Exploring Lexicalized Features for Coreference Resolution

Anders Björkelund
Lund University / LTH
Lund / Sweden
Anders.Bjorkelund@cs.lth.se

Pierre Nugues
Lund University / LTH
Lund / Sweden
Pierre.Nugues@cs.lth.se

Abstract

In this paper, we describe a coreference solver based on the extensive use of lexical features and features extracted from dependency graphs of the sentences. The solver uses Soon et al. (2001)'s classical resolution algorithm based on a pairwise classification of the mentions.

We applied this solver to the closed track of the CoNLL 2011 shared task (Pradhan et al., 2011). We carried out a systematic optimization of the feature set using cross-validation that led us to retain 24 features. Using this set, we reached a MUC score of 58.61 on the test set of the shared task. We analyzed the impact of the features on the development set and we show the importance of lexicalization as well as of properties related to dependency links in coreference resolution.

1 Introduction

In this paper, we present our contribution to the closed track of the 2011 CoNLL shared task (Pradhan et al., 2011). We started from a baseline system that uses Soon et al. (2001)'s architecture and features. Mentions are identified by selecting all noun phrases and possessive pronouns. Then, the resolution algorithm relies on a pairwise classifier that determines whether two mentions corefer or not.

Lexicalization has proved effective in numerous tasks of natural language processing such as part-of-speech tagging or parsing. However, lexicalized models require a good deal of annotated data to avoid overfit. The data set used in the CoNLL 2011 shared task has a considerable size compared to corpora traditionally used in coreference resolution – the training set comprises 2,374 documents. See Pradhan et al. (2007) for a previous work using an earlier version of this dataset. Leveraging this size, we investigated the potential of lexicalized features.

Besides lexical features, we created features that use part-of-speech tags and semantic roles. We also constructed features using dependency tree paths and labels by converting the constituent trees provided in the shared task into dependency graphs. The final feature set was selected through an automated feature selection procedure using cross-validation.

2 System Architecture

During both training and decoding, we employed the same mention detection and preprocessing steps. We considered all the noun phrases (NP) and possessive pronouns (PRP$) as mentions. In order to extract head words from the NP constituents, we converted the constituent trees provided in the data sets to dependency graphs using the Penn treebank converter of Johansson and Nugues (2007). Using the dependency tree, we extracted the head word of all the NPs by taking the word that dominates the subtree constructed from the NP.

The dependency tree is also used later to extract features of mentions based on dependency tree paths, which is further described in Sec. 3.

In the preprocessing step, we assigned a number and a gender to each mention. For the pronominal mentions, we used a manually compiled lists of pronouns, where we marked the number and gender.

Proceedings of the 15th Conference on Computational Natural Language Learning: Shared Task, pages 45–50,
Portland, Oregon, 23-24 June 2011.

For nonpronominal mentions, we used the number and gender data (Bergsma and Lin, 2006) provided by the task organizers and queried it for the head word of the mention. In cases of ambiguity (e.g. the pronoun *you*), or missing entries in the data for nonpronominals, we assigned an *unknown* value.

2.1 Generation of training examples

To create a set of training examples, we used pairs of mentions following the method outlined by Soon et al. (2001). For each anaphoric mention m_j and its closest preceding antecedent m_i, we built a positive example: $P = \{(m_i, m_j)\}$. We constructed the negative examples with noncoreferring pairs of mentions, where the first term is a mention occurring between m_i and m_j and the second one is m_j: $N = \{(m_k, m_j)|i < k < j)\}$.

The training examples collected from the CoNLL 2011 training set consist of about 5.5% of positive examples and 94.5% of negative ones.

2.2 Learning method

We evaluated two types of classifiers: decision trees and logistic regression. We used the decision trees and the C4.5 algorithm from the Weka distribution (Hall et al., 2009) for our baseline system. We then opted for linear logistic regression as it scaled better with the number of features and feature values.

Logistic regression is faster to train and allowed us to carry out an automated feature selection, which is further described in Sec. 3.4. In addition, the logistic classifiers enabled us to interpret their results in terms of probabilities, which we used for the decoding step. We trained the logistic regression classifiers using the LIBLINEAR package (Fan et al., 2008).

2.3 Decoding

The decoding algorithm devised by Soon et al. (2001) selects the closest preceding mention deemed to be coreferent by the classifier. This clustering algorithm is commonly referred to as *closest-first clustering*. Ng and Cardie (2002) suggested a different clustering procedure, commonly referred to as *best-first clustering*. This algorithm selects the most likely antecedent classified as coreferent with the anaphoric mention. During early experiments, we found that while the best-first method increases the performance on nonpronominal anaphoric expressions, it has the opposite effect on pronominal anaphoric expressions. Consequently, we settled on using the closest-first clustering method for pronominal mentions, and the best-first clustering method otherwise. For the best-first clustering, we used the probability output from our logistic classifiers and a threshold of 0.5.

