

Evaluating hybrid versus data-driven coreference resolution

Iris Hendrickx¹, Veronique Hoste², and Walter Daelemans¹

¹ CNTS - Language Technology Group,
University of Antwerp, Universiteitsplein 1, Antwerp
Belgium

`iris.hendrickx@ua.ac.be`, `walter.daelemans@ua.ac.be`

² LT3 - Language and Translation Technology Team,
University College Ghent, Groot-Brittaniëlaan 45, Ghent,
Belgium

`veronique.hoste@hogent.be`

Abstract. In this paper, we present a systematic evaluation of a hybrid approach of combined rule-based filtering and machine learning to Dutch coreference resolution. Through the application of a selection of linguistically-motivated negative and positive filters, which we apply in isolation and combined, we study the effect of these filters on precision and recall using two different learning techniques: memory-based learning and maximum entropy modeling. Our results show that by using the hybrid approach, we can reduce up to 92 % of the training material without performance loss. We also show that the filters improve the overall precision of the classifiers leading to higher F-scores on the test set.

1 Introduction

Coreference resolution, resolving different descriptions or names for the same underlying object, is an important text analysis module for further processing and understanding of text, for example in applications like information extraction and question answering.

As an alternative to knowledge-based approaches, corpus-based machine learning techniques have become increasingly popular for the resolution of coreferential relations. In a typical machine learning approach to coreference resolution, information on pairs of noun phrases is represented in a set of feature vectors. Unsupervised learning techniques, e.g. [1], view coreference resolution as a clustering task of combining noun phrases into equivalence classes. Most learning approaches to coreference resolution, however, are supervised learning techniques, for example the C4.5 decision tree learner [2] as in [3], [4] and [5] or the RIPPER rule learner [6] as in [7]. A supervised learning approach requires an annotated corpus from which for each pair of noun phrases a class (coreferential or not coreferential) can be obtained. The pair of NPs is represented by a feature vector containing distance, morphological, lexical, syntactic and semantic information on the candidate anaphor, its candidate antecedent and also on the

relation between both. In a postprocessing phase of such an approach, a complete coreference chain has to be built between the pairs of NPs that were classified as being coreferential.

As a consequence of recasting the problem as a classification task, coreference resolution data sets reveal large class imbalances: only a small part of the possible relations between noun phrases (NPs) is coreferential. When trained on such imbalanced data sets, classifiers can exhibit a good performance on the majority class instances but a high error rate on the minority class instances. Always assigning the “non coreferential” class will lead to a highly ‘accurate’ classifier, which cannot find any coreferential chain in a text.

In order to cope with these class imbalances, different instance selection techniques have been proposed to rebalance the corpus [8, 9, 7, 10–12]. In [12], rebalancing is done without any a priori linguistic knowledge about the class to be solved. Most approaches, however, aim to produce better performing classifiers through the application of linguistically motivated filters on the training data before application of the classifier. Through the application of these linguistic filters, part of the problem to be solved, viz. coreference resolution, is solved beforehand and only a small part of the instances is handled by the classifier. Unfortunately, previous literature on these filters is relatively vague on their exact implementation and use and no systematic studies of their impact are provided.

In this paper, we investigate the hybrid approach of combined knowledge-based filtering and machine learning in a more systematic way. We apply a selection of linguistically-motivated negative and positive filters. Negative filters filter out negative instances, positive filters do the same with examples of co-referring NPs. We study the effect of these filters on performance and we investigate how much reduction in training material we can obtain without performance loss. The filters are considered separately and in combination. We use two different machine learning techniques to demonstrate the effects of filtering: a memory-based learning approach and a maximum entropy approach. Most existing learning approaches to coreference resolution can be described as eager decision tree or rule learning approaches; we investigate in this paper how a memory-based learning and a maximum entropy approach tackle the problem of coreference resolution and the problem of the skewness of the data. The experiments are performed on the KNACK-2002 Dutch data set [12].

