Detecting reference chains in Norwegian

Anders Nøklestad
Dept. of Linguistics
Univ. of Oslo
P.O. Box 1102 Blindern
NO-0317 Oslo
Norway
Anders.Noklestad@ilf.uio.no

Christer Johansson
Dept. of Linguistics
Univ. of Bergen
Sydnesplassen 7
NO-5007 Bergen
Norway
Christer.Johansson@lili.uib.no

Abstract

This article describes a memory-based mechanism for anaphora resolution and coreference. A program labels the words in the text with part-of-speech tags, functional roles, and lemma forms. This information is used for generating a representation of each anaphor and antecedent candidate. Each potential anaphor-antecedent pair has a match vector calculated, which is used to select similar cases from a database of labeled cases. We are currently testing many feature combinations to find an optimal set for the task. The most recent results show an overall F-measure (combined precision and recall) of 62, with an F-measure of 40 for those cases where anaphor and antecedent are non-identical, and 81 for identical ones. The coreference chains are restricted so that an anaphor is only allowed to link to the last item in a chain.

1 Introduction

This article discusses a method for deciding whether an anaphor is coreferential with a potential antecedent. An early algorithm (Hobbs, 1978) performs searches for antecedents in parsed syntactic trees. Preferences in the model are present in the order of the tree search. Another approach is to model the salience of possible antecedents (Lappin and Leass, 1994). Salience factors are weighted somewhat ad hoc, but with a rank-order inspired by observed ratings of salience in psycho-linguistic experiments. A third influential theory is Centering Theory (Grosz et al., 1995). Centering too, assumes a salience rating, but has an added constraint that there is a single item which is in focus at any given time and therefore most likely to be pronominalized. None of these models can be regarded as robust models, as they rely on fixed methods, which are not changed by experience. Hobbs' method relies heavily on parsing, and depends on having the correct parse trees available. Lappin and Leass' approach relies on a saliency rating, as well as parsing and finding functional roles. Their algorithm also has a heuristics that takes distance into account. Centering depends on maintaining forward and backward looking centers, and selecting the most likely candidate to be in focus.

Recently, anaphora resolution has been performed by machine learning techniques, using resources that are less demanding. For example, Lappin and Leass' algorithm has been modified so that it can operate on the result of a statistical part-of-speech tagger (Kennedy and Boguraev, 1996). Cardie and Wagstaff (1999) introduced an unsupervised clustering technique that performs better than many handcrafted systems on the related task of noun phrase coreference. Because the algorithm is unsupervised, finding coreference does not rely on pre-classified exemplars. Other benefits of using a clustering technique is that it can more easily be adapted to different domains. Coreference can also be viewed as a classification task. A comparison between decision tree learning (classification) and the clustering algorithm, shows, not surprisingly, that training on pre-classified examples can provide better results (Soon et al., 2001). An F-ratio of 62.6 was obtained for decision tree learning, whereas the clustering algorithm produced a measure of 53.6 on the same dataset (MUC-6). There is, however, a slight gap to the best system, which

S. Werner (ed.), *Proceedings of the 15th NODALIDA conference, Joensuu 2005*, Ling@JoY 1, 2006, pp. 156–162
ISBN 952-458-771-8, ISSN 1796-1114. http://ling.joensuu.fi/lingjoy/

produced a measure of 65, according to Cardie and Wagstaff (1999, p.82).

This article will discuss the use of memory-based learning (using the Tilburg Memory Based Learner application, known as TiMBL) in anaphora resolution. Exemplar based models have been tried before, as noted above in the works on clustering (Cardie and Wagstaff, 1999), and decision tree learning (Soon et al., 2001). TiMBL internally uses decision tree techniques to make faster nearest neighbor decisions. The main advantage of, and theoretical point of, memory-based learning is that all information is regarded as useful. All examples may contribute to a correct classification, even rare examples, exceptions and sometimes even examples of misclassifications (Bosch and Daelemans, 1998). A further motivation is to let the exemplars decide the classification as far as possible, and not the prejudices of a biased linguist, who enters the task with (too) much background knowledge about how the phenomenon should work, and (too) little coverage of all the cases that are exceptions, restricted to the situation, or limited by performance demands for efficient communication. We still claim that a linguistic analysis is necessary, as we have no other means of judging which factors might contribute to the solution. The decision is however let to the examples, when it comes to judging how the factors we have pre-selected interact towards a classification decision. We should also acknowledge that there are certainly many relevant factors that have not been selected for representation, either because they did not contribute to the solution with the current set of features, or because we have not thought of them.

