Coreference Semantics from Web Features

Mohit Bansal and **Dan Klein**
Computer Science Division
University of California, Berkeley
{mbansal, klein}@cs.berkeley.edu

Abstract

To address semantic ambiguities in coreference resolution, we use Web n-gram features that capture a range of world knowledge in a diffuse but robust way. Specifically, we exploit short-distance cues to hypernymy, semantic compatibility, and semantic context, as well as general lexical co-occurrence. When added to a state-of-the-art coreference baseline, our Web features give significant gains on multiple datasets (ACE 2004 and ACE 2005) and metrics (MUC and B^3), resulting in the best results reported to date for the end-to-end task of coreference resolution.

1 Introduction

Many of the most difficult ambiguities in coreference resolution are semantic in nature. For instance, consider the following example:

When Obama met Jobs, the president discussed the economy, technology, and education. His election campaign is expected to [...]

For resolving coreference in this example, a system would benefit from the world knowledge that *Obama* is *the president*. Also, to resolve the pronoun *his* to the correct antecedent *Obama*, we can use the knowledge that *Obama* has an *election campaign* while *Jobs* does not. Such ambiguities are difficult to resolve on purely syntactic or configurational grounds.

There have been multiple previous systems that incorporate some form of world knowledge in coreference resolution tasks. Most work (Poesio et al., 2004; Markert and Nissim, 2005; Yang et al., 2005; Bergsma and Lin, 2006) addresses special cases and subtasks such as bridging anaphora, other anaphora, definite NP reference, and pronoun resolution, computing semantic compatibility via Web-hits and counts from large corpora. There is also work on end-to-end coreference resolution that uses large noun-similarity lists (Daumé III and Marcu, 2005) or structured knowledge bases such as Wikipedia (Yang and Su, 2007; Haghighi and Klein, 2009; Kobdani et al., 2011) and YAGO (Rahman and Ng, 2011). However, such structured knowledge bases are of limited scope, and, while Haghighi and Klein (2010) self-acquires knowledge about coreference, it does so only via reference constructions and on a limited scale.

In this paper, we look to the Web for broader if shallower sources of semantics. In order to harness the information on the Web without presupposing a deep understanding of all Web text, we instead turn to a diverse collection of Web n-gram counts (Brants and Franz, 2006) which, in aggregate, contain diffuse and indirect, but often robust, cues to reference. For example, we can collect the co-occurrence statistics of an anaphor with various candidate antecedents to judge relative surface affinities (i.e., (*Obama*, *president*) versus (*Jobs*, *president*)). We can also count co-occurrence statistics of competing antecedents when placed in the context of an anaphoric pronoun (i.e., *Obama's election campaign* versus *Jobs' election campaign*).

All of our features begin with a pair of headwords from candidate mention pairs and compute statistics derived from various potentially informative queries' counts. We explore five major categories of semantically informative Web features, based on (1) general lexical affinities (via generic co-occurrence statistics), (2) lexical relations (via Hearst-style hypernymy patterns), (3) similarity of entity-based context (e.g., common values of y for which *h is a y* is attested), (4) matches of distributional soft cluster ids, and (5) attested substitutions of candidate antecedents in the context of a pronominal anaphor.

We first describe a strong baseline consisting of the mention-pair model of the Reconcile system (Stoyanov et al., 2009; Stoyanov et al., 2010) using a decision tree (DT) as its pairwise classifier. To this baseline system, we add our suite of features in turn, each class of features providing substantial gains. Altogether, our final system produces the best numbers reported to date on end-to-end coreference resolution (with automatically detected system mentions) on multiple data sets (ACE 2004 and ACE 2005) and metrics (MUC and B^3), achieving significant improvements over the Reconcile DT baseline and over the state-of-the-art results of Haghighi and Klein (2010).

2 Baseline System

Before describing our semantic Web features, we first describe our baseline. The core inference and features come from the Reconcile package (Stoyanov et al., 2009; Stoyanov et al., 2010), with modifications described below. Our baseline differs most substantially from Stoyanov et al. (2009) in using a decision tree classifier rather than an averaged linear perceptron.

2.1 Reconcile

Reconcile is one of the best implementations of the *mention-pair* model (Soon et al., 2001) of coreference resolution. The mention-pair model relies on a pairwise function to determine whether or not two mentions are coreferent. Pairwise predictions are then consolidated by transitive closure (or some other clustering method) to form the final set of coreference clusters (chains). While our Web features could be adapted to entity-mention systems, their current form was most directly applicable to the mention-pair approach, making Reconcile a particularly well-suited platform for this investigation.