The remainder of this paper is organized as follows. Section 2 discusses the preparation of the data sets including the selection of positive and negative instances and presents the two machine learning packages we use in the experiments. Section 3 gives an overview of instance selection in the machine learning of coreference resolution literature, and discusses the positive and negative filters. This section also reports on the results obtained for both learners in a hybrid architecture with filters compared to a completely data-driven setting, and to baselines. Section 4 concludes this paper.

2 Experimental setup

2.1 Data

Our experiments are performed on a Dutch coreferentially annotated corpus, KNACK-2002. KNACK is a Flemish weekly news magazine with articles on national and international current affairs. For the annotation of the corpus, the MUC-7 [13] manual, the manual from Davies et al. [14] and the work from van Deemter and Kibble [15] were taken as source. The complete corpus consists of 267 documents annotated with coreference information for NPs. 12,546 noun phrases are annotated with coreferential information. For the experiments, 50 documents are randomly selected, of which 25 are used for training and the other half for testing.

For the construction of the initial data sets, we selected all noun phrases, which could be detected after preprocessing the raw text corpora. The following preprocessing steps were taken: tokenization was performed by a rule-based system using regular expressions. Dutch named entity recognition was performed by looking up the entities in lists of location names, person names, organization names and other miscellaneous named entities. We applied a part-of-speech tagger and text chunker for Dutch that use the memory-based tagger MBT [16], trained on the Spoken Dutch Corpus (<http://lands.let.ru.nl/cgn>). Finally, grammatical relation finding was performed, using a shallow parser to determine the grammatical relation between NP chunks and verbal chunks, e.g. subject, object, etc. The relation finder [17] was trained on the previously mentioned Spoken Dutch Corpus. It offers a fine-grained set of grammatical relations, such as modifiers, verbal complements, heads, direct objects, subjects, predicative complements, indirect objects, reflexive objects, etc. Figure 1 gives an overview of the part-of-speech tags, chunk tags and relation tags for the following KNACK-2002 training sentence.

(1) < COREF ID = "1528" MIN = "conflict" > Het conflict over het grensgebied < /COREF > is zo oud als < COREF ID = "1464" > < COREF ID = "1451" > India < /COREF > en < COREF ID = "1459" > Pakistan < /COREF > < /COREF >.

English: The conflict about the border area is as old as India and Pakistan.

On the basis of the preprocessed texts, instances are created. We create an instance between every NP and its preceding NPs, with a restriction of 20 sentences backwards. A pair of NPs that belongs to the same coreference chain, gets a positive label; all other pairs get a negative label. This is the basic set of instances. In the training set, the positive class accounts for only 8.5% of the total number of 76,920 instances.

Instances describe the relation between a potential anaphor and its antecedent. For each NP pair we create a set of 39 features encoding morphological-lexical, syntactic, semantic, string matching and positional information sources.

Fig. 1. Part-of-speech tags, chunk tags and relation tags for the example sentence (1).

The overview below gives a short impression of the type of information encoded in the features.

- morphological-lexical
 - Is there number agreement between anaphor and antecedent? Is it a definite/indefinite anaphor? Is the anaphor/antecedent a pronoun or proper noun?
- syntactic
 - Is the anaphor/antecedent object or subject of the sentence, is the anaphor an apposition?
- positional
 - The local context of the anaphor, the distance in NPs and in sentences between the anaphor and antecedent.
- semantic
 - Named entity type information. Are the NPs synonyms/hyponyms of each other?
- string matching
 - Are the anaphor and antecedent a complete or partial match or alias from each other? Do they share the same head word?

2.2 Learners

Two machine learning techniques are applied to the task of coreference resolution: a memory-based learning algorithm and a maximum entropy learner.

Memory-based learning (a k -nearest neighbor approach) is a lazy learning approach that stores all training data in memory. At classification time, the algorithm classifies new instances by searching for the nearest neighbors to the new instance using a similarity metric, and extrapolating from their class. In our experiments we use the TIMBL [18] software package³ that implements a version of the k -nn algorithm optimized for working with linguistic datasets and

³ URL:<http://ilk.uvt.nl>

that provides several similarity metrics and variations of the basic algorithm. Lazy learning is claimed to have the right bias for learning language processing problems as it doesn't abstract from exceptions and subregularities as more eager learning approaches do through mechanisms like pruning.

