Machine Learning for Coreference Resolution: Recent Developments

Yimeng Zhang Yangbo Zhu

Abstract

This paper surveys recent developments of machine learning methods for coreference resolution. Accurate coreference resolution requires leveraging all levels of knowledge about human language, including morphological, syntactic, semantic and pragmatic analysis. The success of a machine learning system depends on several major choices: the feature set, the training algorithm and the resolution algorithm. We discuss all three aspects in detail. Because coreference resolution is inherently a clustering problem, the evaluation is less straightforward than that for classification. We compare all widely used evaluation metrics for this task. We also present some empirical evaluation results of existing systems on public data sets, and compare them along different dimensions.

1 Introduction

The goal of coreference resolution is to find all noun phrases (NPs) referring to the same real-world entity. An NP can be a pronominal phrase (e.g. he), a nominal phrase (e.g. the king) or a named entity (e.g. Carnegie Mellon). Coreference resolution is easy for humans, but hard for computers. It can be applied to various tasks including information extraction, information retrieval, question answering, machine translation and text summarization. As in many areas of NLP, the early methods for coreference resolution were built with hand-crafted rules. In the past decade, machine learning methods have shown advantages over knowledge-based ones.

This paper will focus on developments since 2002, highlighting some important ideas from earlier literature. For a comprehensive overview of coreference resolution before 2002, readers are referred to (Ng, 2002) and (Elango, 2005). This paper will not cover cross-document entity resolution (Bhattacharya and Getoor, 2006) and abstract reference resolution (Byron, 2002).

Coreference resolution is essentially a clustering problem. NPs in a document are partitioned into clusters, with each cluster referring to a single entity. A typical coreference resolution system consists of five key components:

(1) NP identification. Named entity recognizer and NP chunkers are used to identify NPs in the document.

(2) Feature generation. Many NLP tools (e.g., POS taggers, parsers) are used to produce a rich set of features, including positional, morphological, syntactic and semantic ones.

(3) Distance measure. The foundation of a clustering algorithm is the distance measure. NPs close to each other are connected to form a cluster. The distance can be calculated by combining features in

different ways. Besides NP-to-NP distance, some clustering algorithms also need NP-to-cluster, or even cluster-to-cluster distance.

(4) Clustering algorithm. Given a distance measure, we can use different clustering algorithms to partition the set of NPs. The early methods are mostly local and greedy. They connect NP to the nearest NP, and incrementally construct the cluster. More recent methods try to find the globally optimal clustering results. Since the search space for clustering is exponentially large, approximation algorithms are needed for global methods. The clustering problem can also be cast to a sequence search problem, which can be solved using beam search algorithms.

(5) Training instance construction. Both the distance measure and clustering algorithm have parameters, which can be learned from training data. Different clustering algorithms require different instance (positive and negative) construction methods, which is not always as straightforward as in classification problems.

(6) Testing instance construction. During testing (decoding), the cluster labels are unknown. Therefore, the instance construction is slightly different from that of training.

The rest of this paper is structured as follows: Section 2 describes commonly used features. Section 3 presents methods for evaluating the likelihood of coreference between NPs. Several clustering algorithms (from local to global) are presented in Section 4. Three learning styles (from supervised to unsupervised) are discussed in Section 5. Section 6 briefly lists some commonly used data sets, analyzes five evaluation methods, and presents empirical evaluation results for different systems. Section 7 concludes the paper and provides directions for future research.

2 Features

Virtually all levels of linguistic analysis can be useful for coreference resolution. Pre-processing is required to produce a rich set of features. Commonly used pre-processing modules include morphological processing, NP chunking, POS taggers, named entity recognizers and parsers. The performance of coreference resolution heavily depends on the quality of pre-processing. In this section, we present a set of basic features commonly used by current coreference resolution systems. Then we introduce semantic similarity and relatedness features derived from large corpora (e.g., Wikipedia and the Web). Finally we present some high-level features including anaphoricity and coreferentiality.

2.1 Basic Features

Soon et al. (2001) present a set of 12 features, which Ng and Cardie (2002b) augment to include 53 features, including positional, morphological, lexical, syntactic, semantic and even pragmatic features. Most current systems are built on this feature set, with some modification and augmentation.

