A Modular Approach to Learning Dutch Co-reference

Véronique Hoste
CNTS-Language Technology Group
University of Antwerp, Belgium
veronique.hoste@ua.ac.be

Antal van den Bosch
Language & Information Science
Tilburg University, The Netherlands
antal.vdnbosch@uvt.nl

Abstract

This paper presents the first machine learning approach to the resolution of co-referential relations between nominal constituents in Dutch. Based on the hypothesis that different types of information sources contribute to a correct resolution of different types (pronominal, proper noun and common noun) of co-referential links, we propose a modular approach in which a separate module is trained per NP type. We present a thorough comparison of two machine learning techniques, a lazy learner and an eager learning approach, trained on the modular tasks as well as on the undecomposed task. In addition, we show that by post-processing the resulting co-reference chains by means of a string-edit distance correction mechanism, we can avoid some unlikely local chainings and thereby improve precision. Lacking comparative results for Dutch, we also report results on the English MUC-6 and MUC-7 data sets, which are widely used for evaluation.

1 Introduction

This paper is concerned with the automatic resolution of co-reference using machine learning techniques. Co-reference can be considered as the act of using a referring expression to point to some discourse entity. Written and spoken texts contain a large number of co-referential relations and a good text understanding largely depends on the correct resolution of these relations. Machine learning approaches have become increasingly popular for this problem. In a typical machine learning approach to co-reference resolution, information on pairs of NPs is represented in a set of feature vectors. Unsupervised learning techniques, e.g. Cardie and Wagstaff (1999), view co-reference resolution as a clustering task of combining noun phrases into equivalence classes. Most learning approaches to co-reference resolution, however, are supervised techniques, which make use of the C4.5 decision tree learner (Quinlan 1993) as in Aone and Bennett (1995), McCarthy (1996) and Soon, Ng and Lim (2001) or the RIPPER rule learner (Cohen 1995) as in Ng and Cardie (2002).

In this paper, we aim to investigate the following issues. In a set of cross-validation experiments, we investigate whether the use of a modular approach in which one learning system is developed separately for pronouns, proper nouns, and common nouns, performs better than a global approach in which the learner is trained on all available data. Furthermore, whereas most existing learning approaches to co-reference resolution can be described as eager learning approaches, we investigate in this paper how a lazy learning approach tackles the problem of co-reference resolution. We present a comparison of two machine learning techniques on the task of co-reference resolution: the lazy learning implementation

TIMBL (Daelemans, Zavrel, van der Sloot and van den Bosch 2002) [1] and the eager rule induction method RIPPER (Cohen 1995). The learning biases of these two approaches provide extremes in the *eagerness* dimension in ML (the degree to which a learning algorithm abstracts from the training data in forming a hypothesis). Based on the observations reported in (Hoste, Hendrickx, Daelemans and van den Bosch 2002, Daelemans and Hoste 2002, Daelemans, Hoste, De Meulder and Naudts 2003a, Decadt, Hoste, Daelemans and van den Bosch 2004, Hoste 2005), which show that a proper comparative experiment requires extensive optimization and that the performance increase obtained by this optimization is considerable, we perform an extensive feature selection and parameter optimization on both learning methods using a genetic algorithm. These results are contrasted with the default results of both learners.

We pay additional attention to correcting errors in the co-reference chains produced by the system. Since decisions on chaining elements are made pairwise, without coordination between different decisions for other pairs that may or may not be part of the same larger chain, a full chain can contain some stray elements that need to be deleted from the chain. We develop a post-processor that compares each produced chain to a gold-standard lexicon of chains from the training material, finds a nearest match in terms of Levenshtein distance, and decides to delete certain stray elements if the nearest match suggests such a deletion. This correction method, adopted from spelling correction, assumes that chains are represented as a sequence of symbols from a small alphabet. We show that a simple alphabet encoding the three different types of NPs and intermediate non-chaining elements as four different symbols attains a modest but noticable boost of 2.5 points on the system's precision.

The remainder of the paper is organised as follows. Section 2 introduces the annotated corpora for Dutch and English. It discusses the preparation of the data sets, including the preprocessing of the data, the selection of positive and negative instances and the construction of three different NP-type data sets instead of one single data set. Section 3 deals with the problem of selection of information sources. Section 4 presents and evaluates the two machine learning packages which we used in our experiments. Both default and optimized results are provided for the cross-validation data. Section 5 is devoted to the testing procedure and results. Section 6 reports on our experiments with Levenshtein-based chain correction post-processing. We conclude with some general observations in Section 7.

2 Construction of the data sets

For the experiments, we selected all noun phrases in the English MUC-6 (MUC-6 1995) and MUC-7 (MUC-7 1998) corpora and the Dutch KNACK-2002 corpus (Hoste 2005). For both MUC-6 and MUC-7, 30 documents annotated with co-reference information were used as training documents. The MUC-6 and MUC-7 training sets contain 1,644 and 1,905 anaphoric NPs, respectively. The test sets for

[1] Available from http://ilk.uvt.nl

MUC-6 and MUC-7 contain 30 documents (1,627 anaphoric NPs) and 20 documents (1,311 anaphoric NPs), respectively.

Lacking a substantial Dutch corpus of anaphoric relations between different types of NPs, including named entities, definite and indefinite NPs and pronouns, we annotated a corpus ourselves. The corpus is based on KNACK, a Flemish weekly magazine with articles on national and international current affairs; 267 documents were annotated. For the experiments, 50 texts were used, of which 25 for training and 25 for testing. The KNACK-2002 training and test set contain 1,688 and 1,326 anaphoric NPs, respectively.

2.1 Preprocessing

The following preprocessing steps were taken. First, tokenisation was performed to split non-abbreviating punctuation from word tokens. The tokenization for both Dutch and English was performed by a rule-based system using regular expressions. For the recognition of names, a memory-based named entity recognition approach (De Meulder and Daelemans 2003) was used, which distinguishes between persons, organisations and locations. Part-of-speech tagging and text chunking was performed by the memory-based tagger MBT (Daelemans, Zavrel, van den Bosch and van der Sloot 2003b). For the grammatical relation finding which determines which chunk has which grammatical relation to which verbal chunk (e.g. subject, object, etc.) a memory-based relation finder was used (Buchholz 2002, Tjong Kim Sang, Daelemans and Höthker 2004) for both languages. For Dutch we also performed a machine learned morphological analysis (De Pauw, Laureys, Daelemans and Van Hamme 2004). Dutch features a more extensive inflection, conjugation and derivation system than English. An example is the use of diminutive suffixes, such as the "tje" in "bureautje" (English: "small office"). Furthermore, also compounds had to be split into their components. Compounding in Dutch can occur through concatenation as in "pensioenspaarfonds" (English: "pension saving fund") and through concatenation in combination with the /s/ infix as in "bedrijfsstructuur" (English: "company structure") or in combination with the /e<n>/ infix as in "studentenorganisatie" (English: "student organization").

The information obtained through this preprocessing was then used in the construction of the feature vectors for the learning techniques. On the basis of the preprocessed texts, we selected positive and negative instances for the training data. Positive instances were generated by combining each anaphor with each preceding element in the co-reference chain (a set of noun phrases referring to the same discourse entity). The negative instances were built by combining each anaphor with each preceding NP which was not part of any co-reference chain, as well as by combining each anaphor with each preceding NP which was part of another co-reference chain. This yielded a highly skewed data set. For example, out of the 171,081 training instances in the MUC-6 data, only 6.6% were positive ones.

