Understanding the Value of Features for Coreference Resolution

Eric Bengtson **Dan Roth**
Department of Computer Science
University of Illinois
Urbana, IL 61801
{ebengt2,danr}@illinois.edu

Abstract

In recent years there has been substantial work on the important problem of coreference resolution, most of which has concentrated on the development of new models and algorithmic techniques. These works often show that complex models improve over a weak pairwise baseline. However, less attention has been given to the importance of selecting strong features to support learning a coreference model.

This paper describes a rather simple pairwise classification model for coreference resolution, developed with a well-designed set of features. We show that this produces a state-of-the-art system that outperforms systems built with complex models. We suggest that our system can be used as a baseline for the development of more complex models – which may have less impact when a more robust set of features is used. The paper also presents an ablation study and discusses the relative contributions of various features.

1 Introduction

Coreference resolution is the task of grouping all the mentions of entities[1] in a document into equivalence classes so that all the mentions in a given class refer to the same discourse entity. For example, given the sentence (where the head noun of each mention is subscripted)

> *An American$_1$ official$_2$ announced that American$_1$ President$_3$ Bill Clinton$_3$ met his$_3$ Russian$_4$ counterpart$_5$, Vladimir Putin$_5$, today.*

the task is to group the mentions so that those referring to the same entity are placed together into an equivalence class.

Many NLP tasks detect attributes, actions, and relations between discourse entities. In order to discover all information about a given entity, textual mentions of that entity must be grouped together. Thus coreference is an important prerequisite to such tasks as textual entailment and information extraction, among others.

Although coreference resolution has received much attention, that attention has not focused on the relative impact of high-quality features. Thus, while many structural innovations in the modeling approach have been made, those innovations have generally been tested on systems with features whose strength has not been established, and compared to weak pairwise baselines. As a result, it is possible that some modeling innovations may have less impact or applicability when applied to a stronger baseline system.

This paper introduces a rather simple but state-of-the-art system, which we intend to be used as a strong baseline to evaluate the impact of structural innovations. To this end, we combine an effective coreference classification model with a strong set of features, and present an ablation study to show the relative impact of a variety of features.

As we show, this combination of a pairwise model and strong features produces a 1.5 percent-

[1] We follow the ACE (NIST, 2004) terminology: A noun phrase referring to a discourse entity is called a *mention*, and an equivalence class is called an *entity*.

Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing, pages 294–303,
Honolulu, October 2008.

age point increase in B-Cubed F-Score over a complex model in the state-of-the-art system by Culotta et al. (2007), although their system uses a complex, non-pairwise model, computing features over partial clusters of mentions.

2 A Pairwise Coreference Model

Given a document and a set of mentions, coreference resolution is the task of grouping the mentions into equivalence classes, so that each equivalence class contains exactly those mentions that refer to the same discourse entity. The number of equivalence classes is not specified in advance, but is bounded by the number of mentions.

In this paper, we view coreference resolution as a graph problem: Given a set of mentions and their context as nodes, generate a set of edges such that any two mentions that belong in the same equivalence class are connected by some path in the graph. We construct this entity-mention graph by learning to decide for each mention which preceding mention, if any, belongs in the same equivalence class; this approach is commonly called the pairwise coreference model (Soon et al., 2001). To decide whether two mentions should be linked in the graph, we learn a pairwise coreference function *pc* that produces a value indicating the probability that the two mentions should be placed in the same equivalence class.

The remainder of this section first discusses how this function is used as part of a document-level coreference decision model and then describes how we learn the *pc* function.

2.1 Document-Level Decision Model

Given a document d and a pairwise coreference scoring function pc that maps an ordered pair of mentions to a value indicating the probability that they are coreferential (see Section 2.2), we generate a coreference graph G_d according to the Best-Link decision model (Ng and Cardie, 2002b) as follows:

For each mention m in document d, let B_m be the set of mentions appearing before m in d. Let a be the highest scoring antecedent:

$$a = \underset{b \in B_m}{\operatorname{argmax}}(pc(b, m)).$$

If $pc(a, m)$ is above a threshold chosen as described in Section 4.4, we add the edge (a, m) to the coreference graph G_d.

