

The VerbOcean file format is a text file format where each line contains a quartet with two verbs linked by a relation, and a strength value, between 0 and 26.46. For the antonymy relation we use "*happens-before*" and "*opposite-of*" relations from this file. Below is a small fragment of VerbOcean file:

```
pursue [happens-before] abandon :: 11.262514
abandon [opposite-of] pursue :: 10.178111
abandon [happens-before] renew :: 10.151251
abandon [opposite-of] renew :: 9.658213
```

Table 20: Examples from VerbOcean file

In cases in which we map the verb from hypothesis to one verb from text and we detect an antonymy relation between them, we apply the "Contradiction rule". As we will see, in these cases the calculation of local fitness is useless.

3.2.3. Extended WordNet

eXtended WordNet[33] is an ongoing project at the Human Language Technology Research Institute, University of Texas at Dallas. The goal of this project is to develop a tool that takes the current or future versions of WordNet[34] as input and automatically generates an eXtended WordNet that provides several important enhancements intended to remedy the present limitations of WordNet.

In the eXtended WordNet, the WordNet glosses are syntactically parsed and transformed into logic forms and the content words are semantically disambiguated. For non-verb nodes of the hypothesis tree, if in the text tree no nodes with the same lemma are found, the system searches for their similarity values in the eXtended WordNet. For every similar value, it checks to see which word appears in the text tree, and selects the mapping with the best value according to the values from eXtended Wordnet. Subsequently, the word from the hypothesis tree is changed with the word from eXtended Wordnet and also its fitness is changed with its indicated similarity value. For example, the relation between "relative" and "niece" is found with a score of 0.078652. In this case we consider the local fitness value equal with the similarity value indicated by eXtended WordNet.

3.2.4. WordNet

WordNet (Fellbaum, 1998) is a large lexical database of English, developed under the direction of George A. Miller at the Princeton University. Nouns, verbs, adjectives and adverbs are grouped into sets of cognitive synonyms (synsets), each expressing a distinct concept. Synsets are interlinked by means of conceptual semantic and lexical relations.

From WordNet, we used the *synonym* relation for nouns (between two words having the same or nearly the same meaning), and the *antonymy* relation for verbs (between two words having opposite meanings). For the synonymy relation, the local fitness value is considered to be

[33] Extended WordNet: http://xwn.hlt.utdallas.edu/
[34] WordNet: http://wordnet.princeton.edu/

the maximum value i.e. 1. In opposition, the antonymy relation inserts a penalty in the local fitness value.

3.2.5. Acronyms

Due to the need to ease and fasten communication, acronyms have become a normal part of everyday speech and newspaper world. Collections of acronyms can be found by a simple Google search, on many Internet sites. Due to the need for complex and complete sets of acronyms, organized according to the topic they belong to, the acronyms database[35] was chosen.

The acronyms' database helps the program to find relations between the acronym and its meaning: "*US - United States*", "*USA - United States of America*", and "*EU - European Union*". Also, the word is changed with the corresponding expression from this database, but because the meaning is the same, the local fitness value is set on maximum, i.e. 1.

3.2.6. Background Knowledge

The Background Knowledge helps our program to find relations between a named entity from hypothesis (without correspondence in the text) and another named entity from text. This background knowledge was not priory available, so its acquirement became a practical issue, whose solving brought significant improvements to the system performance (Iftene and Balahur-Dobrescu, 2008b). Our approach consists of applying grammar induced extraction patterns on a large corpus – Wikipedia – for the extraction of relations between a given Named Entity and other Named Entities.

3.2.6.1. A Grammar of Definitions

Under the framework of the FP6 European project LT4eL[36] (Language Technology for e-Learning), a Romanian grammar was created for the automatic identification of definitions in texts (Iftene, Trandabăţ, Pistol, 2007), (Iftene et al., 2008e). 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 are not present (Mureşan and Klavans 2002, Liu et al., 2003).

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 texts have been classified as follows:

[35] Acronym Database: http://www.acronym-guide.com
[36] LT4eL: http://www.lt4el.eu/

"*is_def*" (definitions containing the verb "is"), "*verb_def*" (definitions containing specific verbs, different then "is"), "*punct_def*" (definitions which use punctuation signs), "*layout_def*" (definitions that can be deduced by the layout), "*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), "*other_def*" (other definitions, which cannot be included in any of the previous categories). For every one of these types, we built similar rules for English.

Our program has the possibility to extract snippets from Wikipedia that contain a specified named entity (NE), or it can extract a list with NEs related to a given NE. *In the first case*, we use this grammar and identify definitions contexts, and *in the second case* we consider relations between types of NEs. For both cases we build relations between NEs and a part of these relations were used by our Textual Entailment system.

3262. Extracting NEs Related to a Specified NE from Wikipedia

In the first case, for a specified NE, we use a module to extract snippets from Wikipedia with information related to it. In the snippets extracted from Wikipedia we try to identify the definition contexts. For each such context:

a. we identify the "core" of the definition (which is either the verb "to be" or another definition introducing a 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 and the patterns built using existing grammar in order to identify the relations between entities. The goal in this endeavour is to identify a known relation between two NEs. If such a relation is found, we make the association and save it to an output file.

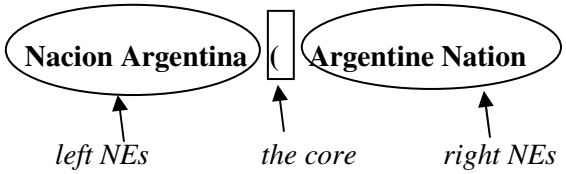


Figure 8: Example for Argentina

In Figure 8, for the sentence "*Argentina, Nacion Argentina (Argentine Nation for many*

legal purposes), is in the world.", we marked these values. Eventually, only the line "Argentina [is] Argentine" is added to the background knowledge.

Starting from a named entity we built an oriented graph on levels, thus: the first level of NEs for the candidate NE is made up of all the NEs extracted for that NE. Further on, all NEs related to NEs from the first level give the second level of NEs. This process is continued until no new NEs related to any of the previous levels for a candidate NE are obtained. A suggestive example is obtained for the NE "Paris", of type LOCATION, used as starting NE. A partial result is shown in Figure 9).

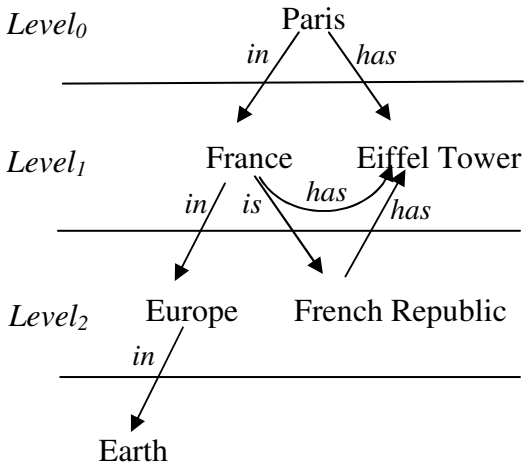


Figure 9: Example acquisition of knowledge issued by Paris

3263. Types of Relations between the Extracted NEs and the Initial NE

In the second case, the NEs extracted from Wikipedia related to a NE are clustered and classified. In order to do that we use GATE and identify the following types of NEs: country, city, ocean, sea, river, mountain, region (small and big), language, money, person name, organization, and job.

Classification depends on the initial NE type and the types of NEs related to it. As an example, for the start entity "Italy" whose type is country, we extract 186 entities, from which 61 are different. In the table below are the NEs with the highest number of appearances in the file obtained for the initial NE Italy.

Initial NE type	Type of related NEs	Relation
Country	Person	Leonardo da Vinci <live in> Italy
	Country	France <neighbour with> Italy
	Language	Italian Language <spoken in> Italy
	Money	Euro <is currency from> Italy
	City	Milan <located in> Italy
	Big Region	Italy <located in> Europe

Table 21: Example for Italy

We can see in the table above that between entities of type person and entities of type *Country,* the considered relation is "live in"; between entities of type countries we consider "neighbour with" and so on. These types of relations were considered only for cases in which the numbers of appearances for found named entity are high. In other cases the relation between named entities remains "unknown".

From all these relations, only few of them are used in the system, related to the entailment relation: "in", "is" from the first case and "located in", "live in" in the second case. The local fitness is set to the maximum value only for the "is" relation (i.e. 1) and receives penalties for "in" cases ("in", "live in", "located in"). We test more values for "in" relations on training data (between 0 and 1), and in the end we decide to use the middle of this interval: i.e. the value 0.5.

3.3. *The Main Module*

Let's see, for every type of possible answer, what are the rules that promote it. All next examples are from the RTE-4 test set[37].

3.3.1. Entailment Cases

3.3.1.1. Basic Positive Rules

In this case any type of mapping will increase the global score and, in the end, will increase the probability to have the final answer "Entailment". I explain in this section how is the local fitness calculated. For every node of the hypothesis tree which can be mapped directly to a node of the

[37] RTE-4 test set: http://www.nist.gov/tac/protected/2008/RTE4_TEST-SET.xml.gz

text tree, we will consider the local fitness value to be 1 (the maximum value).

When it is not possible to do a direct mapping between a hypothesis node and a text node, we will try to use the external resources and to transform the hypothesis node in an equivalent one.

Thus, for *verbs* we use the DIRT resource (Lin, 1998) and transform the hypothesis tree into an equivalent one, with the same nodes, except the verb. This is the case of pair 84:

*T: Barack Obama has **declared** himself "the Democratic nominee for president of the United States". He was speaking to a cheering crowd on the last day of the primary season, as projections showed he had earned enough delegates to clinch the nomination.*

*H: Obama **claims** Democratic victory*.

We found in DIRT a similarity relation between the verbs *declare* and *claim* with a score of 0.290608. After using this resource, the hypothesis has changed into

H': *Obama **declares** Democratic victory*.

and in this form it is easier to compare the text and hypothesis and in the end the value of the global fitness score is increased.

If the word is a *named entity*, we try to use a database of acronyms[38] or obtain information related to it from the background knowledge (Iftene and Balahur-Dobrescu, 2008b). As an example for acronyms we can indicate the pair 528 where in the text we have *EU* and in the hypothesis we have *European Union*. Examples in which the module adding new elements from the English Wikipedia to the background knowledge is used are: pairs 51 (relation between *Jewish* and *Jew*), 104 (between *Buenos Aires* and *Argentina*), 184 (between *Ireland* and *Irish*), 216 (between *UK* and *British*), 280 (between *16* and *sixteen*), 528 (between *Portugal* and *Portuguese*), etc.

For *nouns* and *adjectives* we use WordNet (Fellbaum, 1998) and a part of the relations from the eXtended WordNet[39] to look up synonyms and then we try to map them to nodes from the text tree. We only took relations with high scores of similarity from eXtended WordNet. Examples in which the synonymy relation as given by WordNet is used are: pairs 57 (between *nuclear* and *atomic*), 92 (between *trouble* and *problem*), 114 (between *talk* and *discussion*), etc.

[38] Acronym Guide: http://www.acronym-guide.com
[39] eXtended WordNet: http://xwn.hlt.utdallas.edu/

For every transformation based on DIRT or eXtended WordNet, we consider the similarity value indicated by these resources for local fitness. When we use WordNet, database of acronyms or the background knowledge a local fitness of 1 is considered.

3.3.1.2. Positive Rules for Numbers

As for the 2007 competition, in 2008 a special attention was paid to the name entities of type number. In the test data for the 2008 competition we noticed some special situations involving number type entities. Even if different numbers appear in the text and the hypothesis, we have some quantification expressions that change their meaning and in the end we have equivalent relations between the text and the hypothesis. For example, at pair 304, we have in the text "*at least 80 percent*" which is equivalent to "*more than 70 percent*" from the hypothesis. With our initial rule from 2007, we compare only the numbers (80 and 70 in this case) and the final verdict is a negative entailment based on the fact that these numbers are different. With the new rule introduced in 2008, we determine intervals from those quantification expressions (percents over 80 for text and percents over 70 for hypothesis) and because these intervals overlap, the final answer is a positive entailment in this case. We built a list of quantification expression that contains, for example: "more than", "less than", or words like "over", "under" and in these cases we consider some intervals instead of considering only the numbers.

Another situation is the case of the pair 331 where we have in the text "*killing all 109 people on board and four workers on the ground*" and in the hypothesis we have "*killed 113 people*". As can be seen, adding "109" and "four" from the text, "113" is obtained that represents the value in the hypothesis. For reasons that are easily discovered in the above example, a rule takes into consideration an additional number belonging to one sentence (text or hypothesis) which is obtained as the sum of consecutive numbers separated by the word "and" or commas.

3.3.2. Cases of No Entailment

3.3.2.1. Basic Negative Rules

If after all checks are made, there remain nodes of the hypothesis tree that cannot be mapped onto text tree nodes, local fitness of the unmapped node is penalised. Also, because the stop words of the hypothesis (*the*, *an*, *a*, *at*, *to*, *of*, *in*, *on*, *by*, etc.) would artificially increase the value of global fitness, they are not taken into consideration in the final global fitness.

3322. Negation Rules

For every verb from the hypothesis a Boolean value is considered, which indicates whether the verb is negated or not. To find that, we check inside its tree on its descending branches to see whether one or more of the following words can be found: *not*, *never*, *may*, *might*, *cannot*, etc. For each of these words the initial truth value associated with the verb is successively negated, which by default is "false". The final value depends on the number of such words.

Another rule 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 *believe*, *glad*, *claim* or their synonyms, or adjective like *necessary*, *compulsory*, *free* or their synonyms, or nouns like *attempt*, *trial* and their synonyms, the meaning of the verb in the infinitive form is stressed upon and becomes "certain". For all other cases, the "to" particle diminishes the certainty of the action expressed in the infinitive-form verb.

We will see below more exactly how these rules were split into rules for *contradiction cases* and rules for *unknown cases*. XML files used for identification of negation are given in Appendix 4.

3.3.2.2.1. Contradiction Cases

For negation rules *contradictions cases* are considered: when verbs are negated with words like "never", "not", "no", "cannot", "unsuccessfully", "false" etc. This case is exemplified by the pair 660 where in text is expression "*Aquacell Water, Inc announced today that it has* ***not received*** …" and in hypothesis is expression "*Aquacell Water* ***receives*** …".

Also, we consider cases when before the particle "to" there is one of words "refuse", "deny", "ignore", "plan", "intend", "proposal", "able", etc. These situations appear for example at the pair 54 where the text is "***Plans to detain*** *terrorist suspects for up to 42 days without charge* …" and the hypothesis is "*Police* ***can detain*** *terror suspects for 42 days without charge.*", and at pair 354 where in text is "… *Shin was sacked from the school on June 20 after* ***refusing to resign*** *from his post as director of KBS*." and the hypothesis is "*Shin Tae-seop* ***resigned*** *from his post at Dong-eui University*.".

An important situation in the identification of contradiction is that when there is an *antonymy relation* between words belonging to text and hypothesis. In order to identify such a relation the [*opposite-of*] records from the VerbOcean resource are used (Chklovski and Pantel,

2004) and the *antonymy* relation from WordNet. Examples of opposite relations from VerbOcean are for the pairs 8 (between *increase* and its opposite *decrease*), 885 (between *pay* and its opposite *sell*). Examples of the antonymy relation from WordNet are for the pairs 28 (between *low* and *increase*) and for 48 (between *leave* and *stay*).

In order to catch more situations, a combination between synonyms from WordNet and the antonymy relation from WordNet or the opposite relation from VerbOcean is considered. So, using only the WordNet, for words from the hypothesis, which cannot be mapped to words from the text using the synonymy relation or the antonymy relation, we consider the set of antonyms for each of their synonyms and we check if something from this new set can me mapped to the text. This is the case of the pair 302, where in the hypothesis is the word *separate*, synonym to the word *distinct*. The antonym of the word *distinct* is the word *same* which appears in the text. Thus, between the word *separate* of the hypothesis and the word *same* of the text an antonymy relation is found, even if this is not explicit in WordNet. Similarly, we consider corresponding sets using the synonyms from WordNet and opposite relation from VerbOcean.

Checking the *similarity relation* from DIRT we observe that in some situations it is an *antonymy relation*, and for this reason we do an extra verification of DIRT relations to see if it is antonymy in WordNet or in VerbOcean. If an antonymy relation is identified, the local fitness of the node is changed with a negative value. For example, initially at pair 167 we have, using DIRT, a relation between *convict* and *acquit* with score 0.302455, but because we found in WordNet that *convict* and *acquit* are antonyms, the local fitness score of hypothesis verb is changed and a penalty is inserted.

For all identified *contradiction cases*, since we consider the penalties with the highest values direct in the global fitness, in the end, the final answer for the considered pairs will be "*Contradiction*".

3.3.2.2.2. Unknown Cases

From the negation rules, *unknown cases* are considered: when words are "may", "can", "should", "could", "must", "might", "infrequent", "rather", "probably", etc. In these cases, inserted penalties are not decisive in final answer establishing, which is obtained only after the calculation of global fitness. At pair 198 in the text is "… *could also be linked to* …" and in hypothesis is "… *is linked to* …".

Related to the particle "to" we will consider the cases which are not included in

contradiction cases. So at pair 391 with the text "*It **is hard to like** Will Carling …*" and the hypothesis "*Nobody **likes** Will Carling*" because "hard" is not in the list with contradictions cases, the verb "like" will receive a penalty.

At the cases presented above we only insert penalties and only in the end, after the calculation of the global fitness, we will know the final result for a pair. The things are different regarding *named entities* with problems.

Named Entity Rule: In the event that even after using acronyms database and background knowledge we cannot map one *named entity* from the hypothesis to a named entity from the text tree, we decide that the final result for the current pair is "*Unknown*". This case is at pair 454 of below, in which identify the named entities *U.S.* and *Russia* in the hypothesis without a corresponding value in the text.

T: In 1977 to wide media fanfare, Polanski was charged with a host of sexual crimes for his involvement with a 13-year-old girl. He was subsequently convicted of unlawful intercourse with a minor, but fled the country in 1978 before final sentencing.

*H: Polanski fled from the **U.S.** to **Russia** in 1978.*

For the numbers of the text and of the hypothesis, when it is possible, we also keep their "unit measure". The rule presented above for named entities is also applied in cases in which we have the same numbers in text and in hypothesis, but we have different "unit measures" for them. This is the case for pair 441 where in the hypothesis is *11 troops* and in the text is *11 September*:

*T: Britain deployed troops to Afghanistan shortly after the attacks of **11 September**, 2001. Few then thought that British forces would still be in Afghanistan in far larger numbers seven years on, nor that they would be involved in some of the fiercest fighting British forces have seen in decades, as part of Nato's International Security and Assistance Force (ISAF).*

*H: Britain has **11 troops** that take part in Nato's International Security and Assistance Force.*

An exception from the named entity rule presented above is the case when the type of name entity is *first name*. In this case we only insert a penalty in the global fitness. This is the case of pair 122 from below where in the hypothesis has *Gordon Brown* and in the text has *Mr Brown*:

*T: **Mr Brown**'s courage and determination are not in doubt: he soaks up punishment as if he believes it is good for him. But week after week he gives no sign that he knows how to seize the initiative and dictate the course of the fight.*

*H: **Gordon Brown** is the UK Prime Minister.*

333. Fitness calculation

The above steps indicate what helps our program in identifying the final answer for every pair from our test data. In situations in which we don't identify obvious *unknown* or *contradiction* cases the final answer is decided only after global fitness calculation. Accordingly, with an applied tool or with used resources the local fitness score and implicit the global fitness score are increase or decrease, and in the end the final answer is obtained. We discuss below how the local fitness and the extended local fitness are calculated for every word from the hypothesis, and after that how the global fitness is obtained (a normalized value for all words from hypothesis).

3331. Local Fitness Calculation

For every node from hypothesis tree which can be mapped directly to a node from the text tree, the system will consider the local fitness value to be 1. For nodes without a direct mapping the following possibilities are envisaged:

- If the word is *a verb* and the mapping was based on the DIRT resource and the antonymy relation between words is not identified, the local fitness is considered to be equal with the DIRT similarity score. If any kind of antonymy between verbs with the same value of negation is identified (using VerbOcean or WordNet or DIRT, or a combination of them) the contradiction rule is applied and the final answer is "Contradiction".

- If the word is marked as *named entity* and the acronyms' database is used or the "is" relation from the Background Knowledge is used, then the local fitness is set to 1. For the "in" relations from the Background Knowledge ("in", "located in" and "live in") the local fitness is set to 0.5. For all other relations or when we don't find any relation between a name entity from the hypothesis and all name entities from the text, the name entity rule is applied and the final answer will be "Unknown".

- In case of *nouns* and *adjectives*, the local fitness is considered equal to 1 when

WordNet is used for mapping and corresponding similarity score from eXtended WordNet when it is used.

If after all checks one node from the hypothesis tree cannot be map, its local fitness is set to -0.5 value.

For a word from hypothesis, the local fitness value informs us about the way in which it was mapped to a node from text. High positive values indicate a good node mapping with a similar node from text. Lower positive values or even negative values indicate problems in mapping. *What don't we know about mapping*? The **context** in which the mapping was done: *Do the nodes have the same father*? *If yes, do the nodes have the same relations to it*? etc. The way in which the extended local fitness value is calculated comes to fix this inconvenient.

3332. Extended Local Fitness

The *Extended Local Fitness* is calculated for every node from the hypothesis as the average of the following values:

1. local fitness obtained after the tree transformation and node mapping,
2. parent fitness after parent mapping,
3. mapping of the node edge label from the hypothesis tree onto the text tree,
4. node position (left, right) towards its father in the hypothesis and position of the mapping nodes from the text.

After calculating this *extended local fitness* score, the system computes a *total fitness* for all the nodes in the hypothesis tree and a *negation value* associated to the hypothesis tree. Tests on data have shown that out of these parameters, some are more important (1.) and some have a smaller significance (3.). Below you can observe an example of how the calculations for 3 and 4 are performed. Because all values from (1.) to (4.) are in the range -1 to 1 then their average will also be in this range. Then the extended local fitness value will be in the interval [-1, 1].

3.3.3.2.1. Edge label mapping

After the process of mapping between nodes, the program checks how edge labels from the hypothesis tree are mapped onto the text tree. Thus, having two adjacent nodes in the hypothesis, which are linked by an edge with a certain label, it searches this label on the path between the nodes' mappings in the text tree. (See Figure 10).

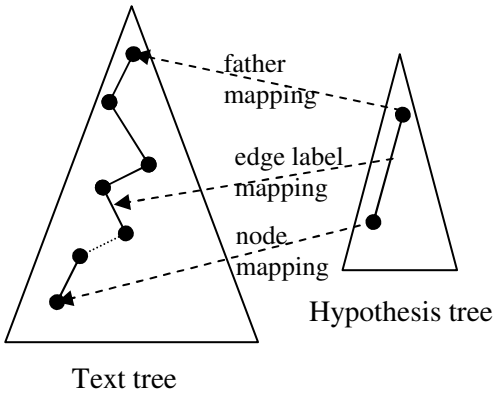


Figure 10: Entity mapping

It is possible that more nodes exist before the label of the edge linking the nodes in the hypothesis, or it is possible that this label is not even found on this path. According to the distance or to the case in which the label is missing, some penalties are inserted in the extended local fitness.

When the distance is minimum (one single edge) the value added to the extended fitness is maximum i.e. 1. For every additional node which appears on this path between nodes the maximum value is divided by 2. If the label is missing, instead to add a positive value to the extended local fitness a value of -0.5 is added.

3.3.3.2.2. Node position

After mapping the nodes, one of the two following possible situations may be encountered:

- The position of the node towards its father and the position of the mapping node towards its father's mapping are the same (*left-left* or *right-right*). In this case, the extended local fitness is incremented by 1.
- The positions are different (*left-right* or *right-left*) and in this case a penalty with value -1 is applied accordingly.

3333. Global fitness calculation

For every node from the hypothesis tree, the value of the extended local fitness was calculated, and afterwards the normalized value relative to the number of nodes from the hypothesis tree was considered except for stop words. This result was denoted by *TF* (*Total Fitness*):

$$TF = \frac{\sum_{word \in H \setminus StopWords} ExtendedLocalFitness_{word}}{|H \setminus StopWords|}$$

Because all extended fitness values are in the [-1, 1] interval, the TF value will be in the same interval. After calculating this value, the *NV* (*negation value*) is computed. This number indicates the percent of positive verbs from the hypothesis. The formula used is the following:

$$NV = \frac{\#\_of\_Positive\_Verbs}{TotalNumberOfVerbs}$$

where the *#_of_Positive_Verbs* is the number of non-negated verbs from the hypothesis as given by the negation rules, and *TotalNumberOfVerbs* is the total number of verbs from the hypothesis.