Table 1 shows the basic features used by current coreference resolution systems. It differentiates hard constraints and soft features. Zhou and Su (2004) use hard constraints to reduce the number of candidate antecedents, and use soft features to pick the most likely antecedent for each anaphor. In practice, hard constraints are often treated as soft features, because the pre-processors required to produce features are not perfect. Schiehlen (2004) shows that most common hard constraints are not truly “hard” (at least for German). Strictly applying any single hard constraint will filter out around 10% of real antecedents. Fortunately, current machine learning algorithms are good at weighting features by their usefulness and combining evidences from different sources.

Table 1: Basic features for coreference resolution

(features with * are hard constraint, others are soft features)

Feature	Description
Positional features	
*Span	In nested NPs (e.g., the captains' daughter), one NP spans the other, which implies they are different entities.
Distance	Number of words, nouns or sentences between two NPs.
Morphological features	
*Gender	Agreement on masculine, feminine or neuter.
*Number	Agreement on singular, dual or plural.
*Animacy	Agreement on humans, animals, plants, natural forces or others.
String matching	Complete or partial string matching.
Alias	For example, IBM is an alias for International Business Machines Corp.
Minimum edit distance	Strube et al. (2002) showed that Wagner and Fischer's minimum edit distance is an effective feature for German coreference resolution.
Part-of-speech	Categories of NPs, including pronoun, proper nouns, demonstrative NPs (this car), definite NPs (the car) and indefinite NPs (a car). Modjeska et al. (2003) specifically studied "other-anaphora" (e.g., "other", "another").
Syntactic features	
*Apposition	Two NPs are placed side by side.
*Predicate nominal construction	Two NPs form a predicate nominal construction.
*Binding	Two NPs conform to principles B and C in the Binding Theory (Chomsky 1981).[1] The Binding Theory defines conditions for two NPs to be coreferent or non-coreferent.
*Contra-indices	NPs cannot be co-indexed based on simple heuristics. For example, in "CMU in Pittsburgh", CMU and Pittsburgh must be different entities.
Maximal NP projection	Two NPs have the same maximal NP projection.
Parse tree similarity	The similarity between the sub parse trees covering the anaphor and the antecedent. (Yang et al., 2006)
Collocation Match	Two NPs precede or follow different appearances of the same verb.

[1] See http://en.wikipedia.org/wiki/Government_and_binding for details.

Table 1: Basic features for coreference resolution

(features with * are hard constraint, others are soft features)

Feature	**Description**
Syntactic parallelism	Two NPs have the same syntactic roles.
Semantic features	
Named Entity class	Common classes provided by NER include person, organization, location, facility and GPE (geographical-political entity).
WordNet Semantic class	Two NPs have the same WordNet semantic class. Versley (2007) uses GermaNet (a part of EuroWordNet) for German coreference resolution.

2.2 Corpus-based Semantic Features

The WordNet semantic class feature is widely used for coreference resolution. However, it has several limitations. First, it is manually constructed, and therefore it has limited coverage. Specifically, it does not cover most proper nouns. Second, WordNet only provides hierarchical hyponyms and synonyms, but cannot measure the general relatedness between two NPs. Third, WordNet provides a list of senses for each word. Most systems just pick the first sense, because it is hard to pick the right sense without any contextual information. In order to overcome this knowledge gap, researchers try to derive semantic relatedness information from large corpora, like Wikipedia and the Web.

Modjeska et al. (2003) use simple patterns to extract co-occurrence frequency of two NPs. For example, if the pattern "X and other Y" occurs frequently, then X and Y are probably synonyms. They get individual frequencies of X and Y from Google, and calculate the pointwise mutual information (PMI) between X and Y. If the PMI is high, they are likely to be coreferent. Martket and Nissim (2005) compare the knowledge extracted from WordNet, British National Corpus and the Web. Their Web-based method outperforms WordNet-based method on other-anaphora resolution, and achieves comparable performance with a WordNet-based method on definite-NP coreference resolution. Poesion et al. (2004a) use Google to count patterns in the Web to extract mereological relationship. For example, if "the X of the Y" appears very often, then X is probably a meronym of Y. Bergsma and Lin (2006) use dependency parser to extract more complex patterns.