The Reconcile system provides baseline features, learning mechanisms, and resolution procedures that already achieve near state-of-the-art results on multiple popular datasets using multiple standard metrics. It includes over 80 core features that exploit various automatically generated annotations such as named entity tags, syntactic parses, and WordNet classes, inspired by Soon et al. (2001), Ng and Cardie (2002), and Bengtson and Roth (2008). The Reconcile system also facilitates standardized empirical evaluation to past work.[1]

In this paper, we develop a suite of simple semantic Web features based on pairs of mention headwords which stack with the default Reconcile features to surpass past state-of-the-art results.

2.2 Decision Tree Classifier

Among the various learning algorithms that Reconcile supports, we chose the decision tree classifier, available in Weka (Hall et al., 2009) as J48, an open source Java implementation of the C4.5 algorithm of Quinlan (1993).

The C4.5 algorithm builds decision trees by incrementally maximizing information gain. The training data is a set of already classified samples, where each sample is a vector of attributes or features. At each node of the tree, C4.5 splits the data on an attribute that most effectively splits its set of samples into more ordered subsets, and then recurses on these smaller subsets. The decision tree can then be used to classify a new sample by following a path from the root downward based on the attribute values of the sample.

We find the decision tree classifier to work better than the default averaged perceptron (used by Stoyanov et al. (2009)), on multiple datasets using multiple metrics (see Section 4.3). Many advantages have been claimed for decision tree classifiers, including interpretability and robustness. However, we suspect that the aspect most relevant to our case is that decision trees can capture non-linear interactions between features. For example, recency is very important for pronoun reference but much less so for nominal reference.

3 Semantics via Web Features

Our Web features for coreference resolution are simple and capture a range of diffuse world knowledge. Given a mention pair, we use the head finder in Reconcile to find the lexical heads of both mentions (for

[1]We use the default configuration settings of Reconcile (Stoyanov et al., 2010) unless mentioned otherwise.

example, the head of *the Palestinian territories* is the word *territories*). Next, we take each headword pair (h_1, h_2) and compute various Web-count functions on it that can signal whether or not this mention pair is coreferent.

As the source of Web information, we use the Google n-grams corpus (Brants and Franz, 2006) which contains English n-grams ($n = 1$ to 5) and their Web frequency counts, derived from nearly 1 trillion word tokens and 95 billion sentences. Because we have many queries that must be run against this corpus, we apply the trie-based hashing algorithm of Bansal and Klein (2011) to efficiently answer all of them in one pass over it. The features that require word clusters (Section 3.4) use the output of Lin et al. (2010).[2]

We describe our five types of features in turn. The first four types are most intuitive for mention pairs where both members are non-pronominal, but, aside from the general co-occurrence group, helped for all mention pair types. The fifth feature group applies only to pairs in which the anaphor is a pronoun but the antecedent is a non-pronoun. Related work for each feature category is discussed inline.

3.1 General co-occurrence

These features capture co-occurrence statistics of the two headwords, i.e., how often h_1 and h_2 are seen adjacent or nearly adjacent on the Web. This count can be a useful coreference signal because, in general, mentions referring to the same entity will co-occur more frequently (in large corpora) than those that do not. Using the n-grams corpus (for n = 1 to 5), we collect co-occurrence Web-counts by allowing a varying number of wildcards between h_1 and h_2 in the query. The co-occurrence value is:

$$bin\left(\log_{10}\left(\frac{c_{12}}{c_1 \cdot c_2}\right)\right)$$

where

$$\begin{aligned} c_{12} &= count(\text{“}h_1 \ \star \ h_2\text{”}) \\ &\quad + count(\text{“}h_1 \ \star \star \ h_2\text{”}) \\ &\quad + count(\text{“}h_1 \ \star \star \star \ h_2\text{”}), \\ c_1 &= count(\text{“}h_1\text{”}), \text{and} \\ c_2 &= count(\text{“}h_2\text{”}). \end{aligned}$$

We normalize the overall co-occurrence count of the headword pair c_{12} by the unigram counts of the individual headwords c_1 and c_2, so that high-frequency headwords do not unfairly get a high feature value (this is similar to computing scaled mutual information MI (Church and Hanks, 1989)).[3] This normalized value is quantized by taking its $\log_{10}$ and binning. The actual feature that fires is an indicator of which quantized bin the query produced. As a real example from our development set, the co-occurrence count c_{12} for the headword pair (*leader, president*) is 11383, while it is only 95 for the headword pair (*voter, president*); after normalization and $\log_{10}$, the values are -10.9 and -12.0, respectively.