Maximum entropy modeling (a kind of exponential or log linear modeling), on the other hand, is a discriminative statistical machine learning approach [19, 20] that derives a conditional probability distribution from labeled training data by assigning a weight to each feature. We use the entropy modeling software package MAXENT by Zhang Le [21]. Maximum Entropy has been shown to provide good results with language data, and can handle large feature vectors and feature redundancy.

Memory-based learning offers several algorithmic parameters such as number of nearest neighbors, the feature weighting and distance weighting. These parameters can, individually and in combination, affect the functioning of the algorithm. We use a heuristic wrapped-based method to set them automatically for all experiments.

Wrapped progressive sampling (WPS)[22] combines classifier wrapping [23] with progressive sampling of training material [24]. WPS starts with a large pool of experiments, each with one systematically generated recombination of tested algorithmic parameter settings. In the first step of WPS, each attempted setting is applied to a small amount of training material and tested on a small amount of held-out training data. Only the best settings are kept; all others are removed from the pool of competing settings. In subsequent iterations, this step is repeated, retaining only the best-performing settings, with an exponentially growing amount of training and held-out data – until all training data is used or one best setting is left. Selecting the best settings at each step is based on classification score on the held-out data; a simple one-dimensional clustering on the ranked list of scores determines which group of settings is selected for the next iteration. The final selected parameters of the WPS procedure are then used to classify the test set.

We did not optimize the parameters of MAXENT as it was shown in [22] that WPS did not increase the generalization performance of maximum entropy modeling. We train MAXENT with L-BFGS parameter estimation, 100 iterations and a Gaussian prior with mean zero and σ^2 of 1.0.

2.3 Evaluation

Defining the coreference resolution process as a classification problem involves the use of a two-step procedure. In a **first step**, the classifier (in our case TIMBL or MAXENT) 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 coreferential 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 the selection of one coreferential link per anaphor.

In our experiments, the two steps are organized as follows. As a first step, in the classification experiments on the instance level, possibly coreferential NPs are classified as being coreferential or not. For the experiments with both learners, we perform 25-fold cross validation on the training data and we evaluate the results of our experiments by computing micro-averaged precision, recall and F-score at the instance level. For the second step, the experiments on the test set of 25 documents, the performance is also reported in terms of precision, recall and F-measure, but this time using the MUC scoring program from Vilain et al. [25]. The program looks for the evaluation at equivalence classes, being the transitive closure of a coreference chain ⁴.

We can illustrate this testing procedure for the coreferential relation between “he” and “President Bush” in the following test sentence.

(2) **President Bush** met Verhofstadt in Brussels. **He** talked with our prime minister about the situation in the Middle East.

For the NP “he” test instances are built for the NP pairs displayed in Table 1. After application of TIMBL or MAXENT, the result of the **first step** might be that the learner classifies the first instance as non-coreferential and the last two instances as being coreferential. Since we start from the assumption that each NP can only corefer with exactly one other preceding NP, a **second step** is required to make a choice between these two positive instances (he - Verhofstadt) and (he - President Bush).

Table 1. Test instances built for the “he” in example (2).

Antecedent	Anaphor	Classification
Brussels	he	no
Verhofstadt	he	yes
President Bush	he	yes

In a second step, the coreferential chains are built on the basis of the positively classified instances. For this step, different directions can be taken: a “closest-first” approach (eg. [5]) in which the first markable found to be coreferent with the anaphor is the antecedent, or an approach [7] which aims to find the most likely antecedent. This is done by selecting the antecedent with the highest confidence value among the candidate antecedent, or a twin-candidate approach [26, 9] in which the antecedent for an anaphor is selected after pairwise comparison of the possible antecedents.

