Semantic Class Induction and Coreference Resolution

Vincent Ng
Human Language Technology Research Institute
University of Texas at Dallas
Richardson, TX 75083-0688
vince@hlt.utdallas.edu

Abstract

This paper examines whether a learning-based coreference resolver can be improved using semantic class knowledge that is automatically acquired from a version of the Penn Treebank in which the noun phrases are labeled with their semantic classes. Experiments on the ACE test data show that a resolver that employs such induced semantic class knowledge yields a statistically significant improvement of 2% in F-measure over one that exploits heuristically computed semantic class knowledge. In addition, the induced knowledge improves the accuracy of common noun resolution by 2-6%.

1 Introduction

In the past decade, *knowledge-lean* approaches have significantly influenced research in noun phrase (NP) coreference resolution — the problem of determining which NPs refer to the same real-world entity in a document. In knowledge-lean approaches, coreference resolvers employ only morpho-syntactic cues as knowledge sources in the resolution process (e.g., Mitkov (1998), Tetreault (2001)). While these approaches have been reasonably successful (see Mitkov (2002)), Kehler et al. (2004) speculate that deeper linguistic knowledge needs to be made available to resolvers in order to reach the next level of performance. In fact, semantics plays a crucially important role in the resolution of common NPs, allowing us to identify the coreference relation between two lexically dissimilar common nouns (e.g., *talks* and *negotiations*) and to eliminate *George W. Bush* from the list of candidate antecedents of *the city*, for instance. As a result, researchers have re-adopted the once-popular *knowledge-rich* approach, investigating a variety of semantic knowledge sources for common noun resolution, such as the *semantic relations* between two NPs (e.g., Ji et al. (2005)), their *semantic similarity* as computed using WordNet (e.g., Poesio et al. (2004)) or Wikipedia (Ponzetto and Strube, 2006), and the *contextual role* played by an NP (see Bean and Riloff (2004)).

Another type of semantic knowledge that has been employed by coreference resolvers is the *semantic class* (SC) of an NP, which can be used to disallow coreference between semantically incompatible NPs. However, learning-based resolvers have not been able to benefit from having an SC agreement feature, presumably because the method used to compute the SC of an NP is too simplistic: while the SC of a proper name is computed fairly accurately using a named entity (NE) recognizer, many resolvers simply assign to a common noun the first (i.e., most frequent) WordNet sense as its SC (e.g., Soon et al. (2001), Markert and Nissim (2005)). It is not easy to measure the accuracy of this heuristic, but the fact that the SC agreement feature is not used by Soon et al.'s decision tree coreference classifier seems to suggest that the SC values of the NPs are not computed accurately by this first-sense heuristic.

Motivated in part by this observation, we examine whether automatically induced semantic class knowledge can improve the performance of a learning-based coreference resolver, reporting evaluation results on the commonly-used ACE corefer-

Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics, pages 536–543,
Prague, Czech Republic, June 2007.

ence corpus. Our investigation proceeds as follows.

Train a classifier for labeling the SC of an NP. In ACE, we are primarily concerned with classifying an NP as belonging to one of the ACE semantic classes. For instance, part of the ACE Phase 2 evaluation involves classifying an NP as PERSON, ORGANIZATION, GPE (a geographical-political region), FACILITY, LOCATION, or OTHERS. We adopt a corpus-based approach to SC determination, recasting the problem as a six-class classification task.

Derive two knowledge sources for coreference resolution from the induced SCs. The first knowledge source (KS) is *semantic class agreement* (SCA). Following Soon et al. (2001), we represent SCA as a binary value that indicates whether the induced SCs of the two NPs involved are the same or not. The second KS is *mention*, which is represented as a binary value that indicates whether an NP belongs to one of the five ACE SCs mentioned above. Hence, the *mention* value of an NP can be readily derived from its induced SC: the value is NO if its SC is OTHERS, and YES otherwise. This KS could be useful for ACE coreference, since ACE is concerned with resolving only NPs that are mentions.

Incorporate the two knowledge sources in a coreference resolver. Next, we investigate whether these two KSs can improve a learning-based baseline resolver that employs a fairly standard feature set. Since (1) the two KSs can each be represented in the resolver as a *constraint* (for filtering non-mentions or disallowing coreference between semantically incompatible NPs) or as a *feature*, and (2) they can be applied to the resolver in *isolation* or in *combination*, we have eight ways of incorporating these KSs into the baseline resolver.