The resulting graph contains connected components, each representing one equivalence class, with all the mentions in the component referring to the same entity. This technique permits us to learn to detect some links between mentions while being agnostic about whether other mentions are linked, and yet via the transitive closure of all links we can still determine the equivalence classes.

We also require that no non-pronoun can refer back to a pronoun: If m is not a pronoun, we do not consider pronouns as candidate antecedents.

2.1.1 Related Models

For pairwise models, it is common to choose the best antecedent for a given mention (thereby imposing the constraint that each mention has at most one antecedent); however, the method of deciding which is the best antecedent varies.

Soon et al. (2001) use the Closest-Link method: They select as an antecedent the closest preceding mention that is predicted coreferential by a pairwise coreference module; this is equivalent to choosing the closest mention whose pc value is above a threshold. Best-Link was shown to outperform Closest-Link in an experiment by Ng and Cardie (2002b). Our model differs from that of Ng and Cardie in that we impose the constraint that non-pronouns cannot refer back to pronouns, and in that we use as training examples all ordered pairs of mentions, subject to the constraint above.

Culotta et al. (2007) introduced a model that predicts whether a pair of equivalence classes should be merged, using features computed over all the mentions in both classes. Since the number of possible classes is exponential in the number of mentions, they use heuristics to select training examples. Our method does not require determining which equivalence classes should be considered as examples.

2.2 Pairwise Coreference Function

Learning the pairwise scoring function pc is a crucial issue for the pairwise coreference model. We apply machine learning techniques to learn from examples a function pc that takes as input an ordered pair of mentions (a, m) such that a precedes m in the document, and produces as output a value that is

interpreted as the conditional probability that m and a belong in the same equivalence class.

2.2.1 Training Example Selection

The ACE training data provides the equivalence classes for mentions. However, for some pairs of mentions from an equivalence class, there is little or no direct evidence in the text that the mentions are coreferential. Therefore, training *pc* on all pairs of mentions within an equivalence class may not lead to a good predictor. Thus, for each mention m we select from m's equivalence class the closest preceding mention a and present the pair (a, m) as a positive training example, under the assumption that there is more direct evidence in the text for the existence of this edge than for other edges. This is similar to the technique of Ng and Cardie (2002b). For each m, we generate negative examples (a, m) for all mentions a that precede m and are not in the same equivalence class. Note that in doing so we generate more negative examples than positive ones.

Since we never apply *pc* to a pair where the first mention is a pronoun and the second is not a pronoun, we do not train on examples of this form.

2.2.2 Learning Pairwise Coreference Scoring Model

We learn the pairwise coreference function using an averaged perceptron learning algorithm (Freund and Schapire, 1998) – we use the regularized version in Learning Based Java[2] (Rizzolo and Roth, 2007).

3 Features

The performance of the document-level coreference model depends on the quality of the pairwise coreference function *pc*. Beyond the training paradigm described earlier, the quality of *pc* depends on the features used.

We divide the features into categories, based on their function. A full list of features and their categories is given in Table 2. In addition to these boolean features, we also use the conjunctions of all pairs of features.[3]

[2]LBJ code is available at `http://L2R.cs.uiuc.edu/~cogcomp/asoftware.php?skey=LBJ`

[3]The package of all features used is available at `http://L2R.cs.uiuc.edu/~cogcomp/asoftware.php?skey=LBJ#features`.

In the following description, the term *head* means the head noun phrase of a mention; the *extent* is the largest noun phrase headed by the head noun phrase.

3.1 Mention Types

The type of a mention indicates whether it is a proper noun, a common noun, or a pronoun. This feature, when conjoined with others, allows us to give different weight to a feature depending on whether it is being applied to a proper name or a pronoun. For our experiments in Section 5, we use gold mention types as is done by Culotta et al. (2007) and Luo and Zitouni (2005).

Note that in the experiments described in Section 6 we predict the mention types as described there and do not use any gold data. The mention type feature is used in all experiments.