Because the maximum value for the extended fitness is 1, the complementary value of the *TF* is *1-TF*. The formula for the **global fitness** used is the following:

$$GlobalFitness = NV * TF + (1 - abs(NV)) * (1 - abs(TF))$$

Because the TF value is in [-1, 1] interval and NV is in [0, 1] interval the *GlobalFitness* value is in [-1, 1] interval.

3.3.4. Obtaining the Final Answer using the Global Fitness Value

The "No entailment" cases are represented by pairs of text and hypothesis for which the value of global fitness is under a threshold identified on training data, and the "Entailment" cases are represented by pairs for which the global fitness is over the same threshold. In order to differentiate contradictions and unknown cases, we considered another threshold, also identified on training data. Since penalties for *contradiction cases* are higher than penalties for *unknown cases*, the order is as follows: the lower level for global fitness, the contradiction cases, first threshold, unknown cases, the second threshold, the entailment cases, and the highest level for global fitness. (See picture below).

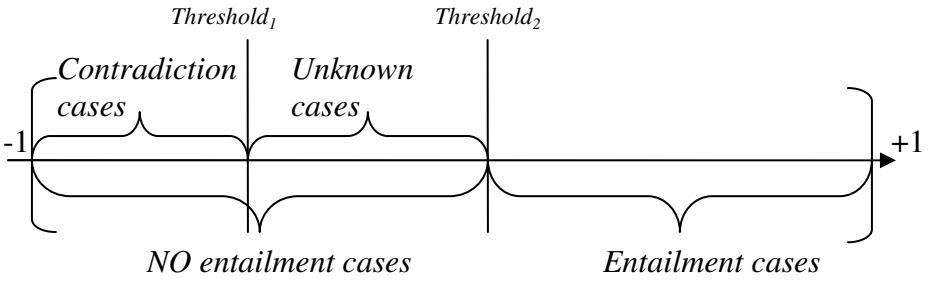


Figure 11: Final answers consideration using Global fitness values

The threshold values were identified on the training set data, and there are small differences accordingly to the task used to obtain the data. Thus, the $threshold_1$ was identified to be the same for all tasks, i.e. -0.9, and the $threshold_2$ values are between 0.01 and 0.07.

3.3.4.1. Example of Entailment Cases

For the pair 245 from the test set we have the greatest fitness score because all entities from hypothesis were found in the text tree in the same form. For the same reason, we don't use external resources in this case. The pair is presented below:

T: A miniature Pinscher dog in Germany has adopted a baby tiger. The tiger, named Siera, was born at the end of December in a circus based in the town of Wülfrath. But her mother couldn't give her any milk so she had to be hand fed from a baby bottle.

H: A dog has adopted a tiger.

The extended local fitness's are as in the next table:

Initial entity	**Node Fitness**	**Extended local fitness**	**Motivation**
(a, dog, det)	*1*	*1*	The same entity was found in the text tree
(dog, adopt, s)	1	1	
(have, adopt, have)	1	1	
(adopt, -, -)	1	1	
(a, tiger, det)	*1*	*1*	
(tiger, adopt, obj)	1	1	

Table 22: Extended local fitness calculation for pair 245

Total_Fitness = (1 + 1 + 1 + 1) / 4 = 1 (The stop words are ignored.)

NV = 1/1 = 1

GlobalFitness = 1 * 1 + (1 – 1) * (1 – 1) = 1

This is an ideal case, a very uncommon one, because in most of the cases the extended local fitness values and global fitness being less than 1. Below is the case of pair 72.

T: A key UN-sponsored summit has opened in Rome aimed at addressing the problem of soaring global food prices.

H: UN summit targets global food crisis.

The extended local fitness's are like next table:

Initial entity	Node Fitness	Extended local fitness	Motivation
(UN, summit, nn)	1	0.515	Another words are on path between "UN" and "summit" in text tree (*)
(summit, target, s)	1	0.812	(*)
(target, -, -)	0.0855	0.761	DIRT recovers the relation between "target" and "address" with the score 0.0855
(global, crisis, mod)	1	0.501	(*)
(food, crisis, nn)	1	0.501	(*)
(crisis, target, obj)	-0.5	-0.5	For "crisis" no mapping can be found in the text tree

Table 23: Extended local fitness calculation for pair 72

Total_Fitness = (0.515 + 0.812 + 0.761 + 0.5 + 0.5 – 0.5) / 6 = 0.4317

NV = 1/1 = 1

GlobalFitness = 1 * 0.4317 + (1 – 1) * (1 – 0.4317) = 0.4317

Because the value is greater than the second threshold the decision in this case for pair 72 is "Entailment".

3342. Example of Unknown Cases

For pair 270 the opposite of the case presented above for pair 245 happens: no mapping can be established for any entity from the hypothesis tree to an entity from the text tree, and the attempts to use external resources are useless. In this case the global fitness value is minimum i.e. -0.5.

T: A senior consultant in the care of older people, Dr Adrian Treloar, stated at the end of last year that elderly patients are being left to starve to death in NHS hospitals because of an unspoken policy of involuntary euthanasia designed to relieve pressure on the NHS (Laville and Hall, 1999).

H: Age discrimination is outlawed.

The extended local fitness's are like in the next table:

Initial entity	Node Fitness	Extended local fitness	Motivation
(age, discrimination, nn)	-0.5	-0.5	only lemma for "be" was found in the text tree, but it has different father lemma
(discrimination, outlaw, s)	-0.5	-0.5	
(be, outlaw, be)	*1*	*0.5*	
(outlaw, -, -)	-0.5	-0.5	

Table 24: Extended local fitness calculation for pair 270

Because "be" is a stop word, the global fitness goes down to -0.5. Because this value is positioned between $threshold_1$ and $threshold_2$ the answer in this case is "*Unknown*".

Usually, cases of "Unknown" correspond to those in which a part of the entities cannot be mapped, and the global fitness is negative. This is the case for pair 4 presented below:

T: Save the Children demanded action after its research found that starving and desperate youngsters as young as six were being coerced to sell sex for food, money, soap and even mobile phones in war zones and disaster areas.

H: UN peacekeepers abuse children.

The extended local fitness's for pair 4 are presented below:

Initial entity	Node Fitness	Extended local fitness	Motivation
(UN, peacekeeper, nn)	-0.5	-0.5	only for "child" was found a corresponded in the text tree
(peacekeeper, abuse, s)	-0.5	-0.5	
(abuse, -, -)	-0.5	-0.5	
(child, abuse, obj)	1	0.325	

Table 25: Extended local fitness calculation for pair 4

Total_Fitness = (– 0.5 – 0.5 – 0.5 + 0.325) / 4 = - 0.2938

NV = 1/1 = 1

GlobalFitness = 1 * (- 0.2938) + (1 – 1) * (1 – 0.2938) = - 0.2938

Again, the value is between $threshold_1$ and $threshold_2$ and the answer in this case is "*Unknown*".

3.4. *Results*

The results presented in this chapter are related to the competitions in which our system has participated: RTE-3 and RTE-4. First the results are presented separately, and after that we compare the results and check the differences.

3.4.1. Results in RTE3

The first edition of our system built for RTE-3 competition was designed to tackle the 2-way task. Based on it, a second version was afterwards developed for the 3-way task. For the 2-way task, the answer distribution is presented in the next table:

Answer Type	# of answers in Gold	# of correct answers given by our system	Total # of answers given by our system	Precision	Recall	F-measure
Yes	410	379	594	63.80 %	92.44 %	75.50 %
No	390	174	206	84.47 %	44.62 %	58.39 %
Total	**800**	**553**	**800**	**69.13 %**		

Table 26: UAIC Textual Entailment System Results in RTE3 in the 2-way task

We can notice the better behaviour for "Yes" answers, where out of the 410 answers in gold we correctly find 379 answers (this gives the highest values for recall: 92.44 % and F-measure: 75.5 %). However, unfortunately, "Yes" answers are given also for a great number of gold-classified "No" answers, which lowers the precision down to 63.8 %. Another problem with this run was the lower number of "No" answers offered by the system: only 206 out of the 390, which brings the recall as down as: 44.62 %. The precision of the "No" answers was, however, high – 84.47 %, which results in an F-measure of 58.39 %.

From these results we can conclude that the rules related to negation and to name entities have a very good precision and identify with a high accuracy the No entailment cases. The high number of Yes answers indicates the fact that the system prefers this answer when it doesn't have clear evidence for identifying the "Yes" or "No" answers.

The 3-way task shows equal results for "Yes" answers as in the core of the 2-way task, and the "Contradiction" and "Unknown" cases split in two categories the "No" answers. This is due to the way contradiction and unknown are made to share the "No" answers by adding a new threshold, as explained in section 3.3.4. Results presented below indicate the weakness of the

rules for identification of “Contradiction” and “Unknown” cases:

Answer Type	# of answers in Gold	# of correct answers given by our system	Total # of answers given by our system	Precision	Recall	F-measure
Entailment	410	379	594	63.80 %	92.44 %	75.50 %
Contradiction	72	10	115	8.70 %	13.89 %	10.70 %
Unknown	318	67	91	73.63 %	21.07 %	32.76 %
Total	**800**	**456**	**800**	**57.00 %**		

Table 27: UAIC Textual Entailment System Results in RTE3 in the 3-way task

Our assumption that the cases in which we have *named entities* problems represent cases of “Contradiction” answers seems to be wrong. From here all lower results for *contradiction* case: we offer more “Contradiction” answers in comparison with number of “Unknown” answers and most of them were wrong. Once again, like in the 2-way case we notice that we offer a low number of “Unknown” answers, but 73.63 % of them were right.

3.4.2. Results in RTE4

In this competition our attention was oriented towards the improvement of the system for the 3-way task. The results for the 2-way task represent mainly a side effect of this specific focus. For this reason we first present the results for the 3-way task and after that the results for the 2-way task. Below are the results from the 3-way task:

Answer Type	# of answers in Gold	# of correct answers given by our system	Total # of answers given by our system	Precision	Recall	F-measure
Entailment	500	466	712	65.45%	93.20%	76.90%
Contradiction	150	69	85	81.18%	46.00%	58.72%
Unknown	350	150	203	73.89%	42.86%	54.25%
Total	**1000**	**685**	**1000**	**68.50%**		

Table 28: UAIC Textual Entailment System Results in RTE4 in the 3-way task

The results show, the highest precision for *Contradiction*: 81.18 % and the lowest precision for *Entailment*: 65.45%, although *Entailment* gets the highest recall and F-measure

93.2% and respectively 76.9 %. The lowest values are for the *Unknown* case, 42.86% and 54.25 %. The meaning of these results is that our system offers very many *Entailment answers* and catches almost all possible *Entailment* cases; also the system offers a lower number of *Contradiction* and *Unknown* answers, but almost all are correct.

The results show that the rules related to contradictions and unknown cases are very efficient and identify with a high degree of precision these types of answers. The high number of Entailment answers indicates the fact that after training, the system prefers to choose the "Entailment" answer when it does not have evidences for clear identification of either of the "Entailment", "Contradiction" or "Unknown" answers.

The 2-way task distribution is presented in the table below:

Answer Type	**# of answers in Gold**	**# of correct answers given by our system**	**Total # of answers given by our system**	**Precision**	**Recall**	**F-measure**
Yes	500	466	712	65.45%	93.20%	76.90%
No	500	255	288	88.54%	51.00%	64.72%
Total	**1000**	**721**	**1000**	**72.10%**		

Table 29: UAIC Textual Entailment System Results in RTE4 in the 2-way task

The results are similar to results from the 3-way task and we notice the high precision for No cases (88.54%), where from 288 answers offered by our system 255 are correct. The meaning of the difference between global precision from 2-way task and 3-way task is that only in 36 cases from 255, we don't distinguish correctly between "*Contradiction*" and "*Unknown*" cases.

3.4.3. Comparison between Results

In comparison to results from RTE3 we can see that accordingly to the provenience of testing data we have significant improvements on IR, SUM and IE tasks, but on QA task, where we got the best results in RTE3, we have the lowest result in RTE4.

Task	**RTE3**	**RTE4**
IR	69.00 %	**82.00 %**
QA	**87.00 %**	63.00 %
SUM	63.50 %	**78.00 %**

Task	RTE3	RTE4
IE	57.00 %	**64.33 %**
Total	69.13 %	**72.10 %**

Table 30: Comparison on tasks between results in RTE3 and RTE4

Following the RTE-3 competition in order to determine each component's relevance, the system was run in turn with each component removed. The same technique was used after RTE-4. Table 31 presents these results in parallel for RTE-3 and RTE-4.

System Description	RTE-3 (69.13 % precision)			RTE-4 (72.1 % precision)		
	Precision (%)	Contribution (%)	Weighted Relevance (%)	Precision (%)	Contribution (%)	Weighted Relevance (%)
Without DIRT	68.76	0.37	0.54	71.40	**0.7**	**0.97**
Without WordNet	68.00	1.13	1.63	69.10	**3.0**	**4.16**
Without Acronyms	68.38	**0.75**	**1.08**	71.80	0.3	0.42
Without BK	67.75	1.38	2.00	70.40	**1.7**	**2.36**
Without the Negation rule	67.63	1.50	2.17	68.70	**3.4**	**4.72**
Without the NE rule	57.58	**11.55**	**16.71**	66.90	5.2	7.21
Without the Contradiction rule	-	-	-	68.10	**4.0**	**5.55**

Table 31: Components' relevance

The meanings of the columns for RTE-3 competition are the following (similar for RTE-4 columns):

- $Precision_{Without\_Component}$ value was obtained running the RTE-3 system without a specific component (for example, $Precision_{Without\_DIRT}$ is 68.76 % and it represents the precision of the RTE-3 system without the DIRT component);
- $Contribution_{Component} = Full\_system\_precision - Precision_{Without\_Component}$ (for example, $Contribution_{DIRT}$ is 69.13 % - 68.76 % = 0.37 % for the DIRT component of the RTE-3 system, where 69.13 % is the precision for the full RTE-3 system and 68.76 % is the precision for RTE-3 system without DIRT component);

- $WeightedRelevance_{Component} = \frac{100 \times Contribution_{Component}}{Full\_system\_precision}$ (for example, for the DIRT component in RTE-3, $WeightedRelevance_{DIRT} = \frac{100 \times Contribution_{DIRT}}{Full\_system\_precision} = \frac{100 \times 0.37}{69.13} = 0.54\%$).

The results in the Table 31 show that the system's rules related to negation, named entities and contradictions are the most important. We can see how in the RTE4 the distinction between these cases is done better and therefore the better results on 3-way task. Also, we noticed that using WordNet and of a part from eXtended WordNet was benefic for the system in the RTE-4 competition.

3.5. *Limitations of the System*

Here, we will present the limits of the system and we will give some examples of the cases in which the system is unable to take the correct decision for different reasons.

3.5.1. Resource limitations

In significant number of cases, the system is unable to find a relation between a word belonging to the hypothesis and a word belonging to the text. For example, when for the word from hypothesis there is in the text a word with a similar meaning, but the system cannot perform the mapping, an artificially penalty is inserted into the local fitness of the hypothesis word and the "Entailment" case is not identified. Similar the "Contradiction" case, especially when the word from the hypothesis represents something else than the word from text (antonyms, different things, etc.) and inserted penalty is not enough for identification the "Contradiction" case. Examples of such cases are presented below.

Entailment cases problems:

The lack of paraphrasing ability: this is the case of pair 137, where in text is expression "*are raised in ways that are ethically and environmentally unsound*" and in hypothesis is the expression "*are reared cruelly*". These expressions represent the same thing, but this cannot be proved with existing used resources. The same situation is at pair 149 where the expression in text is "*when gets wet her body explodes in sore*" and in the hypothesis expression is "*is allergic to water*".

The lack of rules for obtaining extra information: in some cases, from constructions

like "*Ford has sold car brands Jaguar and Land Rover to Indian company Tata*" can be obtained "*Tata takes over Jaguar*" (pair 238). Similar, from "*German retail corporation Rewe*" can be obtained "*Rewe is based in Germany*" (pair 951). The first case can be deduced if the system uses the following rule: if (*X sold Y to Z*) then (*Z takes Y*), and a similar rule can be built in the second case.

Contradiction cases problems:

The lack of paraphrasing ability: at pair 676 the expression in text is "*aren't the only species in danger*", and it is equivalent with expression "*are in danger*" which is opposite to hypothesis expression "*aren't in danger*". Because, the first equivalence cannot be deduced with existing resources, the contradiction between the text and the hypothesis cannot be identified. Similar at pair 689 where the expression belonging to the text "*create the Rapid Disaster Response Blog*" is similar to expression "*create a Response Blog about Rapid Disaster*" which is different than the expression belonging to the hypothesis "*started a rapid disaster*".

The lack of rules for obtaining extra information: at pair 13, the expression of text "*Becker has never played tennis in his life*" is in contradiction with expression of hypothesis "*Becker was a tennis champion*". If a rule to obtain additional information from hypothesis exists in the form: if (*X is SPORT champion*) then *X plays SPORT*, then this new information can be used and it shows contradiction with the information in the text. Similar, at pair 429 where in text is information "*Florida is the second largest nursery crop-producing state*" and a rule that says how "*the second largest*" ≠ "*the largest*" can deduce that this information of text is in contradiction with information of hypothesis "*Florida is the largest producer of nursery crop*".

Missing resources: at pair 64 in text appears the sentence "*All the victims are adults*". Because there does not exist a resource with information "*adults*" are different than "*children*" the system cannot deduce that "*Children were not killed*". This is in contradiction to the hypothesis "*A French train crash killed children*". At pair 587, the text presents how a "*dog bit his hand during a dog show*", and because in existing used resources an explicit relation between "*hand*" and "*foot*" doesn't exist the system cannot deduce how the text is in contradiction to the hypothesis which says "*a dog bit his foot*".

Unknown cases problems:

Missing Resources: at pair 100 in text we have information "*diplomats were today attacked*" and in hypothesis information "*Diplomats were detained*". The system has not a

resource from where to know whether the attacked persons can be *detained* or not after an *attack*. From this reason it cannot do an association between verb *attack* belonging to the text and verb *detained* belonging to the hypothesis.

The lack of rules for obtaining extra information: in some contexts the "Unknown" behaviour can be deduces from rules for pairs of verbs. Thus, if (*X died in Y*) then "*we cannot deduce that X lived in Y or not*" and in this case pair of verbs (*die*, *live*) induce an "Unknown" context. This situation is encored at pair 312 where in text we have "*death of Ben Kinsella in north London*" and in hypothesis "*Ben Kinsella lived in north London*". A similar rule is the following: if (*X talks about Y*) then "*we cannot deduce that X does Y or not*". This is the case of pair 318 where in text we have "*Sara Hiom said the research highlighted the importance of a healthy balanced diet*" and in hypothesis we have "*Sara Hiom is on a diet*". In both cases, the system is unable to identify the "Unknown" context for these pairs of verbs and from this reason; the system is unable to offer the "Unknown" answer.

352. Insufficient exploitation of Geographical Resources

This drawback comes from the fact that the "neighbour" relation is identified between named entities, but we are unable to exploit this additional information. For example at pair 190, where in text it is said that "*A strong earthquake struck off the southern tip of Taiwan*" and in the hypothesis is presented how "*An earthquake strikes Japan*". In this case the "neighbour" relation between "Taiwan" and "Japan" is identified, but we cannot use some additional relations which can be used in the case of *earthquakes*: if (*the geographical regions are sufficiently closed*) then "*accordingly with earthquake power, it is possible that the earthquakes to be felt in both regions*". The variables in this case are the *distance between regions* and the *power of the earthquake*.

A similar situation is encountered at pair 104 reproduced below:

T: Oil leaking from the MS Explorer cruise ship, which sank in Antarctic waters, could cause an environmental disaster near the Antarctic shore, a top-level Argentine foreign ministry official said on Thursday.

H: An oil spill threatens Buenos Aires.

As we can see, the variables in this case are the *distance* between *Antarctic* (which is "neighbour" with *Argentine*) and *Buenos Aires* (which is "in" *Argentine*) and the *dimension of oil leaking*.

353. Missing of Semantic Role Labelling

This drawback affects the correct identification of "Unknown" and "Contradiction" cases, since in those cases, identifying that some words have different semantic roles in text and in hypothesis can help the entailment decision. Some classes of examples are presented below:

Contradiction cases:

At pair 483, the text "*Sheriff's detectives are looking for a 34-year-old former Oceanside resident they say went on a four-day shopping spree in April with her company credit card and racked up more than $150,000 in purchases*" has the verb "*rack*" with arguments A0 "*a 34-year-old former Oceanside resident*" and A3 "*more than $150,000*". In the hypothesis, "*Sheriff's detectives spent $150,000 shopping*" the argument A0 for verb "*spend*" is "*Sheriff's detectives*" and the argument A3 is "*$150,000*". In both cases the *price paid* argument is the same *$150,000*, but the A0 arguments (that represent the persons that paid the price) are different.

A similar situation is met at pair 551 with text "*United Kingdom flag carrier British Airways (BA) has entered into merger talks with Spanish airline Iberia Lineas Aereas de Espana SA. BA is already Europe's third-largest airline*" and hypothesis "*The Spanish airline Iberia Lineas Aereas de Espana SA is Europe's third-largest airline*". From the verb "be" we have the same argument A2 "*Europe's third-largest airline*" but we have different A1 arguments: "*BA*" in text and "*SA*" in the hypothesis.

Unknown cases:

The problems presented above are encored also for the "Unknown" cases. In the pair 447, the text "*Mr. de la Vega worked closely with Apple Chief Executive Steve Jobs on the deal that made AT&T the exclusive U.S. provider of the iPhone, which was a smash hit in its first year*" the A0 (*worker*) argument of the verb "work" is "*Mr. de la Vega*" and the argument A3 (*coworker*) is "*Apple Chief Executive Steve Jobs*". In the hypothesis "*Mr. de la Vega is the AT&T Chief Executive*" the arguments A0 (worker) and A3 (coworker) from text are put in a new relation through the verb *to be*. Thus, the *worker* must be *coworker*, but we need extra information in order to decide that, and the final answer must be "Unknown".

354. Useless Rules

Until now, in the presentation of the system, we insist on the rules that have a positive impact over the system (*Contradiction rule, Negation rule, Named entities rule, etc.*), called **positive**

rules. Additionally, other rules were tested on training data, but because their impact was insignificant we removed them from the final system – **useless rules**. In the following sections, the main characteristics of the *useless rules* are presented, and we hope that in the future, a better detailed specification for them will help the system in taking correct decisions.

3.5.4.1. The "but" rule

We noticed that *but* is one of the words that can change the verb meaning. Usually, in these cases an initial statement is negated by a statement that follows the conjunction *but*. This is the case of pair 993:

> *T: White was approached by Harry M. Miller to work on a screenplay for Voss, but nothing came of it.*
>
> *H: White wrote a screenplay for Voss.*

where the initial statement from text "*White was approached by Harry M. Miller to work on a screenplay for Voss*" is negated in the second part: "*but nothing came of it*". The first part of the text is in the "Entailment" relation with the hypothesis. *But* changes this relation into "Contradiction".

The case presented above is an ideal case. Most cases are more complicated, and when the text is longer and more complicated, we are unable to identify if we have negation or not after *but*, and the hardest part is to identify to which previous verb the identified negation is applied. For the pair 555:

> *T: Puffin numbers are declining on the Farne Islands, where the United Kingdom's largest colony resides. During the past five years, the puffin population on the islands has decreased by a third. Experts had expected numbers to rise this year, but now with news of the decline, are citing an inability to find food as the probable cause.*
>
> *H: Puffin numbers are falling on the UK's Farne Islands.*

we must identify how "*but now with news of the decline, are citing an inability to find food as the probable cause*" is a negation which is applied to the verb "*rise*". In the end the answer is "Entailment" because negated "rise" from text is equivalent to "fall" from hypothesis.

3.5.4.2. Simple Rules for Semantic Role Labelling

In order to cover the drawback that comes from the fact that we don't use a special tool to

identify the semantic role for sentence components, we built our module that identifies only a part of verbs arguments. Thus, we start from the dependency tree built by MINIPAR and identify obvious arguments from it. For example, the A0 and A1 arguments are set using the "s" and "subj" MINIPAR relations (see Table 50 from Appendix 8.1 for more details about MINIPAR relations), AM-LOC argument is set when in dependency tree are identified context words like "from", "in", "region", etc.

With only few arguments, our preliminary tests on set data from RTE-3 improved the system precision with almost 2 %. This rule was useful for short pairs of text and hypothesis and useless for long pairs of text and hypothesis, because in these cases it is possible for the same argument from hypothesis to appear in more places in the text, with different roles.

