Designing Test-beds for General Anaphora Resolution

Oana Postolache†* and Dan Cristea*♦

†University of Saarland, Saarbrücken, Germany
oana@coli.uni-sb.de

* "Al. I. Cuza" University, Iaşi, Romania
♦Romanian Academy - the Iaşi Branch, Institute of Computer Science, Romania
dcristea@infoiasi.ro

Abstract

This paper proposes a framework for evaluating coreference resolution systems, taking into account the contribution of subcomponents. Our goal is to have a way to quickly identify bottlenecks so that the development effort can focus on the weakest part of the processing chain. We describe experiments on the contribution of the module for searching potential referential expressions using two types of input, plain text and Penn Treebank-style syntactically annotated data. We propose a metric for evaluating a coreference resolution system when the set of potential referential expressions in the input is not the same as the set of potential referential expressions in the gold.

1. Introduction

In more and more NLP applications, such as summarization (Alonso and Fuentes, 2003), information extraction (Gaizauskas and Humphreys, 2000; Luo et al., 2004) and question answering (Harabagiu and Maiorano, 2003), anaphora resolution is reported to be a key component. In all these systems, generally employing a pipe-line architecture, evaluation of the system performances should consider the contribution of each component to the improvement or decrease in the performance of the overall system. Although seen as a component of other larger systems, coreference resolution is not an isolated process, since it is often relying on other more basic processes. To give an example, in its most elementary setting, an anaphora resolution (AR) system uses potential anaphors among the referential expressions (also called markables) that it tries to link into chains of coreferences. But the detection of these markables is a process in itself, which is also prone to errors that can impact the performance of the AR resolution system. There are few attempts to study the performance of an AR-systems in correlation to other more basic processes, such as the automatic detection of markables. The majority of approaches consider that the gold coreference chains and the test coreference chains share the same set of markables. For instance, in a recent approach dedicated to the evaluation of the coreference task, Popescu-Belis et al. (2004) show that all the measures they implemented rely on the hypothesis that the set of markables in both gold and test files is the same.

In this paper we report on new experimental data with a model of coreference resolution that works on two different types of input (plain text and Penn Treebank-style syntactically annotated data). To evaluate the components of an anaphora resolution system we propose a methodology, which takes into account the different subtasks, and aims to quantify the contribution of the component parts to the overall system performance in a pipe-line architecture. Moreover, the same procedure can be applied to the evaluation of larger NLP systems with an AR component.

The structure of the paper is as follows. Section 2 presents the evaluation strategy we propose, Section 3 describes the two types of corpora we have used and how the module of detecting the markables works on them. Section 4 is a brief overview of the coreference resolution system we use. In Section 5 we present the methods of evaluation employed by our test-bed, and in Section 6 we discuss the results and conclude.

2. Evaluation strategy

We believe that the evaluation of a complex NLP system should be done in a way that shows the performance contribution of different components. This way bottlenecks can be identified and the development effort can be focussed on the weakest part of the chain. We have used such a methodology for the first time during the development of an incremental discourse parsing and summarization system. In this paper we express the principle arguments and benefits of such a methodology and show how it is implemented for the evaluation of a coreference resolution system.

We see a system intended to extract the coreference chains as being built out of two modules in a pipe-line: one that detects referential expressions (RE), which are potential anaphors (we will refer to this module as the **RE-extractor**), and one that creates the coreference chains (**AR-engine**).

The input to the RE-extractor can be any kind of text, annotated or not. As we will see in Section 3, the RE-extractor is responsible for detecting the markables, using information from either some pre-processing phase or from manual annotation (if provided). The output of the RE-extractor must be, however, for all types of input, a file conformant with the format accepted by the AR-engine.

The AR-engine is described briefly in Section 4, and in greater detail in (Cristea et al., 2002; Postolache, 2004; Postolache and Forascu, 2004; Cristea and Postolache, in press). The output of the AR-engine is a partition of the input markables into coreference chains.

The quality of the final output depends on the quality of both modules and will manifest successive degradation. To blame, in this context, one module more than the other means to compare their behaviour in a setting in which their results are not influenced by their interaction in a chain.