2.2 One vs. three

Besides merging all NPs into one single train and test set (as for example Ng and Cardie (2002) and Soon et al. (2001)), we also built 3 smaller datasets, each specialized in one NP type (pronouns, proper nouns, and common nouns). This resulted in a learning system for pronouns, one for named entities and a third system for the other NPs.

The main motivation for this approach is that different information sources play a role in the resolution of pronominal references than for example in the resolution of references involving proper nouns. Example sentence (1) illustrates the importance of string matching or aliasing in the resolution of proper nouns. These features are less important for the resolution of the first co-referential link between a pronoun and a common noun NP in example (2), but here information on gender, number and distance is crucial.

(1) **Eastern Air** Proposes Date For Talks on Pay-Cut Plan. **Eastern Airlines** executives notified union leaders (...) By proposing a meeting date, **Eastern** moved one step closer toward reopening current high-cost contract agreements with its unions.

(2) **Union representatives who could be reached** said **they** hadn't decided whether **they** would respond.

In order to test our hypothesis that three classifiers, each trained on one specific NP type, would perform better than one single classifier, we built the data sets displayed in Table 1. The 'Pronouns' data set contains the NPs ending in a personal, reflexive or possessive pronoun. The 'Proper nouns' data set contains the NPs which have a proper noun as head, while the 'Common nouns' data set contains all other NPs not in the two other categories. The fourth dataset is the union of the former three datasets.

Additional motivation for the construction of three different data sets was found in the results reported by Ng and Cardie (2002) and Strube, Rapp and Müller (2002). Ng and Cardie (2002) calculated the performance of their system on pronouns, proper nouns, and common nouns, and observed a low precision on common noun resolution (antecedents were found for many non-anaphoric common nouns) and a high precision on pronoun and proper noun resolution. A similar observation was reported by Strube et al. (2002) when experimenting with applying the C5.0 decision-tree induction algorithm (Quinlan 1993) to a corpus of German texts. Their results show that the feature describing the form of the anaphor (definite NP, indefinite NP, personal pronoun, demonstrative pronoun, possessive pronoun, proper noun) is the most important. They furthermore show that the classifier performs poorly on definite NPs and demonstrative pronouns, moderately on proper nouns and quite good on personal pronouns and possessive pronouns. These different error rates reported for the different types of NPs are an additional motivation for building more fine-grained data sets for each NP type.

Table 1: Number of instances per NP type in the MUC-6/7 and KNACK-2002 training corpora.

	MUC-6	
NP type	positive	negative
Pronouns	2,006	26,811
Proper nouns	5,901	68,634
Common nouns	3,359	64,370
Complete	11,266	159,815

	MUC-7	
NP type	positive	negative
Pronouns	2,705	28,952
Proper nouns	3,455	54,109
Common nouns	2,655	60,009
Complete	8,815	143,070

	KNACK-2002	
NP type	positive	negative
Pronouns	3,111	33,155
Proper nouns	2,065	31,370
Common nouns	1,281	31,394
Complete	6,457	95,919

3 Information sources

3.1 Selected information sources

Several information sources contribute to a correct resolution of co-referential relations: morphological, lexical, syntactic, semantic and positional information and also world-knowledge. In order to come to a correct resolution of co-referential relations, existing systems, e.g. (Fisher, Soderland, Mccarthy, Feng and Lehnert 1995, Cardie and Wagstaff 1999, Soon et al. 2001, Strube et al. 2002), use a combination of these information sources. For our co-reference resolution system, we used a common set of features employed by most of the other machine learning resolution systems, but we will also introduce some new features, especially semantic features, such as the hypernym and synonym features. This focus on semantic features has also been recently reported in for example (Markert and Nissim 2005) and (Ji, Westbrook and Grishman 2005). We will now continue with a short description of the features used in the experiments. For a more detailed description we refer to (Hoste 2005).

The feature vectors consist of a combination of positional features indicating the number of sentences or NPs between the anaphor and its possible antecedent; morphological and lexical features, such as features which indicate whether a given anaphor, its candidate antecedent or both are pronouns, proper nouns, demonstrative or definite NPs; syntactic features which inform on the syntactic function of the anaphor and its candidate antecedent and check for syntactic parallelism; string-matching features which look for complete and partial matches; and, finally, several semantic features.

For the semantic features, we took into account gazetteer lists with location names, male and female person names and names of organizations. Furthermore, we looked for female/male pronouns and for gender indicators such as 'Mr.', 'Mrs.' and 'Ms.'. Further information for this feature for the two English corpora was also extracted from the WordNet1.7 (Fellbaum 1998) synonyms and hypernyms. This synonym and hypernym information is provided for each different sense of the given input word, which is often ambiguous. In case of such an input word with more than one possible sense, there were two possible options. The first option was to use word sense disambiguation (WSD) to determine the contextual meaning of a given noun (see for example Hoste et al. (2002)) and to look for a synonym for this specific meaning of the noun. Due to the rather low accuracies on unrestricted word sense disambiguation for English in the Senseval-2 and Senseval-3 tasks, however, we decided not to use WSD for the construction of the semantic features. For Senseval-2, for example, an official top score of 69.0% precision and recall (Mihalcea 2002) was obtained. For Senseval-3, our own WSD system (Decadt et al. 2004) outperformed all other systems, but it only reached a top performance of 65.2% precision and recall, which was merely 2.8% better than the WordNet most frequent sense baseline. Therefore, we decided to leave the word ambiguous, and tried to exploit this ambiguity in the construction of the semantic features.

The following semantic features were used for English:

- **semantic class of anaphor/antecedent (ambiguous)** represents a concatenation of all semantic classes the anaphor or antecedent belong to. For example, the noun 'Washington' can be a person, an organization and a location.

- **semantic class of anaphor/antecedent (most frequent)** gives the most frequent semantic class. E.g. the noun 'Washington' is more frequently used as a location.

- **semantic class agreement** (values: 'male', 'female', 'incomp', 'person', 'object', 'date', 'loc', 'no', 'na'). If the matched constituents are both male or both female, the value of this feature is set to 'male' and 'female', respectively. If one of the constituents is of the male gender, whereas the other constituent is female, or vice versa, the feature is set to 'incomp'. If both NPs are persons, but it is not possible to determine the gender of one of the NPs or of both, the feature takes as value 'person'. If both constituents are an object, a date or a location, the feature is set to 'object', 'date' and 'loc', respectively. If both NPs do not agree on one of the preceding categories, the feature value is set to 'no'. If it is not possible to determine the semantic class of one or both of the constituents, the feature takes as value 'na'. Since this feature already encapsulates gender information, we decided not to use a distinct feature for gender.

- **synonym** (values: 'yes', 'no') looks for a noun in the anaphoric NP with the same meaning as its possible antecedent. The following pairs from the MUC-7 cross-validation data are examples of NPs which are labeled as synonymic: "the three recent crashes" and "accidents"; "noise" and "a brief unidentified sound".

- **hypernym** (values: 'yes', 'no'). A noun is a hypernym of another noun if the concept it denotes is a superconcept of the concept the other noun denotes. The following pairs from the MUC-7 cross-validation data are examples of hypernymic NPs: "the fighter" and "the aircraft"; "the heavy-lift helicopter" and "the craft"; "cockpit" and "that area".

- **named entity agreement** (values: 'I-ORG', 'I-PER', 'I-LOC', 'no'). This feature identifies the exact agreement of the named entity type (organization, person, location) of both NPs.