In our experiments on the ACE Phase 2 coreference corpus, we found that (1) our SC induction method yields a significant improvement of 2% in accuracy over Soon et al.'s first-sense heuristic method as described above; (2) the coreference resolver that incorporates our induced SC knowledge by means of the two KSs mentioned above yields a significant improvement of 2% in F-measure over the resolver that exploits the SC knowledge computed by Soon et al.'s method; (3) the *mention* KS, when used in the baseline resolver as a *constraint*, improves the resolver by approximately 5-7% in F-measure; and (4) *SCA*, when employed as a *feature* by the baseline resolver, improves the accuracy of common noun resolution by about 5-8%.

2 Related Work

Mention detection. Many ACE participants have also adopted a corpus-based approach to SC determination that is investigated as part of the *mention detection* (MD) task (e.g., Florian et al. (2006)). Briefly, the goal of MD is to identify the boundary of a mention, its mention type (e.g., pronoun, name), and its semantic type (e.g., person, location). Unlike them, (1) we do not perform the full MD task, as our goal is to investigate the role of SC knowledge in coreference resolution; and (2) we do not use the ACE training data for acquiring our SC classifier; instead, we use the *BBN Entity Type Corpus* (Weischedel and Brunstein, 2005), which consists of all the Penn Treebank Wall Street Journal articles with the ACE mentions manually identified and annotated with their SCs. This provides us with a training set that is approximately five times bigger than that of ACE. More importantly, the ACE participants do not evaluate the role of *induced* SC knowledge in coreference resolution: many of them evaluate coreference performance on perfect mentions (e.g., Luo et al. (2004)); and for those that do report performance on automatically extracted mentions, they do not explain whether or how the induced SC information is used in their coreference algorithms.

Joint probabilistic models of coreference. Recently, there has been a surge of interest in improving coreference resolution by jointly modeling coreference with a related task such as MD (e.g., Daumé and Marcu (2005)). However, joint models typically need to be trained on data that is *simultaneously* annotated with information required by all of the underlying models. For instance, Daumé and Marcu's model assumes as input a corpus annotated with both MD and coreference information. On the other hand, we tackle coreference and SC induction separately (rather than jointly), since we train our SC determination model on the BBN Entity Type Corpus, where coreference information is absent.

3 Semantic Class Induction

This section describes how we train and evaluate a classifier for determining the SC of an NP.

3.1 Training the Classifier

Training corpus. As mentioned before, we use the BBN Entity Type Corpus for training the SC classifier. This corpus was originally developed to support the ACE and AQUAINT programs and consists of annotations of 12 named entity types and nine nominal entity types. Nevertheless, we will only make use of the annotations of the five ACE semantic types that are present in our ACE Phase 2 coreference corpus, namely, PERSON, ORGANIZATION, GPE, FACILITY, and LOCATION.

Training instance creation. We create one training instance for each proper or common NP (extracted using an NP chunker and an NE recognizer) in each training text. Each instance is represented by a set of lexical, syntactic, and semantic features, as described below. If the NP under consideration is annotated as one of the five ACE SCs in the corpus, then the classification of the associated training instance is simply the ACE SC value of the NP. Otherwise, the instance is labeled as OTHERS. This results in 310063 instances in the training set.

Features. We represent the training instance for a noun phrase, NP_i, using seven types of features:

(1) WORD: For each word w in NP_i, we create a WORD feature whose value is equal to w. No features are created from stopwords, however.

(2) SUBJ_VERB: If NP_i is involved in a subject-verb relation, we create a SUBJ_VERB feature whose value is the verb participating in the relation. We use Lin's (1998b) MINIPAR dependency parser to extract grammatical relations. Our motivation here is to coarsely model subcategorization.

(3) VERB_OBJ: A VERB_OBJ feature is created in a similar fashion as SUBJ_VERB if NP_i participates in a verb-object relation. Again, this represents our attempt to coarsely model subcategorization.

(4) NE: We use BBN's IdentiFinder (Bikel et al., 1999), a MUC-style NE recognizer to determine the NE type of NP_i. If NP_i is determined to be a PERSON or ORGANIZATION, we create an NE feature whose value is simply its MUC NE type. However, if NP_i is determined to be a LOCATION, we create a feature with value GPE (because most of the MUC LOCATION NEs are ACE GPE NEs). Otherwise, no NE feature will be created (because we are not interested in the other MUC NE types).