In what follows, we will first explain the memory-based machine learning approach, and then we will describe our database and the features we have used in order to train our classifier on the task of anaphora resolution. Finally, we report on the results of running 10-fold cross-validation experiments, which are done as a test of classification ability of this method. In 10-fold cross-validation, a tenth of the training data is left out and classified by the rest of the instances. This is repeated until all items have been classified this way. This validation is different from using entirely new test data, since it is guaranteed to be within the types of texts we have selected for training. (In our web demonstrator[1], we have chosen a test text based on a children's story, which is definitely in a different genre than the training examples.)

2 Memory-based learning

We decided to work with memory-based learning, as implemented in the TiMBL software package (Daelemans et al., 2004). This decision was inspired by anaphora resolution results from decision tree learning (Soon et al., 2001), which is a type of exemplar based machine learning mechanism. We have chosen a slightly different approach, using a nearest neighbor mechanism, within the field of memory-based learning. TiMBL is an efficient implementation of this algorithm, and recent results using this software package has shown good results, often better than decision tree learning, for example on chunking (Tjong Kim Sang and Buchholz, 2000). Decision tree learning does not use all the examples. It is what is sometimes classified as a greedy learner (i.e. trying to make generalizations as soon as possible), whereas TiMBL is a lazy learner (i.e. postponing the classification decision until it is needed).

Memory-based learning is a kind of k-nearest neighbor approach, which operates on exemplars, or *instances*, in the form of feature vectors. It is a supervised machine learning method that requires an annotated training corpus. During the training phase, training instances are stored in a database, or *instance base*, along with their annotated categories.

Classifying a new instance amounts to comparing it to the instances in the instance base, finding the ones that are most similar to the new instance (i.e., its *nearest neighbors*), and selecting the category that is shared by the majority of those neighbors. The notion of similarity is taken in a geometrical sense, as examples are represented as points, with an associated category, in an n-dimensional space. The degree of similarity between instances is influenced by the choice of similarity metric and feature weighting scheme. The size of the neighborhood is also influenced by the value of *k* used for the *k*-nearest neighbor classifier. If k=1, only the most similar instances are used, with k=2, the next best matches are also included, and so on.

[1] see demo at http://bredt.uib.no

The classification process is illustrated in general terms in table 1, which shows three training instances and a test instance. The features can take the values plus or minus. The reader is encouraged to look for feature values that are the same for the test instance and the training instance.

The test instance matches Train1 on Feature1 and Feature2, and it matches Train2 on Feature2 and Feature4, while Train3 only matches on Feature2. Thus, Train1 and Train2 constitute the set of nearest neighbors for Test. Since the majority (in this case all) of the nearest neighbors share the C1 category, this category is also selected for the Test instance.

	Train1	Train2	Train3	Test
Feature1	**+**	–	–	+
Feature2	**–**	**–**	**–**	–
Feature3	–	–	–	+
Feature4	+	**–**	+	–
Category	C1	C1	C2	*C1*

Table 1: Classification based on nearest neighbors.

3 Anaphora resolution

We use a corpus which has been automatically annotated for part-of-speech tags and syntactic categories, and hand annotated for coreference. The algorithm was trained on a subset of 41970 tokens, of which 35780 were words and the rest typographical annotations. From these 35780 words, we have 11831 markables (potential members of reference chains).

In order to experiment with a more limited material, we selected a set of pronouns which had all been marked either with or without an antecedent. Pronouns without antecedents refer to something not present in the (preceding) text. The chosen pronouns were *han* "he", *hun*, "she", *seg* "himself/herself/itself/themselves", *den* "it" masc./fem., and *de* "they". The set of pronouns consisted of 2199 anaphor candidates, of which 2129 were marked with a coreferential antecedent in the text. Cataphors were excluded from the training set.

3.1 Salience

The classification of an anaphor–antecedent pair is affected by a number of features, such as whether they agree in number, person, case and gender. All these features create subdivisions of the representational space. Since the model is not probabilistic, it is not correct to associate probabilities to this selection. The method selects from the nearest neighbors, regardless of their likelihood of occurrence, or likelihood of predicting the correct class. Additional constraints may stem from syntactic restrictions (for example, whether a reflexive interpretation is possible), and selectional restrictions (e.g., which are the typical objects of specific verbs). Real knowledge of what is possible, may help. Recency, repeated mention and parallelism are other important factors in deciding coreference (Jurafsky and Martin, 2000, pp.678–684).