For Dutch, similar gazetteer list information could be used for the construction of the first five semantic features for the proper nouns data. For the common nouns, however, we were not able to trace a resource which could provide this type of information. Lacking this type of information for the common noun NPs, we used the Celex lexical data base (Baayen, Piepenbrock and van Rijn 1993) instead to provide gender information for the head nouns of the common noun NPs. There are three basic genders in Dutch: male, female and neuter. In addition, CELEX also names female nouns which can be treated as male and nouns of which the gender depends on the context in which they are used. This makes five feature

values with gender information: ‘male’, ‘female’, ‘neuter’, ‘female(male)’, ‘male-female’.

For the extraction of the synonym and hypernym features, we used all synonyms and hypernyms in the Dutch EuroWordNet[2]) output. The final semantic feature, checking for named entity agreement, is based on the output of the Dutch named entity recognizer. These three semantic features can take the same values as the corresponding English features.

For none of our resolution systems we took into account discourse knowledge (e.g. information on centering or focus, pointing at the most salient element in discourse), or real-world knowledge.

3.2 Feature informativeness

We also calculated the informativeness of the different features. Table 2 shows the previously discussed features for one potential antecedent pair in the sentence

(3) Frans Rombouts verdwijnt als hoofd van de Post (...) Zeker bij de Waalse socialisten was hij niet erg geliefd meer.
Translation: *Frans Rombouts leaves as head of the Post. Especially among the Walloon socialists he lost popularity.*

The last column in the table represents the gain ratio values of each feature calculated on the KNACK-2002 corpus for the pronouns. Gain ratio (Quinlan 1993) is a feature-weighting metric which estimates for each feature, on the basis of the training set, how much it contributes to predicting the class labels. This is estimated by computing two entropy measurements; one of the full data set, and one weighted average of the entropies of all subsets partitioned on all occurring values at the particular feature. In order to avoid that features with many possible values are favoured above features with fewer values, the entropy of the feature values (expressed by split info $si(i)$) is used as a counter-factor:

$$w_i = \frac{H(C) - \sum_{v \in V_i} P(v) \times H(C|v)}{si(i)}$$

$$si(i) = -\sum_{v \in V_i} P(v) log_2 P(v)$$

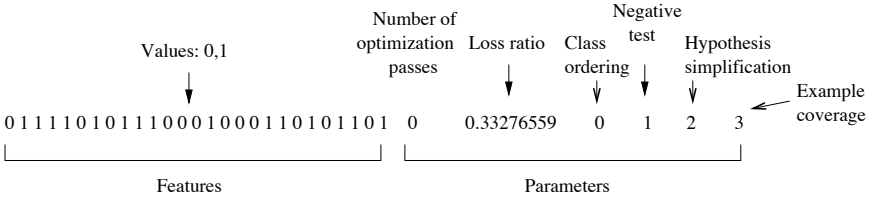
Due to the calculation of the gain ratio on pronominal anaphors only, some features typically designed for proper noun and common noun anaphors have a gain ratio of zero. Furthermore, the string-matching features and the semantic class agreement features are assigned the highest gain ratio values.

[2]EuroWordNet: `http://www.illc.uva.nl/EuroWordNet/`

Table 2: Feature vector for the combination of the anaphor "hij" with its candidate antecedent "Frans Rombouts". The last column gives the gain ratio for each feature calculated on the basis of the complete KNACK-2002 training corpus for the pronouns. Boldface weights represent the four highest weights.

Feature	**value**	**gain ratio** $\times 100$
distance in number of sentences:	3	0.32
distance in number of NPs:	6	0.39
left_wd_3:	Waalse	0.54
left_wd_2:	socialisten	0.68
left_wd_1:	was	0.53
left_pos_3:	ADJ(prenom,basis,met-e,stan)	0.17
left_pos_2:	N(soort,mv,basis)	0.20
left_pos_1:	WW(pv,verl,ev)	0.30
right_wd_1:	niet	0.63
right_wd_2:	erg	0.56
right_wd_3:	geliefd	0.57
right_pos_1:	BW()	0.27
right_pos_2:	ADJ(vrij,basis,zonder)	0.23
right_pos_3:	ADJ(vrij,basis,zonder)	0.18
distance is less than three sentences:	no	1.72
the anaphor is a pronoun:	yes	0.00
the antecedent is a pronoun:	no	3.94
both are pronominal:	no	3.94
anaphor is a demonstrative:	no	0.58
anaphor is a definite NP:	def_yes	0.00
number agreement:	num_yes	3.42
complete match:	no	**14.22**
partial match:	no	**13.86**
named entity agreement:	no	0.00
appositive:	appo_no	0.00
both are proper nouns:	no	0.00
the antecedent is a proper noun:	iproper_yes	1.70
the anaphor is a proper noun:	no	0.00
alias:	no	0.00
semantic class of the anaphor:	male	2.06
semantic class of the antecedent:	male	3.62
semantic class agreement:	male	**11.41**
syntactic function of the anaphor:	SBJ	0.09
syntactic function of the antecedent:	imm_prec_I-SBJ	1.37
are both SBJ/OBJ?:	SBJ	0.64
both share the same head:	no	**13.86**
synonym:	no	0.00
hypernym:	no	0.00
the anaphor is a pronoun refering to a proper noun antecedent:	yes	1.70

Figure 1: Example individual for a genetic algorithm population representing a RIPPER experiment, encoding selected features (left) and algorithmic parameters (right).



4 A lazy and an eager learner

Having built the feature vectors, we experimented with two machine learning techniques on the task of co-reference resolution: the lazy learning implementation TIMBL (Daelemans et al. 2002) and the eager rule induction method RIPPER (Cohen 1995). The motivation for the choice of a lazy or an eager learner lies in their differences, e.g. in the understandability of learned models or abstraction from noise, two hallmarks of eager learning, versus the possibility of learning from low-frequency or exceptional data points, a strong point of lazy learning (Daelemans, van den Bosch and Zavrel 1999). We performed the validation experiments using ten-fold cross-validation on the available training data. In order to have an idea of the performance on the minority class, we evaluated the results of our experiments in terms of precision, recall and $F_{\beta=1}$. The test results are reported in Section 5.

In earlier work (Hoste et al. 2002, Daelemans and Hoste 2002, Daelemans et al. 2003a, Decadt et al. 2004, Hoste 2005) we showed that a proper comparative experiment requires extensive and methodologically correct optimization, and that the performance increase obtained by this optimization has the potential to be considerable. Therefore, for the comparative experiments between TIMBL and RIPPER we used a genetic algorithm (henceforth: GA) to look for the optimal features and parameters for both learning techniques. The principle behind GAs is quite simple: search starts from a population of individuals which all represent a candidate solution to the problem to be solved (in our case, joint feature selection and parameter optimization). Figure 1 displays an example RIPPER individual represented by particular values for all algorithm parameters and features.

A fitness function (for example an F-measure of the individual's generalization performance on validation material) is used to determine how good an individual is at solving a problem. In order to combine effective solutions and maintain diversity in the population, individuals are combined or mutated to breed new individuals. After a number of generations, an optimal individual is selected. Table 3 gives an overview of the default and optimized cross-validation results on the different data sets. In this table, "All" refers to the classifiers trained on all data; "PPC" refers to the combined output of the three separate classifiers for "Pronouns", "Proper

nouns", and "Common nouns".