ACE SC	Keywords
PERSON	person
ORGANIZATION	social group
FACILITY	establishment, construction, building, facility, workplace
GPE	country, province, government, town, city, administration, society, island, community
LOCATION	dry land, region, landmass, body of water, geographical area, geological formation

Table 1: List of keywords used in WordNet search for generating WN_CLASS features.

(5) WN_CLASS: For each keyword w shown in the right column of Table 1, we determine whether the head noun of NP_i is a hyponym of w in WordNet, using only the first WordNet sense of NP_i.[1] If so, we create a WN_CLASS feature with w as its value. These keywords are potentially useful features because some of them are subclasses of the ACE SCs shown in the left column of Table 1, while others appear to be correlated with these ACE SCs.[2]

(6) INDUCED_CLASS: Since the first-sense heuristic used in the previous feature may not be accurate in capturing the SC of an NP, we employ a corpus-based method for inducing SCs that is motivated by research in lexical semantics (e.g., Hearst (1992)). Given a large, unannotated corpus[3], we use IdentiFinder to label each NE with its NE type and MINIPAR to extract all the appositive relations. An example extraction would be <*Eastern Airlines*, *the carrier*>, where the first entry is a proper noun labeled with either one of the seven MUC-style NE types[4] or OTHERS[5] and the second entry is a common noun. We then infer the SC of a common noun as follows: (1) we compute the probability that the common noun co-occurs with each of the eight NE types[6] based on the extracted appositive relations, and (2) if the most likely NE type has a co-occurrence probability above a certain threshold (we set it to 0.7), we create a INDUCED_CLASS fea-

[1] This is motivated by Lin's (1998c) observation that a coreference resolver that employs only the first WordNet sense performs slightly better than one that employs more than one sense.

[2] The keywords are obtained via our experimentation with WordNet and the ACE SCs of the NPs in the ACE training data.

[3] We used (1) the BLLIP corpus (30M words), which consists of WSJ articles from 1987 to 1989, and (2) the Reuters Corpus (3.7GB data), which has 806,791 Reuters articles.

[4] Person, organization, location, date, time, money, percent.

[5] This indicates the proper noun is not a MUC NE.

[6] For simplicity, OTHERS is viewed as an NE type here.

ture for NP_i whose value is the most likely NE type.

(7) NEIGHBOR: Research in lexical semantics suggests that the SC of an NP can be inferred from its distributionally similar NPs (see Lin (1998a)). Motivated by this observation, we create for each of NP_i's ten most semantically similar NPs a NEIGHBOR feature whose value is the surface string of the NP. To determine the ten nearest neighbors, we use the semantic similarity values provided by Lin's dependency-based thesaurus, which is constructed using a distributional approach combined with an information-theoretic definition of similarity.

Learning algorithms. We experiment with four learners commonly employed in language learning:

Decision List (DL): We use the DL learner as described in Collins and Singer (1999), motivated by its success in the related tasks of word sense disambiguation (Yarowsky, 1995) and NE classification (Collins and Singer, 1999). We apply add-one smoothing to smooth the class posteriors.

1-Nearest Neighbor (1-NN): We use the 1-NN classifier as implemented in TiMBL (Daelemans et al., 2004), employing *dot product* as the similarity function (which defines similarity as the number of common feature-value pairs between two instances). All other parameters are set to their default values.

Maximum Entropy (ME): We employ Lin's ME implementation[7], using a Gaussian prior for smoothing and running the algorithm until convergence.

Naive Bayes (NB): We use an in-house implementation of NB, using add-one smoothing to smooth the class priors and the class-conditional probabilities.

In addition, we train an SVM classifier for SC determination by combining the output of five classification methods: DL, 1-NN, ME, NB, and Soon et al.'s method as described in the introduction,[8] with the goal of examining whether SC classification accuracy can be improved by combining the output of individual classifiers in a supervised manner. Specifically, we (1) use 80% of the instances generated from the BBN Entity Type Corpus to train the four classifiers; (2) apply the four classifiers and Soon et al.'s method to independently make predictions for the remaining 20% of the instances; and (3) train an SVM classifier (using the LIBSVM package (Chang and Lin, 2001)) on these 20% of the instances, where each instance, i, is represented by a set of 31 binary features. More specifically, let $L_i = \{l_{i1}, l_{i2}, l_{i3}, l_{i4}, l_{i5}\}$ be the set of predictions that we obtained for i in step (2). To represent i, we generate one feature from each non-empty subset of L_i.