Instead of selecting one single antecedent per anaphor, as in the previously described approaches, we tried to build complete coreference chains for our documents. We will now continue with a description of our selection procedure.

⁴ We did not compute significance scores because the scores given by MUC scoring program are not proper input for significance testing.

2.4 Antecedent selection

We used the following counting mechanism to recover the coreference chains in the test documents.

1. Given an instance base with anaphor - antecedent pairs (ana_i, ant_{ij}) , for which $i = 2$ to N and $j = i - 1$ to 0 . Select all positive instances for each anaphoric NP. Then make groupings by adding the positive ant_{ij} to the group of ana_i and by adding ana_i to the group of ant_{ij} .

The following is an example of such a grouping. The numbers represent IDs of anaphors/antecedents. The number before the colon is the ID of the anaphor/antecedent and the other numbers represent the IDs which relate to this anaphor/antecedent.

```
2: 2 5 6 25 29 36 81 92 99 231 258 259 286
5: 2 5 6 25 29 36 81 92 99 231 258 259 286
6: 2 5 6 25 29 36 81 92 99 231 236 258 259 286
8: 8 43 64 102 103 123 139 144 211 286
20: 20 32 69 79
```

2. Then compare each ID grouping with the other ID groupings by looking for overlap between two groupings. Select the pairs with an overlap value above a predefined threshold. We selected all pairs with an overlap value above 0.1. For example, we computed the overlap between the grouping of ID 2 with the groupings of IDs 5, 8 and 20 in the previous example as can be seen in Table 2. For the groupings of 2 and 5, we can observe a complete overlap. Combining ID 8 with ID 2, however, leads to a very weak overlap (only on one ID) and an overlap value of 0.08. And no overlap is found for the combination of ID 20 and ID 2. If we take into account an overlap threshold of 0.1, this implies that the two last NP pairs in the table below will not be selected.

Table 2. Example of overlap computation between the grouping of ID 2 and the groupings 5, 8 and 20.

Overlap	ID+NP	ID+NP
1	5 Loral Space	2 Loral Space
0.08	8 Globalstar	2 Loral Space
0	20 Lockheed Martin Corp.	2 Loral Space

3. For each pair with an overlap value above the threshold, compute the union of these pairs. Table 3 illustrates this procedure.

Table 3. Example output from the antecedent selection script. The table shows the incremental construction of one coreferential chain.

ID + Anaphor <- ID + Antecedent
8 Globalstar <- 43 Globalstar Telecommunications Ltd.
8 Globalstar <- 43 Globalstar Telecommunications Ltd. <- 64 Globalstar
8 Globalstar <- 43 Globalstar Telecommunications Ltd. <- 64 Globalstar <- 102 Globalstar
8 Globalstar <- 43 Globalstar Telecommunications Ltd. <- 64 Globalstar <- 102 Globalstar <- 103 Globalstar
8 Globalstar <- 43 Globalstar Telecommunications Ltd. <- 64 Globalstar <- 102 Globalstar <- 103 Globalstar <- 123 Globalstar
8 Globalstar <- 43 Globalstar Telecommunications Ltd. <- 64 Globalstar <- 102 Globalstar <- 103 Globalstar <- 123 Globalstar <- 139 Globalstar
8 Globalstar <- 43 Globalstar Telecommunications Ltd. <- 64 Globalstar <- 102 Globalstar <- 103 Globalstar <- 123 Globalstar <- 139 Globalstar <- 144 Globalstar
8 Globalstar <- 43 Globalstar Telecommunications Ltd. <- 64 Globalstar <- 102 Globalstar <- 103 Globalstar <- 123 Globalstar <- 139 Globalstar <- 144 Globalstar

3 Hybrid versus data-driven resolution

In order to rebalance the highly skewed data sets, positive and negative filters can be applied to the data. These filters split the basic set of instances in two parts: one part gets a label automatically assigned by the filter, the other part is classified by a classifier. There are several ways to look at this approach. It can be regarded as a language engineering approach, a preprocessing trick, but it can also be made into a principled approach to creating hybrid knowledge-based and machine learning based systems where both approaches solve the problems they are best at. To be able to do this, a systematic study has to be undertaken of the effect of different possible filters.