Table 3 shows that the joint optimizaton of feature selection and parameter optimization can cause large variation in the results of both classifiers. The bold-faced results in the table show that optimization mainly wipes out the initial weaknesses of both learners: the increase of $F_{\beta=1}$ scores for TIMBL is mainly attained through a marked increase of precision scores (mainly caused by a selection of the proper features), whereas the increase of $F_{\beta=1}$ scores for RIPPER is mainly due to the increase of recall scores (mainly caused by adaptations to the loss ratio parameter).

Furthermore, we can observe that the performance differences inside one single learning method can be much larger than the method-comparing performance differences. The application of TIMBL and RIPPER on the MUC-6 "Pronouns" data set, for example, leads to default $F_{\beta=1}$ scores of 31.97 and 28.70 respectively; a 3% performance difference. In contrast, optimization within one single algorithm, for example TIMBL, leads to a performance increase of about 10% (from 31.97 to 41.80). The TIMBL and RIPPER results on the MUC-7 "Common nouns" data set are another illustrative example. In their default representation, TIMBL and RIPPER yield a 41.03 and a 49.24 $F_{\beta=1}$ score, respectively. Optimization leads to a large performance improvement and to a less prominent performance difference: 52.31 for TIMBL and 53.29 for RIPPER. In conclusion, we can state that we cannot draw conclusions of one classifier being better on a particular task than another classifier, when only taking into account default settings or limited optimization.

In order to determine which of both learning techniques performs better on the task of co-reference resolution, we applied a bootstrap resampling test to estimate significance thresholds. This test was done on the optimized "All" and "PPC" results of both learners and reveals that for half of the results none of the two learners significantly outperforms the other (MUC-6 and MUC-7 "All") and that for the other half RIPPER significantly outperforms TIMBL (MUC-6 "PPC" and KNACK-2002 "All" and "PPC"). These results, which do not reveal a clear supremacy of one learner over the other, confirm the necessity of optimization.

With respect to the use of three classifiers, each trained on the co-referential relations of a specific type of NP, instead of one single classifier covering all co-referential relations, we could observe in the default experiments that the RIPPER results on the combined output of the NP type learners were always higher (MUC7, KNACK-2002: $p << 0.01$) than the results on the data sets as a whole, whereas the TIMBL results on the combined data sets were similar (MUC-6, KNACK-2002) or even significantly below (MUC-7: $p << 0.01$) the results on the complete data set. However, after optimization this tendency becomes less clear. A comparison of the "All" and "PPC" results shows that three classifiers, each trained on one specific NP type perform better than one single classifier in 5 out of 6 cases (not for the Ripper results on KNACK-2002). However, this difference in performance is only significant in 3 out of 6 cases (for TIMBL on KNACK-2002 and RIPPER on MUC-6 and MUC-7). In short, we can conclude that no convincing evidence is found for our initial hypothesis that three more specialized classifiers, each trained on the co-referential relations of a specific type of NP will perform better on the task of co-reference resolution than one single classifier covering all co-referential

Table 3: Cross-validation results in terms of precision, recall and $F_{\beta=1}$ of TIMBL and RIPPER on the complete MUC-6, MUC-7 and KNACK-2002 data sets, on the partial data sets ("Pronouns", "Proper nouns" and "Common nouns") and on the combined output of the partial learners ("PPC"). Columns 2-4 provide the results of both learners without feature selection and in their default parameter settings. Columns 5-7 give the results after joint feature selection and parameter optimization using a genetic algorithm.

MUC-6	DEFAULT			GA OPTIMIZATION		
TIMBL	Prec.	Rec.	$F_{\beta=1}$	Prec.	Rec.	$F_{\beta=1}$
All	56.80	**55.50**	56.15	**83.22**	52.17	**64.14**
PPC	57.19	**56.21**	56.70	**79.13**	54.27	**64.38**
Pronouns	38.33	27.42	31.97	**45.73**	**38.48**	**41.80**
Proper nouns	63.34	**67.53**	65.37	**88.92**	59.57	**71.34**
Common nouns	53.70	53.53	53.62	**87.58**	**54.39**	**67.11**
RIPPER	Prec.	Rec.	$F_{\beta=1}$	Prec.	Rec.	$F_{\beta=1}$
All	**84.65**	49.65	62.59	73.66	**57.36**	**64.49**
PPC	**79.73**	52.59	63.16	76.01	**59.04**	**66.46**
Pronouns	**54.78**	19.44	28.70	50.84	**34.60**	**41.17**
Proper nouns	83.89	61.60	71.04	**86.03**	**62.84**	**72.63**
Common nouns	**79.61**	55.55	65.44	73.12	**66.98**	**69.92**

MUC-7	DEFAULT			GA OPTIMIZATION		
TIMBL	Prec.	Rec.	$F_{\beta=1}$	Prec.	Rec.	$F_{\beta=1}$
All	51.57	**46.09**	48.68	**75.29**	45.24	**56.52**
PPC	50.53	**45.32**	47.78	**76.45**	45.23	**56.84**
Pronouns	42.31	36.60	39.25	**61.28**	**38.96**	**47.64**
Proper nouns	62.36	**56.87**	59.49	**85.92**	55.11	**67.15**
Common nouns	43.06	**39.17**	41.03	**80.45**	38.76	**52.31**
RIPPER	Prec.	Rec.	$F_{\beta=1}$	Prec.	Rec.	$F_{\beta=1}$
All	**77.51**	36.21	49.36	67.19	**48.27**	**56.18**
PPC	**75.89**	38.64	51.21	67.98	**49.54**	**57.31**
Pronouns	**59.50**	22.70	32.86	49.74	**49.61**	**49.68**
Proper nouns	84.58	52.56	64.83	**87.05**	**55.25**	**67.60**
Common nouns	**74.56**	36.76	49.24	72.80	**42.03**	**53.30**

KNACK-2002	DEFAULT			GA OPTIMIZATION		
TIMBL	Prec.	Rec.	$F_{\beta=1}$	Prec.	Rec.	$F_{\beta=1}$
All	48.78	44.93	46.78	**71.83**	**45.50**	**55.71**
PPC	49.75	44.90	47.20	**70.22**	**49.74**	**58.24**
Pronouns	50.11	44.81	47.31	**67.65**	**53.04**	**59.46**
Proper nouns	62.84	54.04	58.11	**80.07**	**54.87**	**65.11**
Common nouns	30.65	30.37	30.51	**59.58**	**33.49**	**42.88**
RIPPER	Prec.	Rec.	$F_{\beta=1}$	Prec.	Rec.	$F_{\beta=1}$
All	**69.49**	34.92	46.49	61.51	**61.93**	**61.72**
PPC	**66.34**	41.75	51.25	60.68	**62.26**	**61.46**
Pronouns	**61.08**	43.14	50.57	58.95	**69.69**	**63.87**
Proper nouns	**76.84**	49.49	60.21	69.36	**62.71**	**65.87**
Common nouns	**61.82**	25.92	36.52	51.57	**43.48**	**47.18**

relations.

5 Testing

Defining the anaphora resolution process as we set out to do, namely to first classify pairs of candidate anaphors and antecedents, necessitates the use of a two-step procedure. In a **first step**, the classifier (in our case TIMBL or RIPPER) decides on the basis of the information learned from the training set whether the combination of a given anaphor and its candidate antecedent in the test set is classified as a co-referential link. Since each NP in the test set is linked with several preceding NPs, this implies that one single anaphor can be linked to more than one antecedent, which for its part can also refer to multiple antecedents, and so on. Therefore, a **second step** is taken, which involves filtering down this link graph to just one co-referential link per anaphor. We expand on this selection procedure in Subsection 5.2. First, we describe how we imposed constraints on our systems in terms of the scope in which antecedents are searched.