	PER	ORG	GPE	FAC	LOC	OTH
Training	19.8	9.6	11.4	1.6	1.2	56.3
Test	19.5	9.0	9.6	1.8	1.1	59.0

Table 2: Distribution of SCs in the ACE corpus.

3.2 Evaluating the Classifiers

For evaluation, we use the ACE Phase 2 coreference corpus, which comprises 422 training texts and 97 test texts. Each text has its mentions annotated with their ACE SCs. We create our test instances from the ACE texts in the same way as the training instances described in Section 3.1. Table 2 shows the percentages of instances corresponding to each SC.

Table 3 shows the accuracy of each classifier (see row 1) for the ACE training set (54641 NPs, with 16414 proper NPs and 38227 common NPs) and the ACE test set (13444 NPs, with 3713 proper NPs and 9731 common NPs), as well as their performance on the proper NPs (row 2) and the common NPs (row 3). We employ as our baseline system the Soon et al. method (see Footnote 8), whose accuracy is shown under the Soon column. As we can see, DL, 1-NN, and SVM show a statistically significant improvement over the baseline for both data sets, whereas ME and NB perform significantly worse.[9] Additional experiments are needed to determine the reason for ME and NB's poor performance.

In an attempt to gain additional insight into the performance contribution of each type of features, we conduct feature ablation experiments using the DL classifier (DL is chosen simply because it is the best performer on the ACE training set). Results are shown in Table 4, where each row shows the accuracy of the DL trained on all types of features except for the one shown in that row (All), as well as accuracies on the proper NPs (PN) and the common NPs (CN). For easy reference, the accuracy of the DL

[7] See http://www.cs.ualberta.ca/∼lindek/downloads.htm

[8] In our implementation of Soon's method, we label an instance as OTHERS if no NE or WN_CLASS feature is generated; otherwise its label is the value of the NE feature or the ACE SC that has the WN_CLASS features as its keywords (see Table 1).

[9] We use Noreen's (1989) Approximate Randomization test for significance testing, with p set to .05 unless otherwise stated.

		Training Set						Test Set					
		Soon	DL	1-NN	ME	NB	SVM	Soon	DL	1-NN	ME	NB	SVM
1	Overall	83.1	**85.0**	84.0	54.5	71.3	84.2	81.1	82.9	83.1	53.0	70.3	**83.3**
2	Proper NPs	83.1	**84.1**	81.0	54.2	65.5	82.2	79.6	**82.0**	79.8	55.8	64.4	80.4
3	Common NPs	83.1	**85.4**	85.2	54.6	73.8	85.1	81.6	83.3	84.3	51.9	72.6	**84.4**

Table 3: SC classification accuracies of different methods for the ACE training set and test set.

	Training Set			Test Set		
Feature Type	PN	CN	All	PN	CN	All
All features	84.1	85.4	85.0	82.0	83.3	82.9
- WORD	84.2	85.4	85.0	82.0	83.1	82.8
- SUBJ_VERB	84.1	85.4	85.0	82.0	83.3	82.9
- VERB_OBJ	84.1	85.4	85.0	82.0	83.3	82.9
- NE	72.9	85.3	81.6	74.1	83.2	80.7
- WN_CLASS	84.1	85.9	85.3	81.9	84.1	83.5
- INDUCED_C	84.0	85.6	85.1	82.0	83.6	83.2
- NEIGHBOR	82.8	84.9	84.3	80.2	82.9	82.1

Table 4: Results for feature ablation experiments.

	Training Set			Test Set		
Feature Type	PN	CN	All	PN	CN	All
WORD	64.0	83.9	77.9	66.5	82.4	78.0
SUBJ_VERB	24.0	70.2	56.3	28.8	70.5	59.0
VERB_OBJ	24.0	70.2	56.3	28.8	70.5	59.0
NE	81.1	72.1	74.8	78.4	71.4	73.3
WN_CLASS	25.6	78.8	62.8	30.4	78.9	65.5
INDUCED_C	25.8	81.1	64.5	30.0	80.3	66.3
NEIGHBOR	67.7	85.8	80.4	68.0	84.4	79.8

Table 5: Accuracies of single-feature classifiers.

classifier trained on all types of features is shown in row 1 of the table. As we can see, accuracy drops significantly with the removal of NE and NEIGHBOR. As expected, removing NE precipitates a large drop in proper NP accuracy; somewhat surprisingly, removing NEIGHBOR also causes proper NP accuracy to drop significantly. To our knowledge, there are no prior results on using distributionally similar neighbors as features for supervised SC induction.