The proposed test-bed is displayed in Figure 1, where (P,R,F) represent precision, recall and F-measure of the RE-extractor, obtained by comparing the results of the RE-extractor (**re-test**) against the corresponding gold-standard (**re-gold**). Then the results of the coreference module (AR-engine) can be measured on the true markables or on the system output markables computed by the RE-extractor: when it receives the **re-test**, it outputs the **re-test-coref-test**; when it receives the **re-gold**, it outputs the **re-gold-coref-test**. In both cases, the evaluation is done against the **re-gold-coref-gold** (in which both markables and coreferences are gold). We evaluate the coreference module, using the success rate (Mitkov, 2002). On one hand, comparing the completely automatic output of the system against the gold standard we obtain success rate SR1, which gives the overall performance of the system, while comparing the output of the AR-engine, fed with a perfect input, against the gold standard, we obtain success rate SR2, which gives only the performance of the second module, unaffected by the degradation of performance due to the processing of the first module. In this way the amelioration of the performance of the whole system can be correlated with that of its individual modules and bottlenecks can be identified.

However, this scheme is not always easy to implement. Difficulties to do that will be discussed in section 5. In the next section we describe how the RE-extractor operates on two types of input: plain text and text manually annotated.

3. Extracting referential expressions from two types of input

For our experiments, we use data from two types of corpora as input for the RE-extractor:

- A plain text corpus of approx. 19,500 words in 1,966 sentences, extracted from the Orwell's novel "1984" (Orwell, 1949);
- A manually annotated corpus for syntactic structure containing approx. 6,250 words in 281 sentences, extracted from the English Penn Treebank (Marcus et al., 1994).

Our markables are generally conformant with the MUC-7 (Hirschman and Chinchor, 1997) and ACE (ACE, 2003) criteria, although there are some differences. In the following we present the types of mentions that we marked as referential expressions:

- **noun phrases**: definite (*the principle*, *the flying object*), indefinite (*a book*, *a future star*) or undetermined (*sole guardian of truth*), singular and plural, and including names (*Winston Smith*, *The Ministry of Love*), dates (*April*), currency expressions (*$40*) and percentages (*48%*);
- **pronouns:** personal (*I*, *you*, *he*, *him*, *she*, *her*, *it*, *they*, *them*), possessive (*his*, *her*, *hers*, *its*, *their*, *theirs*), reflexive (*himself*, *herself*, *itself*, *themselves*) and demonstrative (*this*, *that*, *these*, *those*);
- **wh-pronouns:** relative pronouns *which*, *who*, *whom*, *whose* and *that* when they replace an entity; when they are part of an interrogative sentence they are not marked as referential expressions;
- **numerals:** when they refer to entities (*four of them*, *the first*, *the second*), but not when they have an adjectival or adverbial function (*He was 29*, *three books*);

The most important differences from the MUC-7 conventions are that our markables do not include relative clauses, each term of an apposition is taken separately ([*Big Brother*], [*the primal traitor*]), conjoined expressions are annotated individually ([*John*] and [*Mary*], [*hills*] and [*mountains*]), and modifying nouns appearing in noun-noun modification are not marked separately ([*glass doors*], [*prison food*], [*the junk bond market*]).

In the following subsections we describe the extraction of the markables for each of the two types of input mentioned.

3.1. The Orwell corpus

The corpus consists of chapters 1, 2, 3 and 5 of Orwell's book.. The text was first POS-tagged, then tagged for dependency structure (Järvinen and Tapanainen, 1997). For detecting the markables, we automatically extracted all structures dominated by a noun or a pronoun, using the FDG-structure. We imposed the constraint that relative phrases are not part of a markable. The heads of the markables were also automatically extracted (the root of each such structure). The output of this module was considered a **re-test** file.

In order to obtain the **re-gold** file, a group of annotators manually went through the output of the NP-detector, looking for errors and eliminating them. They also marked, other markables than NPs or pronouns, which have not been identified by the automatic process.

The FDG annotation was also used to extract additional information needed for the anaphora resolution process such as the lemma of words, the lexical number, and the dependency links between words.

Table 1 shows some statistics on the Orwell corpus.