Bean and Riloff (2004) use case frames to represent the contextual role of NPs. A case frame is a frequent pattern surrounding an NP. The intuition is that NPs frequently appearing in the same case frame are likely to be semantically related or equivalent. On the other hand, case frames frequently surrounding the same NP are likely to be related. They propose a semi-supervised method to extract case frames from large corpora.

Yang et al. (2005) try to improve pronominal anaphor resolution using semantic compatibility information, which refers to the compatibility between components of the sentences, such as subject-verb. For example, we want to find the resolution for "it" followed by "collect", and have two candidates: "government" and "money", we would prefer to choose "government" by the semantic compatibility that it is more likely for "government collects" than "money collects". The paper presents an approach to automatically extract such information from a large corpus and the Web, and use the information as features for anaphor resolution. A single-candidate model and a twin-candidate model (see Section 3.3) are used as the resolution algorithm. The experimental results showed that the features give more improvement on the twin-candidate model than the single-candidate one. For semantic compatibility information, three relationships are considered: possessive-noun, subject-verb,

and verb-object. For both corpus-based and web-based algorithms, named entities are replaced with common nouns (e.g., “IBM” -> “company”) to extract the patterns.

Ponzetto and Strube (2006) extract semantic similarity and relatedness from Wikipedia. Wikipedia provides entity taxonomy as a directed graph rather than a tree. The relatedness between two NPs is measured by their position in the taxonomy. The similarity is measured by phrasal overlap between two pages with the NPs as titles. They also obtain semantic parsing information using *semantic role labeling* (SRL). They use the ASSERT parser to get the semantic roles of NPs, and compare the similarity of their corresponding argument-predicate pairs.

Ng (2007a) uses a Named Entity Recognizer (NER) to label proper nouns with semantic classes, and uses the MINIPAR dependency parser to extract appositive relations between proper nouns and common nouns (e.g. <Eastern Airline, the carrier>). All such pairs are extracted from the BLLIP corpus (30M words). The semantic class of a common noun is inferred from the most frequently co-occuring class of proper nouns. Ng (2007b) extracts semantic class from the BBN Entity Type Corpus, which contains all Penn Treebank articles annotated with class information.

Yang and Su (2007) extracts semantic relatedness from Wikipedia. The semantic relatedness is referred to as patterns, like “X such as Y” in this paper. Instead of using some pre-defined patterns, the paper proposes an algorithm to automatically extract patterns from the database from a set of seed NP pairs. To this end, the algorithm utilizes a set of NP pairs with known coreference relationships from the training data as seeds. For each seed pair, it searches the database for the texts where the two NPs co-occur, and collect the surrounding words as the patterns. The patterns derived from the positive seed pairs are reference patterns. To use the reference patterns in the resolution task, two approaches are proposed. For the first one, the 100 most frequent reference patterns are selected as features. For each instance, the feature value is the frequency of each pattern embedded with the NP pair of the instance. In the second strategy, one semantic feature is calculated for each instance, which reflects the confidence that a NP pair is semantically related. Empirical evaluation results show that the algorithm helps the performance for proper names, but does not improve performance for other types of NP pairs.

2.3 Other Features

Ng and Cardie (2002c) use anaphoricity determination as a filter for later clustering. The intuition is that some NPs (e.g., the lawyer) must follow an antecedent and cannot lead a coreference chain. For each coreference chain, the NP at the head is non-anaphoric, and the rest are all anaphoric. They adopt the C4.5 decision tree as a binary classifier to classify NPs into anaphoric and non-anaphoric. For resolution, whenever they encounter a non-anaphoric NP, they create a new coreference chain. Experiments show that anaphoricity determination improves precision and F-measure, but hurts recall. Ng (2004) proposes two methods to make better use of anaphoricity information. First, they optimize the parameter of anaphoricity determination algorithm to improve the global resolution system, rather than the local classification accuracy. Second, anaphoricity is used as a feature, rather than a hard constraint as in their previous work.