The factors we use are taken from a machine annotated corpus, so not all of these factors necessarily have correct values.

The factors are as follows (see also table 3):

- Do the lemmas of the anaphor candidate and the antecedent candidate match?
- Do the anaphor candidate and the antecedent candidate have the same
  - syntactic function?
  - grammatical gender? (We also want to add natural gender in the future.)
  - number?
- Is the anaphor candidate a distinctly human pronoun (*han* "he" or *hun* "she") and the antecedent candidate a proper name?
  - If so, do they have the same (natural) gender?
- Is the anaphor candidate a reflexive and the antecedent candidate its closest subject?
- Is the antecedent candidate a subject?
- Is the number of sentence boundaries, as detected by the tagger (e.g. some conjunctions, and punctuation), between anaphor and antecedent candidates
  - less than 2?
  - less than 3?
- The lemmas of the anaphor and the antecedent candidates concatenated

Note that we have no information about natural gender for lexical noun phrases, which is information we would want to have, in addition to grammatical gender. Various ways of finding the natural gender (Hale and Charniak, 1998, inter al., for English) will be tried out later in our project.

Since we are using memory-based learning for this task, the full impact of each feature is determined after the nearest neighbors have been found. A feature that is highly valuable in some contexts may simply not add anything in other contexts, for example if all the selected nearest neighbors contain the same feature. This is an advantage of memory-based learning, since it does not rely heavily on global statistics. TiMBL may also weigh features by their informativeness, by calculating the information gain of each feature (i.e. how much we gain in classification accuracy by knowing the value of the feature). Still, TiMBL is essentially a selection mechanism and not a probabilistic mechanism.

3.2 Percolation

A strategy to percolate most of the matching features to the next referring expression in a chain was adapted. This means that the most recent antecedent candidate matches the union of its own matching features and those of the preceding antecedent candidates of the same chain. Sometimes information is not available immediately, but will be known after we have established coreference (see table 2) .

3		3.1		3.2
Calvin	→	who	→	he
—		Calvin		—
—		....		—

Table 2: Percolation of *Calvin* to *who*

Take for example the following discourse, adapted and translated from a folk tale: *"[Three brothers lived in a forest.] The oldest was called* $Adam_1$*, the second* $Bert_2$*, and the youngest* $Calvin_3$*,* $who_{3.1}$ *used to tend to the* $ashes_4$*. The* $Sunday_5$ *when the* $notice_6$ *from the* $King_7$ *about the* $ship_8$ *was posted,* $he_{3.2}$ *happened to be there. When* $he_{3.3}$ *came home and told about* $it_{6.1}$*,* $Adam_{1.1}$ *wanted to set out immediately, and prepared some* $food_9$ *for the* $journey_{10}$*. For* $he_?$ *wanted to find out ..."*

The second time that *he* refers to Calvin at point 3.2, the information from the first mention of Calvin at point 3 has been percolated to the *who* at 3.1. After linking up, the information in 3.2 contains the (lemma)/name "Calvin", the functional roles of position 3 (predicate filler), 3.1 (subject), and 3.2 (subject), as well as the number (singular), and the gender (masculine). If we presume that the name Calvin was not found in the lexical resources, then the gender of Calvin may be established by a co-occurrence relation.

feature	3	3.1	$3 \cup 3.1$
match on			
lemma	–	–	–
syn.func.	–	+	+
gender	–	–	–
number	+	+	+
human pro.			
and prop.	+	–	+
and gen	–	–	–
refl+subject	–	–	–
subject-ant	–	+	+
$dist. < 2$	+	+	+
$dist. < 3$	+	+	+

Table 3: Match between *he* at 3.2. and antecedents *Calvin*(3), or *who* (3.1), or $3 \cup 3.1$) with percolation

This strategy is thought to be important for full anaphora resolution. From table 3, we see that we get a match vector with six matching features with percolation, instead of four features for match with Calvin, and five features matching *who*. It is an open question whether there should be a lemma match between a pronoun and the same pronoun only, or if a pronoun should be able to unify with all kinds of strings for surface match. We have decided to allow a lemma match between the same form of pronouns only, but we will try using an unknown value for this type of match. Notice that it would be a good idea to have three values for gender matches: $+$, $-$, and *unknown*. If *Calvin* was found to be a male name, for example from a list of male names, we would be able to access a masculine gender for both *Calvin* and *who*. (This is not to say that "who" is a word with inherent gender.) An unknown value would be good to have when we cannot disprove a match. In addition, we would create and search

for the concatenated lemmas $he/Calvin$, and he/who respectively. These items are not percolated, but contain the value of the candidate antecedent and the anaphor.