	Text 1	**Text 2**	**Text 3**	**Text 4**
No. of sentences	311	175	169	328
No. of words	6935	3317	3260	6008
No. of Res	1942	914	916	1702
Average no. of REs per sentence	6.2	5.2	5.4	5.1
Pronouns	645	281	362	614
No. of Des	921	520	464	863

Table 1: Statistics on the Orwell corpus. REs stands for referential expressions and DEs for discourse entities

3.2. The Penn Treebank corpus

The corpus consists of 7 files (wsj_2428, wsj_2430, wsj_2431, wsj_2435, wsj_2438, wsj_2443, wsj_2444). To obtain the markables we firstly automatically extracted (using the Penn Treebank annotation structure): the nouns, the pronouns, and the noun-phrases. We obtained a set of markables for which we imposed some constraints in order to eliminate some of the NPs that didn't correspond to what we considered a referential expression. The most frequent situation when we had to eliminate a markable was in the case of embedded NPs. For example, for the sentence:

The market for $200 billion of high-risk junk bonds, battered by a succession of defaults and huge price declines this year, practically vanished Friday.

we extracted automatically the following markables:

0: The market
1: $ 200 billion
2: high-risk junk bonds
3: $ 200 billion of high-risk junk bonds
4: The market for $ 200 billion of high-risk junk bonds
5: a succession
6: defaults
7: a succession of defaults
8: huge price declines
9: this year
10: huge price declines this year
11: a succession of defaults and huge price declines this year
12: Friday

The Penn Treebank annotation is done in such a way that, for example, given the construction **A of B**, then **A** is a NP, **B** is an NP and also **A of B** is an NP. The same thing happens also with other types of embedded NPs. Conforming to the view that REs should reflect discourse entities, our belief is that **A** alone should not be considered an RE and we rule it out such that only two of these markables remain, namely **B** and **A of B**.

Thus, for the above example, we consider as markables the following phrases:

2: high-risk junk bonds
3: $ 200 billion of high-risk junk bonds
4: The market for $ 200 billion of high-risk junk bonds
6: defaults
7: a succession of defaults
8: huge price declines
9: this year
12: Friday

We also eliminated the relative clauses from markable spans and we considered the terms of an apposition construction as being two REs (in the Penn Treebank the whole apposition construction is also considered a NP). After running these scripts that eliminated some erroneous markables we obtained the corresponding **re-test** files.

As in the case of the Orwell corpus, annotators corrected the errors and introduced new markables in order to obtain the **re-gold** files.

After detecting the markables we automatically enriched the annotation with basic information needed for coreference:

- choosing the head of the markables: we have used a similar approach as the one described in (Collins, 1999). We consider all the words on the first level of the NP (eliminating the nested NPs) and then search from right to left for the first word that has the part-of-speech equal with NN, NNS, NP, NPS, CD, POS, or JJS[1].
- finding the dependency links between words: we consider that every word is linked to the head of the phrase in which it belongs. We use Collins' rules for detecting the head of the phrases, and then, for each word that is part of a phrase (but not in the head position) the link indicates the head. For a word that is the head of a phrase, the link indicates the phrase parent's head. Some words - the predicative verbs - have no link.
- finding the lemma (stem) of words: we use the Wordnet2.0 API in order to extract the stem of words.

Table 2 shows some statistics on the Penn Treebank corpus we used.

	file 1	**file 2**	**file 3**	**file 4**	**File 5**	**file 6**	**file 7**
No. of sentences	83	9	47	10	19	63	50
No. of words	1837	192	1111	150	534	1483	937
No. of Res	533	65	344	52	165	423	262
Average no. of REs per sentence	6.4	7.2	7.3	5.2	8.6	6.7	5.2
Pronouns	44	5	29	4	11	35	8
No. of Des	483	49	242	30	110	392	240

Table 2: Statistics on the Penn Treebank corpus

4. The AR-engine and the coreference model

Our coreference model is integrated in the anaphora resolution framework of (Cristea and Dima, 2001). Anaphoric references in this approach are processed on a three-layered representation. The text layer retains the markables (REs) from the surface text. Below it, on the projection layer, each markable is represented as a projected structure (PS) with all features that are relevant for the resolution process. Deeper, on the semantic layer, the discourse entities (DEs) are represented. One processing cycle of the engine deals with the resolution of one RE and goes through three compulsory phases and an optional one: during the **projection phase** the current PS is built on the projection layer using the information centred on the current RE obtained from the text layer; the **proposing/evoking phase** is responsible for matching the current PS with one DE, by either proposing a new discourse entity or deciding on the best candidate from the existent ones; in the **completion phase**, the data contained in the resolved PS is combined with the data configuring the found referent; sometimes, an optional **re-evaluation phase** is triggered to resolve postponed PSs left on the projected layer (Cristea and Postolache, in press). Any model under this processing paradigm is seen as a quadruple: **a set of primary attributes** characterising the descriptions on the three layers, **a set of knowledge sources** capable of filling values for the primary attributes of the PSs on the projection layer with information gathered from the text layer, **a set of heuristics or rules** intended to answer one or both of the following two questions, in this order: (1) Does a PS introduce a new discourse entity? (2) If not, which of the existing DE does it co-refer with?, and a **domain of referential accessibility**, which can be implemented also as a set of heuristics/rules able to filter and order the list of candidates DEs. In accordance with most authors, three