The general setup of our experiments on the test set is the following. For all three data sets (MUC-6, MUC-7 and KNACK-2002), we use a held-out test set. Both RIPPER and TIMBL are trained on the complete training set and the resulting classifiers are applied to the held-out test set, which is represented as a set of instances. For the construction of the test instances, all NPs starting from the second NP in a text are considered a possible anaphor, whereas all preceding NPs are considered possible antecedents of the anaphor under consideration. Since this type of instance construction leads to an enormous increase of the data set and since we are only interested in finding one possible antecedent per anaphor, we took into account some search scope limitations.

5.1 Search scope

As a starting point for restricting the number of instances without losing possibly interesting information, we calculated the distance between the references and

their immediately preceding antecedent. For these calculations, we took the MUC-6 and KNACK-2002 training sets as a test case. The distances were calculated as follows: antecedents from the same sentence as the anaphor were at distance 0. Antecedents in the sentence preceding the sentence of the referring expression were at distance 1, and so on. We divided the group of referring expressions into the three categories: (1) pronouns, (2) proper nouns and (3) common nouns. These results are displayed in Figure 2 and Figure 3.

Both figures reveal similar tendencies. With respect to the pronominal anaphors, we can observe that in the MUC-6 training data 97.8% of the antecedents appears in a context of three sentences. From these antecedents, the large majority (73.0%) appears in the sentence itself, 22.7% appears one sentence before and 2.2% of the antecedents of anaphorical pronouns is located two sentences before. In the KNACK-2002 training data, a similar but less prominent observation can be made in the case of pronouns. 77.3% of the immediately preceding antecedents can be found in a context of three sentences. 41.1% of the antecedents appears in the sentence itself, 29.2% appears one sentence before and 6.9% of the antecedents of anaphoric pronouns is located two sentences before. With respect to the proper nouns, we can observe that 79.2% of the proper nouns in the MUC-6 training data occurs in a scope of three sentences. For KNACK-2002, 44.01% of the immediately preceding antecedents of the proper noun NPs can be found in a scope of three sentences. Finally, for the common noun NPs we can observe that in the MUC-6 training data 73.3% occurs in a scope of three sentences. For KNACK-2002, 65.2% of the immediately preceding antecedents can be found in a scope of three sentences.

Although similar tendencies can be observed in both data sets, these tendencies are much more prominent in the MUC-6 data. This difference might be due to a difference in text style (magazine articles for KNACK-2002 as opposed to newspaper articles in MUC-6), a difference in text length (KNACK-2002 has longer texts) and typological differences between the languages.

We will use this search scope information in the construction of the test instances, for example by restricting the number of test instances in the pronouns data set to anaphors with antecedents at distance 0, 1 and 2 (as for example also in Mitkov (1998) and Yang, Zhou, Su and Tan (2003)). For a more elaborate discussion on search scope for English, we refer to Mitkov (2002).

A second motivation for restricting the number of antecedents on the basis of their distance to the anaphor, only for the pronouns, are the decreasing classifier results for the antecedents further away, as shown for MUC-6 in Figure 4. This tendency can be observed for both classifiers. For the other data sets (proper nouns and common nouns), no general conclusion can be drawn concerning the dependency of performance on the distance of the candidate antecedent to the anaphor.

For the construction of the test instances we took into account the search scope observations discussed above and used some simple heuristics:

- **Pronouns**: all NPs in a context of 2 sentences before the pronominal NP are included in the test sets for the pronouns.

Figure 2: Distance in number of sentences between a given referring expression and its immediately preceding antecedent in the MUC-6 training set.

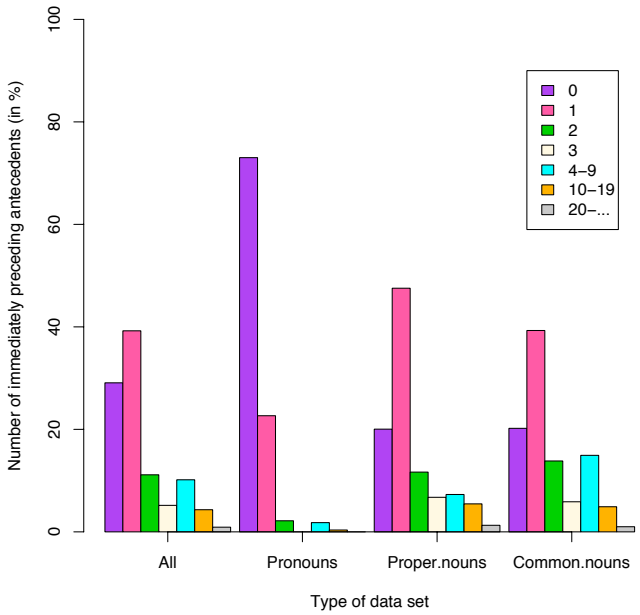


Figure 3: Distance in number of sentences between a given referring expression and its immediately preceding antecedent in the KNACK-2002 training set.

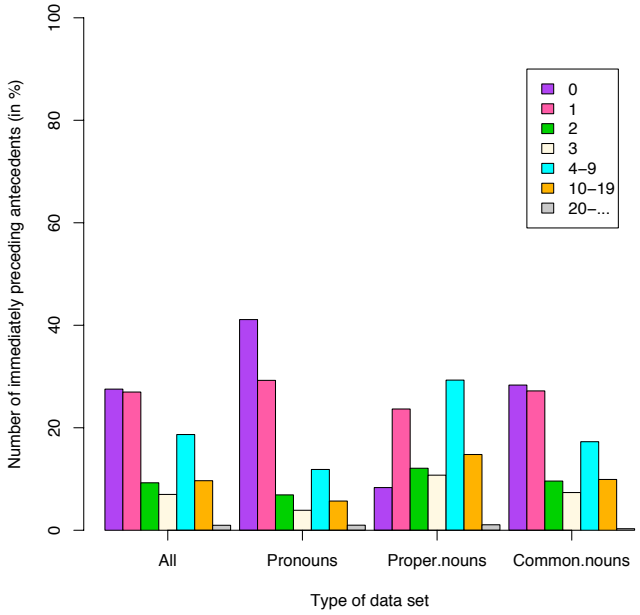
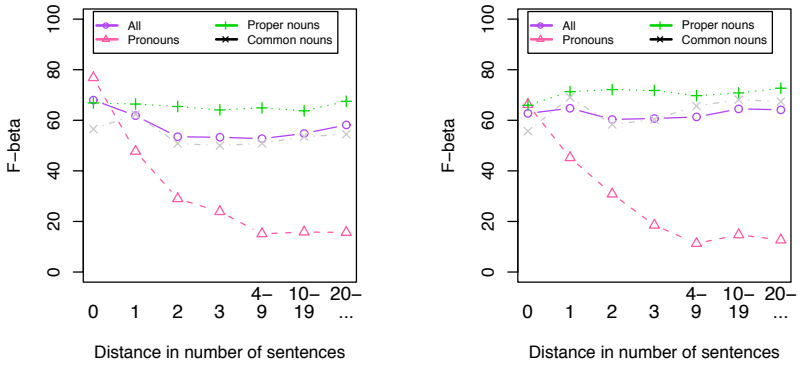


Figure 4: $F_{\beta=1}$ results plotted against the distance in number of sentences between an anaphor and its candidate antecedent after application of TIMBL (left) and RIPPER (right) on the MUC-6 data sets.