These kinds of general Web co-occurrence statistics have been used previously for other supervised NLP tasks such as spelling correction and syntactic parsing (Bergsma et al., 2010; Bansal and Klein, 2011). In coreference, similar word-association scores were used by Kobdani et al. (2011), but from Wikipedia and for self-training.

3.2 Hearst co-occurrence

These features capture templated co-occurrence of the two headwords h_1 and h_2 in the Web-corpus. Here, we only collect statistics of the headwords co-occurring with a generalized Hearst pattern (Hearst, 1992) in between. Hearst patterns capture various lexical semantic relations between items. For example, seeing *X is a Y* or *X and other Y* indicates hypernymy and also tends to cue coreference. The specific patterns we use are:

- h_1 {*is* | *are* | *was* | *were*} {*a* | *an* | *the*}*?* h_2
- h_1 {*and* | *or*} {*other* | *the other* | *another*} h_2
- h_1 *other than* {*a* | *an* | *the*}*?* h_2

[2]These clusters are derived form the V2 Google n-grams corpus. The V2 corpus itself is not publicly available, but the clusters are available at `http://www.clsp.jhu.edu/~sbergsma/PhrasalClusters`

[3]We also tried adding $count(\text{“}h1 \ \ h2\text{”})$ to c_{12} but this decreases performance, perhaps because truly adjacent occurrences are often not grammatical.

- h_1 *such as* $\{$*a* $|$ *an* $|$ *the*$\}$*?* h_2
- h_1 *, including* $\{$*a* $|$ *an* $|$ *the*$\}$*?* h_2
- h_1 *, especially* $\{$*a* $|$ *an* $|$ *the*$\}$*?* h_2
- h_1 *of* $\{$*the*$|$ *all*$\}$*?* h_2

For this feature, we again use a quantized normalized count as in Section 3.1, but c_{12} here is restricted to n-grams where one of the above patterns occurs in between the headwords. We did not allow wildcards in between the headwords and the Hearst-patterns because this introduced a significant amount of noise. Also, we do not constrain the order of h_1 and h_2 because these patterns can hold for either direction of coreference.[4] As a real example from our development set, the c_{12} count for the headword pair (*leader, president*) is 752, while for (*voter, president*), it is 0.

Hypernymic semantic compatibility for coreference is intuitive and has been explored in varying forms by previous work. Poesio et al. (2004) and Markert and Nissim (2005) employ a subset of our Hearst patterns and Web-hits for the subtasks of bridging anaphora, other-anaphora, and definite NP resolution. Others (Haghighi and Klein, 2009; Rahman and Ng, 2011; Daumé III and Marcu, 2005) use similar relations to extract compatibility statistics from Wikipedia, YAGO, and noun-similarity lists. Yang and Su (2007) use Wikipedia to automatically extract semantic patterns, which are then used as features in a learning setup. Instead of extracting patterns from the training data, we use all the above patterns, which helps us generalize to new datasets for end-to-end coreference resolution (see Section 4.3).

3.3 Entity-based context

For each headword h, we first collect context *seeds* y using the pattern

$$h \; \{\textit{is} \mid \textit{are} \mid \textit{was} \mid \textit{were}\} \; \{\textit{a} \mid \textit{an} \mid \textit{the}\}? \; y$$

taking seeds y in order of decreasing Web count. The corresponding ordered *seed list* $Y = \{y\}$ gives us useful information about the headword's entity type. For example, for $h = president$, the top 30 seeds (and their parts of speech) include important cues such as *president is elected* (verb), *president is authorized* (verb), *president is responsible* (adjective), *president is the chief* (adjective), *president is above* (preposition), and *president is the head* (noun).

[4]Two minor variants not listed above are h_1 *including* h_2 and h_1 *especially* h_2.

Matches in the seed lists of two headwords can be a strong signal that they are coreferent. For example, in the top 30 seed lists for the headword pair (*leader, president*), we get matches including *elected*, *responsible*, and *expected*. To capture this effect, we create a feature that indicates whether there is a match in the top k seeds of the two headwords (where k is a hyperparameter to tune).