[1] NN = noun, singular or mass; NNS = noun, plural; NP = proper noun, singular; NPS = proper noun, plural; CD = cardinal number; POS = possessive ending; JJS = adjective, superlative.

types of rules are involved in the proposing/evoking phase: **demolishing rules,** which describe hard constraints prohibiting a coreference link if certain conditions apply, **certifying rules**, which license a coreference link regardless of other criteria, and **promoting/demoting rules**, which increase/decrease a score contributed by a certain attribute and associated with a pair (PS, DE). In our model component one consists of the following set of primary attributes, associated with REs and further projected onto PSs and DEs: `lemma` (the lemma), `number` (lexical number), `pos` (part-of-speech), `role` (the syntactic role of the RE in the sentence), all of these characterizing the head word of the corresponding RE, `link` (syntactic dependency), `npText` (the span of text covered by the RE), `includedNPs` (the set of nested REs in the current RE), `isDefinite` (yes, if the RE is a definite markable), `isUndefinite` (yes, if the RE is an indefinite markable), `isMaleName`, `isFemaleName`, `isFamilyName` (with the evident meaning, decided based on big files with male, female and family names), `HeSheItThey` (the probability of the corresponding RE to be referred to by a *he*, *she*, *it* or *they*, computed from WordNet), `offset` (from the beginning of the text), `sentenceID` (the ID of the sentence the RE belongs to), `isPerson` (if it is a person or not according to WordNet), `predNameOf` (the ID of the subject in case the RE is a predicative noun), `headForm` (the form of the head – not the lemma – in order to distinguish between *they* and *themselves*). The second component implements three types of rules. The demolishing rules section, intended to rule out a possible DE as referent candidate of a PS, includes just one rule: `IncludingRule` which prohibits coreference between nested REs. The certifying rules, which if evaluated to 'true' on a pair (PS, DE) certify without ambiguity the DE as a referent of the PS, include `PredNameRule` and `ProperNameRule`. `PredNameRule` says that if the current RE is on the position of a predicative noun then that DE corresponding to the RE on the position of the subject is certified as an antecedent. In establishing this coreference link between predicative noun and subject we deliberately disregarded in this phase of our model relative identities as state transitions, eventualities, etc. `ProperNameRule` certifies as antecedent a DE displaying the same proper name as the current RE. Promoting/demoting rules are applied after the certifying and demolishing rules and increase/decrease a resolution score associated with a pair (PS, DE). In our model we implemented 8 such rules. `HeSheItTheyRule` gives the probability that the candidate DE can be referred by the current RE, a *he*, *she*, *it* or *they* pronoun. `RoleRule` distinguishes cases of very close scores among antecedents. It returns values between 0 and 1 according to the syntactic role of the antecedents, where the role ranking is consistent with the ranking used in the Centering Theory (Grosz et al, 1995), namely subject > direct object > indirect object > attributive. `NumberRule` prefers antecedents agreeing in the `number` attribute with the current RE. `LemmaRule` is the place where constraints about the word components of the NPs are implemented. `PersonRule` implements the preference of pronoun REs like *themselves* to point person-type antecedents. Besides these, three other rules implement semantic and lexical distance preferences based on WordNet. `SynonymyRule` uses the lemmas of its arguments (a PS and a DE) and interrogates the WordNet in order to find a synset that contains both lemmas. If such a synset is found, the rule returns 1, else 0. `HypernymyRule` is similar to the `SynonymyRule` only that it looks into hypernyms instead of synonyms. `WordnetChainRule` looks in WordNet for a hypernymic or hyponymic chain between the `lemma` attribute of its first argument, the PS, and each of the lemmas of its second argument, the DE. If such a chain is found, the rule return 1, else 0. Since it is possible for the two lemmas, corresponding to the anaphor and candidate antecedent, to have different meanings in the text (in which case the PS corresponding to the anaphor RE does not refer the candidate DE), we attached a relatively low weight to all the WordNet accessing rules. This way an accidental synonymy, hypernymy or chain matching will not influence drastically the final score, but it could be exploited when conjoined with other rules. Also, since there are cases when a DE has a set of different values for an attribute, because it incorporates the values triggered from all the REs that refer to it, a scoring rule tests the inclusion of the value of a PS attribute within the set of values of the corresponding DE attribute. For more details about the coreference model, its attributes and rules see also (Postolache and Forascu, 2004).