In RTE-4 the number of long pairs was greater than in the previous edition, and the number in which this rule was useless was greater than the number in which this rule was useful. For this reason, this rule was not used.

3543. Simple Rules for Anaphora Resolution

Another rule with low influence over the final result was the rule related to the anaphora resolution. Anaphora resolution is the problem of resolving what a pronoun, or a noun phrase refers to.

A simple example in which our program finds the correct solution is at pair 757 where the text is:

> *T: Spencer Dryden was a member of Jefferson Airplane during the band's heyday. He replaced original drummer Skip Spence and played with Jefferson Airplane at the legendary Woodstock and Altamont rock festivals.*

"*He*" from the second sentence is related to the only male name from first sentence "*Spencer Dryden*". Similar situations were solved successfully for the pronouns "*she*" and "*her*", which found female antecedents, or for the pronoun "*his*". In the rest of the situations, we don't replace the pronouns with antecedents, because we haven't yet found a reliable solution for anaphora resolution.

The mapping between the verbs of the hypothesis and the verbs of the text is influenced by the replacement of the pronouns as incurred by a pronoun resolution rule. From our tests on RTE-3 data, we noticed that the number of good mappings is similar to the number of bad mappings, after applying this rule. Anyway, the rule has been kept in the final system, and our

attention will be focused on its improvements in the future.

3.6. *Future Work*

3.6.1. Machine Learning used for Answer Classification

In the paper (Iftene and Balahur-Dobrescu, 2008a), we present a generic system built for recognizing textual entailment. The system is similar to the system used in RTE-4 without some modules and with new additional components. One new component is related to the answers' classification. In this case, 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 (Quinlan, 2000) and SMO (Platt, 1998) for Support Vector Machine (SVM).

In order to classify the pairs, the C4.5 and SMO classifiers was trained, using 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 as characteristics.

Next, the generic system was evaluated using two sets of data from the development and testing parts of the RTE3 competition. Every set consists of 800 texts - hypothesis pairs. The results are close: SVM with precision equal to 0.634 and C4.5 with precision equal to 0.619. For SVM, the results of the predicted classification into Yes and No entailment, against the gold standard, are in the Table 32 below.

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**

Table 32: Detailed results using SVM classification

In order to monitor the system's behaviour, the pairs were trained and classified using SVM on two languages: English and Romanian. For the Romanian system, both the development and test sets of pairs were translated 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.

Language	Development Data	Test Data
English	0.648	0.634
Romanian	0.567	0.561

Table 33: Evaluation results on English and on Romanian using SVM

Similar tests were performed on the output from RTE-3 competition and we notice an improvement of results from 69.13 % during the competition to 72 % after. In the future we want to create our own machine learning algorithm for answer classification.

3.6.2. Resource Building

Starting from RTE-3 and improved in RTE-4, we use our modules and resources in order to identify relations between named entities. The good news is that the use of these modules improves the quality of the system. Unfortunately, from all the identified relations, only few relations are actually used: “is”, “in”, “located-in”, “live-in”, the rest remaining unexploited.

Another inconvenient regarding this part is when a new relation between two specified name entities must be identified. In this case the Wikipedia pages related to one of the name entities are downloaded and statistics on all identified name entities are built. If after this step a relation between initial named entities is not identified, the process presented above is iterated for all new identified named entities, until, eventually, the requested named entity is found. From step to step, the quality of relation between new identified named entities and the initial named entities is lower. It is possible for this process to be time consuming especially if large files from Wikipedia are downloaded. In order to prevent this problem we intend to start from a set of usual name entities (*person names*, *countries*, *cities*, etc) and to run the above algorithm only 3-4 cycles, and to keep only good relations between named entities. The number of cycles and the relation type will be established after tests on some named entities.

3.6.3. Improving the rules

We presented in the previous chapter the need to enhance the quality of rules for anaphora resolution, for semantic role labelling and for negation. For that, we need to do a more detailed analysis of the problems encored and of the cases in which these rules are involved.

The same situation is with *positive rules*, where we want to reduce to minimum the number of cases in which they are incorrectly applied. Error analysis can be very usefully here

and can consistently improve the system behaviour.

Thus, all analysis performed after the two competitions and presented here can substantially improve the result. For example, related to the anaphora resolution part, more simple and efficient rules can be added to the anaphora resolution module. During the competition only rules related to the female and male names were added. After competition, we observed how simple rules can be added for almost all named entities types: City, Country, Organization, Region, etc. For example for type "City" occurrences of words like "city", "capital", "location", "town", etc. must be replaced by previous names of cities. This is the case of pair 734 from below:

T: Maximum sustained winds are reported at 175 mph (78 m/s). Of great concern to New Orleans officials is the predicted coastal storm surge of 18 to 22 feet above normal tide levels. Destruction from flooding is expected since the ***city*** *is situated below sea level.*

H: New Orleans is located below sea level.

As we can see in above text in the part "*Destruction from flooding is expected since the* ***city*** *is situated below sea level*" we can replace the word "city" with the name entity "*New Orleans*" with type "City". After transformation the new text is "*Destruction from flooding is expected since the New Orleans is situated below sea level*". In this new form the text can be easily compared with the hypothesis.

3.7. *Conclusions*

The system's core algorithm is based on mapping between words from hypothesis to words from text using dependency trees, however, focused on transforming the hypothesis. It presently uses wide-spread syntactic analysis tools like MINIPAR and TreeTagger, lexical resources like WordNet and eXtended WordNet, LingPipe and GATE for Named Entities recognition and semantic resources like DIRT and VerbOcean. The system's originality resides firstly in creating a part-of and equivalence ontology using an extraction module for Wikipedia data on NEs (the background knowledge), secondly in using a distinct database of acronyms from different domains, thirdly acquiring a set of important context influencing terms and creating a semantic equivalence set of rules based on English rephrasing concepts and last, but not least, on the technical side, using a distributed architecture for time performance enhancement, as we will see in next chapter.

The approach unveiled some issues related to the dependency of parsing tools, for example separating the verb and the preposition in the case of phrasal verbs, resulting in the change of meaning. The same situation was identified for numbers where the quantification words change numbers in intervals.

Another issue was identifying expressions that change context nuances, which we denoted by “positive” or “negative” words. A careful analysis on these words helps us in identifying individual quantification for every one of these words.

The differences between the system used in RTE-3 and the new system used in RTE-4, comes from how new tools (GATE, TreeTagger), new resources (VerbOcean) and new rules (Contradiction, Named Entities) were used in the last system.

With the new changes, the RTE-4 system precision is improved, and the system is more oriented to an *Entailment-Contradiction-Unknown* system, because the rules for contradictions are more clearly specified. The obtained results from RTE-4 are better than the results from RTE3 participation: 72.1 % on two-way task (with 3 % better than in RTE3) and 68.5 % on three-way task (with 14.5 % better than in RTE3).

The main problems are related to cases in which text and hypothesis are very similar and contain the same set of words, but we have a different order for words which have different semantic roles in text and in hypothesis. The solution for this problem is to use a special tool that identifies semantic roles for words and to insert new rules for cases in which the same word has different roles in text and in hypothesis.

4. Improvements

In order to improve the computation speed, the system architecture adopted the peer-to-peer model, and the system behaviour is similar to that of a computational Grid. After the peer-to-peer network is configured, one computer becomes the *Initiator* and builds the list of available neighbours. Subsequently, the *Initiator* performs some additional tasks such as splitting the initial problem into sub-problems, sending the sub-problems to the list of neighbours for solving, receiving the partial output files and building the final solution. The system functionality was optimized by using a caching mechanism and a quota for synchronizing the end of all processes. The testing network was configured with 7 processes running on 5 computers (3 of them single core and 2 with dual core processor) and a run of the system on 800 pairs from RTE-3 test data takes only 6.7 seconds.

In the last years, computational Grids (Fox and Gannon, 2001), (Gannon and Grimshaw, 1998) have become an important area in large-scale scientific and engineering research. The computational Grids offer a set of services that allow a widely distributed collection of resources to be tied together into a relatively seamless computing framework, teams of researchers being able to collaborate to solve problems that they could not have attempted before. Unfortunately, after years of experience in this area, the task of building Grid applications still remains extremely difficult, mainly because there are only a few tools available to support developers.

Peer-to-peer (P2P) networks are recognized as one of the most used architectures in order to achieve scalability in key components of the Grid systems. One of these components is resource discovery, whose duty is to provide system-wide up-to-date information about resources, a task inherently limited in scalability.

Our aim in this network configuration was to use the computational power of the Grid, especially when we work with large databases like DIRT, WordNet, VerbOcean, etc. In these cases, we created Grid services in order to access and to faster identify the requested information.

We will see how we implemented a computational Grid using the P2P model, and how we used SMB protocol for file transfer. After that we will see how we used this computational Grid, in order to improve the computational speed of a system used in the RTE-3 competition.

One on the most important problems remaining is related to system configuration. For every computer from our network we must configure a local cache zone with information about the network, linguistic databases and tools, and Grid services. Initially, this configuration was

done manually for every computer from our network. At configuration part we lose minutes for a network with around 10 computers, while a single run took only a few seconds. From this reason we implement discovery GRID services using PDP (Peer Discovery Protocol) protocol from JXTA[40] (short for Juxtapose) project.

4.1. *Peer-to-Peer Networks*

The popularity of distributed file systems has continued to grow significantly in the last years, due to the success of peer-to-peer services like file sharing applications, VoIP applications, Instant messaging and TV on the Internet. Peer-to-peer systems offer a decentralized, self-sustained, scalable, fault tolerant and symmetric network of machines providing an effective balance of storage and bandwidth resources.

The term Peer-to-Peer (P2P) refers to a class of systems (hardware and software) which employ distributed resources to perform a function in a decentralized manner. Each node of such a system has the same responsibility.

Basic goals are decentralization, immediate connectivity, reduced cost of ownership and anonymity. P2P systems are defined in (Androutsellis-Theotokis and Spinellis, 2002) as "*applications that take advantages of resources (storage, cycles, content, human presence) available at the edges of the Internet*". The P2P architecture can help reduce storage system costs and allow cost sharing by using the existing infrastructure and resources of several sites. Considering these factors, the P2P systems could be very useful in designing the future generation of distributed file systems.

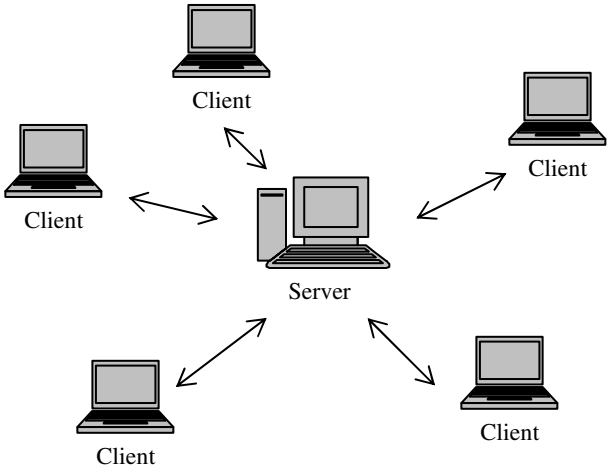


Figure 12: Client/Server architecture

[40] JXTA Project: https://jxta.dev.java.net/

Peer-to-Peer technologies enable any network-aware device to provide services to another network-aware device. A device in a P2P network can provide access to any type of resource that it has at its disposal, whether documents, storage capacity, computing power, or even its own human operator. Most Internet services are distributed using the traditional client/server architecture (see Figure 12).

In this architecture, clients connect to a server using a specific communication protocol, such as the File Transfer Protocol (FTP), to obtain access to a specific resource. Most of the processing involved in delivering a service, usually occurs on the server, leaving the client relatively unburdened. The most popular Internet applications, including the World Wide Web, FTP, telnet, and email, use this service-delivery model.

Unfortunately, this architecture has a major drawback. As the number of clients increases, the load and bandwidth demands on the server also increase, eventually preventing the server from handling additional clients. The advantage of this architecture is that it requires less computational power on the client side.

The client in the client/server architecture acts in a passive role, capable of demanding services from servers but incapable of providing services to other clients. This model of service delivery was developed at a time when most machines on the Internet had a resolvable static IP address, meaning that all machines on the Internet could find each other easily using a simple name. If all machines on the network ran both a server and a client, they formed the foundation of a rudimentary P2P network (see Figure 13).

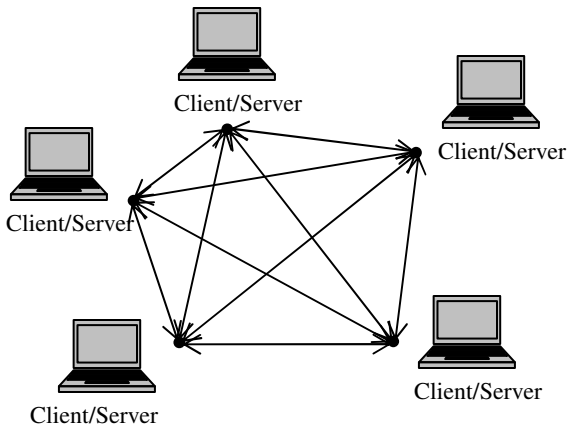


Figure 13: Peer-to-peer architecture

The main advantage of P2P networks is that they distribute the responsibility of providing services among all peers on the network; this eliminates service outages due to a single point of

failure and provides a more scalable solution for offering services. In addition, P2P networks exploit available bandwidth across the entire network by using a variety of communication channels and by filling bandwidth to the "edge" of the Internet. Unlike traditional client/server communication, in which specific routes to popular destinations can become overtaxed, P2P enables communication via a variety of network routes, thereby reducing network congestion.

P2P has the capability of serving resources with high availability at a much lower cost while maximizing the use of resources from every peer connected to the P2P network. While client/server solutions rely on the addition of costly bandwidth, equipment, and co-location facilities to maintain a robust solution, P2P can offer a similar level of robustness by spreading network and resource demands across the P2P network.

4.1.1. Design Issues in P2P Networks

Peer-to-Peer systems have basic properties that separate them from conventional distributed systems. We will describe in this section different design issues and their effect on the system's performance (Ragib and Zahid, 2005).

Symmetry: Symmetry is a property of the participant nodes. We assume that all nodes have the same characteristics. In client-server systems, usually the servers are more powerful than the clients. In our network each node has the ability to be a server and a client at the same time.

Decentralization: P2P systems are decentralized by nature, and they have mechanisms for distributed storage, processing, information sharing. This way, scalability, resilience to faults and availability are increased. But in the same time, our attempts to see the global system and his behaviour are without success.

Operations with Volunteer Participants: An important design issue is that the participants can neither be expected nor enforced. They don't have a manager or a centralized authority and can be inserted or removed from the system at any time. The system should be robust enough to survive the removal or failure of nodes at any moment.

Fast Resource Locating: One of the most important P2P design is the method used for resource locating. Since resources are distributed in diverse peers, an efficient mechanism for object location becomes the deciding factor in the performance of such a system. Currently, object location and routing systems include Chord (Stoica et al., 2001), Pastry (Rowstron and Druschel, 2001), Tapestry (Zhao et al., 2001) and CAN (Ratnasamy et al., 2000). Pastry and Tapestry use Plaxton trees, basing their routing on address prefixes that are a generalization of hypercube

routing. However, Pastry and Tapestry add robustness, dynamism and self-organizing properties to the Plaxton scheme. Chord uses the numerical difference with the destination address to route messages. The Content Addressable Network (CAN) uses a d-dimensional space to route messages. Another important location strategy used in some systems is Distributed Hash Table (DHT). It uses hashing of files or resource names to locate the object.

Load Balancing: Load balancing is very important in a robust P2P system. The system must have optimal distribution of resources based on capability and availability of node resources. The system should prevent the building of locations where the load is high and it should also be possible to re-balance the system based on usage patterns.

Anonymity: With the goal of preventing censorship we need to have anonymity in our system. Our network must ensure anonymity for the producer and for the consumer of information.

Scalability: The number of peers should not affect the network behaviour. Unfortunately, actual systems are not scalable over some hundreds or thousands of nodes.

Persistence of Information: The methods from our system should be able to provide persistent data to consumers; the data stored in the system is safe, protected against destruction, and highly available in a transparent manner.

Security: Security from attacks and system failure are design goals for every system. The system should be able to ensure data security, and this can be achieved with encryption, different coding schemes, etc.

4.1.2. CAN

In order to implement a P2P architecture that respects all these issues, we use the Content Addressable Network (CAN) presented in (Ratnasamy et al., 2000).

The term "CAN" stands for *Content Addressable Network*, which is used as a distributed hash table. The basic operations performed on a CAN are the insertion, lookup and deletion of nodes. Our implementation is based on (Iftene and Croitoru, 2006) and it provides a completely *distributed system* (we have no centralized control, coordination or administration), *scalable* (every node has a small number of neighbours that is independent of the total numbers of nodes of the system), and *fault-tolerant* (we support node failures).

4.1.2.1. CAN Construction

Our design is over a virtual *d-dimensional* Cartesian coordinate space. This space is completely logical and we don't have any relation between it and any physical system. In this *d-dimensional*

coordinate space, two nodes are neighbours if their coordinates are the same along *d-1* dimensions and are different over one dimension. In this way, if all areas from this space are approximately the same, every node has *2d* neighbours. To allow the CAN to grow, a new node that joins to this space must receive its own portion of the coordinate space. Below, we can see the movement between two nodes following the straight-line path through the Cartesian space.

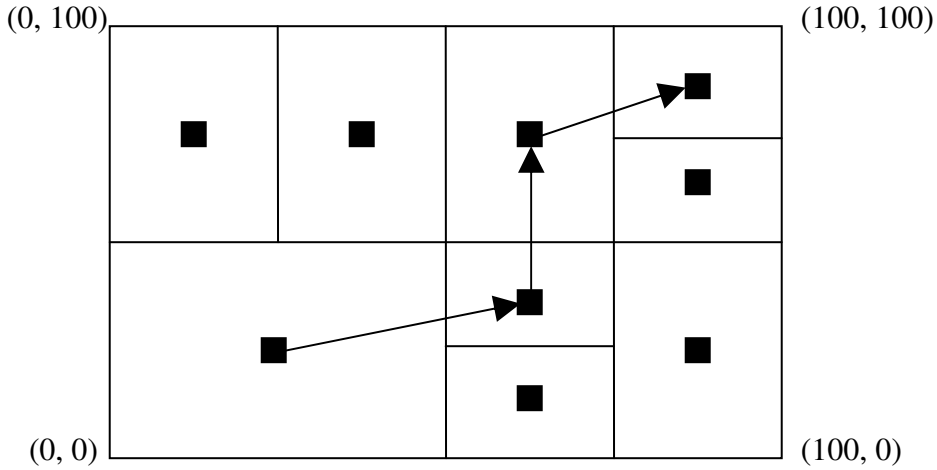


Figure 14: CAN movement

4.1.2.2. CAN Insertion

The process has three steps:

1. The new node must find a node that is already in CAN (*source node*).
2. After that, it knows CAN dimensions and it generates a *d-point* in this space (this point is in a node zone - *destination node*). We route from the *source node* to the *destination node* following the straight-line path through the Cartesian space. The *destination node* then splits its zone in half and assigns one half to the new node.
3. In the last step, the neighbours of the split zone must be notified about the new node of the CAN.

4.1.2.3. CAN Deletion

When nodes leave our CAN, we need to ensure that the remaining neighbours take their zones. If the zone of a neighbour can be merged with the node's zone to produce a valid single zone, then this is done. If not, then the zone is split in areas according with the neighbourhood's structure. In some cases, it is possible that this zone remains unallocated, but it is used with the first occasion.

4.1.3. The System

Solving complex problems has become a usual fact in the Natural Language Processing domain, where it is normal to use large information databases like lexicons, annotated corpora and dictionaries. We will see how we can configure the peer-to-peer system in order to find the global solution for the RTE3 competition task.

The system P2P architecture is based on the CAN model (Iftene, Balahur-Dobrescu, Matei, 2007). The system presented below consists of more core modules (CMs) and databases of linguistic resources. In order to solve a task belonging to the RTE competition we connect to a computer of this computational Grid and initiate the TE system.

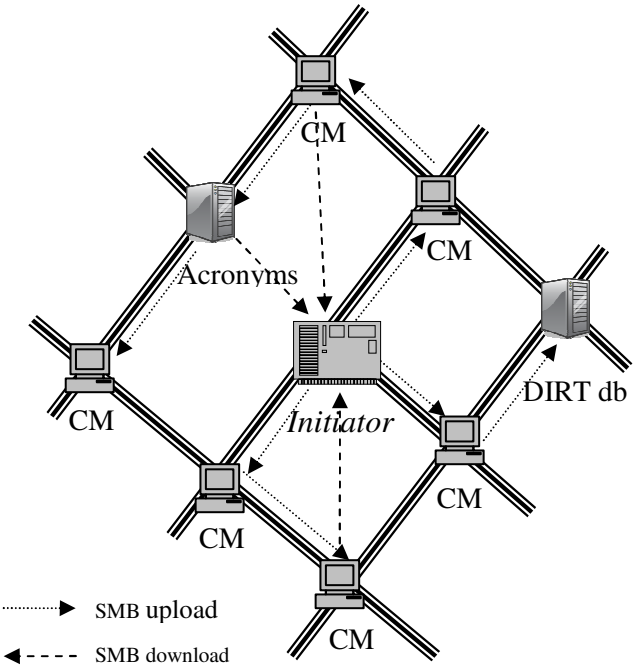


Figure 15: P2P Network

Between CMs, the upload and download operations are done using a special component based on the SMB protocol.

Any computer from this network can initiate the solving of the TE task in which case it becomes the *Initiator*. First of all it checks its list of neighbours in order to know the number of computers which can be involved in the problem solving (the future CMs). After that it searches the latest version of the TE module and updates with it all CMs that have an older version. In parallel, all text and hypothesis pairs are sent to the LingPipe and Minipar modules, which send back the pairs on which the central module can run the TE module. All files are uploaded from the initiator to the other computers and eventually, the results from the other computers are

downloaded to the initiator.

After that, the initial problem (consisting of 800 pairs in RTE-3 or consisting of 1000 pairs in RTE-4) was split in sub-problems (a range between the number of the first and last pair) according to the number of neighbours and using a dynamic quota. At first, this quota has an initial value, but in time it is decreased and eventually becomes the default minimal value. The main goal of using this quota is to send to any neighbour a number of problems according to its computational power and to how busy it is at the moment of run. Our supposition was that the computers, accordingly with their computational power or with their degree of business in the run moment, will solve more or less problems. In order to accomplish this, each computer is given an initial number of problems to solve and when it finishes, it receives another quota according to the remaining number of problems. The code unit for quota updating is:

```
while (quota * NeighboursNumber > RemainingProblems/2 and  quota > 2)
          quota = quota / 2;
```

Table 34: Quota calculation

The execution is performed in a network with different system configurations of computers (dual core - running two processes and single core - running one process) and with a different degree of workload at the execution moment. For that we run Disk Defragmenter on some computers (those marked with * in the next table) to see how the initiator distributes the sub-problems to its neighbours.

quota	**100**	**100**	**100**	**20**	**20**	**20**
Dual1_1	165	163	176	171	178	209
Dual1_2	197	136*	179	145*	159*	221
Dual2_1	175	175	155*	185	138*	108*
Dual2_2	152	183	163	180	129*	130*
Normal	111	143	127	119	196	132

Table 35: Number of problems solved by computers

The table above demonstrates the correctness of our supposition regarding the quota mechanism. In all the cases marked with *, the processor solves fewer problems.

After splitting the initial problem in sub-problems, the “Client Module” belonging to the initiator creates threads for each sub-problem. It also creates a sub-client for every neighbour in

order to communicate with the server components from that neighbour and to solve the sub-problem (see figure 16).

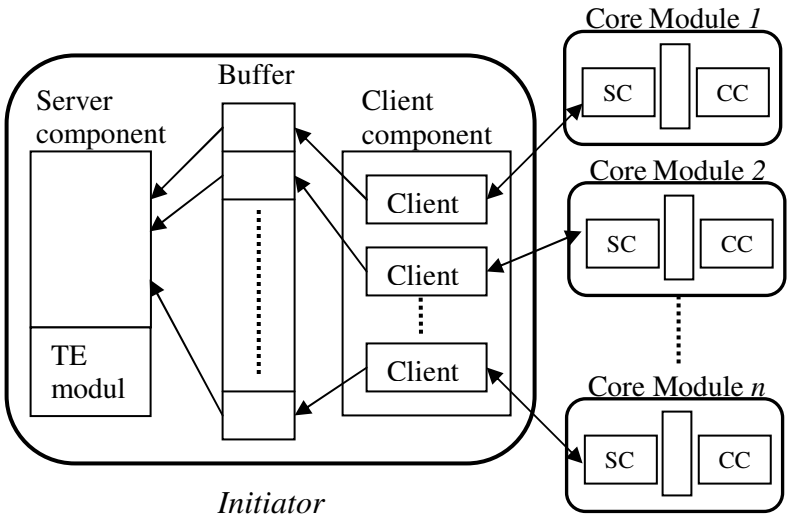


Figure 16: Client Server architecture

This sub-client is responsible for sending sub-problem, for receiving the results and for saving them to a file with partial results, and also for informing the initiator server when the solving of sub-problem is finished. In order to inform the server about the completion of the processing task, the client module uses the buffer zone. The client uses the buffer as a "producer", putting here information about termination of processes, and the server uses the buffer as a "consumer" (Andrews, 1999). The server takes the information and sees how the sub-problem is solved on the corresponding computer and starts solving a new sub-problem when the previous one is finished.