After clustering mentions in a document, we discard all remaining singleton mentions, as they were excluded from the annotation in the CoNLL 2011 shared task.

2.4 Postprocessing

The initial detection of mentions is a direct mapping from two categories of constituents: NP and PRP$. In the postprocessing step, we reclaim some of the mentions that we missed in the initial step.

The automatically generated constituent trees provided in the data set contain errors and this causes the loss of many mentions. Another source of loss is the bracketing of complex NPs, where the internal structure uses the tag NML. In a few cases, these nested nodes participate in coreference chains. However, when we tried to include this tag in the mention detection, we got worse results overall. This is possibly due to an even more skewed distribution of positive and negative training examples.

In the postprocessing step, we therefore search each document for sequences of one or more proper noun tokens, i.e. tokens with the part-of-speech tags NNP or NNPS. If their common ancestor, i.e. the parse tree node that encloses all the tokens, is not already in a mention, we try to match this sequence to any existing chain using the binary features: STRINGMATCH and ALIAS (cf. Sec. 3). If either of them evaluates to true, we add this span of proper nouns to the matched chain.

3 Features

For our baseline system, we started with the feature set described in Soon et al. (2001). Due to space limitations, we omit the description of these features and refer the reader to their paper.

We also defined a large number of feature templates based on the syntactic dependency tree, as well as features based on semantic roles. In the fol-

lowing sections, we describe these features as well as the naming conventions we use. The final feature set we used is given in Sec. 4.

3.1 Mention-based features

On the mention level, we considered the head word (HD) of the mention, and following the edges in the dependency tree, we considered the left-most and right-most children of the head word (HDLMC and HDRMC), the left and right siblings of the head word (HDLS and HDRS), as well as the governor[1] of the head word (HDGOV).

For each of the above mentioned tokens, we extracted the surface form (FORM), the part-of-speech tag (POS), and the grammatical function of the token (FUN), i.e. the label of the dependency edge of the token to its parent. For head words that do not have any leftmost or rightmost children, or left or right siblings, we used a null-value placeholder.

In each training pair, we extracted these values from both mentions in the pair, i.e. both the anaphor and the tentative antecedent. Table 3 shows the features we used in our system. We used a naming nomenclature consisting of the role in the anaphora, where I stands for antecedent and J for anaphor; the token we selected from the dependency graph, e.g. HD or HDLMC; and the value extracted from the token, e.g. POS or FUN. For instance, the part-of-speech tag of the governor of the head word of the anaphor is denoted: J-HDGOVPOS.

The baseline features taken from Soon et al. (2001) include features such as I-PRONOUN and J-DEMONSTRATIVE that are computed using a word list and by looking at the first word in the mention, respectively. Our assumption is that these traits can be captured by our new features by considering the part-of-speech tag of the head word and the surface form of the left-most child of the head word, respectively.

3.2 Path-based features

Between pairs of potentially coreferring mentions, we also considered the path from the head word of the anaphor to the head word of the antecedent in the syntactic dependency tree. If the mentions are not in the same sentence, this is the path from the anaphor to the root of its sentence, followed by the path from the root to the antecedent in its sentence. We differentiate between the features depending on whether they are in the same sentence or in different sentences. The names of these features are prefixed with SS and DS, respectively.

Following the path in the dependency tree, we concatenated either the surface form, the part-of-speech tag, or the grammatical function label with the direction of the edge to the next token, i.e. up or down. This way, we built six feature templates. For instance, DSPATHFORM is the concatenation of the surface forms of the tokens along the path between mentions in different sentences.

Bergsma and Lin (2006) built a statistical model from paths that include the lemma of the intermediate tokens, but replace the end nodes with *noun*, *pronoun*, or *pronoun-self* for nouns, pronouns, and reflexive pronouns, respectively. They used this model to define a measure of coreference likelihood to resolve pronouns within the same sentence. Rather than building an explicit model, we simply included these paths as features in our set. We refer to this feature template as BERGSMALINPATH in Table 3.

3.3 Semantic role features

We tried to exploit the semantic roles that were included in the CoNLL 2011 data set. Ponzetto and Strube (2006) suggested using the concatenation of the predicate and the role label for a mention that has a semantic role in a predicate. They introduced two new features, I_SEMROLE and J_SEMROLE, that correspond to the semantic roles filled by each of the mentions in a pair. We included these features in our pool of feature templates, but we could not see any contribution from them during the feature selection.