We create another feature that indicates whether the dominant parts of speech in the seed lists matches for the headword pair. We first collect the POS tags (using length 2 character prefixes to indicate coarse parts of speech) of the seeds matched in the top k' seed lists of the two headwords, where k' is another hyperparameter to tune. If the dominant tags match and are in a small list of important tags ($\{$JJ, NN, RB, VB$\}$), we fire an indicator feature specifying the matched tag, otherwise we fire a *no-match* indicator. To obtain POS tags for the seeds, we use a unigram-based POS tagger trained on the WSJ treebank training set.

3.4 Cluster information

The distributional hypothesis of Harris (1954) says that words that occur in similar contexts tend to have a similar linguistic behavior. Here, we design features with the idea that this hypothesis extends to reference: mentions occurring in similar contexts in large document sets such as the Web tend to be compatible for coreference. Instead of collecting the contexts of each mention and creating sparse features from them, we use Web-scale distributional clustering to summarize compatibility.

Specifically, we begin with the phrase-based clusters from Lin et al. (2010), which were created using the Google n-grams V2 corpus. These clusters come from distributional K-Means clustering (with $K = 1000$) on phrases, using the n-gram context as features. The cluster data contains almost 10 million phrases and their *soft* cluster memberships. Up to twenty cluster ids with the highest centroid similarities are included for each phrase in this dataset

(Lin et al., 2010).

Our cluster-based features assume that if the headwords of the two mentions have matches in their cluster id lists, then they are more compatible for coreference. We check the match of not just the top 1 cluster ids, but also farther down in the 20 sized lists because, as discussed in Lin and Wu (2009), the soft cluster assignments often reveal different senses of a word. However, we also assume that higher-ranked matches tend to imply closer meanings. To this end, we fire a feature indicating the value $bin(i+j)$, where i and j are the *earliest match* positions in the cluster id lists of h_1 and h_2. Binning here means that match positions in a close range generally trigger the same feature.

Recent previous work has used clustering information to improve the performance of supervised NLP tasks such as NER and dependency parsing (Koo et al., 2008; Lin and Wu, 2009). However, in coreference, the only related work to our knowledge is from Daumé III and Marcu (2005), who use word class features derived from a Web-scale corpus via a process described in Ravichandran et al. (2005).

3.5 Pronoun context

Our last feature category specifically addresses pronoun reference, for cases when the anaphoric mention NP_2 (and hence its headword h_2) is a pronoun, while the candidate antecedent mention NP_1 (and hence its headword h_1) is not. For such a headword pair (h_1, h_2), the idea is to substitute the non-pronoun h_1 into h_2's position and see whether the result is attested on the Web.

If the anaphoric pronominal mention is h_2 and its sentential context is *l' l* h_2 *r r'*, then the substituted phrase will be *l' l* h_1 *r r'*.[5] High Web counts of substituted phrases tend to indicate semantic compatibility. Perhaps unsurprisingly for English, only the right context was useful in this capacity. We chose the following three context types, based on performance on a development set:

- h_1 *r* (R1)
- h_1 *r r'* (R2)
- $h_1 \star r$ (R1Gap)

As an example of the R1Gap feature, if the anaphor h_2 + context is *his victory* and one candidate antecedent h_1 is *Bush*, then we compute the normalized value

$$\frac{count(\text{“}Bush\ 's\ \star\ victory\text{”})}{count(\text{“}\star\ 's\ \star\ victory\text{”})count(\text{“}Bush\text{”})}$$

In general, we compute

$$\frac{count(\text{“}h_1\ 's\ \star\ r\text{”})}{count(\text{“}\star\ 's\ \star\ r\text{”})count(\text{“}h_1\text{”})}$$

The final feature value is again a normalized count converted to $\log_{10}$ and then binned.[6] We have three separate features for the R1, R2, and R1Gap context types. We tune a separate bin-size hyperparameter for each of these three features.

These pronoun resolution features are similar to selectional preference work by Yang et al. (2005) and Bergsma and Lin (2006), who compute semantic compatibility for pronouns in specific syntactic relationships such as possessive-noun, subject-verb, etc. In our case, we directly use the general context of any pronominal anaphor to find its most compatible antecedent.

Note that all our above features are designed to be non-sparse by firing indicators of the quantized Web statistics and not the lexical- or class-based identities of the mention pair. This keeps the total number of features small, which is important for the relatively small datasets used for coreference resolution. We go from around 100 features in the Reconcile baseline to around 165 features after adding all our Web features.