Yang et al. (2004a) propose a coreferentiality feature. For each candidate antecedent of the pronoun, their approach back tracks through the NPs in the coreference chain of the candidate, and the information of the antecedent of the candidate is added as features (backward features) to the instance. One problem is that the antecedent of the candidate is not available in practice. The paper first did experiments by using the antecedent labels in the dataset for instance creation during both training and testing. Then it provided the resolution in a real situation where the antecedents of candidates are not available. Two resolution modules are used in the algorithm. One is for pronoun resolution and the other is for non-pronoun resolution. For both training and testing instances, the antecedents of the candidates are from the results of the two resolution modules, which are first trained on the training data using the instances without the backward features.

3 Distance Measure

Most coreference resolution systems use classifiers to measure the distance (including NP-to-NP, NP-to-cluster and cluster-to-cluster) in the feature space. This section introduces several classifiers widely used for coreference resolution: decision trees, log-linear models, kernel based methods, twin-candidate and multi-candidate methods.

3.1 Decision Trees

Decision trees are the dominating method in this field. C4.5 and its improvement, C5, are the most popular. C4.5 is based on the ID3 algorithm. It carries out iterative greedy feature selection according to the information gain that would result from the addition of a feature to the existing tree. Unlike ID3, C4.5 can also handle continuous features by thresholding. After the construction, it prunes braches that do not improve the performance. Many systems choose decision tree because they can pick out useful features and build a compact tree. In principle, any classifier (such as SVMs) can be used to calculate the distance between NPs, given a set of features.

3.2 Log-linear Models

The weakness of the decision tree method is that it considers one feature at a time, and may miss useful combinations of features. Since the features for coreference resolution are highly overlapping, log-linear models become a natural choice (McCallum and Wellner, 2004) (Culotta et al., 2007). It models the conditional probability of a label given the feature values. People can study the usefulness of each feature by examining its weight. High weights indicate important features.

$$P(y \mid X) = \frac{1}{Z(X)} \exp(\sum_{i=1}^{n} \lambda_i f_i(y, X))$$

where X is the observed data point, y is the label, f_i is a feature, λ_i is the weight, and $Z(X) = \sum_{y} \exp(\sum_{i=1}^{n} \lambda_i f_i(y, X))$ is the partition function, which guarantees that $P(y \mid X)$ sum to one for all possible y.

3.3 Kernel Based Distance

Yang et al. (2006) propose a set of parse-tree based features. They calculate the difference between two parse trees with a tree comparison algorithm, and then use the difference value as the kernel value for SVM. The similarity of two trees is defined as the normalized count of identical sub-trees (Collins and Duffy 2002). SVM is used as the classifier for pronoun resolution. The paper also combines the syntactic features with other normal features based on the kernel methods. Three kinds of trees are extracted from the parse tree. The first is called Min-Expansion, which only includes the nodes occurring in the shortest path connecting the pronoun and the candidate. The second one is called

Simple-Expansion, which also includes the first-level children of the nodes in Min-Expansion. The third one is Full-Expansion, which also includes the nodes that cover the words between the candidate and the pronoun of the nodes in Simple-Expansion. The tree kernel is calculated based on the common sub-trees as the measure of similarity between trees. The tree features are combined with the normal features with the following equation:

$$Kc(x_1, x_2) = \frac{K_n(x_1, x_2)}{\sqrt{K_n(x_1, x_1)K_n(x_2, x_2)}} \frac{K_t(x_1, x_2)}{\sqrt{K_t(x_1, x_1)K_t(x_2, x_2)}}$$

where K_t is the tree kernel, and K_n is the kernel applied to the normal features. The resulting kernel K_c is used as the kernel for SVM. For training, they use the same instance construction method as in (Soon et al., 2001). For testing, the NPs occurring within the same sentence as the current anaphor and the two preceding sentences are taken as the initial antecedent candidates.

3.4 Twin-candidate Method

Yang et al. (2003) points out one problem with single-candidate models, in which the classifier determines the confidence value of whether an antecedent candidate is coreferential to an anaphor. The problem is that it only takes into account the relationships between an anaphor and one individual candidate at a time, and therefore the confidence values may not be accurate and reliable to represent the true competition between the candidates. To overcome this problem, they proposed a twin-candidate model to directly learn the competition between the antecedent candidates. In this model, a training (testing) instance is formed by an anaphor and two candidate antecedents, and the output is which candidate is more likely to be the antecedent of the anaphor. All pairs of the candidates are considered. The candidate that wins the most comparisons is selected as the antecedent. The paper also claimed that by adding a so called 'standard candidate' which is just a blank candidate, the single candidate model is a special case of the twin candidate model.