We also introduced a number of feature templates that only applied to pairs of mentions that occur in the same semantic role proposition. These templates included the concatenation of the two labels of the arguments and the predicate sense label, and variations of these that also included the head words of either the antecedent or anaphor, or both. The only feature that was selected during our feature selection procedure corresponds to the concatenation of the argument labels, the predicate sense, and the head word of the anaphor: SEMROLEPROPJHD in Table 3. In the sentence *A lone protestor parked*

[1] We use the term governor in order not to confuse it with head word of an NP.

herself outside the UN, the predicate *park* has the arguments *A lone protestor*, labeled ARG0, and *herself*, labeled ARG1. The corresponding value of this feature would be *ARG0-park.01-ARG1-herself*.

3.4 Feature selection

Starting from Soon et al. (2001)'s feature set, we performed a greedy forward selection. The feature selection used a 5-fold cross-validation over the training set, where we evaluated the features using the arithmetic mean of MUC, BCUB, and CEAFE. After reaching a maximal score using forward selection, we reversed the process using a backward elimination, leaving out each feature and removing the one that had the worst impact on performance. This backwards procedure was carried out until the score no longer increased. We repeated this forward-backward procedure until there was no increase in performance. Table 3 shows the final feature set.

Feature bigrams are often used to increase the separability of linear classifiers. Ideally, we would have generated a complete bigram set from our features. However, as this set is quadratic in nature and due to time constraints, we included only a subset of it in the selection procedure. Some of them, most notably the bigram of mention head words (I-HDFORM+J-HDFORM) were selected in the procedure and appear in Table 3.

4 Evaluation

Table 1 shows some baseline figures using the binary features STRINGMATCH and ALIAS as sole coreference properties, as well as our baseline system using Soon et al. (2001)'s features.

	MD	MUC	BCUB
STRINGMATCH	59.91	44.43	63.65
ALIAS	19.25	16.77	48.07
Soon baseline/LR	60.79	47.50	63.97
Soon baseline/C4.5	58.96	47.02	65.36

Table 1: Baseline figures using string match and alias properties, and our Soon baseline using decision trees with the C4.5 induction program and logistic regression (LR). MD stands for mention detection.

4.1 Contribution of postprocessing

The postprocessing step described in Sec. 2.4 proved effective, contributing from 0.21 to up to 1 point to the final score across the metrics. Table 2 shows the detailed impacts on the development set.

	MD	MUC	BCUB	CEAFE
No postproc.	66.56	54.61	65.93	40.46
With postproc.	67.21	55.62	66.29	40.67
Increase	0.65	1.01	0.36	0.21

Table 2: Impact of the postprocessing step on the development set.

4.2 Contribution of features

The lack of time prevented us from running a complete selection from scratch and describing the contribution of each feature on a clean slate. Nonetheless, we computed the scores when one feature is removed from the final feature set. Table 3 shows the performance degradation observed on the development set, which gives an indication of the importance of each feature. In these runs, no postprocessing was not used.

Toward the end of the table, some features show a negative contribution to the score on the development set. This is explained by the fact that our feature selection was carried out in a cross-validated manner over the training set.

4.3 Results on the test set

Table 4 shows the results we obtained on the test set. The figures are consistent with the performance on the development set across the three official metrics, with an increase of the MUC score and a decrease of both BCUB and CEAFE. The official score in the shared task is computed as the mean of these three metrics.

The shared task organizers also provided a test set with given mention boundaries. The given boundaries included nonanaphoric and singleton mentions as well. Using this test set, we replaced our mention extraction step and used the given mention boundaries instead. Table 4 shows the results with this setup. As mention boundaries were given, we turned off our postprocessing module for this run.

Metric\Corpus	Development set			Test set			Test set with gold mentions		
	R	P	F1	R	P	F1	R	P	F1
Mention detection	65.68	68.82	67.21	69.87	68.08	68.96	74.18	70.74	72.42
MUC	55.26	55.98	55.62	60.20	57.10	58.61	64.33	60.05	62.12
BCUB	65.07	67.56	66.29	66.74	64.23	65.46	68.26	65.17	66.68
CEAFM	52.51	52.51	52.51	51.45	51.45	51.45	53.84	53.84	53.84
CEAFE	41.02	40.33	40.67	38.09	41.06	39.52	39.86	44.23	41.93
BLANC	69.6	70.41	70	71.99	70.31	71.11	72.53	71.04	71.75
Official CoNLL score	53.78	54.62	54.19	55.01	54.13	54.53	57.38	56.48	56.91

Table 4: Scores on development set, on the test set, and on the test set with given mention boundaries: recall (R), precision (P), and harmonic mean (F1). The official CoNLL score is computed as the mean of MUC, BCUB, and CEAFE.