Table 4 shows the match vectors, after percolation, for $he_?$ matching with either *Adam* or *Calvin*. As can be seen, *Calvin* matches on more features than *Adam*. Still, Adam might be selected as antecedent because it is closer to the anaphor. This is due to the fact that the search for an antecedent moves backwards in the text, starting at the anaphor and stopping as soon as there is a positive classification. Hence, if the match vector for Adam has more positive than negative nearest neighbors in the instance base, *Adam* will be selected as antecedent, and *Calvin* will never be considered.

feature	1.1	3.3
match on		
lemma	–	+
syn.func.	+	+
gender	–	+
number	+	+
human pro.		
and prop.	+	+
and gen	–	–
refl+subject	–	–
subject-ant	+	+
$dist. < 2$	–	–
$dist. < 3$	+	–

Table 4: Match vectors for $he_?$, with $Adam_{1.1}$, and $he_{3.3}$/*Calvin*

Two important points are illustrated. First, that a closer antecedent will have an advantage, because it will be selected before its competitors if TiMBL decides for it, since the search for an antecedent is stopped when there is a positive classification. Second, that the final categorization does not depend on how many matches are recorded, but on how the full vector and its neighbors have been classified previously. A last point is that proper names are assumed to be in the singular; however, for some types of proper names (e.g. organizations) the *number* featuremay have a different default than singular, or may come from a knowledge base. This is crucially an issue of what our lexical resources will deliver.

3.3 Future extensions

All the features we are using are language independent, and can be found with commonly available tools for part-of-speech tagging and functional role assignment. We will therefore be able to extend our work fairly easily to handle other similar languages, where appropriate taggers are available, such as Dutch, English and Swedish. Databases of correctly annotated examples will have to be built up incrementally, but it might help to have a system that can suggest correct anaphor–antecedent relations in a majority of the cases automatically. This procedure is enhanced if a tool is provided that makes it easy to correct mistakes made by the automatic procedure. We have work in progress that will provide such a tool on the internet. If people use it we will also benefit from their work, since we will have access to the annotated data after editing, and because we will get useful feedback on the mechanism. A very interesting research question is whether it is possible to use a common database for many different languages. Will the examples from different languages support or interfere with each other? Factors such as matching functional roles, parallelism, semantic similarity and function, may very well vary in a similar way across many (related) languages. We are further helped by the fact that only factors that are important in the *target* language (the language of the text to be processed) will be important. For example, if a language does not have grammatical gender it can not discriminate on that feature for any of the languages in the database, it will be as if the feature did not exist. When it comes to functional roles, they are designed to be annotated similarly across languages.

4 Training and testing

Several different combinations of the available features were tried, and the previously presented 9 features were those that gave the highest scores. They are most likely not the optimal features, but they are the features that are available in our machine tagged example collection. We have only scored hits on the closest antecedent in a chain, whereas it could be argued that a hit on any antecedent in a chain would suffice.

Selection	Proportion	Recall	Precision	F-measure
Identical anaphor-antecedent	67%	83.95	78.88	81.31
Non-identical anaphor-antecedent	33%	40.75	40.32	40.51
All cases	100%	62.12	62.18	62.14

Table 5: Results from 10-fold cross-validation experiments.

Feature percolation allows previously found antecedents of a chain to influence the decision on anaphoric status. Features percolate and accumulate towards the most recent member of a chain.

Training consists of providing postive and negative examples to the memory-based learner. Positive examples are pairs of anaphor and antecedent candidates, described by how they match on the features we have decided to include. Negative examples are pairs consisting of an anaphor and any markable that is closer to the anaphor than the actual antecedent. Typically, there are many more negative examples than positive examples.

In testing, we start searching from the pronoun (or potential anaphor, in the more general case of coreference detection) backwards in the text until the algorithm finds a markable that is classified as an antecedent for the pronoun. The classification decision is binary, so we assign the first found candidate marked by the mechanism as an antecedent. We have experimented with using the strength of the classification decision (recalculated into z-scores, for general comparison), but this did not improve the results and was abandoned.

The classification decision is therefore simplistic: The memory-based learner is consulted for each markable that is encountered. If the positive nearest neighbor examples in the database outvote the negative examples, the classification is a yes, otherwise a no. When the mechanism says yes, there is no need to search further.