3.2 String Relation Features

String relation features indicate whether two strings share some property, such as one being the substring of another or both sharing a modifier word. Features are listed in Table 1. Modifiers are limited to those occurring before the head.

Feature	Definition
Head match	$head_i == head_j$
Extent match	$extent_i == extent_j$
Substring	$head_i$ substring of $head_j$
Modifiers Match	$mod_i == (head_j$ or $mod_j)$
Alias	$acronym(head_i) == head_j$ or $lastname_i == lastname_j$

Table 1: String Relation Features

3.3 Semantic Features

Another class of features captures the semantic relation between two words. Specifically, we check whether gender or number match, or whether the mentions are synonyms, antonyms, or hypernyms. We also check the relationship of modifiers that share a hypernym. Descriptions of the methods for computing these features are described next.

Gender Match We determine the gender (male, female, or neuter) of the two phrases, and report whether they match (true, false, or unknown). For

Category	Feature	Source
Mention Types	Mention Type Pair	Annotation and tokens
String Relations	Head Match	Tokens
	Extent Match	Tokens
	Substring	Tokens
	Modifiers Match	Tokens
	Alias	Tokens and lists
Semantic	Gender Match	WordNet and lists
	Number Match	WordNet and lists
	Synonyms	WordNet
	Antonyms	WordNet
	Hypernyms	WordNet
	Both Speak	Context
Relative Location	Apposition	Positions and context
	Relative Pronoun	Positions and tokens
	Distances	Positions
Learned	Anaphoricity	Learned
	Name Modifiers Predicted Match	Learned
Aligned Modifiers	Aligned Modifiers Relation	WordNet and lists
Memorization	Last Words	Tokens
Predicted Entity Types	Entity Types Match	Annotation and tokens
	Entity Type Pair	WordNet and tokens

Table 2: Features by Category

a proper name, gender is determined by the existence of *mr*, *ms*, *mrs*, or the gender of the first name. If only a last name is found, the phrase is considered to refer to a person. If the name is found in a comprehensive list of cities or countries, or ends with an organization ending such as *inc*, then the gender is neuter. In the case of a common noun phrase, the phrase is looked up in WordNet (Fellbaum, 1998), and it is assigned a gender according to whether *male*, *female*, *person*, *artifact*, *location*, or *group* (the last three correspond to neuter) is found in the hypernym tree. The gender of a pronoun is looked up in a table.

Number Match Number is determined as follows: Phrases starting with the words *a*, *an*, or *this* are singular; *those*, *these*, or *some* indicate plural. Names not containing *and* are singular. Common nouns are checked against extensive lists of singular and plural nouns – words found in neither or both lists have unknown number. Finally, if the number is unknown yet the two mentions have the same spelling, they are assumed to have the same number.

WordNet Features We check whether any sense of one head noun phrase is a synonym, antonym, or hypernym of any sense of the other. We also check whether any sense of the phrases share a hypernym, after dropping *entity*, *abstraction*, *physical entity*, *object*, *whole*, *artifact*, and *group* from the senses, since they are close to the root of the hypernym tree.

Modifiers Match Determines whether the text before the head of a mention matches the head or the text before the head of the other mention.

Both Mentions Speak True if both mentions appear within two words of a verb meaning *to say*. Being in a window of size two is an approximation to being a syntactic subject of such a verb. This feature is a proxy for having similar semantic types.

3.4 Relative Location Features

Additional evidence is derived from the relative location of the two mentions. We thus measure distance (quantized as multiple boolean features of the

form $[distance \geq i]$) for all i up to the distance and less than some maximum, using units of compatible mentions, and whether the mentions are in the same sentence. We also detect apposition (mentions separated by a comma). For details, see Table 3.

Feature	Definition
Distance	In same sentence # compatible mentions
Apposition	m_1, m_2 found
Relative Pronoun	Apposition and m_2 is PRO

Table 3: Location Features. Compatible mentions are those having the same gender and number.

3.5 Learned Features

Modifier Names If the mentions are both modified by other proper names, use a basic coreference classifier to determine whether the modifiers are coreferential. This basic classifier is trained using Mention Types, String Relations, Semantic Features, Apposition, Relative Pronoun, and Both Speak. For each mention m, examples are generated with the closest antecedent a to form a positive example, and every mention between a and m to form negative examples.