3.1 Related research on instance selection

Some of the filters proposed in literature aim exclusively at the reduction of negative instances, reducing the positive class skewness. Strube et al. [8], for example, apply a number of filters, which reduce up to 50% of the negative instances. These filters are all linguistically motivated, e.g. discard an antecedent-anaphor pair (i) if the anaphor is an indefinite NP, (ii) if one entity is embedded into the other, e.g. if the potential anaphor is the head of the potential antecedent NP, (iii) if either pronominal entity has a value other than third person singular or plural in its agreement feature. And Yang et al. [9] use the following filtering algorithm to reduce the number of instances in the training set: (i) add the NPs in the current and previous two sentences and remove the NPs that disagree in

number, gender and person in case of pronominal anaphors, (ii) add all the non-pronominal antecedents to the initial candidate set in case of non-pronominal anaphors.

Others such as Ng and Cardie [7] and Harabagiu et al. [10] also try to filter out less important or very easy positive instances to force the learning algorithm to specialize on the more difficult cases. Ng and Cardie [7] propose both negative sample selection (the reduction of the number of negative instances) and positive sample selection (the reduction of the number of positive instances), both under-sampling strategies aiming to create a better coreference resolution system. Given the observation that one antecedent is sufficient to resolve an anaphor, they present a corpus-based method for the selection of easy positive instances, which is inspired by the example selection algorithm introduced in [10]. The assumption is that the easiest types of coreference relationships to resolve are the ones that occur with high frequencies in the training data. Harabagiu et al. [10] mine by hand three sets of coreference rules for covering positive instances from the training data by finding the coreference knowledge satisfied by the largest number of anaphor-antecedent pairs. The high confidence coreference rules, for example, look for (i) repetitions of the same expression, (ii) appositions or arguments of the same copulative verb, (iii) name alias recognitions, (iv) anaphors and antecedents having the same head. Whenever the conditions for a rule are satisfied, an antecedent for the anaphor is identified and all other pairs involving the same anaphor can be filtered out. Ng and Cardie [7] write an automatic positive sample selection algorithm that coarsely mimics the [10] algorithm by finding a confident antecedent for each anaphor. They show that system performance improves dramatically with positive sample selection. The application of both negative and positive sample selection leads to even better performance. But they mention a drawback in case of negative sample selection: it improves recall but damages precision.

Uryupina [11] distinguishes between four types of markables (pronouns, definites, named entities, and all the other NPs) and proposes different sample selection mechanisms, reflecting the different linguistic behavior of these anaphors. In cross-comparative results with and without instance selection she shows an increase on both speed and performance.

All previous approaches concentrate on instance selection through the application of linguistically motivated filters. In [12], this rebalancing of the data is done without any a priori knowledge about the task to be solved and linked to the specific learning behavior of a lazy learner (TIMBL) and an eager learner (RIPPER). This work shows that both learning approaches behave quite differently in case of skewness of the classes and they also react differently to a change in class distribution.

The described selection approaches provide very few results on the effect of these filters on performance. In case cross-comparative results are provided, this is done in a coarse-grained manner. In the remainder of this paper, we will discuss our selection of filters and investigate in a fine-grained fashion whether these filters contribute to classification performance and how.

3.2 Positive and negative filtering

Our hybrid approach works as follows. After instance creation, each instance is matched against the filter rule. The subset of instances that match with the filter rule are labeled by the filter. The other part of the instance set is handled by the classifier. The filter rule is applied to both training and test instances.