4.1.4. Results

The idea to use an optimized P2P architecture came after the first run of the first system built for RTE3 (Iftene and Balahur-Dobrescu, 2007a). This first run on all RTE-3 test data had no optimization and took around 26 hours on one computer. In order to notice the system changes faster, we improved this system and during the RTE-3 competition period we used a system which took around 8 hours for a full run.

This system was installed on four computers and after manual splitting of the problems in sub-problems, a single run took only two hours. After all systems finished their runs we manually assembled the final solution out of the four partial solutions.

In order to further increase the computational speeds in the elaboration of current solution,

a common caching mechanism is used for the large databases of Dirt and WordNet. To build the caching resource, the system was run on one computer. After that, the computational Grid was configured on a P2P network with 5 computers (3 with single core and 2 with dual core processors) and a run of the system on all RTE-3 test data took only 6.7 seconds now (see Table 36).

No	Run details	Duration
1	One computer without the caching mechanism	5:28:45
2	One computer with the caching mechanism, with empty cache at start	2:03:13
3	One computer with full cache at start	0:00:41
4	5 computers with 7 processes	0:00:06.7

Table 36: Details on the duration of runs

This approach seems the most appropriate given the complexity of the task and the fact that the impact of any change brought to the system must be quantified quickly (the competition has a duration of 3 days in RTE-3 and 6 days in RTE-4). To our knowledge, there were no similar efforts to optimize TE systems from point of view of the computational speed (presently, to our knowledge, there are systems who run take days to complete).

Another big problem was synchronizing the termination of processes. When we manually split the initial problem in sub-problems, if another process (like Windows update or a scanning computer after viruses) starts on a computer, then this computer works slowly. In this case, we have up to 30 minutes delay due to the fact that we have to wait for it to finish solving its sub-problems to be afterwards assembled into the final solution.

Now, using the quota mechanism for synchronizing the ending of all processes, the difference between the time when the first computer from the network finishes its work and the last computer finishes, is around 0.26 seconds. Also, the powerful computers from the network have time to finish more problems while the slow ones solve less.

4.2. *GRID Systems*

To achieve their envisioned global-scale deployment, Grid systems need to be scalable. Peer-to-peer (P2P) techniques are widely viewed as one of the prominent ways to reach the desired scalability. Resource discovery is one of the most important functionalities of a Grid system and,

at the same time, one of the most difficult to scale. Indeed, the duty of a resource discovery system (such as the Globus MDS[41]) is to provide system-wide up-to-date information, a task which inherently has limited scalability. To add to the challenge, Grid resource discovery systems need to manage not only static resources, but also resources whose characteristics change dynamically over time, making the design critical.

4.2.1. Computational GRID

In distributed computing the syntagm *grid computing* has several meanings[42]:

- A local computer cluster composed of multiple nodes.
- A local computer cluster which can offer online computation or storage as a metered commercial service, known as utility computing, computing on demand, or cloud computing.
- A local computer cluster which can permit the creation of a "virtual supercomputer" by using spare computing resources within an organization.
- A local computer cluster which can create a "virtual supercomputer" by using a network of geographically dispersed computers. Volunteer computing, which generally focuses on scientific, mathematical, and academic problems, is the most common application of this technology.

These varying definitions cover the spectrum of "distributed computing", and sometimes the two terms are used as synonyms.

Functionally, one can also speak of several types of grids:

- *Computational grids* (including CPU Scavenging Grids) which focus primarily on computationally-intensive operations.
- *Data grids* or the controlled sharing and management of large amounts of distributed data.
- *Equipment grids* which have a primary piece of equipment, e.g. a telescope, and where the surrounding Grid is used to control the equipment remotely and to analyze the data produced.

Usually, a *computational Grid* consists of a set of resources, such as computers, networks, on-line instruments, data servers or sensors that are tied together by a set of common services which allow the users of the resources to view the collection as a seamless computing-

[41] Globus MDS: http://www.globus.org/toolkit/mds
[42] Grid Computing: http://en.wikipedia.org/wiki/Grid_computing

information environment. The standard Grid services include (Gannon et al., 2002):

- security services which support user authentication, authorization and privacy,
- information services, which allow users to see what resources (machines, software, other services) are available for use,
- job submission services, which allow a user to submit a job to any computation resource that the user is authorized to use,
- co-scheduling services, which allow multiple resources to be scheduled concurrently,
- user support services, which provide users access to "trouble ticket" systems that span the resources of an entire grid.

Our P2P system can be considered a computational Grid focused on complex linguistic operations. All standard services were implemented except the security services which are supported by our operating system.

422. Peer to Peer Grid Concepts

P2P systems are divided into two categories in (Gannon et al., 2002):

1. *File sharing utilities* as we can see in FreeNet (Clarke, 2000), which can be characterized as providing a global namespace and file caching and a directory service for sharing files in a wide-area distributed environment. In the most interesting cases, the resources and all services are completely distributed. Each user client program, which can access local files, is also a data server for files local to that host.
2. *CPU cycle sharing* of unused user resources usually managed by a central system, which distributes work in small pieces to contributing clients. Examples of this include Seti-athome[43], Entropia[44] and Parabon[45].

Both of these cases are interesting distributed system architectures. From the perspective of Grid computing there are several compelling features to these systems. First, the deployment model of P2P systems is purely user-space based. They require no system administrator.

Security, when it exists in these systems, is based on users deciding how much of their resource they wish to make public to the community or to the central resource manager. Also P2P systems are designed to be very dynamic, with peers coming and going from the collective constantly as users machines go on and off the network. This stands in contrast to large-scale

[43] Seti@home: http://setiathome.berkeley.edu/
[44] Entropia Distributed Computing: http://www.entropia.com
[45] Parabon Computation: http://www.parabon.com

scientific Grid systems, which manage large expensive resources and must be constantly maintained and managed by the system administration staff.

4.2.3. GRID Services

GRID services, beside classical computational systems, come with two new characteristics:

- The *computational power* is very high and it is highly extensible relative to adding of new nodes;
- The possibility of sharing *information on a large scale* exists and we have the possibility of using it for complex problem solving.

Our implementation uses the Globus toolkit and is related to implementation of three types of Grid services corresponding to the P2P system presented in chapter 4.1 (see Appendix 6 for GRID services implementation details) (Pistol and Iftene, 2008). The first type of Grid services represent the basic services and are responsible of accessing the basic resources like DIRT, WordNet, VerbOcean and Wikipedia. The second type of services is the complex services, which is based on basic services and implements specific algorithms. The third type of Grid service is related to the system configuration and it is responsible with service discovery through the network. In the following we will describe the main characteristics of these Grid services.

4.2.3.1. Basic GRID Services

Basic Grid services use data collections memorized as text files. These text files are available in English and Romanian language:

- The **Lemmatizing Service** is used in order to identify the word lemma. It receives as input one word and after searching it in the database with all flexional forms, it returns the word lemma (success) or the initial word (fail).
- The **Synonym Service** has the aim of identifying the synonyms for one word. It receives as input one word (noun or adjective) and after searching in WordNet and in eXtended WordNet resources, it returns a list of synonyms for the initial word. If it fails, it returns the empty list.
- The **Antonymy Service** has the aim of identifying the antonyms for one verb. It receives a verb as input and after searching in the WordNet and in the VerbOcean resources, it returns a list with antonyms for the initial word. If it fails, it returns the empty list.
- The **DIRT Service** is used for identification of similarity verbs from the DIRT database. This service receives a verb as input and it returns a list of verbs from DIRT which is in

the similarity relation with the initial verb. If it fails, it will return an empty list.

- The **Acronym Service** is used for identification of acronym meanings. For an input word it returns from the database of acronyms the meaning of the word, when it exists. When the input word does not exist in the acronym database it returns *null*.
- The **Background Knowledge Service** is used to extract useful information from Wikipedia related to an initial named entity. Thus, starting from an initial named entity, the service extracts other named entities from Wikipedia and, accordingly, with named entity types this service sets the relation between them.

4232. Complex GRID Services

Complex Grid Services use basic Grid services and implement specific algorithms for identification of entailment relation between text and hypothesis. These algorithms are language independent and use the output of basic services. Complex Grid services are:

- **Hypothesis Transformation Service** receives a sentence as input and returns an array of arrays, built by words expanding. The expanding process assumed identification of the lemma and part-of-speech for all words, the synonyms for nouns and adjectives, similarity forms and antonyms for verbs, equivalent forms for named entities, and the meaning for acronyms. Also, the arrays contain the dependency tree generated by MINIPAR.
- **Text Transformation Service** receives a sentence as input and returns an array of arrays, built by words expanding. In this case, the expanding process assumes only the lemma, part-of-speech and dependency tree identification.
- **Calculation of Global Fitness Service** receives as input two sets of arrays corresponding to the text and hypothesis from the above services. The output is a value between -1 and 1 and it represents the global fitness value for the initial text and the hypothesis pair.

4233. Discovery Services

The discovery service was implemented using the PDP (Peer Discovery Protocol) protocol from JXTA project (Iftene, 2008a). We will see the main characteristics of this project and the solution adopted by us in order to discover services through the P2P network.

4.2.3.3.1. JXTA Basic Components

The JXTA[46] project is an open source project from 2001 when the creator Bill Joy puts it in the hands of the P2P development community. Currently, Project JXTA has a reference implementation available in Java, with implementations in C, Objective-C, Ruby, and Perl 5.0 under way. The basic components of a JXTA P2P network are *peers*. The information transfer between peers in this P2P network is done using *network transport* components. The functionality of peers is done through the *services*. Their main characteristics are:

Peers

A *peer* is a node on a P2P network that forms the fundamental processing unit of any P2P solution. Definition from (Brendon, 2002) is: "*Any entity capable of performing some useful work and communicating the results of that work to another entity over a network, either directly or indirectly*."

The definition of *useful work* depends on the type of peer. Three possible types of peers exist in any P2P network:

- **Simple peers** are designed to serve a single end user, allowing that user to provide services from his device and consuming services provided by other peers on the network.
- **Rendezvous peers** provide peers with a network location to use to discover other peers and peers resources. A rendezvous peer can augment its capabilities by caching information on peers for future use or by forwarding discovery requests to other rendezvous peers.
- **Router peers** provide mechanisms for peers to communicate with other peers separated from the network by firewall or Network Address Translation (NAT) equipment.

Network Transport

To exchange data, peers must employ some type of mechanism to handle the transmission of data over the network. This layer, called the *network transport*, is responsible for all aspects of data transmission, including breaking the data into manageable packets, adding appropriate headers to a packet to control its destination, and in some cases, ensuring that a packet arrives at its destination.

The concept of a network transport in P2P can be broken into three constituent parts:

- **Endpoints** – The initial source or final destination of any piece of data being transmitted

[46] JXTA Project: https://jxta.dev.java.net/

over the network.

- **Pipes** – Unidirectional, asynchronous, virtual communication channels connecting two or more endpoints.
- **Messages** – Containers for data being transmitted over a pipe from one endpoint to another.

To communicate using a pipe, a peer first needs to find the endpoints, one for the source of the message and one for each destination of the message, and connect them by binding a pipe to each of the endpoints.

Services

Services provide functionality based on which peers can engage in performing "useful work" on a remote peer. This work might include transferring a file, providing status information, performing a calculation, or basically doing anything that you might want a peer in a P2P network to be capable of doing. Services can be divided into two categories: *Peer services* (when functionality is offered by a particular peer) and *Peer group services* (when functionality is offered by a peer group to members of the peer group).

4.2.3.3.2. P2P Communication in JXTA

The fundamental problem in P2P is how to enable the exchange of services between networked devices. Solving this problem first requires finding answers to the question: "*How does a device find peers and services on a P2P network*?" This question is important because, without the knowledge of the existence of a peer or a service on the network, there's no possibility for a device to engage that service.

Finding Advertisements

Any of the basic building blocks discussed in the last section can be represented as an advertisement, and that characteristic considerably simplifies the problem of finding peers, peer groups, services, pipes, and endpoints. Instead of worrying about the specific case, such as finding a peer, we need to consider only the general problem of finding advertisements on the network. A peer can discover an advertisement in three ways:

- **No discovery** – Instead of actively searching for advertisements on the network, a peer can rely on a cache of previously discovered advertisements to provide information on

peer resources.

- **Direct discovery** – Peers that exist on the same LAN might be capable of discovering each other directly without relying on an intermediate rendezvous peer to aid the discovery process. Direct discovery requires peers to use the broadcast or multicasting abilities of their native network transport.
- **Indirect discovery** – Indirect discovery requires using a rendezvous peer to act as a source of known peers and advertisements, and to perform discovery on a peer's behalf.

Rendezvous peers provide peers with two possible ways of locating peers and other advertisements:

- **Propagation** – A rendezvous peer passes the discovery request to other peers on the network that it knows about, including other rendezvous peers that also propagate the request to other peers.
- **Cached advertisements** – In the same manner that simple peers can use cached advertisements to reduce network traffic, a rendezvous can use cached advertisements to respond to a peer's discovery queries.

Discovering Rendezvous and Routing Peers

For most peers existing on a private internal network, finding rendezvous and router peers is critical to participating in the P2P network. In most P2P applications, the easiest way to ensure that a simple peer can find rendezvous and router peers is to seed the peer with a hard-coded set of rendezvous and router peers. These rendezvous and router peers usually exist at static, resolvable IP addresses and are used by a peer as an entrance point to the P2P network. A peer located behind a firewall can use these static rendezvous peers as a starting point for discovering other peers and services and can connect to other peers using the static set of router peers to traverse firewalls.

4.2.3.3.3. JXTA Peer Discovery Protocol

Every data exchange relies on a protocol to dictate what data gets sent and in what order it gets sent. A *protocol* is simply this (Brendon, 2002): "*A way of structuring the exchange of information between two or more parties using rules that have previously been agreed upon by all parties.*"

In JXTA, the team designed a set of six protocols based on XML messages: *Peer*

discovery protocol, *Peer information protocol*, *Pipe binding protocol*, *Peer resolver protocol*, *Rendezvous protocol*, *Peer endpoint protocol*.

The Peer Discovery Protocol (PDP) defines a protocol for requesting advertisements from other peers and responding to other peers' requests for advertisements. We already saw that peers discover resources by sending a request to another peer, usually a rendezvous peer, and receiving responses containing advertisements describing the available resources on the P2P network. The Peer Discovery Protocol consists of only two messages that define the following:

- A *request format* used to discover advertisements,
- A *response format* for responding to a discovery request.

These two message formats, the *Discovery Query Message* and the *Discovery Response Message*, define all the elements required to perform a discovery transaction between two peers.

The Discovery Query Message

The Discovery Query Message is sent to other peers to find advertisements. It has a simple format, as shown in Figure 17.

```
<?xml version="1.0" encoding="UTF-8"?>
<jxta:DiscoveryQuery>
    <Type> . . . </Type>
    <Threshold> . . . </Threshold>
    <PeerAdv> . . .</PeerAdv>
    <Attr> . . . </Attr>
    <Value> . . .</Value>
</jxta:DiscoveryQuery>
```

Figure 17: The Discovery Query Message XML

The elements of the Discovery Query Message describe the discovery parameters for the query. Only advertisements that match all the requirements described by the query's discovery parameters are returned by a peer.

The Discovery Response Message

To reply to a Discovery Query Message, a peer creates a Discovery Response Message that contains advertisements that match the query's search criteria, such as the Attribute/Value combination or Type of advertisement. The Discovery Response Message is formatted as shown in Figure 18.

```
<?xml version="1.0" encoding="UTF-8"?>
<jxta:DiscoveryResponse>
    <Type> . . . </Type>
    <Count> . . . </Count>
```

```
    <PeerAdv> . . . </PeerAdv>
    <Attr> . . . </Attr>
    <Value> . . . </Value>
    <Response Expiration="expiration time">  . . .   </Response>
</jxta:DiscoveryResponse>
```

Figure 18: The Discovery Response Message XML

The elements of the Discovery Response Message closely correspond to those of the Discovery Query Message.

4.2.3.3.4. Our Implementation

An important step in building a computational Grid based on the P2P architecture was the initial system configuration (Iftene, 2008a). At this step, for every computer from this network the following preparation steps have been performed:

- We add in the first zone of the computer cache the "**neighbours**", in the CAN terminology (IPs of computers from immediate proximity). The computer node will do a static discovery of network through this cache, but in all the following requests this computer will access the entire network only via these computers (indirect discovery).
- We add in the second zone of the computer cache the addresses of the "**NLP tools and resources**" (IPs of computers from entire P2P network). These tools and resources will be used by current CM in order to solve sub-problems assigned by the *Initiator*.
- We add in the third zone of computer cache the addresses of "**GRID services**" (IPs of computers from P2P network or from Internet). These Grid services are used for solving sub-problems or for sending and receiving sub-problems (Pistol and Iftene, 2008).

The big problems of this step come from time consuming during the configuration of all CMs and from problems that appear in failure cases. If one computer from this network is temporarily unavailable and another computer wants to access its tools, resources or services, after some trials without success, the address of the computer with problems will be removed from the computer cache. This leads to the impossibility of using this computer in the future, without a reconfiguration of all neighbours caching.

Our solution comes to solve two drawbacks of the system: identification of addresses for "NLP tools and resources" and identification of addresses for "GRID services". In our initial solution, these parts were done manually and required a couple of minutes for a network with 10 computers. Now, using the PDP protocol, this entire configuration is done automatically.

In order to do this configuration automatically, three components with specific tasks were

implemented: *Publisher*, *Manager* and *Finder* (see next figure):

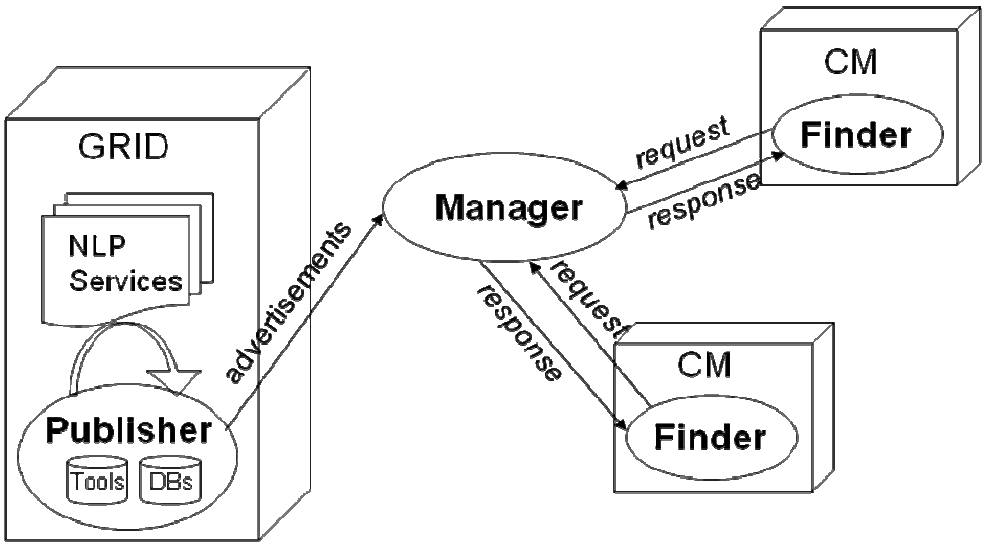


Figure 19: Discovery Services in GRID environment

- The *Publisher* component runs on a GRID server and is able to identify, in a dynamic way, the available GRID services on it. The *Publisher* also has a list of available tools and resources through the network. When it is started, it builds a list with available GRID services and available tools and resources, and it sends advertisements through the network.
- The *Finder* components run on every CMs and search for specific advertisements (about services or resources) in the P2P network.
- The *Manager* component can run anywhere in the network (on GRID server, on a CM, on a computer with NLP resources or tools or even on another computer) and facilitate communication between *Publisher* and *Finder*.

In the current work the JXTA Shell[47] was used and through this shell the PDP protocol for cache building was used. The JXTA Shell is a demo application built on top of the JXTA platform that allows users to experiment with the functionality made available through the Java reference implementation of the JXTA protocols. The JXTA Shell provides a UNIX-like command-line interface that allows the user to perform P2P operations by manipulating peers, peer groups, and pipes using simple commands.

[47] JXTA Shell: http://download.java.net/jxta/jxta-jxse/2.5/jnlp/shell.jnlp

4.3. *Transfer Protocol*

The transfer protocol used in our approach is based on CIFS[48] (Common Internet File System), the Microsoft version of SMB[49] (Server Message Block). The protocol is based on request and response messages called SMBs, as presented in the following figure.

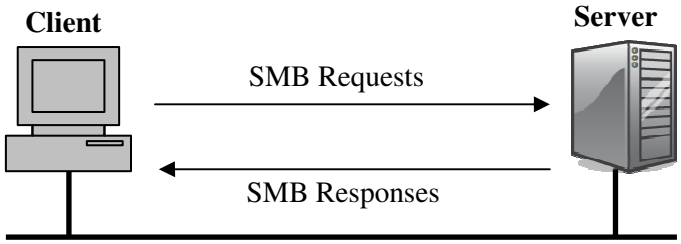


Figure 20: CIFS protocol

The main goal of using this transfer protocol is to manage the download and upload of files between nodes from our P2P network. Using our protocol, advantages come from the possibility of retrying in case of failures and from the possibility of using bandwidth partially in order to by-pass network overload. Details about how Cifs works are presented in Appendix 5.

Download and Upload files

Our CIFS implementation respects the following design issues:

1. Possibility to split and transfer files in blocks;
2. Possibility to retry and resume when connection is lost, in case of power failures;
3. Upload/Download files to/from one computer which needs user/password login;
4. Possibility to use the bandwidth partially, in order not to overload the network.

In order to respect all these requests we implemented all CIFS protocol issues and we worked with it at a low level. For that we sent request packets to the remote computer (it is seen as server) using SMB Packet Header and it was sent to our computer (it is seen as client) the response packet, also using the SMB Packet Header.

Download a file

In order to download a file, the following steps are performed in our implementation:

1. Login: First the dialect of CIFS to use is negotiated and the Protocol to use is obtained. Second, the connection was set up to the server using credentials and set the UID assigned

[48] CIFS or Public SMB Information on Common Internet File System: http://support.microsoft.com/kb/199072
[49] Server Message Block: http://en.wikipedia.org/wiki/Server_Message_Block

by the server to the client.

2. TreeConnect - connect to the specified folder and set TID identifier.
3. Open File - open file for read and set FID (*file identifier*).
4. ReadFromFile - read from remote file starting with position offset, length bytes and put the result in buffer.

Upload a file

In order to upload a file, the first two steps described above are performed, followed by:

3'. Open File - create file or open file for write, append and set FID (flags should be FILE_OPEN_IF).

4'. WriteToFile - write to remote file starting with position offset, length bytes from buffer.

With this protocol we transfer the new versions of our core module (CM) between P2P peers and the new versions of resources on computers where we have basic GRID services.

4.4. *Conclusions*

In this chapter was presented the solution adopted in order to improve the computational speed of the TE system. The architecture used in the RTE competitions from 2007 and 2008 was based on a P2P network and has implemented a caching mechanism for large resources and a quota mechanism for end synchronization. For file transfer between peers we implemented our SMB protocol. The tests performed on a P2P network with 7 processes shown good improvements in speed.

Another direction adopted was related to the implementation of Grid services, useful in building of real-time systems and in configuration of P2P networks. We implemented three types of services: basic and complex Grid services for NLP resources and tools, and discovery Grid service for network configuration.

5. Applications of Textual Entailment

This chapter presents the use of the Textual Entailment system in order to improve the quality of a Question Answering system. The system was successfully used in the 2007 and 2008 editions of QA@CLEF competition, in two tasks from the Multiple Language Question Answering track: the Main task and the Answer Validation Exercise.