5. Evaluation

Figure 1 displays the processing chain for a coreference resolution task, the input, the output and the intermediate results, as well as the gold files against which the results of the processing steps are compared. The evaluation results are shown between the two main processing steps, performed by the RE-extractor and the AR-engine, as well as at the end of the processing chain. As can be seen in this figure, to evaluate the overall process of coreference resolution and the intermediary steps, three types of evaluations have been made:

1. the module for the extraction of the markables (RE-extractor) is evaluated;
2. the coreference module is evaluated when both the test file and the gold file have the same sets of markables (the gold set);
3. in order to measure the influence of the RE-extractor over the overall process, the coreference model is evaluated by comparing a test file that has as markables the output of the RE-extraction module against the gold coreferences file build on the gold set of markables.

In what follows we discuss methods of evaluation for these three steps.

5.1. Evaluating the referential expressions extractor

For both types of the input (plain text data and syntactically annotated data) the RE-extractor produces the same format accepted by the AR-engine. One of the constraints that the AR-engine imposes on its input is that markables should have set the head and also there should not exist two different markables having the same head.

We evaluated the RE-extractor, comparing the **re-test** files against the **re-gold** files. In Figure 1, the results are displayed in terms of precision (P), recall (R) and F-measure (F) for both types of input. We have used two techniques for evaluation: **head matching**, where two markables were considered to match if they have the same

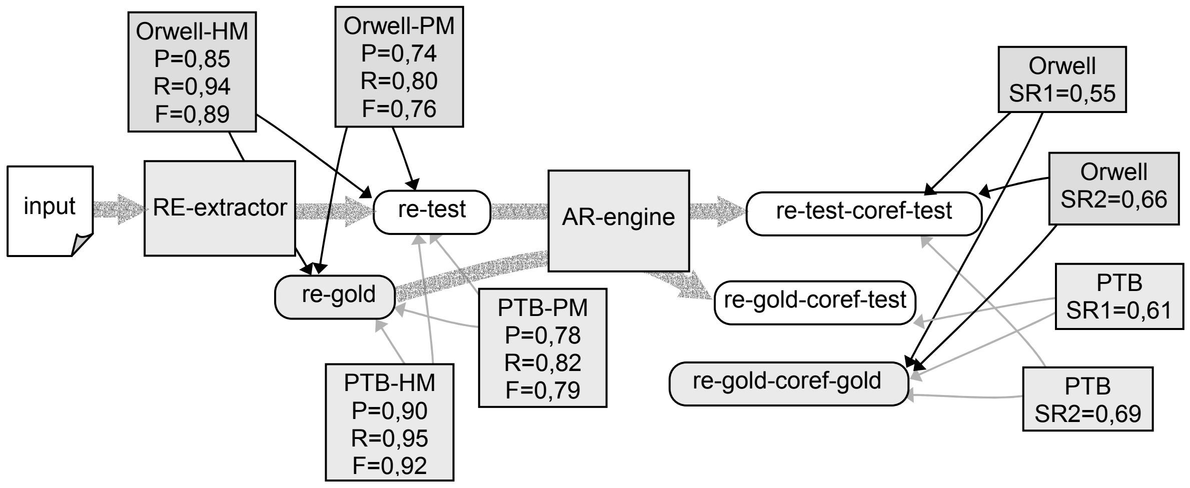

Figure 1: An example of processing and its evaluation points: Orwell stands for the "1984"corpus, PTB – for Penn Treebank corpus, HM – for head matching, PM – partial matching, SR1 is the Success Rate for coreference resolution when the sets of markables are different and SR2 is the Success Rate when the sets of markables are the same.

head, and we computed precision, recall and F-measure; and **partial matching**, in which we considered that two markables match if they have the same head and **the minimal mutual overlap** was higher than 50%. The overlapping score was computing dividing the number of common words over the size of the longest markable out of the two. The results are shown in Figure 1, where the dark squared diagrams have titles that indicate the original corpora and the type of evaluation. They show, as expected, that when we imposed a lower bound of 50% on the mutual overlap of markables the performance of the extractor is poorer.