Note, however, that these results do not imply that the remaining feature types are not useful for SC classification; they simply suggest, for instance, that WORD is not important in the presence of other feature types. To get a better idea of the utility of each feature type, we conduct another experiment in which we train seven classifiers, each of which employs exactly one type of features. The accuracies of these classifiers are shown in Table 5. As we can see, NEIGHBOR has the largest contribution. This again demonstrates the effectiveness of a distributional approach to semantic similarity. Its superior performance to WORD, the second largest contributor, could be attributed to its ability to combat data sparseness. The NE feature, as expected, is crucial to the classification of proper NPs.

4 Application to Coreference Resolution

We can now derive from the induced SC information two KSs — *semantic class agreement* and *mention* — and incorporate them into our learning-based coreference resolver in eight different ways, as described in the introduction. This section examines whether our coreference resolver can benefit from any of the eight ways of incorporating these KSs.

4.1 Experimental Setup

As in SC induction, we use the ACE Phase 2 coreference corpus for evaluation purposes, acquiring the coreference classifiers on the 422 training texts and evaluating their output on the 97 test texts. We report performance in terms of two metrics: (1) the *F-measure* score as computed by the commonly-used MUC scorer (Vilain et al., 1995), and (2) the *accuracy* on the anaphoric references, computed as the fraction of anaphoric references correctly resolved. Following Ponzetto and Strube (2006), we consider an anaphoric reference, NP_i, correctly resolved if NP_i and its closest antecedent are in the same coreference chain in the resulting partition. In all of our experiments, we use NPs automatically extracted by an in-house NP chunker and IdentiFinder.

4.2 The Baseline Coreference System

Our baseline coreference system uses the C4.5 decision tree learner (Quinlan, 1993) to acquire a classifier on the training texts for determining whether two NPs are coreferent. Following previous work (e.g., Soon et al. (2001) and Ponzetto and Strube (2006)), we generate training instances as follows: a positive instance is created for each anaphoric NP, NP_j, and its closest antecedent, NP_i; and a negative instance is created for NP_j paired with each of the intervening NPs, NP_{i+1}, NP_{i+2}, ..., NP_{j-1}. Each instance is represented by 33 lexical, grammatical, semantic, and positional features that have been employed by high-performing resolvers such as Ng and Cardie (2002) and Yang et al. (2003), as described below.

Lexical features. Nine features allow different types of string matching operations to be performed on the given pair of NPs, NP_x and NP_y[10], including (1) exact string match for pronouns, proper nouns, and non-pronominal NPs (both before and after determiners are removed); (2) substring match for proper nouns and non-pronominal NPs; and (3) head noun match. In addition, one feature tests whether all the words that appear in one NP also appear in the other NP. Finally, a nationality matching feature is used to match, for instance, *British* with *Britain*.

Grammatical features. 22 features test the grammatical properties of one or both of the NPs. These include ten features that test whether each of the two NPs is a pronoun, a definite NP, an indefinite NP, a nested NP, and a clausal subject. A similar set of five features is used to test whether both NPs are pronouns, definite NPs, nested NPs, proper nouns, and clausal subjects. In addition, five features determine whether the two NPs are compatible with respect to gender, number, animacy, and grammatical role. Furthermore, two features test whether the two NPs are in apposition or participate in a predicate nominal construction (i.e., the IS-A relation).

Semantic features. Motivated by Soon et al. (2001), we have a semantic feature that tests whether one NP is a name alias or acronym of the other.

Positional feature. We have a feature that computes the distance between the two NPs in sentences.

After training, the decision tree classifier is used to select an antecedent for each NP in a test text. Following Soon et al. (2001), we select as the antecedent of each NP, NP_j, the *closest* preceding NP that is classified as coreferent with NP_j. If no such NP exists, no antecedent is selected for NP_j.

Row 1 of Table 6 and Table 7 shows the results of the baseline system in terms of F-measure (F) and accuracy in resolving 4599 anaphoric references (All), respectively. For further analysis, we also report the corresponding recall (R) and precision (P) in Table 6, as well as the accuracies of the system in resolving 1769 pronouns (PRO), 1675 proper NPs (PN), and 1155 common NPs (CN) in Table 7. As we can see, the baseline achieves an F-measure of 57.0 and a resolution accuracy of 48.4.