5.1. *Question Answering*

In '80, one of the first Question Answering systems (IURES) was designed as an interface to the knowledge which was represented as a semantic network (Tufiş and Cristea, 1985).

Nowadays, Question Answering (QA) Systems are one of the main research topics in the Natural Language Processing field. These systems not only employ various discourse analysis and processing tools, but also require theoretical studies and formalizations of many language issues, such as 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 heavily relies on two things: (1) identification of the required information and (2) the quantity and quality of the available information, thus the characteristics of the corpus in which the information can potentially be found.

One of the competitions for QA systems is organized within CLEF (Cross-Language Evaluation Forum). CLEF[50] supports the development of applications for digital libraries by creating an infrastructure for testing, tuning and evaluation of Information Retrieval systems operating in European languages, both in monolingual and cross-lingual contexts. Within the QA@CLEF[51] evaluation track, we have been participating since 2006 with a Romanian-English multilingual system. In editions from 2007 and 2008, we participated in the monolingual task. Having been confronted with the problem of semantic variability, we have decided to include an English TE module in the QA system.

5.1.1. Introduction

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

[50] CLEF: http://www.clef-campaign.org/
[51] QA@CLEF: http://clef-qa.itc.it/

has emerged as a result of a monolingual English QA track being introduced at TREC[52].

QA systems normally adhere to the pipeline architecture composed of three main modules: *question analysis*, *paragraph retrieval* and *answer extraction* (Harabagiu and Moldovan, 2003).

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.

According to the answer type, we have the following type of questions (Harabagiu and Moldovan, 2003):

- ***Factoid*** – The question refers to a single answer, as for instance: "*Who discovered the oxygen?*", "*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 summarizing more documents: "*What is a quasar?*" or "*What is a question-answering system?*"

5.1.2. Main Components of a QA System

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

In what follows, we will briefly describe the main components of the systems built by the UAIC-FII team, lead by the author of this thesis, for the QA competitions in 2006 (Puşcaşu et al., 2006), 2007 (Iftene et al., 2008h) and 2008 (Iftene et al., 2008f). The architecture of the system used in the 2008 competition (Iftene et al., 2008f) is presented in Figure 21:

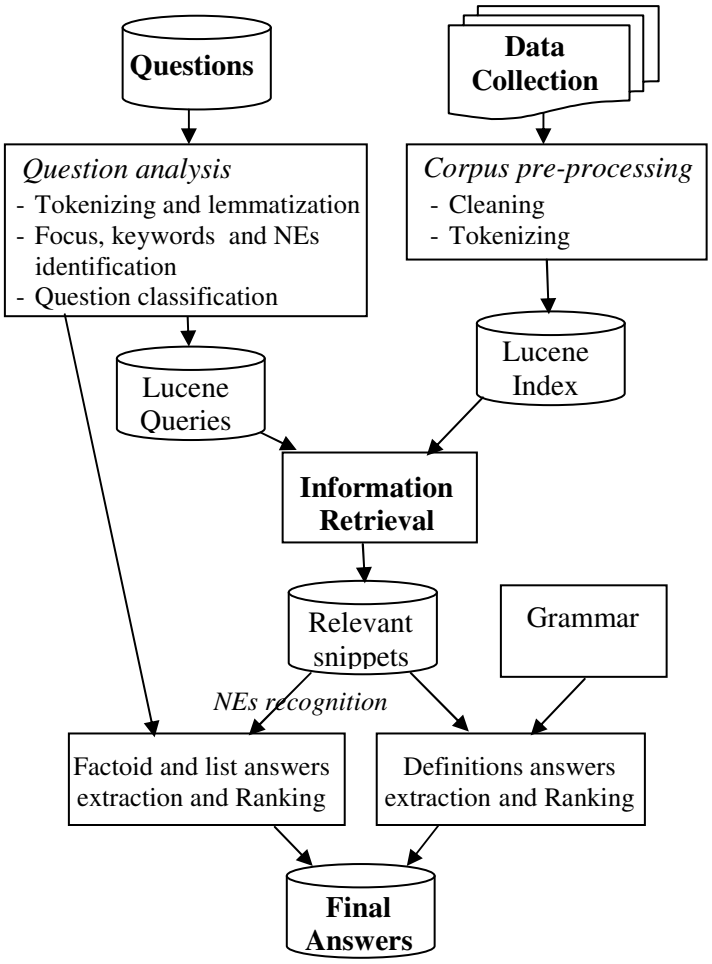


Figure 21: Romanian Question answering system participating at CLEF2008

5.1.2.1. Corpus Pre-processing

This component is responsible for the preparation of the corpus used by the paragraph retrieval module. In the 2007 and 2008 editions, these corpora consisted of frozen version of the Romanian Wikipedia from 2006.

The set of available Wikipedia documents includes topic related information, as well as forum discussions, images and user profiles. In order to transform the corpus into a format more manageable by the available processing tools, two pre-processing steps were performed:

- Documents considered irrelevant for the answer extraction task were removed from the set.

These documents include images, user profiles and forum discussions. The filtering was performed automatically using a pattern based cleaning method.

- In a second step, all remaining documents were converted to a simpler format by stripping off most of the *html* mark-ups. The only mark-ups kept are the name of the article and those indicating paragraphs.

After these two filtering steps, the size of the corpus was reduced significantly. This significant reduction contributed to making the linguistic processing steps more time-efficient.

Both the Wikipedia corpus and the set of test questions went through the same linguistic processing steps prior to the actual analysis: tokenization, POS-tagging, lemmatization and Named Entities Recognition using GATE (Cunningham et al., 2001) for the types Person, Location, Measure, Organization and Date.

5.1.2.2. Question Analysis

This stage is mainly concerned with the identification of the semantic type of the entity sought by the question (*expected answer type*). In addition, it also provides the question focus, the question type and the set of keywords relevant for the question. To achieve these goals, our question analyzer performs the following steps:

i) NP-chunking, Named Entity extraction - Using the pre-processed set of questions as input, on the basis of the morpho-syntactic annotation provided by our tools, a rule-based shallow noun phrase identifier was implemented. The same NE recognizer used during the pre-processing of the corpus provides the set of NEs in the question.

ii) Question focus - The question focus is the word or word sequence that defines or disambiguates the question, in the sense that it pinpoints what the question is searching for or what it is about. The question focus is considered to be either the noun determined by the question stem (as in *What country*) or the head noun of the first question NP if this NP comes before the question's main verb or if it follows the verb "*to be*", as in "*What is Wikipedia*?".

iii) The expected answer type - Our system is able to identify the following types of answers: Person, Location (City, Country, and Region), Organization, Temporal (Date and Year), Measure (Length, Surface and Other), Definition and Other. The assignment of a type of an analyzed question is performed using specific patterns for every type. In the case of ambiguous questions (e.g. *What*), the answer type is computed starting from the question focus. We extracted from

WordNet a set of lists with all hyponyms of the first five[53] answer types and tried to identify the question focus among elements of these lists. For example, in the case of the question *What city is usually identified with the Homeric Troy*?, the question focus is city, noun found in the Location list, therefore the associated expected answer type is Location. Thus, the answer of this question is searched in a database set of city names.

iv) Inferring the question type - Question type can be: *Factoid*, *List* or *Definition*. In order to identify the question type we used three simple rules: if the expected answer type is *Definition*, then the question type is definition; if the question focus is a plural noun, then the question type is *List*, otherwise it is *Factoid*.

v) Keyword generation - The set of keywords is composed of the question focus, the identified NEs in the question, nouns, adjectives, and non-auxiliary verbs belonging to the question.

vi) Anaphora resolution - Every year new features are added in the QA@CLEF competitions. Starting with 2007, a new feature was the grouping of questions into series. All questions belonging to a series address the same general topic, the domain of which is usually specified by either the first question of the series or by its answer. Mainly, questions of a series are tightly coupled by anaphoric links that involve entities mentioned in previous questions or their answers.

This new feature for QA@CLEF competition, grouping questions on topics, requires the addition of a new module in the general architecture of the system, responsible for anaphora resolution. For solving this problem we adopted two methods of anaphora resolution presented below:

1. Antecedent identification

We classified the possible anaphoric relations into the following cases:

- questions with anaphors that refer to a previous question answer;
- questions with anaphors that refer to a previous question focus.

Empirical methods were used to decide if the focus or the answer of the first question in a group is needed.

2. The Backup solution

Since the first method depends on the ability of our system to correctly identify the answer, we

[53] Definition type is computed only by pattern matching methods and Other type is assigned to questions that didn't enter in any other category.

also considered a second method. The second solution was adding all keywords of the first question in the group to the keywords of the other questions.

5.1.2.3. Index Creation and Information Retrieval

The purpose of this module is to retrieve the relevant snippets of text for every question. Below is a brief description of the module:

i) Query creation - Queries are created using the sequences of keywords and Lucene[54] (Hatcher and Gosnopedic, 2004) mandatory operator "+". Thus, we obtain a regular expression for every question, which is then used in the search phase. The queries have also been enhanced by the addition of the question focus and synonyms for all the keywords.

If a word is not preceded by any operator, then that word is optional. The words between the brackets are connected by the logical operator OR, which means that at least one of these elements should be found in a snippet in order for it to be returned by the retrieval module.

ii) Index creation - We have created the index of the document collection using the lemmas determined in the pre-processing phase. We have created two indexes, one at paragraph level and one at document level.

Index creation at the paragraph level

The main purpose of this type of indexing is to identify and retrieve a minimum useful amount of information related to a question. The drawback of this method is that, in some cases, a paragraph is just a single phrase. Of course the advantage is that from a reduced amount of information, we could easier identify and extract the answer from the retrieved paragraph.

Index creation at document level

An alternative indexing method was indexing at the article level. The disadvantage of this method showed up at the moment of answer extraction, when more refined algorithms were necessary.

iii) Relevant paragraph extraction - Using the queries and the index, we extracted, using Lucene, a ranked list of articles / paragraphs for every question.

5.1.2.4. Answer extraction and Ranking

The retrieving process depends on the expected answer type: the answer retrieval module identifies the named entities in every snippet provided by Lucene and matches them to the answer

[54] Lucene: http://lucene.apache.org/

type. When the answer type is not an entity type name, the answer retrieval syntactic patterns are based on the question focus.

The retrieving process depends on the expected answer type: the answer retrieval module identifies the named entities in every snippet provided by Lucene and matches them to the answer type. When the answer type is not an entity type name, the answer retrieval syntactic patterns are based on the question focus.

For identification of the name entities we have used GATE[55], which is able to identify the following types of name entities: JOB, PERSON (Family, Male, and Female), LOCATION (Region, City, Country, River, and Ocean), ORGANIZATION and COMPANY. For the MEASURE and DATE types we start from the patterns built in the competition of 2007, and complete and split them into subtypes. Thus, we split the MEASURE pattern that was a number followed by a measure unit in three patterns: LENGTH, SURFACE and OTHER_MEASURE. For the LENGTH pattern, we consider numbers followed by one of the following unit measures: *kilometre*, *metre*, and *centimetre* singular, plural and short form (km, m and cm). For SURFACE, we consider the same pattern from LENGTH and we put the condition to be followed by one of the surface indicators *2* or ^{2} or *square*. The other unit measures were added in the rest of OTHER_MEASURE pattern (A, kHz, Hz, radian, rad, ohm, grade, Kelvin, K, N, joule, J, watt, W, volt, V, min, mg, Pascal, Pa, etc.). The same split operation was done for the DATE type, where we consider YEAR and FULL_DATE.

This year we built a special module in order to extract answers for DEFINITION questions. This module is based on a Romanian grammar (Iftene, Trandabăţ, Pistol, 2007) built for the LT4eL project[56]. Definitions have been separated into six types in order to reduce the search space and the complexity of grammar rules:

- "**is_def**" – Definitions containing the verb "is".
- "**verb_def**" – Definitions containing specific verbs, different from "is". These verbs are "denote", "show", "state", "represent", "define", "specify", "consist", "name", "permit".
- "**punct_def**" – Definitions which use punctuation signs like "-", "()", "," etc.
- "**layout_def**" – Definitions that can be deduced from the html layout (eg. they can be included in tables).
- "**pron_def**" – Anaphoric definitions, when the defining term is expressed in a previous

[55] GATE: http://gate.ac.uk/
[56] LT4eL: http://www.let.uu.nl/lt4el/

sentence and is only referred in the definition.

- "**other_def**" – Other definitions, which cannot be included in any of the previous categories (eg. "i.e.").

5.1.3. Using TE System in a QA System

The motivation of using the TE system as a module in the general architecture of a QA system is improving the ranking between possible answers for factoid questions in which the expected answer type is Measure, Person, Location, Date and Organization (Iftene, 2008b).

The idea is to select all relevant named entities from the extracted snippets for one question and to replace variables from the patterns associated to the question, with the extracted NEs, in a similar way as in (Bar-Haim et al., 2006). Thus, we will obtain more hypotheses for one text (represented by extracted snippets). For every hypothesis, we calculate the global fitness and then select the named entity for which we have the highest value. Lastly, we compare the highest value for every snippet and select the best global value.

5.1.3.1. Experiments

In the following experiments, we used the file with our submitted results from the Romanian-English cross-lingual task at the QA@CLEF2006[57] competition. For each of the 200 questions given, the system had the possibility to output ten possible answers, among which the ranking was established according to the competitor's own confidence assignment method. Eventually, since the evaluation of answers was done manually (a very difficult task, given the short time at hand), the organizers decided to evaluate only the first answer from the set of given answers for one question.

Our results had a precision of around 9.47%, and we ranked 9*th* among 13 competitors. After the competition, we sought the gold answer for each of the questions in the set of answers provided by our system, and noticed that in over 35% of cases, the gold answer was found among the given answers.

Therefore, we have decided to use a TE module in order to better rank the possible answers and obtain improvements in precision for the questions for which the system returned the right answer in the set of provided answers, but the answer was not ranked first. Thus, we considered 46 questions.

[57] QA@CLEF2006: http://clef-qa.itc.it/2006bis/CLEF-2006.html

In order to use the TE system for ranking, all these questions are first transformed according to the algorithm presented in (Bar-Haim et al., 2006). For example the following question:

Who is John Lennon's widow?

can be transformed in a pattern:

PERSON *is John Lennon's widow.*

where the "PERSON" represent a variable. This variable is the question expected answer type associated to this type of question.

The steps performed by our system are as follows:

- We *build a pattern* with variables for every question according to the question type;
- We *extract all possible candidates* from the snippets extracted for the current question, according to the expected answer type;
- Using the pattern and all possible candidates we *build a set of hypotheses* for every question: H_1, H_2, H_3, etc.;
- We assign to the extracted snippet the role of text *T* and we *run the TE system* for all obtained pairs: (*T,* H_1), (*T,* H_2), (*T,* H_3), etc.;
- Lastly, we *consider the correct answer* for the current question as being the candidate from the hypothesis for which we obtain the greatest score.

Example

In what follows, we will consider question 008 from the question set QA@CLEF2006[58]:

Q: How many passengers does the ship the "*Canadian Empress" carry?*

for which, the gold answer is "66 passengers". Our QA system uses the Lucene[59] tool for corpora indexing and for information retrieval. Using the query generated from keywords identified for this question, Lucene returns two snippets:

Snippet$_1$*: The Empress Canadian (66 p) sails spring through fall on St. Lawrence and Ottawa River cruises to scenic and historic areas; shore excursions are included and most passengers are seniors. (800) 267-7868. Seabourn Cruise Line*

Snippet$_2$*: Call (800) 422-8000 for '94 Queen Sailings, (800) 622-0538 for '95 cruises. St.*

[58] Ro-En test set: http://clef-qa.itc.it/down/qa test06/clefqa06 test ROEN.txt
[59] Lucene: http://lucene.apache.org/

Lawrence Cruise Lines The Empress Canadian (66 p) sails spring through fall on St. Lawrence and Ottawa River cruises to scenic and historic areas; shore excursions are included and most passengers are seniors.

The snippets are ordered using the Lucene score. It is calculated depending on *Lucene query* that is built from identified *keywords* and from *focus* (which represents the most important word of the question that is related to the final answer). For the above question the keywords, the question focus and the Lucene query are:

Keywords: ship Canadian Empress carry

Question Focus: passengers

Lucene Query: +(passengers passenger) ship Canadian Empress carry

The meaning of this query is: search for snippets that mandatory contain one of the words "passengers" or "passenger" (which are the *question focus* and its lemma), and contain optional words "ship", "Canadian", "Empress", and "carry". Both returned snippets contain the same words: "passengers", "Canadian" and "Empress", but the first snippet receives a greater score since it is shorter, as number of words.

The QA system also correctly identifies the answer type, which is MEASURE. For this reason, we extract all numerical expressions from the snippets and the sets with possible answers are:

$Candidates_1$: *66*, *800*, *267-7868*

$Candidates_2$: *800*, *422-8000*, *94*, *800*, *622-0538*, *95*, *66*

Answer ranking using the QA System

The QA system uses two rules to classify answers: first according to the Lucene score and second according to the distance in words from the possible answer to the question focus. Since the focus for this question was the word "passengers", we ranked the candidates according to their distance to this word. For the first snippet, the order is *800*, *267-7868* and *66* and for the second snippet the order is *66*, *95*, *622-0538*, *800*, *94*, *422-8000*, *800*. Since Lucene indicates a greater score for the first snippet, the order of the candidate answers will be: first, the answers from the first snippet will be considered and secondly, the answers from the second snippet. In the end, the answer order was: *800*, *267-7868*, *66*, *66*, *95*, *622-0538*, *800*, *94*, *422-8000*, *800*. We can see that the correct answer appears on the fourth and fifth places.

Answer ranking using the TE System

In this case, the determining factor is the score returned by the TE system. Using the question, we built the following pattern:

Pattern: *MEASURE passengers were carrying by the ship the "Canadian Empress".*

For the first snippet, denoted with T_1, we build three possible hypotheses:

H_{1_1}*: 800 passengers were carried by the ship the "Canadian Empress".*

H_{1_2}*: 267-7868 passengers were carried by the ship the "Canadian Empress".*

H_{1_3}*: 66 passengers were carried by the ship the "Canadian Empress".*

Further, we run the TE system on the pairs (T_1, H_{1_1}), (T_1, H_{1_2}), (T_1, H_{1_3}) and obtain the following global fitness scores: 1.97, 1.99 and 2.01. The results indicate 66 as the best answer. The reasons for which the fitness for the pair (T_1, H_{1_3}) is higher are the following:

- For the first snippet, the MINIPAR tree has three main branches according to the number of sentences from the snippet: on the first main branch is the first sentence from the snippet (*The Empress Canadian (66 p) sails spring through fall on St. Lawrence and Ottawa River cruises to scenic and historic areas; shore excursions are included and most passengers are seniors.*), on the second main branch is the second sentence from the snippet (*(800) 267-7868.*) and on the third main branch is the third sentence (*Seabourn Cruise Line*).
- The distance between the number used in H_{1_3}, *66*, and common words from *H* and *T* (*Empress, Canadian* and *passengers*) is lower (all nodes are on the first main branch of the text tree).
- Since, for H_{1_1} and H_{1_2} the distance from the common words from *H* and *T* (which are the same) to possible answers nodes (*800* and *267-7868*) is bigger (are on separated main branch trees), the inserted penalty is higher and at the end, the global fitness for these pairs is lower.

We perform the same steps for the second snippet, and the results also indicate that value *66* is the best answer, with a global fitness of 2.11. Thus, we choose *66* as final answer.

Certainly, not all cases showed the same good results, but as a general trend, we noticed that the correct answers ascend in the final order of answers. Using the new ranking, the overall precision of the QA system increased with 9.5%, providing 19 new correct answers for the 46 questions targeted.

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.

5.2. *Answer Validation Exercise*

5.2.1. Exercise Description

AVE[60] is a task introduced at QA@CLEF in 2006 (Peñas et al., 2007) with the aim of validating the correctness of the answers given by QA systems. From the beginning, the organizers wanted AVE to improve the quality of QA systems and to check if the answers correspond to the existing supporting texts or not.

Year to year, the evaluation methodology was improved and oriented to the useful factors for QA systems improvement. Thus, in 2007 the AVE systems must select only one VALID answer for every question from a set of possible answers, in contradiction with the edition of 2006, when it was possible to select several VALID answers. In 2008, the organizers tried to fix the problems raised by the possibility to have all answers incorrect. In this case, the solution is to ask for other possible answers from the QA systems, and to try to identify at least one correct answer.

In the following we will present the characteristics of the 2008 edition of 2008 and our system for English monolingual QA validation.

Data Input Format

Similar to previous editions, in 2008 (Rodrigo et al., 2008) the competitors received a set of triplets (*Question*, *Answer*, and *Supporting Text*) and they had to specify the answers correctness. Thus, for every triplet, the competitors had to specify if it is VALIDATED or if it is REJECTED. For English, the input file contained 160 questions, between 1 and 13 possible answers for every question, with a total of 1055 triplets. A part of the input data for question 13 is presented below:

```
<q id="0013" lang="EN">
  <q_str>What is the occupation of Richard Clayderman?</q_str>
  <a id="0013_1" value="">
    <a_str>Number</a_str>
     <t_str doc="LA091294-0245">U.S. Job Growth  Top 10 occupations with the
      largest projected job growth: 04,27,09,10,08  *2*Number of Jobs* Jobs*
      Occupation 1992 2005 Gained Retail salespeople 3,660  4,446 786
      ...</t_str>
    </a>
   <a id="0013_2" value="">
     <a_str>teacher     Qualifications</a_str>
      <t_str doc="LA103194-0191">* Esther H. Wallace Age: 66     Occupation:
       Retired teacher Qualifications: Worked as a high school teacher in
       Whittier.</t_str>
```

[60] AVE: http://nlp.uned.es/clef-qa/ave/

```
    </a>
   <a id="0013_3" value="">
     <a_str>ways</a_str>
      <t_str doc="LA103194-0191">Open seats: Three Candidates: * Theresa K.
       Dieriex Age: 51 Occupation: Family resource consultant
       Qualifications: Heads a committee examining ways of improving school
       grounds.</t_str>
   </a>
   <a id="0013_8" value="">
     <a_str>pianist</a_str>
      <t_str doc="Richard.html">, fictional character from the television
       series Veronica Mars
       Richard Chamberlain, American actor
       Richard Cheney, Vice President Of The United States
       Richard Clayderman, French pop pianist
       Richard Corben, American comic-strip artist ...</t_str>
   </a>
...
</q>
```

Table 37: AVE - Data Input Format

In the above table, we can see the input data format (where "**q_str**" tag contains the question, "**a**" tags correspond to every possible answer: "**a_str**" tag contains the answer itself, and justification text is in the "**t_str**" tag).

Data Output Format

Participant systems must return one of the following values (Rodrigo et al., 2008) for each answer according to the response format (see Table 38):

- **VALIDATED** indicates that the answer is correct and supported by the given supporting text. There is no restriction in the number of VALIDATED answers returned per question (from zero to all).
- **SELECTED** indicates that the answer is VALIDATED and it is the one chosen as the output to the current question by a hypothetical QA system. The SELECTED answers are evaluated against the QA systems of the Main Track. No more than one answer per question can be marked as SELECTED. At least one of the VALIDATED answers must be marked as SELECTED.
- **REJECTED** indicates that the answer is incorrect or there is not enough evidence of its correctness. There is no restriction in the number of REJECTED answers per question (from zero to all).

q_id a_id [VALIDATED| SELECTED|REJECTED] confidence

Table 38: AVE - Data Output Format

Input Data Collection

Similar to previous editions, the input data collections were built from answers offered by competitors in QA@CLEF main track. Additionally, some transformation was performed as follows (Rodrigo et al., 2008):

- Answers judged as Correct in QA have a value equal to VALIDATED in AVE.
- Answers judged as Wrong or Unsupported in QA have a value equal to REJECTED in AVE.
- Answers judged as Inexact in QA have a value equal to UNKNOWN in AVE and are ignored for evaluation purposes.
- Answers not evaluated at the QA main track (if any) are also tagged as UNKNOWN in AVE and they are also ignored in the evaluation.