- **Proper nouns**: all NPs which partially match the proper nouns NP are included. For the non-matching NPs, the search scope is restricted to two sentences.
- **Common nouns**: same selection as for the proper nouns.

5.2 Experimental results

Before proceeding to the experimental results on the test data, we will first discuss the problem of antecedent selection and describe the evaluation procedure as used on the test data.

Antecedent selection As set out in the introduction of this section, a classification-based co-reference resolution system as described here necessitates a two-step procedure. In a first step, possibly co-referential NPs are classified as being co-referential or not. In a second step, the co-referential chains are built on the basis of the positively classified instances. For this second step, different directions can be taken, such as a "closest-first" approach as used by (Soon et al. 2001), a selection approach which performs a right-to-left search to find the most likely antecedent as used by (Ng and Cardie 2002) or a so-called twin-candidate learning model, in which the antecedent for a given anaphor is identified through pairwise comparisons as used by (Connolly, Burger and Day 1994) and (Yang et al. 2003). Instead of selecting one single antecedent per anaphor, as in the previously described approaches, we tried to build complete co-reference

chains for our documents. For an extensive description of this selection procedure, we refer to (Hoste 2005).

Evaluation procedure For all experiments reported on the training data, the performance was reported in terms of precision, recall and F-measure. For all experiments on the test set, the performance is also reported in terms of precision, recall and F-measure, but this time using the MUC scoring program from Vilain, Burger, Aberdeen, Connolly and Hirschman (1995). The program focuses on evaluating equivalence classes, i.e. transitive closures of a co-reference chain – not on evaluating complete chains.

In the Vilain et al. (1995) algorithm, the **recall** for an entire set T of equivalence classes is computed as follows:

$$R_T = \frac{\sum(c(S)-m(S))}{\sum(c(S))}$$

where $c(S)$ is the minimal number of correct links necessary to generate the equivalence class S: $c(S) = (|S| - 1)$. $m(S)$ is the number of missing links in the response relative to equivalence set S generated by the key: $m(S) = (|p(S)| - 1)$. $p(S)$ is a partition of S relative to the response: each subset of S in the partition is formed by intersecting S and the responses sets R_i that overlap S. For the computation of the **precision**, the roles for the answer key and the response are reversed. For example, equivalence class S can consist of the following elements S = {1 2 3 4}. If the response is $< 1 - 2 >$, then $p(S)$ is {1 2}, {3} and {4}.

This algorithm, however, has two major shortcomings according to Baldwin, Morton, Bagga, Baldridge, Chandraseker, Dimitriadis, Snyder and Wolska (1998). The algorithm does not give any credit for separating out singletons (entities occurring in chains only consisting of one element, such as 3 and 4 in the preceding example). Furthermore, it does not distinguish between different types of errors. In the following example, the key consists of three equivalence classes and two responses are given. The two responses yield the same precision (88.9%) and recall score (100%) according to the MUC scoring program. Baldwin et al. (1998) argue that the error made in response 2 is more damaging, since it makes more entities erroneously co-referent:

key	0←1←2←3
	4←5
	6←7←8←9←10
response 1	0←1←2←3←4←5
	6←7←8←9←10
response 2	0←1←2←3←6←7←8←9←10
	4←5

Despite these shortcomings, we used this scoring software since it has been widely used for evaluation on the MUC-6 and MUC-7 data sets and it thus enables comparison with the results of other systems on these data sets.

Table 4 gives an overview of the results obtained by TIMBL and RIPPER on the "Pronouns", "Proper nouns" and "Common nouns" test sets in terms of precision, recall and $F_{\beta=1}$. It shows that our results obtained on the MUC-6 and MUC-7 data sets are comparable to the results reported by Soon et al. (2001). They report a precision of 67.3%, a recall of 58.6% and an $F_{\beta=1}$ of 62.6% on the MUC-6 data. For MUC-7, they report a precision of 65.5%, a recall of 56.1% and an $F_{\beta=1}$ of 60.4%. The best results reported to date on the MUC-6 and MUC-7 data are by Ng and Cardie (2002a,2002b,2002c): 63.3% recall, 76.9% precision and 69.5% $F_{\beta=1}$ on MUC-6 and 54.2% recall, 76.3% precision and 63.4% $F_{\beta=1}$ on MUC-7[3]. Their extensions to the approach of Soon et al. (2001) include (i) the expansion of the feature set with additional lexical, semantic and knowledge-based features, (ii) a modification of the clustering algorithm favoring the 'highest likely antecedent', (ii) a learning-based method to determine anaphoricity, (iii) positive and negative sample selection in order to handle the problem of skewed class distributions, and (iv) pruning of the rule sets.

For KNACK-2002, no comparative results are yet available since this is a new corpus. Table 4 shows that both TIMBL and RIPPER obtain a $F_{\beta=1}$ score of 51% on the Dutch data. As for English, the precision scores for the "Pronouns" (64.9% for TIMBL and 66.7% for RIPPER) and the "Proper nouns" data sets (79.4% for TIMBL and 79.0% for RIPPER) are much higher than those obtained on the "Common nouns" data set (47.6% for TIMBL and 47.5% for RIPPER). Furthermore, the recall scores are about 20% lower than the precision scores: 42.2% recall vs. 65.9% precision for TIMBL and 40.9% recall vs. 66.3% precision for RIPPER.

6 Post-correcting co-reference chains based on Levenshtein distance

The two-step procedure described in the previous sections operates from local classifications in the first step to more global antecedent selection and chaining in the second step. Errors in both steps can lead to incorrect equivalence classes and chains; errors in the first step are irrepairable by the second and percolate to the end result, and errors in the second step immediately lead to mispredicted or missed equivalence classes. Nonetheless, it is conceivable that certain errors in the resulting chains could be identified by a third step as violating certain basic syntactic properties of chains, if there would be a third step that has knowledge of the constraints on the composition of chains. For example, chains may typically not start with a pronoun. Or, two adjacent proper nouns may not link to each other when later in the text one of the two proper nouns is repeated.

As these informal example rules indicate, these validity constraints could be formulated at the level of types of NPs: pronouns, proper nouns, common nouns, and some concept of distance. Once formulated at this abstract level, a chain could then be checked against a grammar or a reference list (or trusted lexicon) of possible chain structures, and be marked as violating some grammar constraint or not occurring in the list.

[3] On MUC-6, they report a top performance of 70.4% $F_{\beta=1}$ when doing manual feature selection.

Table 4: Results from TIMBL and RIPPER on the test set in terms of precision, recall and $F_{\beta=1}$. No recall and $F_{\beta=1}$ scores could be provided on the NP type data sets, since the scoring software does not distinguish between the three NP types.