The results of our cross-validation experiments are shown in Table 5. The overall F-measure is 62.14. It should be noted, however, that the system performs much better in those cases where the anaphor and the antecedent are identical than in cases where they are not identical. Closely related to this observation is the fact that in 78% of the test cases, the classifier selects an antecedent that is identical to the anaphor. This strong tendency to select identical pairs is likely due to the fact that 67% of all manually annotated anaphor–antecedent pairs in the training data fit this characteristic. This is particularly noticeable for *han* “he” and *hun* “she”, which also account for 78% of the relevant pronouns. The problem with this pattern is that we often miss many of the more interesting pairs, where there is a non-pronoun as antecedent. We simply do not have enough recall to find these more interesting cases. The examples favor a negative classification, simply because this is correct most of the time. A simple bias towards positive answers will hurt our precision, but this could be worth it if we could quantify how much more such a pair is worth. This is a highly task dependent decision. In many tasks, chains containing only pronouns would not make much difference, whereas a correct chain that links with keywords would be highly valuable (for example in estimating the topic of a paragraph).

We performed our 10-fold cross-validation experiments on a corpus of novels, which is a complex type of text with typically elaborate and long reference chains. Newspaper texts, in contrast, select for shorter, coherent paragraphs since space is a more limited resource in this domain.

5 Conclusion

We have presented a system for automatic resolution of pronominal anaphora in Norwegian that is based on memory-based learning. The system reaches a performance level, which is comparable to that of earlier systems developed for English in the context of the Message Understanding Conferences.

The performance of the system is considerably better on identical anaphor-antecedent pairs than on non-identical pairs, reflecting the higher proportion of identical pairs in the training data. We are currently testing new feature combinations, and the addition of more lexical resources, some of which are found by statistical association, in order to improve results.

We plan to apply our system to related languages, in order to see how language independent our selected features are. We would like to investigate whether patterns learned from the Norwegian data could actually be applied directly in order to perform anaphora resolution in these other languages without having to re-train the system for each language.

Acknowledgements The BREDT project is supported by a grant from the Norwegian Research Council under the KUNSTI programme. The first author is supported by a Ph.D. grant from the University of Oslo. We thank Dr Kaja Borthen for many valuable comments. A demonstrator of early results for anaphora resolution is available from http://bredt.uib.no

References

A. van den Bosch and W. Daelemans. 1998. Do not forget: Full memory in memory-based learning of word pronunciation. In D.M.W. Powers, editor, *proceedings of NeMLap3/CoNLL98*, pages 195–204, Sydney, Australia.

C. Cardie and K. Wagstaff. 1999. Noun phrase coreference as clustering. In *Proc. of the joint SIGDAT Conf. on Empirical Methods in NLP and Very Large Corpora*, pages 82–89.

W. Daelemans, J. Zavrel, K. van der Sloot, and A. van der Bosch. 2004. TiMBL: Tilburg Memory-Based Learner, Version 5.1, Reference Guide. Technical Report ILK 04–02, the ILK Group, Tilburg, the Netherlands.

B.J. Grosz, A. Joshi, and S. Weinstein. 1995. Centering: A framework for modeling the local coherence of discourse. *Computational Linguistics*, 21(2):203–225.

J. Hale and E. Charniak. 1998. Getting Useful Gender Statistics from English Text. Technical Report CS-98-06, Comp. Sci. Dept. at Brown University, Providence, Rhode Island.

J.R. Hobbs. 1978. Resolving pronoun references. *Lingua*, 44:311–338.

D. Jurafsky and J.H. Martin. 2000. *Speech and Language Processing*. Prentice Hall, New Jersey.

C. Kennedy and B. Boguraev. 1996. Anaphora for everyone: Pronominal anaphora resolution without a parser. In *Proceedings of the 16th International Conference on Computational Linguistics.*, Copenhagen, Denmark.

S. Lappin and H. J. Leass. 1994. An algorithm for pronominal anaphora resolution. *Computational Linguistics*, 20(4):535–561.

Wee Meng Soon, Hwee Tou Ng, and D. Chung Yong Lim. 2001. A machine learning approach to coreference resolution of noun phrases. *Computational Linguistics*, 27(4):521–544.

E. F. Tjong Kim Sang and S. Buchholz. 2000. Introduction to the CoNLL-2000 Shared Task: Chunking. In *Proceedings of CoNLL-2000 and LLL-2000*, pages 127–132. Lisbon, Portugal.