In order to assess the effect of filtering on classification results, we investigate the following filters:

- *fdef*: The first filter rule we investigated, filters out all instances containing an indefinite anaphor and assigns a negative label to these instances.
- The filter *fhead* filters out instances in which the anaphor and antecedent are located at a distance of more than three sentences from each other. Instances beyond the scope of three sentence, which share the same head word, are retained in the data set.
- The filter *fagree* applies to pronouns only and demands agreement between anaphor and antecedent. The filter removes instances in which the antecedent does not have the same number or an incompatible gender type.
- The filter rule *fmatch* is based on [7] and is the only filter that also assigns positive labels. The filter assigns a positive label to an instance that describes an anaphor and antecedent which have a complete string match (discarding determiner information). All other instances containing that same anaphor with another antecedent get a negative label.
- The filter *f3s* restricts the search space for pronouns to three sentences. The filter rule assigns all pronoun-antecedent pairs at a larger sentence distance a negative label. So this filter deliberately can remove part of the positive instances, based on the observation that most pronouns refer to a close-by antecedent.

These filters are also combined: the filter combination 1 combines the four negative filters, whereas the filter combination 2 combines all filters together.

On the one hand, these simple filter rules are aimed at the removal of negative instances to change the balance between positive and negative instances. On the other hand, some of the filter rules also deliberately remove positive instances. As explained in Section 2.3 one anaphor can be preceded by multiple coreferential antecedents, but we only need to resolve one antecedent to build up the coreferential chain. The filters *f3s*, *fhead* and *fmatch* are based upon this principle. The rule *f3s* is based on the observation that pronouns usually find an antecedent within three sentences. The filter therefore can filter out positive antecedents that are at a larger distance from the anaphor. The filter *fmatch* removes other possible antecedents if one confident antecedent (complete match) has been found. The *fhead* rule assigns a negative label to potentially coreferential NPs at a sentence distance larger than 3, if they do not share the same head word with the anaphor under consideration.

Table 4 gives an overview of the number of training instances that are left to be handled by the classifier after the different filters have been applied. The last column of the table shows the number of positive instances in those training

sets. The results show that these filters account for a large part of the data and indeed lead to a less skewed data set, except for the filters *f3s* and *fmatch*. The *fhead* filter has also removed a part of the positive examples but a much larger part of the negatives, leading in the end to a positive effect on the class balance. The combination 2 filter, for example, accounts for 91.7% of the data, only leaving 6,286 instances to be classified by the learner. These instances also have a less skewed distribution.

Table 4. Number of training instances after application of the filters. The second column gives the absolute numbers, whereas the third column shows the percentage of training instances left to be treated by the classifier. The last column shows the skewness of the class distribution in those data sets.

Filter	number	%inst	%pos
normal	76,920	100	8.5
fdef	64,656	84.1	9.2
fagree	66,786	86.8	9.2
f3s	59,183	76.9	7.5
fhead	15,041	19.6	19.5
fmatch	57,479	74.4	8.0
combi1	9,723	12.6	20.3
combi2	6,286	8.3	17.7

We go on to investigate whether the skewness of the data is indeed harmful for the classifiers and whether filtering leads to better classification results and a better overall performance.

3.3 Results

We first consider the results of the 25-fold cross validation experiments on the training set in which we evaluate the performance of the first step of our approach: classifying NP pairs as being coreferential or not. Table 5 shows the micro-averaged F-scores of both classifiers, on the left hand side computed on all training instances (the joint effect of filters and machine learning) and on the right hand side on the subset of instances classified by the learner (the work that is left after the application of the filters). In general, the overall F-scores (in the right column) of the hybrid systems measured are lower than the F-score of the systems without filtering (*default*). Specially the *f3s* filter has a low score which can be explained by the fact that this rule deliberately labels part of the positive training instances as being negative (the instances where the distance between pronouns and antecedents is larger than three). When we look at the subset of instances classified by a learner, we observe for MAXENT that each filter improves the F-score on the subset. For TIMBL we observe only for some of the filters an improvement. This observation for TIMBL is in line with earlier findings in [27, 12].

Another observation relates to skewness. Three of the filters (*fhead*, *combi1*, *combi2*) change the class balance between positive and negative instances drastically as shown in Table 4. We can observe for both MAXENT and TIMBL that these three filters lead to the highest classifier F-scores on the cross-validation data. We do not see this clear effect on the test set. We believe this is due to the difference in measurement. On the test set, we measure F-scores at coreference chain level, and we do not need to retrieve all positive instances to build the complete coreference chain.