To get a better sense of how strong our baseline is, we re-implement the Soon et al. (2001) coreference resolver. This simply amounts to replacing the 33 features in the baseline resolver with the 12 features employed by Soon et al.'s system. Results of our Duplicated Soon et al. system are shown in row 2 of Tables 6 and 7. In comparison to our baseline, the Duplicated Soon et al. system performs worse according to both metrics, and although the drop in F-measure seems moderate, the performance difference is in fact highly significant (p=0.002).[11]

4.3 Coreference with Induced SC Knowledge

Recall from the introduction that our investigation of the role of induced SC knowledge in learning-based coreference resolution proceeds in three steps:

Label the SC of each NP in each ACE document. If a noun phrase, NP_i, is a proper or common NP, then its SC value is determined using an SC classifier that we acquired in Section 3. On the other hand, if NP_i is a pronoun, then we will be conservative and posit its SC value as UNCONSTRAINED (i.e., it is semantically compatible with all other NPs).[12]

Derive two KSs from the induced SCs. Recall that our first KS, *Mention*, is defined on an NP; its value is YES if the induced SC of the NP is not OTHERS, and NO otherwise. On the other hand, our second KS, *SCA*, is defined on a pair of NPs; its value is YES if the two NPs have the same induced SC that is not OTHERS, and NO otherwise.

Incorporate the two KSs into the baseline resolver. Recall that there are eight ways of incorporating these two KSs into our resolver: they can each be represented as a *constraint* or as a *feature*, and they can be applied to the resolver in *isolation* and in *combination*. Constraints are applied during the antecedent selection step. Specifically, when employed as a constraint, the *Mention* KS disallows coreference between two NPs if at least one of them has a *Mention* value of NO, whereas the *SCA* KS disallows coreference if the *SCA* value of the two NPs involved is NO. When encoded as a feature for the resolver, the *Mention* feature for an NP pair has the

[10] We assume that NP_x precedes NP_y in the associated text.

[11] Again, we use Approximate Randomization with p=.05.

[12] The only exception is pronouns whose SC value can be easily determined to be PERSON (e.g., *he*, *him*, *his*, *himself*).

	System Variation	R	P	F	R	P	F	R	P	F	R	P	F
1	Baseline system	60.9	53.6	57.0	–	–	–	–	–	–	–	–	–
2	Duplicated Soon et al.	56.1	54.4	55.3	–	–	–	–	–	–	–	–	–
	Add to the Baseline	Soon's SC Method			Decision List			SVM			Perfect Information		
3	Mention(C) only	56.9	69.7	**62.6**	59.5	70.6	**64.6**	59.5	70.7	**64.6**	61.2	83.1	70.5
4	Mention(F) only	60.9	54.0	57.2	61.2	52.9	56.7	60.9	53.6	57.0	62.3	33.7	43.8
5	SCA(C) only	56.4	70.0	62.5	57.7	71.2	63.7	58.9	70.7	64.3	61.3	86.1	71.6
6	SCA(F) only	62.0	52.8	57.0	62.5	53.5	57.6	63.0	53.3	57.7	71.1	33.0	45.1
7	Mention(C) + SCA(C)	56.4	70.0	62.5	57.7	71.2	63.7	58.9	70.8	64.3	61.3	86.1	71.6
8	Mention(C) + SCA(F)	58.2	66.4	62.0	60.9	66.8	63.7	61.4	66.5	63.8	71.1	76.7	**73.8**
9	Mention(F) + SCA(C)	56.4	69.8	62.4	57.7	71.3	63.8	58.9	70.6	64.3	62.7	85.3	72.3
10	Mention(F) + SCA(F)	62.0	52.7	57.0	62.6	52.8	57.3	63.2	52.6	57.4	71.8	30.3	42.6

Table 6: Coreference results obtained via the MUC scoring program for the ACE test set.