5.2.2. Using the TE System in the AVE track

The system architecture for the edition of 2008 is presented below:

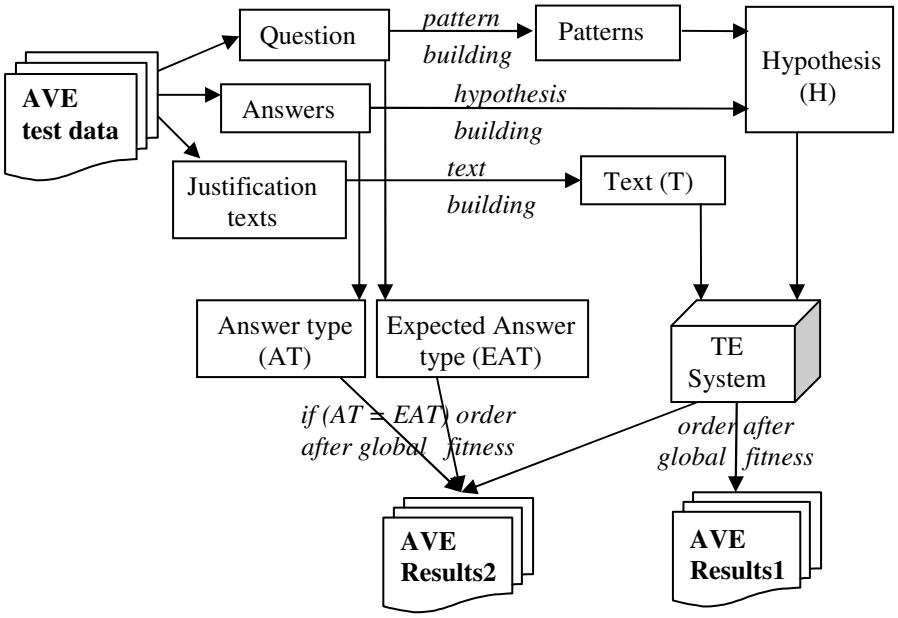


Figure 22: The AVE System for the English track in 2008

The steps executed by our system are:

- Similar to the system built for AVE 2007, the following steps are performed (Iftene and Balahur-Dobrescu, 2008d):
  - We build a pattern with variables for every question according to the question type;
  - Using a pattern and all possible answers, a set of hypotheses for each of the questions:

H_1, H_2, H_3 etc. was built;

- We assign to the justification snippet, the role of text T and we run the TE system for all obtained pairs: (T_1, H_1), (T_2, H_2), (T_3, H_3), etc.

- Additionally, in the 2008 edition the next steps were performed (Iftene and Balahur-Dobrescu, 2008c):
  - Identify the Answer Type (AT);
  - Identify the Expected Answer Type (EAT) in the question.

Lastly, we submit two results for our system:

1. In the first one we consider the correct answer for the current question to be the candidate from the hypothesis for which we obtain the greatest global fitness;
2. In the second one, we consider the correct answer for the current question to be the candidate with AT equal with EAT and for which we obtain the greatest global fitness.

To solve this task, we employ the same strategies as in the experiments presented above at QA system, except for the part involving the extraction of possible answers from the question, which is useless in this case. Thus, we built the *Hypotheses* from the pattern associated to the questions and from the answers, and the *Texts* from the justification texts (see Figure 22) and we calculate the global fitness value for these pairs using our TE system. Additionally, we mark with "NE Problem" the pairs having problems with named entities (the pairs for which the NE rule from TE system can be applied).

Global Fitness Calculation

We consider the pairs built for question 13 from Table 35 (*What is the occupation of Richard Clayderman*?) as input for our Textual Entailment system. The associated pattern is "*The occupation of Richard Clayderman is JOB*." For question 13, pairs (T, H) for the TE system are obtained thus: for *H* the variable JOB from the pattern was replaced with "a_str" values from Table 35 and for *T* we consider corresponding "t_str" values from Table 37.

After running the TE system, the global fitness values and the values marked with "NE Problems" for these pairs are the following:

Pair	Global Fitness	NE Problem
13_1	1.5	Clayderman

Pair	Global Fitness	NE Problem
13_2	2.35	Clayderman
13_3	2.31	Clayderman
13_8	1.92	
13_11	1.82	
13_12	1.86	
13_13	1.89	Clayderman

Table 39: TE System output

Answers Type and Expected Answer Type Identification

The aim of the additional module introduced in the AVE system is to eliminate the cases in which there are differences between Answer Type and Expected Answer Type. For example, in the case of question 13 (*What is the occupation of Richard Clayderman*?), since the expected answer type is JOB, it is normal to try to identify the correct answer in the sub-set of answers of type JOB.

The patterns used in the identification of the expected answer type (EAT) are similar to the patterns used by QA systems in identification of expected answer type (5.1.2.2 iii). For the identification of the answer type (AT), we use GATE[61] for the following types: Job, City, Country, Location, Person, Organization and we build specific patterns in order to identify the following types: Date, Measure, and Count. When an answer cannot be classified with GATE or with our patterns, it is considered to be of type Other. For question number 13, we have:

Pair	EAT	Answer	AT	Match score
13_1	JOB	Number	OTHER	0.25
13_2	JOB	teacher Qualifications	OTHER	0.25
13_3	JOB	Ways	OTHER	0.25
13_8	JOB	Pianist	JOB	1
13_11	JOB	Artist	JOB	1
13_12	JOB	Composer	JOB	1
13_13	JOB	teachers	JOB	1

Table 40: EAT and AT comparison

[61] GATE: http://www.gate.ac.uk/

On the last column we have the matching score between EAT and AT. In order to compute this value, we use a set of rules. The most important rules are:

Rule	Match score
AT = EAT	1
(EAT = "DEFINITION") and (AT = "OTHER")	1
EAT and AT are in the same class of entities: {CITY, COUNTRY, REGION, LOCATION} or {YEAR, DATE} or {COUNT, MEASURE, YEAR}	0.5
(AT = "OTHER") or (EAT = "OTHER")	0.25
OTHERWISE	0

Table 41: Rules for matching score calculation

Answers classification

We submitted two runs for English, the difference being due to some specific system components.

The First run: is based on the TE system output. The answers for which we had NE problems were considered REJECTED (for question 13, using Table 39, we can deduce that answers 1, 2, 3 and 13 are REJECTED). Answers without NE problems are considered VALIDATED (answers 8, 11, 12) and the answer with the highest global fitness is considered SELECTED (answer 8). If all answers contain NE problems, then all answers are considered REJECTED, except for the answer with the highest global fitness, which will be considered SELECTED.

The Second run: in addition to the first run, we added the comparison between EAT and AT. In the cases where we have NE Problems, the answers are considered REJECTED as well, and we also take into consideration if the matching score between EAT and AT is 0 (incompatible types). Out of the remaining answers, if the matching score is not 0, then all answers are VALIDATED. For the identification of the SELECTED answer, we select the answers with the highest matching score (8, 11, 12) and the highest global fitness. In this case, the results are the same.

523. Results in AVE2007 and in AVE2008

With the AVE system based on TE system, we participated in the AVE 2007 and 2008 editions and we obtained good results. The organizers used for ranking a measure that verifies the number

of correctly selected answers (*qa_accuracy*). Additionally, in 2008 they also used a measure that verifies the number of correctly rejected answers (*qa_rej_accuracy*). The formulas for these two measures are presented below (Rodrigo et al., 2008):

$$qa\_accuracy = \frac{answers\_SELECTED\_corectly}{questions}$$

$$qa\_rej\_accuracy = \frac{answers\_REJECTED\_corectly}{questions}$$

In both 2007 and 2008 editions, seven groups participated in the AVE challenge on English, and our system was ranked the first according to the first measure *qa_accuracy*, at equality with another system. In 2008, using the second measure *qa_rej_accuracy*, we were on the fourth place (see below tables with results).

Group	***qa_accuracy***
"Al. I. Cuza" University of Iasi, Romania (Iftene and Balahur-Dobrescu, 2008d)	**0.21 (70%)**
Saarland University and DFKI Lab, Germany (Wang and Neumann, 2007)	0.21 (70 %)
University of Alicante, Spain (Ferrández et al., 2007)	0.19 (65 %)
UNED, Spain (Rodrigo et al., 2007)	0.16 (46.30 %)

Table 42: Results in AVE2007 competition (Peñas et al., 2007)

Group	***qa_accuracy***	***qa_rej_accuracy***
Saarland University and DFKI Lab, Germany (Wang and Neumann, 2008)	0.24 (70.37 %)	0.44
"Al. I. Cuza" University of Iasi, Romania (Iftene and Balahur-Dobrescu, 2008c)	**0.24 (70.37 %)**	**0.01**
University of Alicante, Spain (Ferrández et al., 2008)	0.19 (57.41 %)	0.4
National University of Cordoba, Argentina (Castillo, 2008)	0.16 (46.30 %)	0.1

Table 43: Results in AVE2008 competition (Rodrigo et al., 2008)

The incorrect classifications in our runs are regarding the *qa_rejected_accuracy* (the number of correct REJECTED answers), where the results placed us as the fourth. The

explanation is given by the fact that our AVE system tries to rank the answers in every situation and obtain the most likely answer for the current question, not using conditions for the identification of REJECT cases. The reason for this approach is that in the QA competition, the number of NIL questions is very low (8 out of 200) and therefore, we did not pay special attention to these cases. In the AVE competition, on the other hand, the number of rejected answers was so high because the test data was taken from answers given by the QA participating systems and in many cases these answers were not correct.

5.3. *Applications to Romanian Language*

Encouraged by the good results obtained in the QA competition using the English TE system for answer ranking, we have built a similar system for Romanian. The aim is to use this system in similar Romanian tracks belonging to the CLEF competition.

5.3.1. The Romanian TE system

The main idea is to see where to find the keywords from the hypothesis in the text. The keywords represent the words of the sentences, from which Romanian stop words have been removed. The first step after eliminating the stop words consists in expanding the keywords from the hypothesis using the following resources: WordNet, Acronyms database, and Background Knowledge. Similar to the English system, we use named entities rule, negation rules, and we calculate the global fitness value in order to decide the answer for (text, hypothesis) pairs.

The main characteristics of this system are related to the resources and tools used, which have a correspondent into Romanian language: GATE (Cunningham et al., 2001), WordNet (Tufiş et al., 2004), Wikipedia (Iftene and Balahur-Dobrescu, 2007d), acronyms database, etc. In some cases the quality of these resources and tools was quite low, and in other cases the resources are missing. In both cases, we applied specific algorithms in order to increase their quality or we adapted some existing resources for what we need (Iftene and Balahur-Dobrescu, 2007c).

In order to evaluate the Romanian system, we translated from the RTE-3 datasets both the development and test sets of pairs into Romanian. The results on Romanian (56.7 % on development data and 56.1 % on test data) are lower in comparison with the similar results on English (Iftene and Balahur-Dobrescu, 2008a). The reasons for this difference are the volume and quality of the resources available for Romanian: WordNet and Wikipedia.

5.3.2. Applications

5.3.2.1. Using the Romanian TE system in QA

Similar to the English system, the aim in using the Romanian TE system as a module in the general architecture of a QA system is to improve the ranking between possible answers for questions in which the answer type is Measure, Person, Location, Date and Organization.

In 2007, in the Romanian-Romanian track at QA@CLEF, only the TE system was used to find the correct answer for questions which have answers of type Person and Location. In these cases, the results show an increase in accuracy of up to 5% (Iftene and Balahur-Dobrescu, 2007b).

In the similar edition of 2008, once again, the TE system was used for answer ranking, but since the quality of the QA system was substantially improved, the results show an increase in accuracy up to only 2.5% (Iftene et al., 2008f).

5.3.2.2. Using the Romanian TE system in AVE

The AVE system for Romanian is very similar to the English system (Iftene et al., 2008g). In the two runs submitted the *qa_accuracy* were 0.17 and 0.24 respectively.

Interestingly in the second run the value the value of the *estimated_qa_performance* (the estimated precision of a QA system that use this AVE system for ranking) was equal to 0.25. This value is bigger than the highest precision obtained in the QA competition on Romanian language, and from here we can deduce that this component can improve the quality of the Question Answering systems. The *estimated_qa_performance* is calculated with the following formula (Rodrigo et al., 2008):

$$estimated\_qa\_performance = qa\_accuracy + qa\_accuracy * qa\_rej\_accuracy$$

5.4. *Conclusions*

This chapter demonstrates the utility of a textual entailment system in two important tracks belonging to the QA@CLEF challenge: Question Answering (QA) and Answer Validation Exercise (AVE), both on English and on Romanian.

We showed how the TE system can be used as a module within a QA system, producing an improved ranking of the possible answers for questions of type Measure, Person, Location, Date and Organization. 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 (from 9.47 % to 19 %). Similarly, introduction of a TE system in a Romanian QA system resulted in an

increase in the precision of the Romanian QA system with 5 % percents in 2007 and with 2.5 % in 2008.

In the AVE competition, we showed that the TE system used in the RTE-3 competition can successfully be employed as part of an AVE system to improve ranking between the possible answers, especially in the case of questions with answers of type Measure, Person, Location, Date and Organization. In the edition of 2008, adding the question and answer type classification and the matching component, we showed how we improved, on one hand, the correct classification of the answers, and on the other hand, the validation of more answers. The interesting fact is that on Romanian language our system performed better than the best QA system in the QA competition.

6. Conclusions

This chapter summarizes, gives the main contributions of the thesis, and presents possible future directions of this research.

6.1. *Contributions of the Thesis*

Contributions of the thesis are related to four main directions: (1) presentation of RTE competitions from 2005 to 2008, (2) building of a complex TE system with promising results in the RTE-3 and RTE-4 tracks, (3) improvements in speed using Peer-to-Peer networks and GRID services, and (4) adapting an RTE system in order to improve the quality of QA systems and AVE systems. In what follows, we will see for every direction the most important contributions of the author:

Recognising Textual Entailment:

- *Challenges from 2005 to 2008* are presented with their main characteristics, the new innovations from edition to edition and the best results.
- In the chapter *Trends in Textual Entailment* are presented the main directions followed by competitors from 2005 to 2008.

The Textual Entailment system:

- *The mapping method* is new related to other systems that use trees or graphs to represent the text and the hypothesis. Thus, the mapping method starts from the hypothesis verb and at every step it identifies for the current word the mapping with the maximal value for extended fitness (*context mapping*). The global fitness value is calculated using all extended fitness's values, in comparison with other systems that calculate the distance between text and hypothesis only on basis of local fitness.
- *Using Wikipedia and an English grammar* the definitions contexts are extracted, for identification of relations between named entities. In RTE-3, our system was the first that used Wikipedia to extract additional world knowledge about named entities. In RTE-4, four systems used Wikipedia.
- *Rules for named entities* are more and more specific in order to do a clear distinction between the unknown and contradiction cases. Thus, the *unknown cases* are cases in which a named entity from the hypothesis cannot be mapped to a named entity from the text. For the *contradiction cases* we check also the contexts in which the named entities appear in the hypothesis and in the text, and on basis of these we decide if a contradiction

exists or not.

- *Rules for negations* are responsible with identification of contexts that contradict the meaning of the verbs. In this case, we built two lists of words that change the verbs meaning. The first list contains the words that change clearly the verb meaning: “not”, “never”, “no”, etc. and that corresponds to cases of *contradiction*. The second list corresponds to the *unknown* cases, and it contains words that diminish the verb meaning, as “can”, “could”, “may”, “should”, etc.

System improvements:

- A *Peer-to-Peer network* was used in order to improve the computational speed of the systems used in the RTE competitions from 2007 and 2008. The network architecture is based on the CAN model. It uses caching mechanism for large databases, a quota mechanism for synchronization and uses SMB protocol for file transfer between network peers.
- *GRID services* offer the possibility to have components that determine the entailment relation between two texts in real-time. We implement three types of GRID services: basic and complex NLP services and discovery services.

Applications:

- In a *Question Answering* system a TE module improves the capability 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. The experiments showed how the precision of the QA system is substantially improved using this module.
- In the *Answer Validation Exercise* the TE system was used with success in editions from 2007 and 2008. The results placed our system between the best in these competitions.
- We built a similar TE system on the *Romanian language*, using the tools and resources available on this language. With this system we improved the quality of the answers offered by our QA systems in the QA@CLEF competitions from 2007 and 2008. In 2008 we used the Romanian TE system in the AVE competition on the Romanian language.

6.2. *Future Work*

The future work will be related to the improvement of existing TE, QA and AVE systems, but also to building of new GRID services and also to integration of the current work in ALPE - a complex NLP workflow management system (Cristea and Pistol, 2008). Another important

direction will be publication of current work (tools and resources) as open source under GNU public license. Other future objectives are:

Related to the Textual Entailment system:

- *Building of new resources* can be a source of improvement of the system's precision. The type of resources can be very diversified, from pairs of paraphrases to pairs of rules or pairs of words:
  - *Pairs of paraphrases* which represent the same thing (for entailment cases) or which represent different things (for no entailment cases).
  - *Rules for obtaining extra information* similar with the following forms:
    - if (*X sold Y to Z*) then "*Z takes Y*",
    - if (*X died in Y*) then "*we cannot deduce that X lived in Y*",
    - if (*X talks about Y*) then "*we cannot deduce that X does Y*",
    - if (*X is SPORT champion*) then "*X plays SPORT*".
  - *Pairs of words or expressions from the same class that represent different things* like the following pairs: "*the second largest*" ≠ "*the largest*", "*adults*" ≠ "*children*", "*hand*" ≠ "*foot*", etc.
- *Efficient exploitation of Geographical Resources* assume to use world knowledge when we have neighbour regions like in the following rule:
  - if (*two geographical regions are sufficiently closed*) then "*a disaster can affect both regions*", where *disaster* ∈ {*earthquake, volcanic eruption, tsunami, fire, explosion,* etc.}
- *Using Semantic Role Labelling* in order to identify cases when some words have different semantic roles in the text and in the hypothesis, and for this reason the meanings are different for text and hypothesis.
- The existing *Useless Rules* must be improved in order to become positive rules. These rules are related to the word "but", simple rules for semantic role labelling and simple rules for anaphora resolution. In the case of these rules, we must specify more clearly the contexts when they can be applied.

Related to GRID Services

- *Offering GRID services* on Romanian or English to the NLP community regarding:
  - *Textual Entailment* – this GRID service must receive a pair of text and hypothesis and it must return an answer which specifies if there is entailment or not between

them.

- *Question Answering* – for a question entered in natural language this service will return an exact answer after a search on Internet.
- *Semantic Relations between Named Entities* – starting from a named entity this service will return a list with other named entities related to it.

- *Integration in ALPE* of all implemented GRID services. ALPE (Automated Linguistic Processing Environment) (Cristea and Pistol, 2008) is a system designed to facilitate the management and usage of large and dynamic collections of linguistic resources and tools, using a hierarchy of annotation schemas (Cristea and Butnariu, 2004). The work is ongoing (Pistol and Iftene, 2008).

Open Source Projects

- An important aspect in our next work will be related to the possibility to offer public access to our code, products and knowledge bases. In order to do that, an accurate documentation for these tools and resources will be created as well as useful information.

7. Bibliography

Adams, R. 2006. Textual Entailment Through Extended Lexical Overlap. The Second PASCAL Recognising Textual Entailment Challenge. In *Proceedings of the Second Challenge Workshop Recognising Textual Entailment*, Pages 128-132, 10 April, 2006, Venice, Italia

Adams, R., Nicolae, G., Nicolae, C. and Harabagiu, S. 2007. Textual Entailment Through Extended Lexical Overlap and Lexico-Semantic Matching. . *In Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing.* Pp.119-124. 28-29 June, Prague, Czech Republic

Akhmatova, E. 2005. Textual Entailment Resolution via Atomic Propositions. In *Proceedings of the First Challenge Workshop Recognising Textual Entailment*, Pages 61-64, 33–36 April, 2005, Southampton, U.K

Alfonseca, E. and Pérez, D. 2004. Automatic assessment of short questions with a BLEU-inspired algorithm and shallow nlp. In *Advances in Natural Language Processing*, volume 3230 of *Lecture Notes in Computer Science*, pages 25–35. Springer Verlag.

Andrews, G.R. 1999. Foundations of parallel and distributed programming. Addison-Wesley Longman Publishing Co., Inc., 1999.

Androutsellis-Theotokis, S. and Spinellis, D. 2002. A survey of peer-to-peer files sharing technologies. In *White paper*, Electronic Trading Research Unit (ELTRUN), Athens, University of Economics and Business.

Baker, C., Fillmore, C. and Lowe, J. 1998. The Berkeley FrameNet Project. In Proceedings of 36th ACL.

Bar-Haim, R., Dagan, I., Dolan, B., Ferro, L., Giampiccolo, D., Magnini, B. and Szpector, I. 2006. The Second PASCAL Recognising Textual Entailment Challenge. In *Proceedings of the Second Challenge Workshop Recognising Textual Entailment*, Pages 1-10, 10 April, 2006, Venice, Italia

Bar-Haim, R., Dagan, I., Greental, I., Szpektor, I., Friedman, M. 2007. Semantic Inference at the Lexical-Syntactic Level for Textual Entailment Recognition. *In Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing.* Pp.1-9. 28-29 June, Prague,

Czech Republic

Bar-Haim, R., Berant, J., Dagan, I., Greental, I., Mirkin, S., Shnarch, E., Szpektor, I. 2008. Efficient Semantic Deduction and Approximate Matching over Compact Parse Forests. In *Text Analysis Conference (TAC 2008) Workshop - RTE-4 Track. National Institute of Standards and Technology* (NIST). November 17-19, 2008. Gaithersburg, Maryland, USA.

Basili, R. and Zanzotto, F. M. 2002. Parsing engineering and empirical robustness. *Natural Language Engineering*, 8/2-3.

Bayer, S., Burger, J., Ferro, L., Henderson, J., Yeh, A. 2005. MITRE's Submissions to the EU Pascal RTE Challenge. In *Proceedings of the First PASCAL Challenge Workshop for Recognising Textual Entailment*, pages 41–44, 11–13 April, 2005, Southampton, U.K.

Bensley, J. and Hickl, A. 2008. Workshop: Application of LCC's GROUNDHOG System for RTE-4. In *Text Analysis Conference (TAC 2008) Workshop - RTE-4 Track. National Institute of Standards and Technology* (NIST). November 17-19, 2008. Gaithersburg, Maryland, USA.

Bergmair, R. 2008. Monte Carlo Semantics: MCPIET at RTE4. In *Text Analysis Conference (TAC 2008) Workshop - RTE-4 Track. National Institute of Standards and Technology* (NIST). November 17-19, 2008. Gaithersburg, Maryland, USA.

Bos, J. and Markert, K. 2005. Recognising Textual Entailment with Logical Inference. In *HLT '05: Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, Pages 628-635, Association for Computational Linguistics, Morristown, NJ, USA.

Bos, J., Clark, S., Steedman, M., Curran, J. and Hockenmaier. J. 2004. Wide-coverage semantic representations from a ccg parser. In *Proc of the 20*th *International Conference on Computational Linguistics;* Geneva, Switzerland; 2004.

Brendon, J. W. 2002. JXTA. In *Pearson Education* 2002.

Bunke, H. and Shearer, K. 1998. A graph distance metric based on the maximal common subgraph. *Pattern Recogn. Lett.*, 19(3-4), pages 255–259.

Burchardt, A. and Frank, A. 2006. Approaching Textual Entailment with LFG and FrameNet Frames. In *Proceedings of the Second Challenge Workshop Recognising Textual Entailment*,

Pages 92-97, 10 April, 2006, Venice, Italia

Burchardt, A., Reiter, N., Thater, S., Frank, A. 2007. A Semantic Approach To Textual Entailment: System Evaluation and Task Analysis. In *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*. Pages 10-15. 28-29 June, Prague, Czech Republic.

Burger, J. and Ferro, L. 2005. Generating an Entailment Corpus from News Headlines. *Proceedings of the ACL Workshop on Empirical Modeling of Semantic Equivalence and Entailment*, 49–54. Association for Computational Linguistics, Ann Arbor, Michigan.

Castillo, J. J. 2008. The Contribution of FaMAF at QA@CLEF 2008.Answer Validation Exercise. *In Working Notes of the CLEF 2008 Workshop*. 17-19 September. Aarhus, Denmark.

Chambers, N., Cer, D., Grenager, T., Hall, D., Kiddon, C., MacCartney, B., Marneffe, M. C., Ramage, D., Yeh, E., Manning, C. D. 2007. Learning Alignments and Leveraging Natural Logic. *In Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing.* Pp.165-170. 28-29 June, Prague, Czech Republic

Chierchia, G, and McConnell-Ginet, S. 2001. Meaning and grammar: An introduction to semantics. MIT Press.

Chklovski, T. and Pantel, P. 2004. Verbocean: Mining the web for fine-grained semantic verb relations. In Proceedings of EMNLP 2004, pages 33–40, Barcelona, Spain, July. Association for Computational Linguistics.

Clark, P., Murray, W.R., Thompson, J., Harrison, P., Hobbs, J., Fellbaum. 2007. On the Role of Lexical and World Knowledge in RTE3. *In Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing.* Pp.54-59. 28-29 June, Prague, Czech Republic

Clark, P. and Harrison, P. 2008. Recognizing Textual Entailment with Logical Inference. In *Text Analysis Conference (TAC 2008) Workshop - RTE-4 Track. National Institute of Standards and Technology* (NIST). November 17-19, 2008. Gaithersburg, Maryland, USA.