[5] Possessive pronouns are replaced with an additional apostrophe, i.e., h_1 *'s*. We also use features (see R1Gap) that allow wildcards ($\star$) in between the headword and the context when collecting Web-counts, in order to allow for determiners and other filler words.

[6] Normalization helps us with two kinds of balancing. First, we divide by the count of the antecedent so that when choosing the best antecedent for a fixed anaphor, we are not biased towards more frequently occurring antecedents. Second, we divide by the count of the context so that across anaphora, an anaphor with rarer context does not get smaller values (for all its candidate antecedents) than another anaphor with a more common context.

Dataset	docs	dev	test	ment	chn
ACE04	128	63/27	90/38	3037	1332
ACE05	81	40/17	57/24	1991	775
ACE05-ALL	599	337/145	482/117	9217	3050

Table 1: Dataset characteristics – *docs*: the total number of documents; *dev*: the train/test split during development; *test*: the train/test split during testing; *ment*: the number of gold mentions in the test split; *chn*: the number of coreference chains in the test split.

4 Experiments

4.1 Data

We show results on three popular and comparatively larger coreference resolution data sets – the ACE04, ACE05, and ACE05-ALL datasets from the ACE Program (NIST, 2004). In ACE04 and ACE05, we have only the newswire portion (of the original ACE 2004 and 2005 training sets) and use the standard train/test splits reported in Stoyanov et al. (2009) and Haghighi and Klein (2010). In ACE05-ALL, we have the full ACE 2005 training set and use the standard train/test splits reported in Rahman and Ng (2009) and Haghighi and Klein (2010). Note that most previous work does not report (or need) a standard development set; hence, for tuning our features and its hyper-parameters, we randomly split the original training data into a training and development set with a 70/30 ratio (and then use the full original training set during testing). Details of the corpora are shown in Table 1.[7]

Details of the Web-scale corpora used for extracting features are discussed in Section 3.

4.2 Evaluation Metrics

We evaluated our work on both MUC (Vilain et al., 1995) and B^3 (Bagga and Baldwin, 1998). Both scorers are available in the Reconcile infrastructure.[8] MUC measures how many predicted clusters need to be merged to cover the true gold clusters. B^3 computes precision and recall for each mention by computing the intersection of its predicted and gold cluster and dividing by the size of the predicted and gold cluster, respectively. It is well known (Recasens and Hovy, 2010; Ng, 2010; Kobdani et al., 2011) that MUC is biased towards large clusters (chains) whereas B^3 is biased towards singleton clusters. Therefore, for a more balanced evaluation, we show improvements on both metrics simultaneously.

[7]Note that the development set is used only for ACE04, because for ACE05, and ACE05-ALL, we directly test using the features tuned on ACE04.

[8]Note that B^3 has two versions which handle twinless (spurious) mentions in different ways (see Stoyanov et al. (2009) for details). We use the B^3All version, unless mentioned otherwise.

	MUC			B^3		
Feature	P	R	F1	P	R	F1
AvgPerc	69.0	63.1	65.9	82.2	69.9	75.5
DecTree	**80.9**	61.0	69.5	**89.5**	69.0	77.9
+ Co-occ	79.8	62.1	69.8	88.7	69.8	78.1
+ Hearst	80.0	62.3	70.0	89.1	70.1	78.5
+ Entity	79.4	63.2	70.4	88.1	70.9	78.6
+ Cluster	79.5	63.6	70.7	87.9	71.2	78.6
+ Pronoun	79.9	**64.3**	**71.3**	88.0	**71.6**	**79.0**

Table 2: Incremental results for the Web features on the ACE04 development set. AvgPerc is the averaged perceptron baseline, DecTree is the decision tree baseline, and the +Feature rows show the effect of adding a particular feature incrementally (not in isolation) to the DecTree baseline. The feature categories correspond to those described in Section 3.

4.3 Results

We start with the Reconcile baseline but employ the decision tree (DT) classifier, because it has significantly better performance than the default averaged perceptron classifier used in Stoyanov et al. (2009).[9] Table 2 compares the baseline perceptron results to the DT results and then shows the incremental addition of the Web features to the DT baseline (on the ACE04 development set).