	System Variation	PRO	PN	CN	All	PRO	PN	CN	All	PRO	PN	CN	All
1	Baseline system	59.2	54.8	22.5	48.4	–	–	–	–	–	–	–	–
2	Duplicated Soon et al.	53.4	45.7	16.9	41.4	–	–	–	–	–	–	–	–
	Add to the Baseline	Soon's SC Method				Decision List				SVM			
3	Mention(C) only	58.5	51.3	16.5	45.3	59.1	54.1	20.2	47.5	59.1	53.9	20.6	47.5
4	Mention(F) only	59.2	55.0	22.5	48.5	59.2	56.1	22.4	48.8	59.4	55.2	22.6	48.6
5	SCA(C) only	58.1	50.1	16.4	44.7	58.1	51.8	17.1	45.5	58.5	52.0	19.6	46.3
6	SCA(F) only	59.2	54.9	27.8	**49.7**	60.4	56.7	30.1	**51.5**	60.8	56.4	29.4	51.3
7	Mention(C) + SCA(C)	58.1	50.1	16.4	44.7	58.1	51.8	17.1	45.5	58.5	51.9	19.5	46.3
8	Mention(C) + SCA(F)	58.9	52.0	22.3	47.2	60.2	55.9	28.1	50.6	60.7	55.3	27.4	50.4
9	Mention(F) + SCA(C)	58.1	50.3	16.3	44.8	58.1	52.4	16.7	45.6	58.6	52.4	19.7	46.6
10	Mention(F) + SCA(F)	59.2	55.0	27.6	**49.7**	60.4	56.8	30.1	**51.5**	60.8	56.5	29.5	**51.4**

Table 7: Resolution accuracies for the ACE test set.

value YES if and only if the *Mention* value for both NPs is YES, whereas the *SCA* feature for an NP pair has its value taken from the *SCA* KS.

Now, we can evaluate the impact of the two KSs on the performance of our baseline resolver. Specifically, rows 3-6 of Tables 6 and 7 show the F-measure and the resolution accuracy, respectively, when exactly one of the two KSs is employed by the baseline as either a constraint (C) or a feature (F), and rows 7-10 of the two tables show the results when both KSs are applied to the baseline. Furthermore, each row of Table 6 contains four sets of results, each of which corresponds to a different method for determining the SC value of an NP. For instance, the first set is obtained by using Soon et al.'s method as described in Footnote 8 to compute SC values, serving as sort of a baseline for our results using induced SC values. The second and third sets are obtained based on the SC values computed by the DL and the SVM classifier, respectively.[13] The last set corresponds to an oracle experiment in which the resolver has access to perfect SC information. Rows 3-10 of Table 7 can be interpreted in a similar manner.

From Table 6, we can see that (1) in comparison to the baseline, F-measure increases significantly in the five cases where at least one of the KSs is employed as a constraint by the resolver, and such improvements stem mainly from significant gains in precision; (2) in these five cases, the resolvers that use SCs induced by DL and SVM achieve significantly higher F-measure scores than their counterparts that rely on Soon's method for SC determination; and (3) none of the resolvers appears to benefit from *SCA* information whenever *mention* is used as a constraint.

Moreover, note that even with perfectly computed SC information, the performance of the baseline system does not improve when neither *MD* nor *SCA* is employed as a constraint. These results provide further evidence that the decision tree learner is not exploiting these two semantic KSs in an optimal manner, whether they are computed automatically or perfectly. Hence, in machine learning for coreference resolution, it is important to determine not only *what* linguistic KSs to use, but also *how* to use them.

While the coreference results in Table 6 seem to suggest that *SCA* and *mention* should be employed as constraints, the resolution results in Table 7 sug-

[13]Results using other learners are not shown due to space limitations. DL and SVM are chosen simply because they achieve the highest SC classification accuracies on the ACE training set.

gest that *SCA* is better encoded as a feature. Specifically, (1) in comparison to the baseline, the accuracy of common NP resolution improves by about 5-8% when *SCA* is encoded as a feature; and (2) whenever *SCA* is employed as a feature, the overall resolution accuracy is significantly higher for resolvers that use SCs induced by DL and SVM than those that rely on Soon's method for SC determination, with improvements in resolution observed on all three NP types.

Overall, these results provide suggestive evidence that both KSs are useful for learning-based coreference resolution. In particular, *mention* should be employed as a constraint, whereas *SCA* should be used as a feature. Interestingly, this is consistent with the results that we obtained when the resolver has access to perfect SC information (see Table 6), where the highest F-measure is achieved by employing *mention* as a constraint and *SCA* as a feature.