Clarke, I. 2000. Freenet: a distributed anonymous information storage and retrieval system. In *Workshop on Design Issues in Anonymity and Unobservability*, pp. 46-66. Conference, San Diego, California, March.

Cristea, D. and Butnariu, C. 2004. Hierarchical XML representation for heavily annotated

corpora. In *Proceedings of the LREC 2004 Workshop on XML-Based Richly Annotated Corpora*, (2004), Lisbon, Portugal.

Cristea, D. and Pistol, I. 2008. Managing Language Resources and Tools Using a Hierarchy of Annotation Schemas. Proceedings of the Workshop on Sustainability of Language Resources, LREC-2008, Marakesh.

Cruse, D. 1992. Antonymy Revisited: Some Thoughts on the Relationship between Words and Concepts. In *A. Lehrer and E.V. Kittay (eds.), Frames, Fields, and Contrasts*, Hillsdale, NJ, Lawrence Erlbaum associates, pp. 289-306.

Cunningham, H., Maynard, D., Bontcheva, K., Tablan, V. 2001. *GATE: an architecture for development of robust HLT applications.* In ACL '02: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics, 2001, 168--175, Association for Computational Linguistics, Morristown, NJ, USA

Daelemans, W., Zavrel, J., Ko van der Sloot, and Antal van den Bosch. 1998. Timbl: Tilburg memory based learner, version 1.0, reference guide.

Dagan, I. and Glickman, O. 2004. Probabilistic textual entailment: Generic applied modeling of language variability. In *Learning Methods for Text Understanding and Mining*, Grenoble, France.

Dagan, I., Magini, B., and Glickman, O. 2005. The PASCAL Recognising Textual Entailment Challenge. In *Proceedings of the First PASCAL Challenge Workshop for Recognising Textual Entailment*, pages 1–9, 11–13 April, 2005, Southampton, U.K.

Deerwester, S., Dumais, S. T., Furna, G.W., Landauer, T. K. and Harshman, R. 1990. Indexing by Latent Semantic Analysis. *In Journal of the American Society for Information Science*, 1990.

Delmonte, R., Bristot, A., Piccolino Boniforti, M. A., Tonelli, S. 2007. Entailment and Anaphora Resolution in RTE3. *In Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing.* Pages 48-53. 28-29 June, Prague, Czech Republic

Delmonte, R., Tonelli, S., Aldo Piccolino Boniforti, M., Bristot, A., Pianta, E. 2005. VENSES – a Linguistically-Based System for Semantic Evaluation. In *Proceedings of the First PASCAL Challenge Workshop for Recognising Textual Entailment*, pages 49–52, 11–13 April, 2005,

Southampton, U.K.

Erk, K. and Pado, S. 2006. Shalmaneser - a toolchain for shallow semantic parsing. *In Proceedings of LREC-2006*, Genoa, Italy.

Fellbaum, C. 1998. *Semantic network of English verbs*. In Fellbaum, (ed). WordNet: An Electronic Lexical Database, MIT Press.

Fellbaum, C. 1998. *WordNet: An Electronic Lexical Database*. MIT Press, Cambridge, Mass.

Ferrández, Ó., Micol, D., Muñoz, R. and Palomar, M. 2007. The Contribution of the University of Alicante to AVE 2007. *In Working Notes of the CLEF 2007 Workshop*. 19-21 September. Budapest, Hungary.

Ferrández, Ó., Muñoz, R. and Palomar, M. 2008. A Lexical-Semantic Approach to AVE. *In Working Notes of the CLEF 2008 Workshop*. 17-19 September. Aarhus, Denmark.

Ferrés, D., Rodríguez, H. 2007. Machine Learning with Semantic-Based Distances Between Sentences for Textual Entailment. *In Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing.* Pp.60-65. 28-29 June, Prague, Czech Republic

Fowler, A., Hauser, B., Hodges, D., Niles, I., Novischi, A., Stephan, J. 2005. Applying COGEX to Recognize Textual Entailment. In *Proceedings of the First PASCAL Challenge Workshop for Recognising Textual Entailment*, pages 69-72, 11–13 April, 2005, Southampton, U.K.

Fox, G. C. and Gannon, D. 2001. Computational Grids. *In IEEE Computer Science Eng*. Vol 3, No. 4, Pp. 74-77.

Gannon, D., and Grimshaw, A. 1998. Object-Based Approaches. *The Grid: Blueprint for a New Computing Infrastructure*, Ian Foster and Carl Kesselman (Eds.), Pp. 205-236, Morgan-Kaufman, 1998.

Gannon, D., Bramley, R., Fox, G., Smallen, S., Rossi, A., Ananthakrishnan, R., Bertrand, F., Chiu, K., Farrellee, M., Govindaraju, M., Krishnan, S., Ramakrishnan, L., Simmhan, Y., Slominski, A., Ma, Y., Olariu, C., Rey-Cenvaz, N. 2002. Programming the Grid: Distributed Software Components, P2P and Grid Web Services for Scientific Applications. In *Cluster Computing journal*, Volume 5, Number 3, Pp. 325-336.

Giampiccolo, D., Dang, H. T., Magnini, B., Dagan, I., Dolan, B. 2008. The Fourth PASCAL Recognising Textual Entailment Challenge. In *Text Analysis Conference (TAC 2008)*

Workshop - RTE-4 Track. National Institute of Standards and Technology (NIST). November 17-19, 2008. Gaithersburg, Maryland, USA.

Giampiccolo, D., Magnini, B., Dagan, I., Dolan, B. 2007. The Third PASCAL Recognising Textual Entailment Challenge. *In Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*. Pp.1-9. 28-29 June, Prague, Czech Republic

Glickman, O., Dagan, I. and Koppel, M. 2005b. Web Based Probabilistic Textual Entailment. In *Proceedings of the First Challenge Workshop Recognising Textual Entailment*, Pages 17-20, 33–36 April, 2005, Southampton, U.K

Glickman, O., Dagan, I., and Koppel, M. 2006. A lexical alignment model for probabilistic textual entailment. *In Quinonero-Candela, J.; Dagan, I.; Magnini, B.; d'Alché-Buc, F. (Eds.) Machine Learning Challenges.* Lecture Notes in Computer Science, Vol. 3944, pp. 287-298, Springer, 2006.

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

Harabagiu, S., Miller, G.A., and Moldovan, D. I. 1999. Wordnet 2 - a morphologically and semantically enhanced resource. In *Proceedings of ACL-SIGLEX99: Standardizing Lexical Resources*, pages 1–8, Maryland, June.

Harabagiu, S., Pasca, M., and Maiorano, S. 2000. Experiments with Open-Domain Textual Question Answering. COLING 2000.

Harmeling, S. 2007. An Extensible Probabilistic Transformation-based Approach to the Third Recognising Textual Entailment Challenge. In *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*. Pages 137-142. 28-29 June, Prague, Czech Republic.

Harris, Z. 1954. Distributional structure. *Word*, 10(23): 146-162.

Harris, Z. 1985. Distributional structure. In: *Katz, J. J. (ed.) The Philosophy of Linguistics*. New York: Oxford University Press. pp. 26–47.

Hatcher, E. and Gosnopedic, O. 2004. Lucene in Action. Manning.

Herrera, J., Peas, A. and Verdejo, F. 2005. Textual Entailment Recognition Based on

Dependency Analysis and WordNet. In *Proceedings of the First Challenge Workshop Recognising Textual Entailment*, Pages 21-24, 33–36 April, 2005, Southampton, U.K

Hickl, A. and Bensley, J. 2007. A Discourse Commitment-Based Framework for Recognising Textual Entailment. In *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*. Pages 185-190. 28-29 June, Prague, Czech Republic.

Hickl, A., Bensley, J., Williams, J., Roberts, K., Rink, B., Shi, Y. 2006. Recognising Textual Entailment with LCC's GROUNDHOG System. In *Proceedings of the Second Challenge Workshop Recognising Textual Entailment*, Pages 80-85, 10 April, 2006, Venice, Italia

Iftene, A. 2008a. Discovery Linguistic Services in a GRID Environment. In proceedings Scientific and Educational GRID Applications. 5th European Conference on Intelligent Systems and Technologies (ECIT 2008). Publishing House "Politehnium", Iasi. Pages 49-60. ISBN 978-973-621-236-9. 10-12 July. Iasi, Romania.

Iftene, A. 2008b. Building a Textual Entailment System for the RTE3 Competition. Application to a QA System. In *proceedings of 10th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing* (SYNASC 2008). September 26-29, Timişoara, România.

Iftene, A. 2008c. UAIC Participation at RTE4. In *Text Analysis Conference (TAC 2008) Workshop - RTE-4 Track. National Institute of Standards and Technology* (NIST). November 17-19, 2008. Gaithersburg, Maryland, USA.

Iftene, A., Balahur-Dobrescu, A. 2007a. Hypothesis Transformation and Semantic Variability Rules Used in Recognising Textual Entailment. In *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*. Pages 125-130. 28-29 June, Prague, Czech Republic.

Iftene, A., Balahur-Dobrescu, A. 2007b. Improving a QA System for Romanian Using Textual Entailment. In *Proceedings of RANLP workshop "A Common Natural Language Processing Paradigm For Balkan Languages"*. ISBN 978-954-91743-8-0, Pp. 7-14, September 26, 2007, Borovets, Bulgaria.

Iftene, A., Balahur-Dobrescu, A. 2007c. Textual Entailment on Romanian. The third Workshop on Romanian Linguistic Resources and Tools for Romanian Language Processing. ISSN 1843-911X. Pp. 109-118, 14-15 December. Iaşi, România.

Iftene, A., Balahur-Dobrescu, A. 2007d. Name entity relation discovery using Wikipedia for Romanian. The third Workshop on Romanian Linguistic Resources and Tools for Romanian Language Processing. ISSN 1843-911X. Pp. 99-108, 14-15 December. Iaşi, România.

Iftene, A., Balahur-Dobrescu, A. 2008a. A Language Independent Approach for Recognising Textual Entailment. In *journal "Research in Computing Science"*. Vol. 334, Pp. 3-14. Instituto Politecnico Nacional, Centro de Investigacion en Computacion, Mexico 2007. ISSN: 1870-4069. *Poster at 9th International Conference on Intelligent Text Processing and Computational Linguistics* (CICling 2008). 17-23 February. Haifa, Israel.

Iftene, A., Balahur-Dobrescu, A. 2008b. Named Entity Relation Mining Using Wikipedia. In *Proceedings of the Sixth International Language Resources and Evaluation (LREC'08).* ISBN: 2-9517408-4-0, EAN: 9782951740846. 28-30 May, Marrakech, Morocco.

Iftene, A., Balahur-Dobrescu, A. 2008c. Answer Validation on English and Romanian Languages. *In Working Notes of the CLEF 2008 Workshop*. 17-19 September. Aarhus, Denmark.

Iftene, A., Balahur-Dobrescu, A. 2008d. UAIC Participation in AVE 2007. In CLEF 2007. C. Peters et al. (Eds.), Lecture Notes in Computer Science, LNCS 5152, Pp. 395-403, Springer-Verlag Berlin Heidelberg 2008

Iftene, A., Balahur-Dobrescu, A. and Matei, D. 2007. A Distributed Architecture System for Recognising Textual Entailment. In *proceedings of 9th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing* (SYNASC 2007). Pp. 219--226. Published by the IEEE Computer Society. ISBN 0-7695-3078-8. September 26-29, Timişoara, România.

Iftene, A., Croitoru, C. 2006. Graph Coloring using Peer-to-Peer Networks. In *Proceedings of 5th International Conference RoEduNet IEEE*. Pp. 181-185. Sibiu, România. 1-3 June, 2006.

Iftene, A., Pistol, I., Trandabăţ, D. 2008e. Grammar-based Automatic Extraction of Definitions. In proceedings of 10th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC 2008). September 26-29, Timişoara, România.

Iftene, A., Pistol, I., Trandabăţ, D. 2008f. UAIC Participation at QA@CLEF2008. *In Working*

Notes of the CLEF 2008 Workshop. 17-19 September. Aarhus, Denmark

Iftene, A., Rotaru, A., Marcu, D. A. 2008g. The evaluation of the answers for a Romanian Question Answering system. The fourth Workshop on Romanian Linguistic Resources and Tools for Romanian Language Processing. 19-20 November. Iaşi, România.

Iftene, A., Trandabăţ, D. and Pistol, I. 2007. Grammar-based Automatic Extraction of Definitions and Applications for Romanian. In *Proceedings of RANLP workshop "Natural Language Processing and Knowledge Representation for eLearning environments"*. ISBN 978-954-452-002-1, Pp. 19-25, September 26, 2007, Borovets, Bulgaria.

Iftene, A., Trandabăţ, D., Pistol, I., Moruz, A., Balahur-Dobrescu, A., Cotelea, D., Dornescu, I., Drăghici, I., Cristea, D. 2008h. UAIC Romanian Question Answering system for QA@CLEF. In *CLEF 2007. C. Peters et al. (Eds.), Lecture Notes in Computer Science, LNCS 5152*, Pp. 336-343, Springer-Verlag Berlin Heidelberg 2008

Inkpen, D., Kipp, D. and Nastase, V. 2006. Machine Learning Experiments for Textual Entailment. In *Proceedings of the Second Challenge Workshop Recognising Textual Entailment*, Pages 17-20, 10 April, 2006, Venice, Italia

Jagadeesh, J., Prasad, P. and Varma, V. 2005. A relevance-based language modeling approach to duc 2005. In Document Understanding Conference 2005.

Jijkoun, V. and de Rijke, M. 2005. Recognising Textual Entailment Using Lexical Similarity. Proceedings of the PASCAL Challenges Workshop on Recognising Textual Entailment.

Joachims, T. 2002. Learning to Classify Text Using Support Vector Machines. Kluwer.

Jurafsky, D. and Martin, J. H. 2000. Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics and Speech Recognition. Prentice-Hall. Chapter 14-15.

Kamp, H. and Reyle, U. 1993. From Discourse to Logic. Introduction to Model theoretic Semantics of Natural Language*, Formal Logic and Discourse Representation Theory*. Kluwer, Dordrecht, Netherlands.

Katrenko, S. and Adriaans, P. 2006. Using Maximal Embedded Syntactic Subtrees for Textual Entailment Recognition. In *Proceedings of the Second Challenge Workshop Recognising Textual Entailment*, Pages 33-37, 10 April, 2006, Venice, Italia

Kouylekov, M. and Magnini, B. 2005. Recognising Textual Entailment with Tree Edit Distance Algorithms. In *Proceedings of the First Challenge Workshop Recognising Textual Entailment*, Pages 17-20, 25–28 April, 2005, Southampton, U.K.

Kozareva, Z. and Montoyo, A. 2006. MLEnt: The Machine Learning Entailment System of the University of Alicante. In *Proceedings of the Second Challenge Workshop Recognising Textual Entailment*, Pages 17-20, 10 April, 2006, Venice, Italia

Krestel, R., Bergler, S. and Witte, R. 2008. A Belief Revision Approach to Textual Entailment Recognition. In *Text Analysis Conference (TAC 2008) Workshop - RTE-4 Track. National Institute of Standards and Technology* (NIST). November 17-19, 2008. Gaithersburg, Maryland, USA.

Li, B., Irwin, J., Garcia, E.V. and Ram, A. 2007. Machine Learning Based Semantic Inference: Experiments and Observations at RTE-3. In *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*. Pages 159-164. 28-29 June, Prague, Czech Republic.

Li, F., Zheng, Z., Tang, Y., Bu, F., Ge, R., Zhu, X., Zhang, X., Huang, M. 2008. THU QUANTA at TAC 2008 QA and RTE track. In *Text Analysis Conference (TAC 2008) Workshop - RTE-4 Track. National Institute of Standards and Technology* (NIST). November 17-19, 2008. Gaithersburg, Maryland, USA.

Lin, C. and Hovy, E. 2003. Automatic evaluation of summaries using n-gram co-occurrence statistics. In *Proceedings of the Human Technology Conference 2003 (HLT-NAACL-2003).*

Lin, D. 1998. Dependency-based evaluation of minipar. In Workshop on the Evaluation of Parsing Systems, Granada, Spain.

Lin, D. 2001. *LaTaT: Language and Text Analysis Tools*. Proc. Human Language Technology

Lin, D. and Pantel, P. 2001. Dirt - discovery of inference rules from text. In *Proceedings of ACM Conference on Knowledge Discovery and Data Mining*. San Francisco, CA., pages 323–328.

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

Martínez, D., Agirre, E., Màrquez, L. 2002. *Syntactic features for high precision Word Sense Disambiguation*. In Proceedings of the 19th International Conference on Computational Linguistics, COLING'02. Taipei, Taiwan.

Miller, G. 1995. WordNet: A Lexical Database for English, Communications of the ACM, 38(11): Pages 39-41.

Montejo-Ráez, A., Perea, J.M, Martínez-Santiago, F., García-Cumbreras, M. Á., Martín-Valdivia, M., Ureña-López, A. 2007. Combining Lexical-Syntactic Information with Machine Learning for Recognising Textual Entailment. In *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*. Pages 78-82. 28-29 June, Prague, Czech Republic.

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

Padó, S., Marneffe, M. C., MacCartney, B., Rafferty, A. N., Yeh, E., Manning, C. D. 2008. Deciding Entailment and Contradiction with Stochastic and Edit Distance-based Alignment. In *Text Analysis Conference (TAC 2008) Workshop - RTE-4 Track. National Institute of Standards and Technology* (NIST). November 17-19, 2008. Gaithersburg, Maryland, USA.

Papineni, K., Roukos, S., Ward, T. and Zhu, W. 2001. BLEU: a method for automatic evaluation of machine translation. Research report, IBM.

Pazienza, M.T., Pennacchiotti, M., Zanzotto, F. M. 2005. Textual Entailment as Syntactic Graph Distance: a rule based and a SVM based approach. In *Proceedings of the First Challenge Workshop Recognising Textual Entailment*, Pages 9-12, 25–28 April, 2005, Southampton, U.K.

Peñas, A., Rodrigo, Á., Verdejo, F. 2007. Overview of the Answer Validation Exercise 2007. *In Working Notes of the CLEF 2007 Workshop*. 19-21 September, Budapest, Hungary.

Pérez, D. and Alfonseca, E. 2005. Application of the Bleu algorithm for recognising textual entailments. In *Proceedings of the First Challenge Workshop Recognising Textual Entailment*, Pages 9-12, 11–13 April, 2005, Southampton, U.K.

Pistol, I., **Iftene, A.** 2008. Linguistic Processing Architecture in a GRID Environment. In *proceedings Scientific and Educational GRID Applications. 5th European Conference on Intelligent Systems and Technologies* (ECIT 2008). Publishing House "Politehnium", Iasi. Pages 61-74. ISBN 978-973-621-236-9. 10-12 July. Iasi, Romania.

Platt, J. 1998. 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

Puşcaşu, G., **Iftene, A.,** Pistol, I., Trandabăţ, D., Tufiş, D., Ceauşu, A., Ştefănescu, D., Ion, R., Orasan, C., Dornescu, I., Moruz, A., Cristea, D. 2006. Developing a Question Answering System for the Romanian-English Track at CLEF 2006. In *Proceedings of the CLEF 2006 Workshop. Lecture Notes in Computer Science*, Vol. 4730, pp. 385-394, Springer. 22-24 September. Alicante, Spain.

Quinlan, J.R. 2000. C5.0 Machine Learning Algorithm http://www.rulequest.com

Ragib, H. and Zahid, A. 2005. A survey of peer-to-peer storage techniques for distributed file system. *National Center for Supercomputing Applications Department of Computer Science*, April 2005.

Raina, R., Haghighi, A., Cox, C., Finkel, J., Michels, J., Toutanova, K., MacCartney, B.,, Marneffe, M.C., Manning, C., Ng, A., Y. 2005. Robust Textual Inference using Diverse Knowledge Sources. In *Proceedings of the First Challenge Workshop Recognising Textual Entailment*, Pages 57-60, 11–13 April, 2005, Southampton, U.K.

Ratnasamy, S., Francis, P., Handley, M., Karp, R. and Shenker, S. 2000. A scalable content addressable network. In *TR-00-010*, 2000.

Richardson, S. D. 1997. Determining Similarity and the Inferring Relations in a Lexical Knowledge-Base. *Ph.D. Thesis*. The City University of New York.

Rodrigo, Á., Peñas, A. and Verdejo, F. 2007. UNED at Answer Validation Exercise 2007. In *Working Notes of the CLEF 2007 Workshop*. 19-21 September. Budapest, Hungary.

Rodrigo, Á., Peñas, A. and Verdejo, F. 2008. Overview of the Answer Validation Exercise 2008. In *Working Notes of the CLEF 2008 Workshop*. 17-19 September. Aarhus, Denmark.

Rowstron, A. and Druschel, P. 2001. Pastry: Scalable, decentralized object location and routing for large-scale peer-to-peer systems. *IFIP/ACM International Conference on Distributed Systems Platforms (Middleware), Heidelberg, Germany*: 329–350, http://research.microsoft.com/~antr/PAST/pastry.pdf.

Santorini, B. 1990. Part-of-Speech Tagging Guidelines for the Penn Treebank Project (3rd Revision, 2nd printing), ftp://ftp.cis.upenn.edu/pub/treebank/doc/tagguide.ps.gz

Schubert, L. K. and Hwang, C. H. 2000. Episodic Logic meets Little Red Riding Hood: A comprehensive, natural representation for language understanding. In *L. Iwanska and S.C. Shapiro (eds.), Natural Language Processing and Knowledge Representation*. MIT/AAAI Press.

Siblini, R. and Kosseim, L. 2008. Using Ontology Alignment for the TAC RTE Challenge. In *Text Analysis Conference (TAC 2008) Workshop - RTE-4 Track. National Institute of Standards and Technology* (NIST). November 17-19, 2008. Gaithersburg, Maryland, USA.

Sleator, D. D. and Temperley, D. 1993. Parsing English with a link grammar. *In Third International Workshop on Parsing Technologies*. Tilburg, the Netherlands, August 1993.

Stoica, I., Morris, R., Karger, D., Kaashoek, M. F. and Balakrishnan, H. 2001. Chord: A scalable peer-to-peer lookup service for Internet applications. *In ACM SIGCOMM*, August 2001.

Tatu, M. and Moldovan, D. 2007. COGEX at RTE3. In *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*. Pages 22-27. 28-29 June, Prague, Czech Republic.

Tatu, M., Iles, B., Slavick, J., Novischi, A., Moldovan, D. 2006. COGEX at the Second Recognising Textual Entailment Challenge. In *Proceedings of the Second Challenge Workshop Recognising Textual Entailment*, Pages 17-20, 10 April, 2006, Venice, Italia

Tufiş, D. and Cristea, D. 1985. IURES: A Human Engineering Approach to Natural Language Question-Answering Systems. *In Artificial Intelligence. Methodology, Systems, Applications*. Editors: Bibel, W., Petkoff, B. North-Holland, Amsterdam, pag.177-184.

Tufiş, D., Barbu, E., Barbu Mititelu, V., Ion, R., Bozianu, L. 2004. The Romanian Wordnet. *Romanian Journal of Information Science and Technology*, Volume 7, Numbers 1-2, pp. 107-124.

Vanderwende, L., Coughlin, D., Dolan, B. 2005. What Syntax can Contribute in Entailment Task. In *Proceedings of the First Challenge Workshop Recognising Textual Entailment*, Pages 13-16, 11–13 April, 2005, Southampton, U.K.

Verhagen, M., Mani, I., Sauri, R., Knippen, R., Littman, J. and Pustejovsky, J. 2005. Automating Temporal Annotation with TARSQI. *In Proceedings of ACL 2005*. Demo Session.

Voorhees, E.M. and Harman, D. 1999. Overview of the seventh text retrieval conference. In

Proceedings of the Seventh Text Retrieval Conference (TREC-7). NIST Special Publication.

Wang, R. and Neumann, G. 2007. DFKI–LT at AVE 2007: Using Recognising Textual Entailment for Answer Validation. In *Working Notes of the CLEF 2007 Workshop*. 19-21 September. Budapest, Hungary.

Wang, R. and Neumann, G. 2008. Information Synthesis for Answer Validation. In *Working Notes of the CLEF 2008 Workshop*. 17-19 September. Aarhus, Denmark.

Wang, R. and Neumann, G. 2008b. An Accuracy-Oriented Divide-and-Conquer Strategy for Recognising Textual Entailment. In *Text Analysis Conference (TAC 2008) Workshop - RTE-4 Track. National Institute of Standards and Technology* (NIST). November 17-19, 2008. Gaithersburg, Maryland, USA.