A different instance creation procedure is adopted by this method. An instance in the training set is formed by an anaphor, a single positive candidate and a single negative candidate. The pair of candidates is ordered such that the first one is closer to the anaphor than the second one. The label for the instance is positive when the first candidate is positive, and the label is negative when the second candidate is positive.

The twin-candidate method generates more instances than a single-candidate method does. The paper applies a filter to the candidates during both training and testing to reduce the computational cost and data noise. During both training and testing, the candidate antecedents of a pronoun come from the current or preceding two sentences, and agree with the pronoun in number, gender and person. The candidates of a non-pronoun are filtered based on the confidence value of a single-candidate model. Those whose confidence values are less than 0.5 are filtered.

3.5 Multi-candidate Ranking

Denis and Baldridge (2007) generalize the twin-candidate idea to multi-candidate ranking. Twin-candidate methods aggregate pairwise comparison to pick the most likely antecedent, multi-candidate methods rank all candidate antecedents using a log-linear model. During training, each instance is constructed using the closest antecedent to the pronoun, and a set of non-antecedents. The training algorithm optimizes feature weights to assign higher probability to the real antecedent. During testing, all possible antecedents are fed to the ranker, and the most probable antecedent is selected. Single-

candidate and twin-candidate methods are restricted to local comparison, while ranking method makes global comparison.

4 Clustering

This section presents several clustering algorithms, from local to global. Local clustering only considers NP-to-NP distance, and attaches an NP to its most likely antecedent. Incremental clustering considers NP-to-cluster distance. It can take advantage of features defined over clusters, rather than over a pair of NPs. Agglomerative clustering keeps merging small clusters into big ones, until some termination criterion is met. Therefore, it needs to measure cluster-to-cluster distance. Global clustering goes to another extreme in that it separates NPs into clusters in one step. It can be viewed as maximizing inter-cluster distance, and minimizing intra-cluster distance. Local methods have small search space and are easy to implement, while Global methods need to deal with exponentially large search spaces, which requires efficient approximation methods for it to be practical. The clustering problem can also be treated as a sequence search problem, which can be solved by beam search algorithms. Meta-clustering evaluates the output of several clustering algorithm, and pick the best partition.

4.1 Local Clustering

Coreference resolution is different from general clustering in that the NPs are naturally ordered in the text. An NP can either start a new coreference chain, or refer to a preceding NP.

First-Match: Soon et al. (2001) describe a *first-match* clustering algorithm. It processes NPs from left to right. Whenever it encounters an NP, it pairs the NP with each preceding NP from right to left, and throws the pair into the classifier. If the classifier decides that they are coreferent, the current NP is connected to its antecedent. Otherwise, a new coreference chain is created with the current NP as its head. This method is very greedy but surprisingly effective. This is because the position of NPs plays a vital role in coreference.

Best-Match: Ng and Cardie (2002b) presents a less greedy method called *best-match*. Similar to *first-match*, it scans NPs from left to right. It pairs the current NP with all preceding NPs (until reaching the beginning of the document), and picks the pair with the highest output from the classifier. It connects the current NP with its most likely antecedent.

4.2 Incremental Clustering

Yang et al.(2004b) point out that it is often difficult to judge whether two NPs are talking about the same entity simply from the properties of the pair alone, since the individual NP usually lacks adequate descriptive information about its refereed entity. Therefore, the paper proposes to do resolution by determining the links of NPs to the existing coreferential clusters rather than the reference relationships between NP pairs. The idea is based on the intuition that a coreferential cluster offers more information to describe an entity than a single NP does.

This clustering method calls for a special instance construction method: an instance is formed by the NP under consideration, one existing cluster, and one of the NPs in the cluster (the reference NP). During training time, for each anaphoric NP, the preceding coreferential clusters are ordered by their distance to the NP. For each cluster, one instance is created by taking the last NP in the cluster as the reference NP. This process continues until the cluster to which the anaphoric NP belongs is found. All intervening clusters are labeled as negative. During testing time, an instance is created by joining the

current NP with each of the existing clusters and each of the NPs in the cluster. Multiple instances are created for each cluster. The confidence value of the cluster is the maximal confidence value of its instances.