	MD	MUC	BCUB
All features	66.56	54.61	65.93
I-HDFORM+J-HDFORM	-1.35	-2.66	-1.82
STRINGMATCH†	-1.12	-1.32	-1.55
DISTANCE†	-0.16	-0.62	-0.59
J-HDGOVPOS	-0.51	-0.49	-0.13
I-HDRMCFUN	-0.27	-0.39	-0.2
ALIAS†	-0.47	-0.36	-0.06
I-HDFORM	-0.42	-0.18	0.04
I-GENDER+J-GENDER	-0.3	-0.15	0.05
NUMBERAGREEMENT†	0.01	-0.14	-0.41
I-HDPOS	-0.32	-0.14	0.05
J-PRONOUN†	-0.25	-0.08	-0.09
I-HDLMCFORM+ J-HDLMCFORM	-0.41	-0.04	0.08
I-HDLSFORM	-0.01	0.01	0
SSBERGSMALINPATH	-0.04	0.02	-0.13
I-HDGOVFUN	-0.09	0.09	0.01
J-HDFUN	-0.01	0.13	-0.04
I-HDLMCPOS	-0.08	0.13	-0.09
DSPATHFORM	-0.03	0.16	-0.02
J-HDGOVFUN	-0.04	0.16	-0.05
J-DEMONSTRATIVE†	-0.03	0.18	0.03
GENDERAGREEMENT†	0	0.18	-0.01
SEMROLEPROPJHD	0.01	0.2	0.01
I-PRONOUN†	0.01	0.22	0.04
I-HDFUN	0.05	0.22	-0.06

Table 3: The final feature set and, for each feature, the degradation in performance when leaving out this feature from the set. All evaluations were carried out on the development set. The features marked with a dagger † originate from the Soon et al. (2001) baseline feature set.

5 Conclusions

The main conclusions and contributions of our work to the CoNLL 2011 shared task concern the detection of mention boundaries, feature lexicalization, and dependency features.

The mention boundaries are relatively difficult to identify. Although far from perfect, we applied a direct mapping from constituents to extract the mentions used in the resolution procedure. We then reclaimed some mentions involving proper nouns in a postprocessing step. Using the gold-standard mention boundaries in the test set, we saw an increase in all metrics with up to 3.51 for the MUC score.

The lexicalization of the feature set brings a significant improvement to the scores. By order of performance loss in Table 3, the first feature of our model is a lexical one. This property does not seem to have been systematically explored before, possibly because of a tradition of using corpora of modest sizes in coreference resolution.

Grammatical dependencies seem to play an important role in the anaphoric expressions. Results in Table 3 also show this, although in a less pronounced manner than lexicalization. Features extracted from dependencies are implicit in many systems, but are not explicitly mentioned as such. We hope our work helped clarified this point through a more systematic exploration of this class of features.

Acknowledgements

This research was supported by Vetenskapsrådet, the Swedish research council, under grant 621-2010-4800.

References

Shane Bergsma and Dekang Lin. 2006. Bootstrapping path-based pronoun resolution. In *Proceedings of the*

21st International Conference on Computational Linguistics and 44th Annual Meeting of the ACL, pages 33–40, July.

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874.

Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. 2009. The WEKA data mining software: An update. *SIGKDD Explorations*, 11(1):10–18, July.

Richard Johansson and Pierre Nugues. 2007. Extended constituent-to-dependency conversion for English. In Joakim Nivre, Heiki-Jaan Kaalep, Kadri Muischnek, and Mare Koit, editors, *NODALIDA 2007 Conference Proceedings*, pages 105–112, Tartu, May 25-26.

Vincent Ng and Claire Cardie. 2002. Improving machine learning approaches to coreference resolution. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 104–111.

Simone Paolo Ponzetto and Michael Strube. 2006. Semantic role labeling for coreference resolution. In *Proceedings of the 11th Conference of EACL: Posters and Demonstrations*, pages 143–146, April.

Sameer Pradhan, Lance Ramshaw, Ralph Weischedel, Jessica MacBride, and Linnea Micciulla. 2007. Unrestricted coreference: Identifying entities and events in OntoNotes. In *Proceedings of the IEEE International Conference on Semantic Computing (ICSC)*, Irvine, CA, September 17-19.

Sameer Pradhan, Lance Ramshaw, Mitchell Marcus, Martha Palmer, Ralph Weischedel, and Nianwen Xue. 2011. CoNLL-2011 shared task: Modeling unrestricted coreference in OntoNotes. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning (CoNLL 2011)*, Portland, Oregon, June.

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