Table 5. Summary of micro-averaged F-scores of 25-fold cross validation experiments on the training set for MAXENT and TIMBL with and without the different filters. The left part of the table shows the results of the combined filters and learners, whereas the right part of the table only considers the subset of instances classified by the learners.

	MAXENT	TIMBL	#num.	MAXENT	TIMBL
default	37.6	46.7	76,920	37.6	46.7
fdef	37.6	44.2	64,656	40.0	46.8
fagree	37.9	44.7	66,786	39.5	46.4
f3s	31.6	35.2	59,183	41.5	45.2
fhead	34.8	39.7	15,041	58.3	67.0
fmatch	43.1	43.6	57,479	39.0	39.7
combi1	29.3	31.3	9,723	65.9	70.8
combi2	31.5	30.5	6,286	55.6	54.0

We now discuss the results on the test set showing MUC-scores computed at the coreference chain level. We computed a baseline score by assigning each NP in the test set its most nearby NP as antecedent. This gives us a baseline score with a high recall of 63.1%, a precision of 22.7% and an F-score of 33.4%.

The results on the test set for MAXENT and TIMBL are shown in Table 6. For TIMBL, we observe that all hybrid systems improve the precision of the system at the expense of recall. Only in the case of the *combi1* filter this leads to a lower F-score; in all other cases the shift leads to a higher F-score. For MAXENT, all hybrid systems except *fagree* have a higher F-score on the test set than the default system. Each filter produces a higher precision and in the case of *f3s*, *fhead* and *fmatch* also a higher recall.

4 Concluding remarks

We have shown that two distinct learning techniques benefit from a combined approach of knowledge-based filtering and machine-learning based classification. We observe that our simple filter rules can provide a large reduction in the number of instances to be classified. The filters improve the overall precision of the system on the test set leading to higher F-scores in almost all experiments.

Table 6. MUC scores on the test set of TIMBL and MAXENT with and without the different filters.

	TIMBL			MAXENT		
	recall	precision	F-score	recall	precision	F-score
normal	60.0	35.2	44.4	41.7	42.2	42.0
fdef	49.2	46.7	47.9	39.5	46.4	42.7
f3s	58.0	36.8	45.1	51.2	43.8	47.2
fagree	50.2	40.4	44.7	41.3	42.3	41.8
fhead	39.8	60.3	47.9	45.5	42.7	44.1
fmatch	46.7	48.4	47.5	51.2	42.4	46.4
combi1	40.7	46.1	43.2	38.5	51.6	44.1
combi2	36.7	61.0	45.8	40.0	51.8	45.1

Most successful is the *combi2 filter* which combines five simple filter rules and leads to a large instance reduction of up to 92%, and produces a better F-score on the test set for both MAXENT and TIMBL.

As future work we plan to investigate the filter rules in contrast to a machine learning approach in which the feature weights correspond to the filters are boosted. It would also be interesting to investigate whether similar filter rules have a similar positive effect for other languages. In contrast to a pure machine learning approach, a hybrid approach has the disadvantage that it may require careful re-engineering of the knowledge-based part for different languages.

References

1. Cardie, C., Wagstaff, K.: Noun phrase coreference as clustering. In: Proceedings of the 1999 joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora. (1999) 82–89
2. Quinlan, J.: C4.5: Programs for machine learning. Morgan Kaufmann, San Mateo, CA (1993)
3. Aone, C., Bennett, S.: Evaluating automated and manual acquisition of anaphora resolution strategies. In: Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics (ACL-1995). (1995) 122–129
4. McCarthy, J.: A Trainable Approach to Coreference Resolution for Information Extraction. PhD thesis, Department of Computer Science, University of Massachusetts, Amherst MA (1996)
5. Soon, W., Ng, H., Lim, D.: A machine learning approach to coreference resolution of noun phrases. Computational Linguistics **27** (2001) 521–544
6. Cohen, W.W.: Fast effective rule induction. In: Proceedings of the 12th International Conference on Machine Learning (ICML-1995). (1995) 115–123
7. Ng, V., Cardie, C.: Combining sample selection and error-driven pruning for machine learning of coreference rules. In: Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP-2002). (2002) 55–62
8. Strube, M., Rapp, S., Müller, C.: The influence of minimum edit distance on reference resolution. In: Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP-2002). (2002) 312–319