Anaphoricity Ng and Cardie (2002a) and Denis and Baldridge (2007) show that when used effectively, explicitly predicting anaphoricity can be helpful. Thus, we learn a separate classifier to detect whether a mention is anaphoric (that is, whether it is not the first mention in its equivalence class), and use that classifier's output as a feature for the coreference model. Features for the anaphoricity classifier include the mention type, whether the mention appears in a quotation, the text of the first word of the extent, the text of the first word after the head (if that word is part of the extent), whether there is a longer mention preceding this mention and having the same head text, whether any preceding mention has the same extent text, and whether any preceding mention has the same text from beginning of the extent to end of the head. Conjunctions of all pairs of these features are also used. This classifier predicts anaphoricity with about 82% accuracy.

3.6 Aligned Modifiers

We determine the relationship of any pair of modifiers that share a hypernym. Each aligned pair may have one of the following relations: match, substring, synonyms, hypernyms, antonyms, or mismatch. Mismatch is defined as none of the above. We restrict modifiers to single nouns and adjectives occurring before the head noun phrase.

3.7 Memorization Features

We allow our system to learn which pairs of nouns tend to be used to mention the same entity. For example, *President* and *he* often refer to *Bush* but *she* and *Prime Minister* rarely do, if ever. To enable the system to learn such patterns, we treat the presence or absence of each pair of final head nouns, one from each mention of an example, as a feature.

3.8 Predicted Entity Type

We predict the entity type (*person*, *organization*, *geo-political entity*, *location*, *facility*, *weapon*, or *vehicle*) as follows: If a proper name, we check a list of personal first names, and a short list of honorary titles (e.g. *mr*) to determine if the mention is a person. Otherwise we look in lists of personal last names drawn from US census data, and in lists of cities, states, countries, organizations, corporations, sports teams, universities, political parties, and organization endings (e.g. *inc* or *corp*). If found in exactly one list, we return the appropriate type. We return *unknown* if found in multiple lists because the lists are quite comprehensive and may have significant overlap.

For common nouns, we look at the hypernym tree for one of the following: *person*, *political unit*, *location*, *organization*, *weapon*, *vehicle*, *industrial plant*, and *facility*. If any is found, we return the appropriate type. If multiple are found, we sort as in the above list.

For personal pronouns, we recognize the entity as a person; otherwise we specify unknown.

This computation is used as part of the following two features.

Entity Type Match This feature checks to see whether the predicted entity types match. The result is true if the types are identical, false if they are different, and unknown if at least one type is unknown.

Entity Type Conjunctions This feature indicates the presence of the pair of predicted entity types for the two mentions, except that if either word is a pronoun, the word token replaces the type in the pair. Since we do this replacement for entity types, we also add a similar feature for mention types here. These features are boolean: For any given pair, a feature is active if that pair describes the example.

3.9 Related Work

Many of our features are similar to those described in Culotta et al. (2007). This includes Mention Types, String Relation Features, Gender and Number Match, WordNet Features, Alias, Apposition, Relative Pronoun, and Both Mentions Speak. The implementations of those features may vary from those of other systems. Anaphoricity has been proposed as a part of the model in several systems, including Ng and Cardie (2002a), but we are not aware of it being used as a feature for a learning algorithm. Distances have been used in e.g. Luo et al. (2004). However, we are not aware of any system using the number of compatible mentions as a distance.

4 Experimental Study

4.1 Corpus

We use the official ACE 2004 English training data (NIST, 2004). Much work has been done on coreference in several languages, but for this work we focus on English text. We split the corpus into three sets: Train, Dev, and Test. Our test set contains the same 107 documents as Culotta et al. (2007). Our training set is a random 80% of the 336 documents in their training set and our Dev set is the remaining 20%.

For our ablation study, we further randomly split our development set into two evenly sized parts, Dev-Tune and Dev-Eval. For each experiment, we set the parameters of our algorithm to optimize B-Cubed F-Score using Dev-Tune, and use those parameters to evaluate on the Dev-Eval data.