4.3 Agglomerative Clustering

Culotta et al. (2007) propose a *first-order probabilistic model*. They measure the probability of a set of NPs being coreferent using a log-linear model. The parameters are learned by gradient ascent to maximize the log-likelihood over training data. Note that in their model, each instance is a set rather than a pair of NPs. During training time, positive examples are generated by sampling a subset from a true cluster. Negative examples are generated by sampling two subsets from two true clusters and merging them together. During testing time, they use standard agglomerative clustering. In each step, they merge two clusters into a single cluster that yields the highest gain in likelihood. They continue merging NP clusters until the likelihood starts to drop.

4.4 Global Clustering

McCallum and Wellner (2004) model the joint probability of a partition using a log-linear model.

$$P(y \mid X) = \frac{1}{Z(X)} \exp\left(\sum_{i,j,l} \lambda_l f_l(x_i, x_j, y_{ij}) + \sum_{i,j,k} \lambda_c f_c(y_{ij}, y_{jk}, y_{ik})\right)$$

In the above equation, $f_c(y_{ij}, y_{jk}, y_{ik})$ ensures that the clustering result is consistent (ensuring transitivity of coreference relationship): if A and B are coreferent and B and C are coreferent, then A and C must be coreferent.

They show that finding the most probable resolution to this model is equivalent to finding the optimal partition of a graph. In this graph, NPs are nodes, edges are weighted as the log potential of their two ends $\sum_l \lambda_l f_l(x_i, x_j, y_{ij})$, where $f_l(x_i, x_j, 1) = -f_l(x_i, x_j, 0)$. However, graph partition with negative weights is NP-hard. Given n NPs, the number of possible partitions is the nth Bell number, which increases exponentially with respect to n. They use the minimizing-disagreements Correlational Clustering Algorithm to get approximate solutions.

Daume and Marcu (2005) combine entity detection and coreference resolution in one system, and solve the two problems simultaneously. For almost all previous work on coreference resolution, the result mentions of an entity or mention detection system are used as the input to the coreference resolution system. Luo et al. (2004) have discussed their experiment errors resulted from the errors on mention detection system. It is intuitive that we can benefit a lot if the mention detection module can take advantage of information from the coreference module. This is also true for the tasks of linking classification and clustering for coreference resolution, which are usually done separately by previous works. The integrated system will accept a large collection of features that one cannot consider in a pipelined approach. To make this modeling feasible, the authors used a *Learning as Search Optimization* (LaSO) framework.

The LaSO framework accepts a set of input structures X, and output structure Y. In the mention detection and coreference resolution task, X would be the input documents, and Y would be the documents marked with mentions and their coreference sets. The key idea for learning in LaSO is to

update the weight vectors whenever we reach a point at which it becomes impossible to reach the correct solution during searching. This paper used a large margin updating scheme.

The search procedure first chooses the number of words it is going to consume (for instance, to form a mention "Bill Clinton", it would need to consume two words). Then it decides on an entity type and a mention type. Finally assuming it chooses to form an entity, it decides on which of the previous coreference chains this mention belongs to. All these decisions are made simultaneously, and the given hypothesis is scored.

Nicolae and Nicolae (2006) try to find the optimal cut for an undirected weighted graph representation for the coreference search space. Each node of the graph represents a mention, and the edge weights are the confidence values derived from the coreference classification phase. The problem of clustering mentions into entities is considered as a graph-cut problem. Each sub-graph represents an entity. The authors claim that the graph approach is a more accurate representation than a tree. The tree approach only deals with anaphora resolution, trying to connect mentions with preceding candidates. The graph approach can also handle cataphora resolution. Moreover, the approach with Bell trees uses some heuristics and tree pruning during searching, thus may not result in a real global optimization. Conversely, the graph approach can take advantage of efficiency and simplicity of graph algorithms.

In their algorithm, a graph is created for each entity type (person, organization, etc). A score for a cut in a graph is calculated base on the weights associated with each edge. The best cut is the cut with the largest score. A SVM-based classifier is used to decide whether to perform a cut or not. The input to the classifier is the features extracted from the best cut. An example feature would be the largest weight of the edges cross the cut. When the classifier decide to make a cut, the procedure of cutting the graph will be continued, otherwise, the procedure would be stopped, and output the sub-graphs as the resulting entities.