Wu, D. 2005. Textual Entailment Recognition Based on Inversion Transduction Grammars. In *Proceedings of the First Challenge Workshop Recognising Textual Entailment*, Pages 13-16, 37–40 April, 2005, Southampton, U.K.

Zanzotto, F. M., Moschitti, A., Pennacchiotti, M., Pazienza, M. T. 2006. Learning textual entailment from examples. In *Proceedings of the Second Challenge Workshop Recognising Textual Entailment*, Pages 50-55, 10 April, 2006, Venice, Italia

Zhang K., Shasha D. 1990. Fast algorithm for the unit cost editing distance between trees. *Journal of algorithms, vol. 11, p. 1245-1262, December 1990*

Zhao, B.Y., Kubiatowicz, J. D. and Joseph, A. D. 2001. Tapestry: An infrastructure for fault-resilient wide-area location and routing. Technical Report UCB-CSD-01-1141, U. C. Berkeley, April 2001.

8. Appendixes

8.1. *Appendix 1 – Example of the MINIPAR output*

All following results are obtained on MINIPAR demo page[62].

Sentence Input: *Le Beau Serge was directed by Chabrol.*

XML Result:

```
<?xml version="1.0" encoding="utf-8" ?>
<node label="E1" category="U">
  <node label="E0" category="C" word="" root="fin">
    <node label="5" category="V" word="directed" root="direct" relation="i">
      <node label="3" category="N" word="Serge" root="Le Beau Serge" relation="s">
        <node label="1" category="U" word="Le" root="Le" relation="lex-mod" />
        <node label="2" category="U" word="Beau" root="Beau" relation="lex-mod" />
      </node>
      <node label="4" category="be" word="was" root="be" relation="be" />
      <node label="E2" category="N" word="" root="Le Beau Serge" relation="obj" />
      <node label="6" category="Prep" word="by" root="by" relation="by-subj">
       <node label="7" category="N" word="Chabrol" root="Chabrol" relation="pcomp-n"/>
      </node>
    </node>
  </node>
  <node label="8" category="U" word="." root="." relation="punc" />
</node>
```

Table 44: MINIPAR – XML output

Tree Result:

```
E1      (U
E0        (C  (root fin)
5           (V directed (root direct) (relation i)
3             (N Serge (root Le Beau Serge) (relation s)
1               (U Le (root Le) (relation lex-mod))
2               (U Beau (root Beau) (relation lex-mod)))
4             (be was (root be) (relation be))
E2            (N  (root Le Beau Serge) (relation obj))
6             (Prep by (root by) (relation by-subj)
7               (N Chabrol (root Chabrol) (relation pcomp-n)))))
8         (U . (root .) (relation punc)))
```

Table 45: MINIPAR – Tree result

Triple Result:

[62] MINIPAR demo page: http://202.113.229.160/miniparweb/Default.aspx

parent_root	parent_category:relation:category	root
fin	C:i:V	direct
direct	V:s:N	Le Beau Serge
Le Beau Serge	N:lex-mod:U	Le
Le Beau Serge	N:lex-mod:U	Beau
direct	V:be:be	be
direct	V:obj:N	Le Beau Serge
direct	V:by-subj:Prep	by
by	Prep:pcomp-n:N	Chabrol

Table 46: MINIPAR – Triple result

Text Result

```
(
E0      (()     fin             C       *       )
1       (Le     ~               U       3       lex-mod         (gov Le Beau Serge))
2       (Beau   ~               U       3       lex-mod         (gov Le Beau Serge))
3       (Serge  Le Beau Serge   N       5       s               (gov direct))
4       (was    be              be      5       be              (gov direct))
5       (directed       direct  V       E0      i               (gov fin))
E2      (()     Le Beau Serge   N       5       obj             (gov direct)    (antecedent 3))
6       (by     ~               Prep    5       by-subj         (gov direct))
7       (Chabrol        ~       N       6       pcomp-n         (gov by))
8       (.      ~               U       *       punc)
)
```

Table 47: MINIPAR – Text result

Table Result

label	word	root	category	parent label	relation	parent root	antecedent label	attributes
E0		fin	C	*				
1	Le	~	U	3	lex-mod	Le Beau Serge		
2	Beau	~	U	3	lex-mod	Le Beau Serge		
3	Serge	Le Beau Serge	N	5	s	direct		
4	was	be	be	5	be	direct		

label	word	root	category	parent label	relation	parent root	antecedent label	attributes
5	directed	direct	V	E0	i	fin		
E2		Le Beau Serge	N	5	obj	direct	3	
6	by	~	Prep	5	by-subj	direct		
7	Chabrol	~	N	6	pcomp-n	by		
8	.	~	U	*	punc			

Table 48: MINIPAR – Table result

Help - Grammatical Categories

Category	Description
Det	Determiners
PreDet	Pre-determiners (search for PreDet in data/wndict.lsp for instances)
PostDet	Post-determiners (search for PostDet in data/wndict.lsp for instances)
NUM	numbers
C	Clauses
I	Inflectional Phrases
V	Verb and Verb Phrases
N	Noun and Noun Phrases
NN	noun-noun modifiers
P	Preposition and Preposition Phrases
PpSpec	Specifiers of Preposition Phrases (search for PpSpec in data/wndict.lsp for instances)
A	Adjective/Adverbs
Have	have
Aux	Auxiliary verbs, e.g. should, will, does, ...
Be	Different forms of be: is, am, were, be, ...
COMP	Complementizer
VBE	be used as a linking verb. E.g., I am hungry
V_N	verbs with one argument (the subject), i.e., intransitive verbs

Category	Description
V_N_N	verbs with two arguments, i.e., transitive verbs
V_N_I	verbs taking small clause as complement

Table 49: MINIPAR – Grammatical Categories

Help - Grammatical Relationships

Relationship	Description
appo	"ACME president, --appo-> P.W. Buckman"
aux	"should <-aux-- resign"
be	"is <-be-- sleeping"
c	"that <-c-- John loves Mary"
comp1	first complement
det	"the <-det `-- hat"
gen	"Jane's <-gen-- uncle"
have	"have <-have-- disappeared"
i	the relationship between a C clause and its I clause
inv-aux	inverted auxiliary: "Will <-inv-aux-- you stop it?
inv-be	inverted be: "Is <-inv-be-- she sleeping"
inv-have	inverted have: "Have <-inv-have-- you slept"
mod	the relationship between a word and its adjunct modifier
pnmod	post nominal modifier
p-spec	specifier of prepositional phrases
pcomp-c	clausal complement of prepositions
pcomp-n	nominal complement of prepositions
post	post determiner
pre	pre determiner
pred	predicate of a clause
rel	relative clause
vrel	passive verb modifier of nouns
wha, whn, whp	wh-elements at C-spec positions

Relationship	Description
obj	object of verbs
obj2	second object of ditransitive verbs
subj	subject of verbs
s	surface subject

Table 50: MINIPAR – Grammatical Relationships

8.2. *Appendix 2 – MINIPAR relations*

No	Relation	Direct	Indirect	Description
1	By-subj	X		Subject with passives
2	C		X	Clausal complement
3	Cn		X	Nominalized clause
4	compl	X		Complement (PP, inf/fin clause) of noun
5	Desc	X		Description
6	Fc	X		Finite complement
7	I		X	See c and fc, dep between clause and main verb
8	Mod	X		Modifier
9	Obj	X		Object
10	pcomp-c	X		Clause of pp
11	Pcomp-n	X		Nominal head of pp
12	Pnmod	X		Post-nominal modifier
13	Pred	X		Predicative (can be A or N)
14	Sc	X		Sentential complement
15	Subj	X		Subject
16	Vrel	X		Passive verb modifier of nouns

Table 51: MINIPAR relations (Martínez et al., 2002)

8.3. *Appendix 3 – Penn-Treebank-Tagset*

The following table contains a list of tags from Penn-Treebank in alphabetical order and their corresponding POS (Santorini, 1990).

No	Tag	Corresponding part-of-speech
1	CC	Coordinating conjunction
2	CD	Cardinal number
3	DT	Determiner
4	EX	Existential there
5	FW	Foreign word
6	IN	Preposition or subordinating conjunction
7	JJ	Adjective
8	JJR	Adjective, comparative
9	JJS	Adjective, superlative
10	LS	List item marker
11	MD	Modal
12	NN	Noun, singular or mass
13	NNS	Noun, plural
14	NP	Proper noun, singular
15	NPS	Proper noun, plural
16	PDT	Predeterminer
17	POS	Possessive ending
18	PP	Personal pronoun
19	PP$	Possessive pronoun
20	RB	Adverb
21	RBR	Adverb, comparative
22	RBS	Adverb, superlative
23	RP	Particle
24	SYM	Symbol
25	TO	to
26	UH	Interjection
27	VB	Verb, base form
28	VBD	Verb, past tense
29	VBG	Verb, gerund or present participle

No	Tag	Corresponding part-of-speech
30	VBN	Verb, past participle
31	VBP	Verb, non-3rd person singular present
32	VBZ	Verb, 3rd person singular present
33	WDT	Wh-determiner
34	WP	Wh-pronoun
35	WP$	Possessive wh-pronoun
36	WRB	Wh-adverb

Table 52: Penn-Treebank-Tagset

8.4. *Appendix 4 – Negation Words*

For the identification of negation the words given in the next table are considered to negate the sense of the verb when are placed in front of it. These words were categorized on two classes: *contradiction*, with a score of negation set maxim i.e. 1, and *unknown* with the score for negation set to 0.5.

```
<?xml version="1.0" encoding="UTF-8" ?>
<negations>
   <word><lemma>not</lemma><type>C</type></word>
   <word><lemma>refuse</lemma><type>C</type></word>
   <word><lemma>wrong</lemma><type>C</type></word>
   <word><lemma>deny</lemma><type>C</type></word>
   <word><lemma>no</lemma><type>C</type></word>
   <word><lemma>false</lemma><type>C</type></word>
   <word><lemma>ignore</lemma><type>C</type></word>
   <word><lemma>cannot</lemma><type>C</type></word>
   <word><lemma>never</lemma><type>C</type></word>
   <word><lemma>unsuccessfully</lemma><type>C</type></word>

   <word><lemma>could</lemma><type>U</type></word>
   <word><lemma>might</lemma><type>U</type></word>
   <word><lemma>must</lemma><type>U</type></word>
   <word><lemma>infrequent</lemma><type>U</type></word>
   <word><lemma>rather</lemma><type>U</type></word>
   <word><lemma>probably</lemma><type>U</type></word>
   <word><lemma>likely</lemma><type>U</type></word>

   <word><lemma>to</lemma><type>U</type></word>
</negations>
```

Table 53: Words which used before a verb change its meaning

As can be seen for every word the lemma and the negation type (C from “Contradiction” and U from “Unknown”) are recorded. A special case is the word “to” to which correspond additional types, as given in the table below:

```
<?xml version="1.0" encoding="UTF-8" ?>
<to_words>
  <word><pos>V</pos><lemma>allow</lemma><type>P</type></word>
  <word><pos>V</pos><lemma>like</lemma><type>P</type></word>
  <word><pos>V</pos><lemma>move</lemma><type>P</type></word>
  <word><pos>V</pos><lemma>impose</lemma><type>P</type></word>
  <word><pos>V</pos><lemma>galvanize</lemma><type>P</type></word>
  <word><pos>V</pos><lemma>go</lemma><type>P</type></word>
  <word><pos>A</pos><lemma>necessary</lemma><type>P</type></word>
  <word><pos>A</pos><lemma>compulsory</lemma><type>P</type></word>
  <word><pos>A</pos><lemma>glad</lemma><type>P</type></word>

  <word><pos>V</pos><lemma>believe</lemma><type>P</type></word>
  <word><pos>V</pos><lemma>aim</lemma><type>P</type></word>
```

```
  <word><pos>V</pos><lemma>mean</lemma><type>P</type></word>
  <word><pos>V</pos><lemma>claim</lemma><type>P</type></word>
  <word><pos>A</pos><lemma>free</lemma><type>P</type></word>

  <word><pos>N</pos><lemma>attempt</lemma><type>C</type></word>
  <word><pos>N</pos><lemma>proposal</lemma><type>C</type></word>
  <word><pos>N</pos><lemma>plan</lemma><type>C</type></word>
  <word><pos>N</pos><lemma>refused</lemma><type>C</type></word>
  <word><pos>V</pos><lemma>refuse</lemma><type>C</type></word>
  <word><pos>N</pos><lemma>refusing</lemma><type>C</type></word>
  <word><pos>N</pos><lemma>intend</lemma><type>C</type></word>
  <word><pos>N</pos><lemma>not</lemma><type>C</type></word>
  <word><pos>N</pos><lemma>rejected</lemma><type>C</type></word>

  <word><pos>N</pos><lemma>stand</lemma><type>U</type></word>
  <word><pos>N</pos><lemma>approached</lemma><type>U</type></word>
</to_words>
```

Table 54: Words that used before "to" change the sense of the infinitive verb

In the table above are types for words that "diminish" the sense of the verbs (type U, from Unknown and C, from Contradiction) and types for words that as "certain" the sense of the verb (type P, from Positive).

8.5. *Appendix 5 – An example of how CIFS works*

SMBs have a specific format that is very similar for both requests and responses. Each consists of a fixed size header portion, followed by a variable sized parameter and data portion:

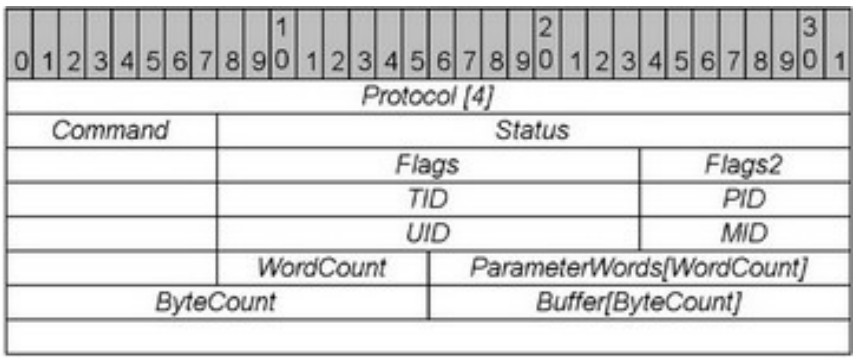


Figure 23: SMB Header

After connecting at the network level, the client is ready to request services from the server. However, the client and server must first identify which protocol variant they each understand. The client negotiates the protocol which will be further used in its communication with the server.

Once a protocol has been established, the client can proceed to login to the server, if required. One of the most important aspects of the response is the UID (*user identifier*) of the logged on user. This UID must be submitted with all subsequent SMBs on that connection to the server.

The client sends a SMB specifying the network name of the share that they wish to connect to, and if all are correct, the server responds with a TID (*tree identifier*) that the client will use in all future SMBs relating to that share.

Having connected to a tree, the client can now open a file with an open SMB, followed by reading it with read SMBs, writing it with write SMBs, and closing it with close SMBs.

8.6. *Appendix 6 – GRID services implementation*

For writing and deploying a WSRF Web Service we must follow five steps:

1. **Define the service's interface** with *WSDL*
2. **Implement the service** with *Java*.
3. **Define the deployment parameters** with *WSDD* and *JNDI*
4. **Compile everything and generate a GAR file** with *Ant*
5. **Deploy of the service** with *a GT4 tool*

Define the service's interface. The first step in writing a web service (including those that use WSRF to keep state) is to define the *service interface*. The service needs to specify what is going to provide to the outer world. At this point we are not concerned with the inner workings of that service (what algorithms it uses, other systems it interacts with, etc.). We just need to know what *operations* will be available to our users. In Web Services lingo, the service interface is usually called the *port type* (usually written *portType*). There is a special XML language which can be used to specify what operations a web service offers: the Web Service Description Language (WSDL). At this step a description of the NLPServices is writing using WSDL.

```
<?xml version="1.0" encoding="UTF-8"?>
<definitions name="NLPServices" …>

<!--=====   T Y P E S   =======-->
<types>
<xsd:schema      targetNamespace="http://www.globus.org/namespaces/uaic/fii
/NLPServices_instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:tns="http://www.globus.org/namespaces/uaic/fii/NLPServices_instance">
        <!-- REQUESTS AND RESPONSES -->

        <xsd:element name="lemma" type="xsd:string"/>
        <xsd:element name="addResponse">
                <xsd:complexType/>
        </xsd:element>

        <!-- RESOURCE PROPERTIES -->
        <xsd:element name="Word" type="xsd:string"/>
        <xsd:element name="NLPResourceProperties">
        <xsd:complexType>
          <xsd:sequence>
            <xsd:element ref="tns:Value" minOccurs="1" maxOccurs="1"/>
            <xsd:element ref="tns:LastOp" minOccurs="1" maxOccurs="1"/>
          </xsd:sequence>
        </xsd:complexType>
        </xsd:element>
</xsd:schema>
</types>

<!--===  M E S S A G E S  ===-->
<message name="LemmaInputMessage">
        <part name="parameters" element="tns:lemma"/>
```

```
</message>
<message name="LemmaOutputMessage">
        <part name="parameters" element="tns:lemmaResponse"/>
</message>
<message name="GetValueRPOutputMessage">
        <part name="parameters" element="tns:getValueRPResponse"/>
</message>

<!--====    P O R T T Y P E ====-->
<portType name="NLPPortType" wsdlpp:extends="wsrpw:GetResourceProperty"
    wsrp:ResourceProperties="tns:NLPResourceProperties">
        <operation name="lemma">
                <input message="tns:LemmaInputMessage"/>
                <output message="tns:LemmaOutputMessage"/>
        </operation>

        <operation name="getValueRP">
                <input message="tns:GetValueRPInputMessage"/>
                <output message="tns:GetValueRPOutputMessage"/>
        </operation>
</portType>
</definitions>
```

Implement the service: After defining the service interface ("*what* the service does"), the next step is implementing that interface. The implementation is "*how* the service does what it says it does". The methods written in the service are the interface by which users can access and modify the resource values beneath. It can be thought of as a gateway or a "public face" to the service. The class with implementation of the NLP services can be broken down into different sections and methods as follows:

1. Implements the values and resource properties from the namespace interface
2. Has methods for creating of resources
3. Has (private) methods for retrieving of resources
4. Have specific service operations – like *lemma, WordNet or Wikipedia* methods that obtain for a given word its lemma, synonyms or relations from Wikipedia.

```
package uaic.fii.nlp.impl;
import javax.xml.namespace.QName;
public interface NLPQNames {
      public static final String NS = "http://www.globus.org/
                          namespaces/uaic/fii /NLPServices_instance ";
       public static final QName RP_VALUE = new QName(NS, "Value");
       public static final QName RP_WORD = new QName(NS, "Word");
       public static final QName RESOURCE_PROPERTIES = new QName(NS,
                  "LemmaResourceProperties");
}
```

Configuring the deployment in WSDD: Up to this point, the two most important parts of state full Web service are written: the service interface (WSDL) and the service implementation

(Java). This step makes the web service available to the client connections. For that we must take all the loose pieces we have written up to this point and make them available through a *Web services container*. This step is called the *deployment* of the web service. One of the key components of the deployment phase is a file called the *deployment descriptor*. It's the file that tells the Web Services container how it should publish the web service (for example, telling it what our service's URI will be). The deployment descriptor is written in WSDD format (Web Service Deployment Descriptor). The deployment descriptor for our Web service will look like this:

```
<?xml version="1.0" encoding="UTF-8"?>
<deployment name="defaultServerConfig"
    xmlns="http://xml.apache.org/axis/wsdd/"
    xmlns:java="http://xml.apache.org/axis/wsdd/providers/java"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema">

    <service    name="core/NLPService"    provider="Handler"    use="literal"
style="document">
     <parameter name="className" value="uaic.fii.services.NLPService"/>
       <wsdlFile>share/schema/NLPService_instance/NLP_service.wsdl</wsdlFile>
        <parameter name="allowedMethods" value="*"/>
        <parameter name="handlerClass"
value="org.globus.axis.providers.RPCProvider"/>
        <parameter name="scope" value="Application"/>
        <parameter name="providers" value="GetRPProvider"/>
        <parameter name="loadOnStartup" value="true"/>
    </service>
</deployment>
```

Create a GAR file with Ant: At this point we have a service interface in WSDL, a service implementation in Java, and a deployment descriptor in WSDD telling the Web Services container how to present and to the outer world. Using those three files we will generate a *Grid Archive*, or *GAR file*. This GAR file is a single file which contains all the files and information the Web services container needs to *deploy* our service and make it available to the whole world. The creating of a GAR file is a pretty complex task which involves the following:

- Processing the WSDL file to add missing pieces (such as bindings)
- Creating the stub classes from the WSDL
- Compiling the stubs classes
- Compiling the service implementation
- Organize all the files into a very specific directory structure.

This task is performed with Ant. Ant, an Apache Software Foundation[63] project, is a Java *build tool*. It allows programmers to forget about the individual steps involved in obtaining an executable from the source files, which will be taken care of by Ant.

Deploy the service into a Web Services container: The GAR file, as mentioned above, contains all the files and information the web server needs to deploy the web service. Deployment is done with a GT4 tool that, using Ant, unpacks the GAR file and copies the files within (WSDL, compiled stubs, compiled implementation, WSDD) into key locations in the GT4 directory tree.

[63] Apache: http://www.apache.org/

8.7. *Appendix 7 – Terms*[64]

BLEU = **BLEU** (acronym for ***Bil****ingual* ***e****valuation* ***u****nderstudy*) is a method for evaluating the quality of text which has been translated using machine translation.

CD = Comparable Documents = The CD task definition can essentially be characterized as recognition of noisy word-aligned sentence pairs

FOL = **First-order logic (FOL)** is a system of deduction, extending propositional logic (equivalently, sentential logic), which is in turn extended by second-order logic. It is also called **first-order predicate calculus** (**FOPC**).

IE = **Information extraction** (IE) is a type of information retrieval whose goal is to automatically extract structured or semi-structured information from unstructured machine-readable documents. It is a sub-discipline of language engineering, a branch of computer science.

IR = **Information retrieval (IR)** is the science of searching for information in documents, searching for documents themselves, searching for metadata which describe documents, or searching within databases, whether relational stand-alone databases or hypertext networked databases such as the Internet or intranets, for text, sound, images or data. There is a common confusion, however, between data retrieval, document retrieval, information retrieval, and text retrieval, and each of these has its own bodies of literature, theory, praxis and technologies.

MT = **Machine translation**, sometimes referred to by the acronym **MT**, is a sub-field of computational linguistics that investigates the use of computer software to translate text or speech from one natural language to another. At its basic level, MT performs simple substitution of atomic words in one natural language for words in another. Using corpus techniques, more complex translations may be attempted, allowing for better handling of differences in linguistic typology, phrase recognition, and translation of idioms, as well as the isolation of anomalies.

[64] English Wikipedia: http://en.wikipedia.org/wiki/Main_Page

NLP = Natural language processing (NLP) is a subfield of artificial intelligence and linguistics. It studies the problems of automated generation and understanding of natural human languages. Natural language generation systems convert information from computer databases into normal-sounding human language, and natural language understanding systems convert samples of human language into more formal representations that are easier for computer programs to manipulate.

POS[65] = Part-of-Speech = lexical category (also word class, lexical class, or in traditional grammar part of speech) = a linguistic category of words (or more precisely lexical items), which is generally defined by the syntactic or morphological behaviour of the lexical item in question. Schools commonly teach that there are 8 parts of speech in English: noun, verb, adjective, preposition, pronoun, adverb, conjunction, and interjection. However, there are clearly many more categories and sub-categories. For nouns, plural, possessive, and singular forms can be distinguished. In many languages words are also marked for their "case" (role as subject, object, etc.), grammatical gender, and so on; while verbs are marked for tense, aspect, and other things.

QA = Question answering (QA) is a type of information retrieval. Given a collection of documents (such as the World Wide Web or a local collection) the system should be able to retrieve answers to questions posed in natural language. QA is regarded as requiring more complex natural language processing (NLP) techniques than other types of information retrieval such as document retrieval, and it is sometimes regarded as the next step beyond search engines.

SVM = Support vector machines (SVMs) are a set of related supervised learning methods used for classification and regression. Their common factor is the use of a technique known as the "kernel trick" to apply linear classification techniques to non-linear classification problems.

S-V-O = Subject, Verb, Object order

TE = Textual Entailment

[65] POS: http://en.wikipedia.org/wiki/Parts_of_speech

WordNet[66] = WordNet is a semantic lexicon for the English language. It groups English words into sets of synonyms called *synsets*, provides short, general definitions, and records the various semantic relations between these synonym sets.

[66] WordNet: http://wordnet.princeton.edu/