9. Yang, X., Zhou, G., Su, S., Tan, C.: Coreference resolution using competition learning approach. In: Proceedings of the 41th Annual Meeting of the Association for Computational Linguistics (ACL-03). (2003) 176–183
10. Harabagiu, S., Bunescu, R., Maiorano, S.: Text and knowledge mining for coreference resolution. In: Proceedings of the 2nd Meeting of the North American Chapter of the Association of Computational Linguistics (NAACL-2001). (2001) 55–62
11. Uryupina, O.: Linguistically motivated sample selection for coreference resolution. In: Proceedings of DAARC-2004. (2004)
12. Hoste, V.: Optimization Issues in Machine Learning of Coreference Resolution. PhD thesis, Antwerp University (2005)
13. MUC-7: Muc-7 coreference task definition. version 3.0. In: Proceedings of the Seventh Message Understanding Conference (MUC-7). (1998)
14. Davies, S., Poesio, M., Bruneseaux, F., Romary, L.: Annotating coreference in dialogues: Proposal for a scheme for mate. http://www.hcrc.ed.ac.uk/poesio/MATE/anno_manual.htm (1998)
15. van Deemter, K., Kibble, R.: On coreferring: Coreference in muc and related annotation schemes. *Computational Linguistics* **26** (2000) 629–637
16. Daelemans, W., Zavrel, J., Berck, P., Gillis, S.: Mbt: A memory-based part of speech tagger generator. In: Proceedings of the 4th ACL/SIGDAT Workshop on Very Large Corpora. (1996) 14–27
17. Tjong Kim Sang, E., Daelemans, W., Höthker, A.: Reduction of dutch sentences for automatic subtitling. In: *Computational Linguistics in the Netherlands 2003. Selected Papers from the Fourteenth CLIN Meeting*. (2004) 109–123
18. Daelemans, W., van den Bosch, A.: *Memory-based Language Processing*. Cambridge University Press (2005)
19. Guiasu, S., Shenitzer, A.: The principle of maximum entropy. *The Mathematical Intelligencer* **7** (1985)
20. Berger, A., Della Pietra, S., Della Pietra, V.: Maximum Entropy Approach to Natural Language Processing. *Computational linguistics* **22** (1996)
21. Le, Z.: Maximum Entropy Modeling Toolkit for Python and C++ (version 20041229). Natural Language Processing Lab, Northeastern University, China. (2004)
22. van den Bosch, A.: Wrapped progressive sampling search for optimizing learning algorithm parameters. In: Proceedings of the 16th Belgian-Dutch Conference on Artificial Intelligence. (2004) 219–226
23. Kohavi, R., John, G.H.: Wrappers for feature subset selection. *Artificial Intelligence* **97** (1997) 273–323
24. F. Provost, D.J., Oates, T.: Efficient progressive sampling. In: Proceedings of the Fifth International Conference on Knowledge Discovery and Data Mining. (1999) 23–32
25. Vilain, M., Burger, J., Aberdeen, J., Connolly, D., Hirschman, L.: A model-theoretic coreference scoring scheme. In: Proceedings of the Sixth Message Understanding Conference (MUC-6). (1995) 45–52
26. Connolly, D., Burger, J., Day, D.: A machine learning approach to anaphoric reference. In: Proceedings of the International Conference on ‘New Methods in Language Processing’. (1994)
27. Daelemans, W., van den Bosch, A., Zavrel, J.: Forgetting exceptions is harmful in language learning. *Machine Learning* **34** (1999) 11–41