5.2. Evaluating the AR-engine

The metric used to evaluate the AR-engine was based on the success rate (the ratio between the number of correctly resolved anaphors and the number of all anaphors) defined in (Mitkov, 2002). We have considered each of the referential expressions that we have marked as a potential anaphor and instead of deciding whether a certain anaphor was correctly resolved or not, for each anaphor we assign a **correctness value** between 0 (meaning incorrectly solved) and 1 (meaning correctly solved). We explain below how this value is computed when the sets of markables in the text file and the gold file are identical.

Since our output consists of coreference chains (sets of anaphors which refer to the same entity), we must compare *the gold set* of chains (obtained from the manually annotated corpus) with the system output chains (we will call it *the test set*). For each anaphor:

- if, in the gold set, it belongs to a chain that doesn't contain any other anaphor, then we look in the test set to see if it belongs to a similar trivial chain, in which case it will get the value 1; else (the test chain corresponding to the current anaphor contains more than one anaphors) it will get the value 0;
- if, in the gold set, the anaphor belongs to a chain containing other *n* anaphors, then we look in the test set and count how many of these *n* anaphors belong to the chain corresponding to the current anaphor (we note this number with *m*). The ratio *m/n* will be the value assigned to the current anaphor.

We will then sum up these values for the whole set of anaphor and divide by the cardinality of this set.

For the third type of evaluation needed, when the sets of markables in the test file and the gold file are different, there are three possibilities, as follows:

- there is a common set of markables (on the identity of head criterion), but which could possibly cover different spans of text;
- in the test there are markables which have no corresponding markables in the gold standard (false alarms) and vice-versa (misses);
- the test file and the gold file have different chains of markables which refer to the entities in the text, different with respect not only to the partition of the common markables, but also with respect to the markables of the chains (for example, the test chains may contain some markables that are not even considered as markables in the gold).

We described above the way of evaluating the coreference chains when the two sets of markables are the same. To extend this method to the case when the two sets of markables are the same (on the identity of head criterion) but the markables' spans are different, we adjust the correctness score as follows. In counting the scores corresponding to trivial chains anaphors (a singleton coreference chain consisting of a single referential expression) we replace the 1 values in the sum (meaning correctly solved) with the ratio between the number of common tokens and the length of the longest markable, giving a **mutual overlapping score** between the two markables. In counting the scores corresponding to anaphors belonging to non-trivial chains (longer than one markable), the *m* value is no more the number of common markables in the test and gold chain, but the sum of the mutual overlapping scores (as above) between markables belonging to test and gold chains.

For the other cases, when there are differences between the two sets of markables (also identified by the identity of head criterion), in the computation of the correctness score, the non-intersecting markables in the test and gold files will not contribute to the sum, thus deteriorating the overall rating.

6. Conclusions

Our results show (as we expected) that the overall performance of the AR system is affected by the performance of the previous modules. In the case of a coreference resolution system, the main point, which can influence the results, is the detection of the markables. In our test-bed, we obtained a difference of 0,11 (in the case of the plain text input), and 0,09 (in the case of the annotated corpus) between the performance of the AR-engine when run with gold markables and the performance of it when run with test markables. For sure the most part of the blame must be put on the RE-detector, but not only: other pre-processing phases may also have 'helped' at the deterioration of the final score. For example, all models of AR rely heavily on the information extracted from the heads of the markables, that means that the script for finding the head of a markable also have a responsibility.

An interesting thing to observe is the difference between the values obtained for the two types of corpus we worked with. In the case of the plain text we obtained smaller values when evaluating the RE-extractor, than in the case of the manually annotated treebank. One explanation of this fact is enforcing our belief: we have used an automatic pre-processing phase (the FDG-parser) in order to obtain the syntactic trees. For sure, the output of the FDG-parser is not perfect, or at least it doesn't have the same quality as the manually annotated corpus. So this weak point in the extraction of the markable (in the case of the plain text corpus) had impact on the performance of the RE-extractor.