5 Conclusions

We have shown that (1) both *mention* and *SCA* can be usefully employed to improve the performance of a learning-based coreference system, and (2) employing SC knowledge induced in a supervised manner enables a resolver to achieve better performance than employing SC knowledge computed by Soon et al.'s simple method. In addition, we found that the MUC scoring program is unable to reveal the usefulness of the *SCA* KS, which, when encoded as a feature, substantially improves the accuracy of common NP resolution. This underscores the importance of reporting both resolution accuracy and clustering-level accuracy when analyzing the performance of a coreference resolver.

References

D. Bean and E. Riloff. 2004. Unsupervised learning of contextual role knowledge for coreference resolution. In *Proc. of HLT/NAACL*, pages 297–304.

D. M. Bikel, R. Schwartz, and R. M. Weischedel. 1999. An algorithm that learns what's in a name. *Machine Learning* 34(1–3):211–231.

C.-C. Chang and C.-J. Lin, 2001. LIBSVM: a library for support vector machines. Software available at http://www.csie.ntu.edu.tw/∼cjlin/libsvm.

M. Collins and Y. Singer. 1999. Unsupervised models for named entity classification. In *Proc. of EMNLP/VLC*.

W. Daelemans, J. Zavrel, K. van der Sloot, and A. van den Bosch. 2004. TiMBL: Tilburg Memory Based Learner, version 5.1, Reference Guide. ILK Technical Report.

H. Daumé III and D. Marcu. 2005. A large-scale exploration of effective global features for a joint entity detection and tracking model. In *Proc. of HLT/EMNLP*, pages 97–104.

R. Florian, H. Jing, N. Kambhatla, and I. Zitouni. 2006. Factorizing complex models: A case study in mention detection. In *Proc. of COLING/ACL*, pages 473–480.

M. Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *Proc. of COLING*.

H. Ji, D. Westbrook, and R. Grishman. 2005. Using semantic relations to refine coreference decisions. In *Proc. of HLT/EMNLP*, pages 17–24.

A. Kehler, D. Appelt, L. Taylor, and A. Simma. 2004. The (non)utility of predicate-argument frequencies for pronoun interpretation. In *Proc. of NAACL*, pages 289–296.

D. Lin. 1998a. Automatic retrieval and clustering of similar words. In *Proc. of COLING/ACL*, pages 768–774.

D. Lin. 1998b. Dependency-based evaluation of MINIPAR. In *Proc. of the LREC Workshop on the Evaluation of Parsing Systems*, pages 48–56.

D. Lin. 1998c. Using collocation statistics in information extraction. In *Proc. of MUC-7*.

X. Luo, A. Ittycheriah, H. Jing, N. Kambhatla, and S. Roukos. 2004. A mention-synchronous coreference resolution algorithm based on the Bell tree. In *Proc. of the ACL*.

K. Markert and M. Nissim. 2005. Comparing knowledge sources for nominal anaphora resolution. *Computational Linguistics*, 31(3):367–402.

R. Mitkov. 2002. *Anaphora Resolution*. Longman.

R. Mitkov. 1998. Robust pronoun resolution with limited knowledge. In *Proc. of COLING/ACL*, pages 869–875.

V. Ng and C. Cardie. 2002. Improving machine learning approaches to coreference resolution. In *Proc. of the ACL*.

E. W. Noreen. 1989. *Computer Intensive Methods for Testing Hypothesis: An Introduction*. John Wiley & Sons.

M. Poesio, R. Mehta, A. Maroudas, and J. Hitzeman. 2004. Learning to resolve bridging references. In *Proc. of the ACL*.

S. P. Ponzetto and M. Strube. 2006. Exploiting semantic role labeling, WordNet and Wikipedia for coreference resolution. In *Proc. of HLT/NAACL*, pages 192–199.

J. R. Quinlan. 1993. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Mateo, CA.

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

J. Tetreault. 2001. A corpus-based evaluation of centering and pronoun resolution. *Computational Linguistics*, 27(4).

M. Vilain, J. Burger, J. Aberdeen, D. Connolly, and L. Hirschman. 1995. A model-theoretic coreference scoring scheme. In *Proc. of MUC-6*, pages 45–52.

R. Weischedel and A. Brunstein. 2005. BBN pronoun coreference and entity type corpus. Linguistica Data Consortium.

X. Yang, G. Zhou, J. Su, and C. L. Tan. 2003. Coreference resolution using competitive learning approach. In *Proc. of the ACL*, pages 176–183.

D. Yarowsky. 1995. Unsupervised word sense disambiguation rivaling supervised methods. In *Proc. of the ACL*.