MUC-6	Prec.	Rec.	$F_{\beta=1}$
Timbl			
PPC	70.5	59.1	64.3
Pronouns	77.3	—	—
Proper nouns	83.0	—	—
Common nouns	56.4	—	—
Ripper			
PPC	66.2	60.9	63.4
Pronouns	79.9	—	—
Proper nouns	82.2	—	—
Common nouns	50.4	—	—

MUC-7	Prec.	Rec.	$F_{\beta=1}$
Timbl			
PPC	67.1	54.5	60.2
Pronouns	68.4	—	—
Pronouns	68.4	—	—
Proper nouns	78.0	—	—
Common nouns	54.3	—	—
Ripper			
PPC	68.7	49.5	57.6
Pronouns	67.8	—	—
Proper nouns	82.5	—	—
Common nouns	57.2	—	—

KNACK-2002	Prec.	Rec.	$F_{\beta=1}$
Timbl			
PPC	65.9	42.2	51.4
Pronouns	64.9	—	—
Proper nouns	79.4	—	—
Common nouns	47.6	—	—
Ripper			
PPC	66.3	40.9	50.6
Pronouns	66.7	—	—
Proper nouns	79.0	—	—
Common nouns	47.5	—	—

While a grammar of co-reference chains may be manually written, it would require manual labor for each language and possibly each domain, since different types of text use different conventions for chaining and the use of pronouns, such as fictional novels versus newspaper text. Instead, we explored the option of using a trusted lexicon and an error detection module that checks whether a produced chain conforms with the list. We define this procedure as a spelling correction mechanism, where our "spelling" is the abstract representation of a chain into a small alphabet of symbols representing types of NPs, their distance, and their linkage. We tested five increasingly complex alphabets, starting from a two-symbol code only encoding linkage, to a four-symbol code encoding types, linkage, and distance. The alphabet encodings have in common that they encode a chain as a string that starts with the first linked element of the chain, and ends with the final linked element. The further definition of the five alphabets is the following:

1. The LOLOL alphabet encodes any linked element, whether it is a pronoun, a proper noun, or a common noun, as an L, and any NP between two linked elements as an O. Even if several NPs occur between two Ls, they are represented by a single O.

2. The LOOLOL alphabet is similar to LOLOL, except that it represents each intermediary non-linked NP between two Ls as an individual O.

3. The LOLOP alphabet is again similar to LOLOL, but it distinguishes between pronouns (the most frequent type of linked element, represented by P) and other NPs (proper and common nouns, L).

4. The LOOLOP alphabet inherits its definition from both LOLOP and LOOLOL; it encodes the numbers of non-linked intermediary elements, and discerns between proper nouns and other NPs.

5. The NOOLOP alphabet is an extension of LOOLOP in that it further discerns between proper nouns (N) and common nouns (L).

In example text fragment 4 the bold-faced words represent the elements of a co-reference chain, while the italicized words belong to other chains. The simple chain from "the space" to "space" has five intermediary NPs that do not belong to the chain, so while the LOLOL coding would encode this chain as LOL, the LOOLOL coding would encode it as LOOOOOL.

(4) The *earth* is not disconnected from **the space** from which *she* originated. *The Astrophysical Journal* even reports that *the evolution* of *life* on *earth* can be influenced by what happens in **space**.

The post-processing step we applied on our data uses all chains in the training set (encoded by one of these alphabets) as the trusted lexicon, and treats any new predicted chain as a possibly incorrect sequence that needs to be corrected to match its most likely correct form. To find the most likely correct form, a

classical method is to find the most similar form according to Levenshtein distance (Levenshtein 1965). This classical distance metric counts the numbers of deletions, insertions, and substitutions needed to convert one string to the other, whereby in its simplest variant all three operations are counted as contributing a distance of 1. Levenshtein distance can be computed efficiently for small strings using dynamic programming.

A small alphabet and no representation of numbers of intermediate elements, such as the LOLOL coding, leads to just a small number of possible patterns of alternations of Ls and Os, and it is likely that a new chain predicted on test data will have been seen in the training set. This would spoil the intended functionality of the method; the goal is to be able to detect incorrect sequences by detecting a strong similarity with a known sequence in the reference list, to be able to spot the difference and suggest a correction. To attain this goal, the alphabet coding should be expressive enough to allow differences to occur between sufficiently different chains, but should not be too expressive to avoid the suggestion of corrections from too distant neighbors.

We implemented the checking procedure as follows. Given each generated chain, we first converted it according to the selected alphabet coding, and searched for the most similar chain in the trusted lexicon according to Levenshtein distance. In case of equally-distant neighbors, we heuristically preferred the neighbor with an equal number of symbols. If the most similar chain was in fact an identical chain we left the predicted chain unchanged. If the best match differed, we adapted the predicted chain only if the correction involved a change of a link element into an element outside the chain (O). This implies that our correction only involves the deletion of chain elements, and not the insertion or replacement – it will therefore only improve precision if it correctly deletes superfluous link elements. The motivation for not performing insertions is that insertions, if incorrect, would do double damage, both to the current chain as well as to any other chain it is necessarily deleted from (since elements cannot belong to two chains). The straightforward reason for not replacing link elements is that the link element types cannot be corrected as such - a pronoun cannot be "corrected" into a proper noun. In this sense our correction differs from orthographical spelling correction.

We applied the Levenshtein-based post-processing correction method, using the five alphabet codings, to the KNACK-2002 corpus, and compared its overall performance when applied to the output of Timbl on the "All" task. The results are given in Table 5, which also offers the MUC score on the original output before correction for comparison. None of the correction methods improves on recall, which is to be expected since the correction only involves the deletion of chain elements; any incorrect deletion diminishes recall. Precision, however, is indeed slightly boosted by post-processing. The best precision, 70.0, 2.5 points above the baseline precision, is attained using the LOOLOP coding; the best F-measure, 51.8, is attained by the similar NOOLOP coding since it does not harm recall as much as LOOLOP. It appears that the LOOLOL and LOOLOP codings (encoding the actual number of intermediate symbols) trigger relatively many false-alarm corrections compared to LOLOL and LOLOP. The most expressive alphabet tested, NOOLOP,

Table 5: Comparative results of Levenshtein-distance-based correction, as measured by the MUC scorer on the "All" task, based on Timbl's output on KNACK-2002: the baseline score (*no coding*), and the scores attained by correction using five different alphabet codings. Bold-face results indicate the highest score in the column.

Correction coding	Recall	Precision	$F_{\beta=1}$
no coding	**41.4**	67.5	51.3
LOLOL	41.1	67.7	51.1
LOOLOL	38.4	67.3	48.9
LOLOP	41.4	67.6	51.3
LOOLOP	40.9	**70.0**	51.6
NOOLOP	41.3	69.9	**51.8**

appears to strike the best balance.

An error analysis reveals that certain recurring types of errors are consistently repaired by the method. For instance, enumerations of more than two proper nouns with commas would often trigger Timbl to link the second proper noun to the first, while they are in fact enumerated non-linked names. Another typical correction is the deletion of a link between an NP and an immediately adjacent pronoun, followed at some distance by a non-pronominal link element. In code terms, LPOOOL would for instance be corrected to LOOOOL.

7 Concluding remarks

We have presented a modular machine learning approach to the resolution of coreferential relations between nominal constituents in English and Dutch. The system has three components; in the first, local decisions are made among pairs of pronominal, common noun and proper noun NPs to determine whether they refer to the same. In the second step a more global selection is made among all pairwise linkages to form entire co-reference chains. In the third step, a global post-processing procedure based on spelling correction methods checks if each generated chain contains superfluous linked elements.

On the local decision level we showed that optimization of feature selection and parameter optimization can cause large variation in the results of a classifier. Furthermore, the performance differences inside one single learning algorithm could be much larger than the method-comparing performance differences. With respect to the use of three classifiers, each trained on the co-reference relations of a specific type of NP, instead of one single classifier covering all relations, we could not find convincing evidence for our initial hypothesis that three more specialized classifiers would outperform one single classifier.