The DT classifier, in general, is precision-biased. The Web features somewhat balance this by increasing the recall and decreasing precision to a lesser extent, improving overall F1. Each feature type incrementally increases both MUC and B^3 F1-measures, showing that they are not taking advantage of any bias of either metric. The incremental improvements also show that each Web feature type brings in some additional benefit over the information already present in the Reconcile baseline, which includes alias, animacy, named entity, and WordNet

[9]Moreover, a DT classifier takes roughly the same amount of time and memory as a perceptron on our ACE04 development experiments. It is, however, slower and more memory-intensive ($\sim$3x) on the bigger ACE05-ALL dataset.

	MUC			B^3		
System	P	R	F1	P	R	F1
ACE04-TEST-RESULTS						
Stoyanov et al. (2009)	-	-	62.0	-	-	76.5
Haghighi and Klein (2009)	67.5	61.6	64.4	77.4	69.4	73.2
Haghighi and Klein (2010)	67.4	**66.6**	67.0	81.2	73.3	77.0
This Work: Perceptron Baseline	65.5	61.9	63.7	84.1	70.9	77.0
This Work: DT Baseline	**76.0**	60.7	67.5	**89.6**	70.3	78.8
This Work: DT + Web Features	74.8	64.2	**69.1**	87.5	**73.7**	**80.0**
This Work: Δ of DT+Web over DT	($p < 0.05$)		1.7	($p < 0.005$)		1.3
ACE05-TEST-RESULTS						
Stoyanov et al. (2009)	-	-	67.4	-	-	73.7
Haghighi and Klein (2009)	73.1	58.8	65.2	82.1	63.9	71.8
Haghighi and Klein (2010)	74.6	62.7	68.1	83.2	68.4	75.1
This Work: Perceptron Baseline	72.2	61.6	66.5	85.0	65.5	73.9
This Work: DT Baseline	**79.6**	59.7	68.2	**89.4**	64.2	74.7
This Work: DT + Web Features	75.0	**64.7**	**69.5**	81.1	**70.8**	**75.6**
This Work: Δ of DT+Web over DT	($p < 0.12$)		1.3	($p < 0.1$)		0.9
ACE05-ALL-TEST-RESULTS						
Rahman and Ng (2009)	75.4	64.1	69.3	54.4	70.5	61.4
Haghighi and Klein (2009)	72.9	60.2	67.0	53.2	73.1	61.6
Haghighi and Klein (2010)	77.0	**66.9**	**71.6**	55.4	**74.8**	63.8
This Work: Perceptron Baseline	68.9	60.4	64.4	80.6	60.5	69.1
This Work: DT Baseline	**78.0**	60.4	68.1	**85.1**	60.4	70.6
This Work: DT + Web Features	77.6	64.0	70.2	80.7	65.9	**72.5**
This Work: Δ of DT+Web over DT	($p < 0.001$)		2.1	($p < 0.001$)		1.9

Table 3: Primary test results on the ACE04, ACE05, and ACE05-ALL datasets. All systems reported here use automatically extracted system mentions. B^3 here is the B^3All version of Stoyanov et al. (2009). We also report statistical significance of the improvements from the Web features on the DT baseline, using the bootstrap test (Noreen, 1989; Efron and Tibshirani, 1993). The perceptron baseline in this work (Reconcile settings: 15 iterations, threshold = 0.45, SIG for ACE04 and AP for ACE05, ACE05-ALL) has different results from Stoyanov et al. (2009) because their current publicly available code is different from that used in their paper (*p.c.*). Also, the B^3 variant used by Rahman and Ng (2009) is slightly different from other systems (they remove all and only the singleton twinless system mentions, so it is neither B^3All nor B^3None). For completeness, our (untuned) B^3None results (DT + Web) on the ACE05-ALL dataset are P=69.9|R=65.9|F1=67.8.

class / sense information.[10]

Table 3 shows our primary *test* results on the ACE04, ACE05, and ACE05-ALL datasets, for the MUC and B^3 metrics. All systems reported use automatically detected mentions. We report our results (the 3 rows marked 'This Work') on the perceptron baseline, the DT baseline, and the Web features added to the DT baseline. We also report statistical significance of the improvements from the Web features on the DT baseline.[11] For significance testing, we use the bootstrap test (Noreen, 1989; Efron and Tibshirani, 1993).

Our main comparison is against Haghighi and Klein (2010), a mostly-unsupervised generative approach that models latent entity types, which generate specific entities that in turn render individual mentions. They learn on large datasets including

[10]We also initially experimented with smaller datasets (MUC6 and MUC7) and an averaged perceptron baseline, and we did see similar improvements, arguing that these features are useful independently of the learning algorithm and dataset.