Also the performance of the AR-engine is better in the case of the Penn Treebank corpus. At least two reasons can be thought of: 1). the annotation is better; 2). Penn Treebank consists of newspapers and the task of coreference resolution on newspapers seems to be easier than on belles-lettres register.

The original work reported in this paper is the following:

- it proposes a methodology to evaluate pipe-line architectures when the gold and test data are available in-between intermediate steps in the processing chain. The method allows to appreciate the contribution of individual modules on the overall results irrespective of the depreciation of the results due to the weakness of the preceding modules;
- it reports and compares new anaphora resolution results on input belonging to two different registers: belles-lettres and finance, and to two different levels of input: plain-text and treebank annotation.

Acknowledgements

Part of this work has been accomplished under the IST-2000 29388 EU project Balkanet and with a support from the Romanian Ministry of Education and Research under the CORINT programme. We are grateful to our colleagues from the Laboratory of Computational Linguistics of the University of Wolverhampton, who have provided us the FDG parsing of the Orwell text. We would also like to thank Dr. Nicolas Nicolov for advice, suggestions and comments on an earlier draft of the paper.

References

ACE 2003. Entity Detection and Tracking – Phase 1, ACE Pilot study definition available from the ACE site http://www.itl.nist.gov/iaui/894.01/tests/ace/phase1/index.htm.

Alonso, L. and M. Fuentes, 2003. Integrating Cohesion and Coherence for Text Summarization. *Proceedings of the EACL Student Session*, Budapest, Hungary.

Collins, M., 1999. *Head-Driven Statistical Models for Natural Language Parsing*. PhD Thesis, University of Pennsylvania.

Cristea, D. and G.E. Dima, 2001. An Integrating Framework for Anaphora Resolution. *Information Science and Technology,* Bucharest 4:3-4, 273-291.

Cristea, D, O. Postolache, G.E. Dima and C. Barbu, 2002. AR-Engine – a framework for unrestricted coreference resolution. *Proceedings of LREC 2002*, Las Palmas, vol. VI, 2000-2007.

Cristea, D. and O. Postolache, 2004. How to deal with wicked anaphora. In A. Branco, T. McEnery and R. Mitkov (eds), *Anaphora Processing: linguistic, cognitive and computational modeling*. Amsterdam: John Benjamins (to appear).

Gaizauskas, R. and K. Humphreys, 2000. Quantitative Evaluation of Coreference Algorithms in an Information Extraction System. In S. Botley and T. McEnery (eds.), *Corpus-based and Computational Approaches to Discourse Anaphora*. Amsterdam/New-York: John Benjamins.

Harabagiu, S. and S. Maiorano, 2003. Relational Meaning Used for Processing Complex Questions. *Proceedings of the International Symposium on Reference Resolutionand Its Applications to Question Answering and Summarization*, Venice, Italy.

Hirschman, L. and N. Chinchor, 1997. MUC-7 Coreference Task Definition, version 3.0. *MUC-7 Proceedings*. See also: http://www.muc.saic.com.

Luo, X., A. Ittycheriah, H. Jing, N. Kambhatla, S. Roukos, 2004. A Mention-Synchronous Coreference Resolution Algorithm Based on Bell Tree. Proceeding of ACL, Barcelona, Spain.

Järvinen, T. and P. Tapanainen, 1997. A dependency Parser for English. *The Technical Reports of the Department of General Linguistics*, University of Helsinki.

Marcus, G., G. Kim, M.A. Marcinkiewicz, R. MacIntyre, A. Bies, M. Ferguson, K. Katz and B. Schasberger, 1994. The Penn Treebank: Annotating predicate argument structure. *Proceedings of Human Language Technology Workshop*.

Mitkov, R., 2002. *Anaphora resolution*. London: Longman.

Orwell, G., 1949. *1984*. London: Secker and Warburg.

Popescu-Belis, A., L. Rigouste, S. Salmon-Alt and L. Romary, 2004. Online Evaluation of Coreference Resolution. *Proceedings of LREC'04*. Lisbon, Portugal

Postolache, O., 2004. *RARE- Robust Anaphora Resolution Engine*. Master Thesis, University of Iaşi.

Postolache, O., C. Forascu, 2004. A Coreference Resolution Model On Excerpts from a Novel. Proceedings of ESSLLI Student Session, Nancy, France.