4.2 Preprocessing

For the experiments in Section 5, following Culotta et al. (2007), to make experiments more comparable across systems, we assume that perfect mention boundaries and mention type labels are given. We do not use any other gold annotated input at evaluation time. In Section 6 experiments we do not use any gold annotated input and do not assume mention types or boundaries are given. In all experiments we automatically split words and sentences using our preprocessing tools.[4]

4.3 Evaluation Scores

B-Cubed F-Score We evaluate over the commonly used B-Cubed F-Score (Bagga and Baldwin, 1998), which is a measure of the overlap of predicted clusters and true clusters. It is computed as the harmonic mean of precision (P),

$$P = \frac{1}{N} \sum_{d \in D} \left(\sum_{m \in d} \left(\frac{c_m}{p_m} \right) \right), \quad (1)$$

and recall (R),

$$R = \frac{1}{N} \sum_{d \in D} \left(\sum_{m \in d} \left(\frac{c_m}{t_m} \right) \right), \quad (2)$$

where c_m is the number of mentions appearing both in m's predicted cluster and in m's true cluster, p_m is the size of the predicted cluster containing m, and t_m is the size of m's true cluster. Finally, d represents a document from the set D, and N is the total number of mentions in D.

B-Cubed F-Score has the advantage of being able to measure the impact of singleton entities, and of giving more weight to the splitting or merging of larger entities. It also gives equal weight to all types of entities and mentions. For these reasons, we report our results using B-Cubed F-Score.

MUC F-Score We also provide results using the official MUC scoring algorithm (Vilain et al., 1995). The MUC F-score is also the harmonic mean of precision and recall. However, the MUC precision counts precision errors by computing the minimum number of links that must be added to ensure that all mentions referring to a given entity are connected in the graph. Recall errors are the number of links that must be removed to ensure that no two mentions referring to different entities are connected in the graph.

[4] The code is available at `http://L2R.cs.uiuc.edu/~cogcomp/tools.php`

4.4 Learning Algorithm Details

We train a regularized average perceptron using examples selected as described in Section 2.2.1. The learning rate is 0.1 and the regularization parameter (separator thickness) is 3.5. At training time, we use a threshold of 0.0, but when evaluating, we select parameters to optimize B-Cubed F-Score on a held-out development set. We sample all even integer thresholds from -16 to 8. We choose the number of rounds of training similarly, allowing any number from one to twenty.

5 Results

	Precision	Recall	B^3 F
Culotta et al.	86.7	73.2	79.3
Current Work	88.3	74.5	**80.8**

Table 4: Evaluation on unseen Test Data using B^3 score. Shows that our system outperforms the advanced system of Culotta et al. The improvement is statistically significant at the $p = 0.05$ level according to a non-parametric bootstrapping percentile test.

In Table 4, we compare our performance against a system that is comparable to ours: Both use gold mention boundaries and types, evaluate using B-Cubed F-Score, and have the same training and test data split. Culotta et al. (2007) is the best comparable system of which we are aware.

Our results show that a pairwise model with strong features outperforms a state-of-the-art system with a more complex model.

MUC Score We evaluate the performance of our system using the official MUC score in Table 5.

MUC Precision	MUC Recall	MUC F
82.7	69.9	75.8

Table 5: Evaluation of our system on unseen Test Data using MUC score.

5.1 Analysis of Feature Contributions

In Table 6 we show the relative impact of various features. We report data on Dev-Eval, to avoid the possibility of overfitting by feature selection. The parameters of the algorithm are chosen to maximize the BCubed F-Score on the Dev-Tune data. Note that since we report results on Dev-Eval, the results in Table 6 are not directly comparable with Culotta et al. (2007). For comparable results, see Table 4 and the discussion above.

Our ablation study shows the impact of various classes of features, indicating that almost all the features help, although some more than others. It also illustrates that some features contribute more to precision, others more to recall. For example, aligned modifiers contribute primarily to precision, whereas our learned features and our apposition features contribute to recall. This information can be useful when designing a coreference system in an application where recall is more important than precision, or vice versa.