[11]All improvements are significant, except on the small ACE05 dataset with the MUC metric (where it is weak, at $p < 0.12$). However, on the larger version of this dataset, ACE05-ALL, we get improvements which are both larger and more significant (at $p < 0.001$).

Wikipedia, and their results are state-of-the-art in coreference resolution. We outperform their system on most datasets and metrics (except on ACE05-ALL for the MUC metric). The other systems we compare to and outperform are the perceptron-based Reconcile system of Stoyanov et al. (2009), the strong deterministic system of Haghighi and Klein (2009), and the cluster-ranking model of Rahman and Ng (2009).

We develop our features and tune their hyper-parameter values on the ACE04 development set and then use these on the ACE04 test set.[12] On the ACE05 and ACE05-ALL datasets, we directly transfer our Web features and their hyper-parameter values from the ACE04 dev-set, without any retuning. The test improvements we get on all the datasets (see Table 3) suggest that our features are generally useful across datasets and metrics.[13]

5 Analysis

In this section, we briefly discuss errors (in the DT baseline) corrected by our Web features, and analyze the decision tree classifier built during training (based on the ACE04 development experiments).

To study error correction, we begin with the mention pairs that are coreferent according to the gold-standard annotation (after matching the system mentions to the gold ones). We consider the pairs that are wrongly predicted to be non-coreferent by the baseline DT system but correctly predicted to be coreferent when we add our Web features. Some examples of such pairs include:

Iran ; the country
the EPA ; the agency
athletic director ; Mulcahy
Democrat Al Gore ; the vice president
Barry Bonds ; the best baseball player
Vojislav Kostunica ; the pro-democracy leader
its closest rival ; the German magazine Das Motorrad
One of those difficult-to-dislodge judges ; John Marshall

These pairs are cases where our features on Hearst-style co-occurrence and entity-based context-match are informative and help discriminate in favor of the correct antecedents. One advantage of using Web-based features is that the Web has a surprising amount of information on even rare entities such as proper names. Our features also correct coreference for various cases of pronominal anaphora, but these corrections are harder to convey out of context.

Next, we analyze the decision tree built after training the classifier (with all our Web features included). Around 30% of the decision nodes (both non-terminals and leaves) correspond to Web features, and the average error in classification at the Web-feature leaves is only around 2.5%, suggesting that our features are strongly discriminative for pairwise coreference decisions. Some of the most discriminative nodes correspond to the general co-occurrence feature for most (binned) log-count values, the Hearst-style co-occurrence feature for its zero-count value, the cluster-match feature for its zero-match value, and the R2 pronoun context feature for certain (binned) log-count values.

6 Conclusion

We have presented a collection of simple Web-count features for coreference resolution that capture a range of world knowledge via statistics of general lexical co-occurrence, hypernymy, semantic compatibility, and semantic context. When added to a strong decision tree baseline, these features give significant improvements and achieve the best results reported to date, across multiple datasets and metrics.

Acknowledgments

We would like to thank Nathan Gilbert, Adam Pauls, and the anonymous reviewers for their helpful suggestions. This research is supported by Qualcomm via an Innovation Fellowship to the first author and by BBN under DARPA contract HR0011-12-C-0014.

[12] Note that for the ACE04 dataset only, we use the 'SmartInstanceGenerator' (SIG) filter of Reconcile that uses only a filtered set of mention-pairs (based on distance and other properties of the pair) instead of the 'AllPairs' (AP) setting that uses all pairs of mentions, and makes training and tuning very slow.

[13] For the ACE05 and ACE05-ALL datasets, we revert to the 'AllPairs' (AP) setting of Reconcile because this gives us baselines competitive with Haghighi and Klein (2010). Since we did not need to retune on these datasets, training and tuning speed were not a bottleneck. Moreover, the improvements from our Web features are similar even when tried over the SIG baseline; hence, the filter choice doesn't affect the performance gain from the Web features.

References

Amit Bagga and Breck Baldwin. 1998. Algorithms for scoring coreference chains. In *Proceedings of MUC-7 and LREC Workshop*.

Mohit Bansal and Dan Klein. 2011. Web-scale features for full-scale parsing. In *Proceedings of ACL*.