On the co-reference chain level we could observe that the results on Dutch, which are the first results ever to be reported on this scale for this language, are lower than for English. This might be due to a difference in the types of training and testing material, to a more difficult feature construction for Dutch due to less

developed syntactic and semantic resources, but also to differences between Dutch and English. For Dutch, for example, the resolution of male and female pronouns suffers from a much larger search space of possible candidate antecedents since pronouns can also refer to the linguistic gender of the antecedent. However, disregarding these differences, we observe rather similar results to those of the English systems.

In a final post-processing step which uses Levenshtein distance to check if the generated chains contain superfluous linked elements, a modest increase in precision could be obtained.

References

Aone, C. and Bennett, S. (1995), Evaluating automated and manual acquisition of anaphora resolution strategies, *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics (ACL-1995)*, pp. 122–129.

Baayen, R. , Piepenbrock, R. and van Rijn, H. (1993), The CELEX lexical data base on CD-Rom.

Baldwin, B. , Morton, T. , Bagga, A. , Baldridge, J. , Chandraseker, R. , Dimitriadis, A. , Snyder, K. and Wolska, M. (1998), Description of the upenn camp system as used for coreference, *Proceedings of the Seventh Message Understanding Conference (MUC-7)*.

Buchholz, S. (2002), *Memory-based Grammatical Relation finding*, PhD thesis, Tilburg University.

Cardie, C. and Wagstaff, K. (1999), Noun phrase coreference as clustering, *Proceedings of the 1999 joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, pp. 82–89.

Cohen, W. W. (1995), Fast effective rule induction, *Proceedings of the 12th International Conference on Machine Learning (ICML-1995)*, pp. 115–123.

Connolly, D. , Burger, J. and Day, D. (1994), A machine learning approach to anaphoric reference, *Proceedings of the International Conference on 'New Methods in Language Processing'*.

Daelemans, W. and Hoste, V. (2002), Evaluation of machine learning methods for natural language processing tasks, *Proceedings of the Third International Conference on Language Resources and Evaluation (LREC-2002)*, pp. 755–760.

Daelemans, W. , Hoste, V. , De Meulder, F. and Naudts, B. (2003a), Combined optimization of feature selection and algorithm parameter interaction in machine learning of language, *Proceedings of the 14th European Conference on Machine Learning (ECML-2003)*, pp. 84–95.

Daelemans, W. , van den Bosch, A. and Zavrel, J. (1999), Forgetting exceptions is harmful in language learning, *Machine Learning* **34**(1-3), 11–41.

Daelemans, W. , Zavrel, J. , van den Bosch, A. and van der Sloot, K. (2003b), Memory based tagger, version 2.0, reference guide, *Technical Report ILK Technical Report - ILK 03-13*, Tilburg University.

Daelemans, W. , Zavrel, J. , van der Sloot, K. and van den Bosch, A. (2002), Timbl:

Tilburg memory-based learner, version 4.3, reference guide, *Technical Report ILK Technical Report - ILK 02-10*, Tilburg University.

De Meulder, F. and Daelemans, W. (2003), Memory-based named entity recognition using unannotated data, *Proceedings of the Seventh Conference on Natural Language Learning (CoNLL-2003)*, pp. 208–211.

De Pauw, G. , Laureys, T. , Daelemans, W. and Van Hamme, H. (2004), A comparison of two different approaches to morphological analysis of Dutch, *Proceedings of the Seventh Meeting of the ACL Special Interest Group in Computational Phonology*, pp. 62–69.

Decadt, B. , Hoste, V. , Daelemans, W. and van den Bosch, A. (2004), GAMBL, genetic algorithm optimization of memory-based WSD, *Proceedings of the Third International Workshop on the Evaluation of Systems for Semantic Analysis of Text (SENSEVAL-3)*, pp. 108–112.

Fellbaum, C. (1998), *WordNet: An Electronic Lexical Database*, MIT Press.

Fisher, F. , Soderland, S. , Mccarthy, J. , Feng, F. and Lehnert, W. (1995), Description of the UMass system as used for MUC-6, *Proceedings of the Sixth Message Understanding Conference (MUC-6)*, pp. 127–140.

Hoste, V. (2005), *Optimization Issues in Machine Learning of Coreference Resolution*, PhD thesis, Antwerp University.

Hoste, V. , Hendrickx, I. , Daelemans, W. and van den Bosch, A. (2002), Parameter optimization for machine-learning of word sense disambiguation, *Natural Language Engineering, Special Issue on Word Sense Disambiguation Systems* **8**, 311–325.

Ji, H. , Westbrook, D. and Grishman, R. (2005), Using semantic relations to refine coreference decisions, *Proceedings of HLT/EMNLP*, pp. 17–24.

Levenshtein, V. I. (1965), Binary codes capable of correcting insertions, deletions, and reversals, *Doklady Akademii Nauk SSSR* **163**(4), 845–848. Original article in Russian. English translation in *Soviet Physics Doklady*, **10**(8):707–710, 1966.

Markert, K. and Nissim, M. (2005), Comparing knowledge sources for nominal anaphora resolution, *Computational Linguistics* **31**(3), 367–401.

McCarthy, J. (1996), *A Trainable Approach to Coreference Resolution for Information Extraction*, PhD thesis, Department of Computer Science, University of Massachusetts, Amherst MA.

Mihalcea, R. (2002), Word sense disambiguation with pattern learning and automatic feature selection, *Natural Language Engineering, Special Issue on Word Sense Disambiguation Systems* **8**, 343–358.

Mitkov, R. (1998), Robust pronoun resolution with limited knowledge, *Proceedings of the 17th International Conference on Computational Linguistics (COLING-1998/ACL-1998)*, pp. 869–875.

Mitkov, R. (2002), *Anaphora Resolution*, Longman.

MUC-6 (1995), Coreference task definition. version 2.3., *Proceedings of the Sixth Message Understanding Conference (MUC-6)*, pp. 335–344.

MUC-7 (1998), Muc-7 coreference task definition. version 3.0., *Proceedings of the Seventh Message Understanding Conference (MUC-7)*.

Ng, V. and Cardie, C. (2002), Combining sample selection and error-driven pruning for machine learning of coreference rules, *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP-2002)*, pp. 55–62.

Quinlan, J. (1993), *C4.5: Programs for machine learning*, Morgan Kaufmann, San Mateo, CA.

Soon, W. , Ng, H. and Lim, D. (2001), A machine learning approach to coreference resolution of noun phrases, *Computational Linguistics* **27**(4), 521–544.

Strube, M. , Rapp, S. and Müller, C. (2002), The influence of minimum edit distance on reference resolution, *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP-2002)*, pp. 312–319.

Tjong Kim Sang, E. , Daelemans, W. and Höthker, A. (2004), Reduction of Dutch sentences for automatic subtitling, *Computational Linguistics in the Netherlands 2003. Selected Papers from the Fourteenth CLIN Meeting*, pp. 109–123.

Vilain, M. , Burger, J. , Aberdeen, J. , Connolly, D. and Hirschman, L. (1995), A model-theoretic coreference scoring scheme, *Proceedings of the Sixth Message Understanding Conference (MUC-6)*, pp. 45–52.

Yang, X. , Zhou, G. , Su, S. and Tan, C. (2003), Coreference resolution using competition learning approach, *Proceedings of the 41th Annual Meeting of the Association for Compuatiational Linguistics (ACL-03)*, pp. 176–183.