We examine the effect of some important features, selecting those that provide a substantial improvement in precision, recall, or both. For each such feature we examine the rate of coreference amongst mention pairs for which the feature is active, compared with the overall rate of coreference. We also show examples on which the coreference systems differ depending on the presence or absence of a feature.

Apposition This feature checks whether two mentions are separated by only a comma, and it increases B-Cubed F-Score by about one percentage point. We hypothesize that proper names and common noun phrases link primarily through apposition, and that apposition is thus a significant feature for good coreference resolution.

When this feature is active 36% of the examples are coreferential, whereas only 6% of all examples are coreferential. Looking at some examples our system begins to get right when apposition is added, we find the phrase

> *Israel's Deputy Defense Minister, Ephraim Sneh.*

Upon adding apposition, our system begins to correctly associate *Israel's Deputy Defense Minister* with *Ephraim Sneh*. Likewise in the phrase

> *The court president, Ronald Sutherland*,

the system correctly associates *The court president* with *Ronald Sutherland* when they appear in an appositive relation in the text. In addition, our system

	Precision	Recall	B-Cubed F
String Similarity	86.88	67.17	75.76
+ Semantic Features	85.34	69.30	**76.49**
+ Apposition	89.77	67.53	77.07
+ Relative Pronoun	88.76	68.97	77.62
+ Distances	89.62	71.93	79.81
+ Learned Features	87.37	74.51	**80.43**
+ Aligned Modifiers	88.70	74.30	80.86
+ Memorization	86.57	75.59	80.71
+ Predicted Entity Types	87.92	76.46	81.79

Table 6: Contribution of Features as evaluated on a development set. Bold results are significantly better than the previous line at the $p = 0.05$ level according to a paired non-parametric bootstrapping percentile test. These results show the importance of Distance, Entity Type, and Apposition features.

begins correctly associating relative pronouns such as *who* with their referents in phrases like

Sheikh Abbad, who died 500 years ago.

although an explicit relative pronoun feature is added only later.

Although this feature may lead the system to link comma separated lists of entities due to misinterpretation of the comma, for example *Wyoming* and *western South Dakota* in a list of locations, we believe this can be avoided by refining the apposition feature to ignore lists.

Relative Pronoun Next we investigate the relative pronoun feature. With this feature active, 93% of examples were positive, indicating the precision of this feature. Looking to examples, we find *who* in

the official, who wished to remain anonymous

is properly linked, as is *that* in

nuclear warheads that can be fitted to missiles.

Distances Our distance features measure separation of two mentions in number of compatible mentions (quantized), and whether the mentions are in the same sentence. Distance features are important for a system that makes links based on the best pairwise coreference value rather than implicitly incorporating distance by linking only the closest pair whose score is above a threshold, as done by e.g. Soon et al. (2001).

Looking at examples, we find that adding distances allows the system to associate the pronoun *it* with *this missile* not separated by any mentions, rather than *Tehran*, which is separated from *it* by many mentions.

Predicted Entity Types Since no two mentions can have different entity types (person, organization, geo-political entity, etc.) and be coreferential, this feature has strong discriminative power. When the entity types match, 13% of examples are positive compared to only 6% of examples in general. Qualitatively, the entity type prediction correctly recognizes *the Gulf region* as a geo-political entity, and *He* as a person, and thus prevents linking the two. Likewise, the system discerns *Baghdad* from *ambassador* due to the entity type. However, in some cases an identity type match can cause the system to be overly confident in a bad match, as in the case of a *palestinian state* identified with *holy Jerusalem* on the basis of proximity and shared entity type. This type of example may require some additional world knowledge or deeper comprehension of the document.

6 End-to-End Coreference

The ultimate goal for a coreference system is to process unannotated text. We use the term *end-to-end coreference* for a system capable of determining coreference on plain text. We describe the challenges associated with an end-to-end system, describe our approach, and report results below.