Eric Bengtson and Dan Roth. 2008. Understanding the value of features for coreference resolution. In *Proceedings of EMNLP*.

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

Shane Bergsma, Emily Pitler, and Dekang Lin. 2010. Creating robust supervised classifiers via web-scale n-gram data. In *Proceedings of ACL*.

Thorsten Brants and Alex Franz. 2006. The Google Web 1T 5-gram corpus version 1.1. *LDC2006T13*.

Kenneth Ward Church and Patrick Hanks. 1989. Word association norms, mutual information, and lexicography. In *Proceedings of ACL*.

Hal Daumé III and Daniel Marcu. 2005. A large-scale exploration of effective global features for a joint entity detection and tracking model. In *Proceedings of EMNLP*.

B. Efron and R. Tibshirani. 1993. An introduction to the bootstrap. *Chapman & Hall CRC*.

Aria Haghighi and Dan Klein. 2009. Simple coreference resolution with rich syntactic and semantic features. In *Proceedings of EMNLP*.

Aria Haghighi and Dan Klein. 2010. Coreference resolution in a modular, entity-centered model. In *Proceedings of NAACL-HLT*.

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).

Zellig Harris. 1954. Distributional structure. *Word*, 10(23):146162.

Marti Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of COLING*.

Hamidreza Kobdani, Hinrich Schutze, Michael Schiehlen, and Hans Kamp. 2011. Bootstrapping coreference resolution using word associations. In *Proceedings of ACL*.

Terry Koo, Xavier Carreras, and Michael Collins. 2008. Simple semi-supervised dependency parsing. In *Proceedings of ACL*.

Dekang Lin and Xiaoyun Wu. 2009. Phrase clustering for discriminative learning. In *Proceedings of ACL*.

Dekang Lin, Kenneth Church, Heng Ji, Satoshi Sekine, David Yarowsky, Shane Bergsma, Kailash Patil, Emily Pitler, Rachel Lathbury, Vikram Rao, Kapil Dalwani, and Sushant Narsale. 2010. New tools for web-scale n-grams. In *Proceedings of LREC*.

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

Vincent Ng and Claire Cardie. 2002. Improving machine learning approaches to coreference resolution. In *Proceedings of ACL*.

Vincent Ng. 2010. Supervised noun phrase coreference research: The first fifteen years. In *Proceedings of ACL*.

NIST. 2004. The ACE evaluation plan. In *NIST*.

E.W. Noreen. 1989. Computer intensive methods for hypothesis testing: An introduction. *Wiley, New York*.

Massimo Poesio, Rahul Mehta, Axel Maroudas, and Janet Hitzeman. 2004. Learning to resolve bridging references. In *Proceedings of ACL*.

J. R. Quinlan. 1993. C4.5: Programs for machine learning. *Morgan Kaufmann Publishers Inc., San Francisco, CA, USA*.

Altaf Rahman and Vincent Ng. 2009. Supervised models for coreference resolution. In *Proceedings of EMNLP*.

Altaf Rahman and Vincent Ng. 2011. Coreference resolution with world knowledge. In *Proceedings of ACL*.

Deepak Ravichandran, Patrick Pantel, and Eduard Hovy. 2005. Randomized algorithms and NLP: Using locality sensitive hash functions for high speed noun clustering. In *Proceedings of ACL*.

Marta Recasens and Eduard Hovy. 2010. Coreference resolution across corpora: Languages, coding schemes, and preprocessing information. In *Proceedings of ACL*.

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.

Veselin Stoyanov, Nathan Gilbert, Claire Cardie, and Ellen Riloff. 2009. Conundrums in noun phrase coreference resolution: Making sense of the state-of-the-art. In *Proceedings of ACL/IJCNLP*.

Veselin Stoyanov, Claire Cardie, Nathan Gilbert, Ellen Riloff, David Buttler, and David Hysom. 2010. Reconcile: A coreference resolution research platform. In *Technical report, Cornell University*.

Marc Vilain, John Burger, John Aberdeen, Dennis Connolly, and Lynette Hirschman. 1995. A model-theoretic coreference scoring scheme. In *Proceedings of MUC-6*.

Xiaofeng Yang and Jian Su. 2007. Coreference resolution using semantic relatedness information from automatically discovered patterns. In *Proceedings of ACL*.

Xiaofeng Yang, Jian Su, and Chew Lim Tan. 2005. Improving pronoun resolution using statistics-based semantic compatibility information. In *Proceedings of ACL*.