6.1 Challenges

Developing an end-to-end system requires detecting and classifying mentions, which may degrade coreference results. One challenge in detecting mentions is that they are often heavily nested. Additionally, there are issues with evaluating an end-to-end system against a gold standard corpus, resulting from the possibility of mismatches in mention boundaries, missing mentions, and additional mentions detected, along with the need to align detected mentions to their counterparts in the annotated data.

6.2 Approach

We resolve coreference on unannotated text as follows: First we detect mention heads following a state of the art chunking approach (Punyakanok and Roth, 2001) using standard features. This results in a 90% F_1 head detector. Next, we detect the extent boundaries for each head using a learned classifier. This is followed by determining whether a mention is a proper name, common noun phrase, prenominal modifier, or pronoun using a learned mention type classifier that. Finally, we apply our coreference algorithm described above.

6.3 Evaluation and Results

To evaluate, we align the heads of the detected mentions to the gold standard heads greedily based on number of overlapping words. We choose not to impute errors to the coreference system for mentions that were not detected or for spuriously detected mentions (following Ji et al. (2005) and others). Although this evaluation is lenient, given that the mention detection component performs at over 90% F_1, we believe it provides a realistic measure for the performance of the end-to-end system and focuses the evaluation on the coreference component. The results of our end-to-end coreference system are shown in Table 7.

	Precision	Recall	B^3 F
End-to-End System	84.91	72.53	78.24

Table 7: Coreference results using detected mentions on unseen Test Data.

7 Conclusion

We described and evaluated a state-of-the-art coreference system based on a pairwise model and strong features. While previous work showed the impact of complex models on a weak pairwise baseline, the applicability and impact of such models on a strong baseline system such as ours remains uncertain. We also studied and demonstrated the relative value of various types of features, showing in particular the importance of distance and apposition features, and showing which features impact precision or recall more. Finally, we showed an end-to-end system capable of determining coreference in a plain text document.

Acknowledgments

We would like to thank Ming-Wei Chang, Michael Connor, Alexandre Klementiev, Nick Rizzolo, Kevin Small, and the anonymous reviewers for their insightful comments. This work is partly supported by NSF grant SoD-HCER-0613885 and a grant from Boeing.

References

A. Bagga and B. Baldwin. 1998. Algorithms for scoring coreference chains. In *MUC7*.

A. Culotta, M. Wick, R. Hall, and A. McCallum. 2007. First-order probabilistic models for coreference resolution. In *HLT/NAACL*, pages 81–88.

P. Denis and J. Baldridge. 2007. Joint determination of anaphoricity and coreference resolution using integer programming. In *HLT/NAACL*, pages 236–243, Rochester, New York.

C. Fellbaum. 1998. *WordNet: An Electronic Lexical Database*. MIT Press.

Y. Freund and R. E. Schapire. 1998. Large margin classification using the Perceptron algorithm. In *COLT*, pages 209–217.

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

X. Luo and I. Zitouni. 2005. Multi-lingual coreference resolution with syntactic features. In *HLT/EMNLP*, pages 660–667, Vancouver, British Columbia, Canada.

X. Luo, A. Ittycheriah, H. Jing, N. Kambhatla, and S. Roukos. 2004. A mention-synchronous coreference resolution algorithm based on the bell tree. In *ACL*, page 135, Morristown, NJ, USA.

V. Ng and C. Cardie. 2002a. Identifying anaphoric and non-anaphoric noun phrases to improve coreference resolution. In *COLING-2002*.

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

NIST. 2004. The ace evaluation plan. www.nist.gov/speech/tests/ace/index.htm.

V. Punyakanok and D. Roth. 2001. The use of classifiers in sequential inference. In *The Conference on Advances in Neural Information Processing Systems (NIPS)*, pages 995–1001. MIT Press.

N. Rizzolo and D. Roth. 2007. Modeling Discriminative Global Inference. In *Proceedings of the First International Conference on Semantic Computing (ICSC)*, pages 597–604, Irvine, California.

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

M. Vilain, J. Burger, J. Aberdeen, D. Connolly, and L. Hirschman. 1995. A model-theoretic coreference scoring scheme. In *MUC6*, pages 45–52.