Two classifiers are used in this paper. One is to evaluate the coreferential relationship between pairs of mentions. The other is to decide when to stop the graph cutting. The training instances for the latter are created as follows. Each training instance is a certain cut (S,T), where S and T are set of mentions. The label for the training instance is to perform the cut (negative) or not (positive). A negative example is constructed by combining two coreference chains together. A positive example is constructed by taking a subset of all mentions in the same coreference chain.

4.5 Meta-clustering

Ng (2005a) tries a meta-clustering method. He compares all 54 combinations of three learning algorithms (C4.5 Decision tree, RIPPER and Maximum Entropy), three instance creation methods, two feature sets and three clustering algorithms. He constructs partition-based features using simple functions (e.g., min, max, mean) of NP-based features. They use different coreference resolution systems to produce a set of partitions, and pick the best partition using an SVM-based ranker.

4.6 Sequence Search

Luo et al. (2004) and Florian et al. (2004) cast coreference resolution to a search problem, where the search space is represented with the Bell tree. Finding the coreference chain is equivalent to find the best path from the Bell tree. They aim to make the decision by finding the global optimal entity cluster or coreference chain among the search space. The process of forming entities from mentions can be represented in a tree structure. The Bell tree is formed by traversing mentions in a document from left to right. The root node is a partial entity containing the first mention of a document. The second mention is added in the next step by either linking to the existing entity or starting a new entity. The

two options form the two children of the root node, which is the second layer of the tree nodes. Subsequent mentions are added to the tree in the same manner. Each node includes several entity clusters formed by the mentions added till this step. Each leaf node corresponds to a possible coreference outcome, and the coreference resolution is therefore equivalent to find the best leaf node.

A maximum entropy model is used to calculate the probability of linking one mention to a previously formed entity. The probability of forming a new entity is approximated by one minus the maximum of linking the mention to any existing entities. The score of a path is the value of multiplying all the linking probabilities of that path. In order to reduce the search space, pruning is performed at each time adding a new mention. Local pruning prunes the children nodes whose linking probabilities are too smaller than a threshold. Global pruning prunes the nodes whose scores are small enough.

5 Learning

This section introduces three kinds of learning styles: supervised, semi-supervised and unsupervised. Supervised learning trains the model using labeled data. Semi-supervised learning takes advantage of unlabeled data. Unsupervised learning relies on unlabeled data only.

5.1 Supervised Learning

Most systems use supervised learning to estimate parameters for classifiers, clustering algorithms or rankers. The objective function for optimization can be classification accuracy, log-likelihood over training data, and similarity with the perfect ranking.

Finley and Joachims (2005) propose a framework using SVMs to do supervised clustering, and apply the algorithm to coreference resolution. It is an extension to the more general structural SVM algorithm. Given the gold standard clusters, the clustering SVM learns the similarity measure that maximizes overall intra-cluster similarity. Although they use correlation clustering as their clustering algorithm, the framework works with any clustering algorithm whose objective function can be written as a linear function of feature weights.

5.2 Semi-supervised Learning

Muller et al. (2002) apply co-training (Blum and Mitchell, 1998) to coreference resolution. They build two decision trees on top of two non-overlapping sets of features (two views). The classifiers are first trained on two subsets of labeled data, then the training set is iteratively augmented with unlabeled data. In each iteration, one classifier processes all of the unlabeled instances, and picks the instances with the highest confidence value (either positive or negative), and feeds them to the other classifier as new training data. The success of co-training relies on two basic assumptions: redundancy and conditional independence. Redundancy means any one of the two views is sufficient to build a reasonably good classifier. Conditional independence means the two views are independent given the class label. While the conditional independence assumption is hard to meet, they try to satisfy the sufficiency assumption by evenly splitting the features into two views. They first pick the best two features as seeds, and iteratively build up the two views. In each iteration, they pick the best feature given the existing features, and add that feature to one view.

Ng and Cardie (2003a, 2003b) tries three other semi-supervised methods: one-view co-training, self-training and EM. In one-view co-training, they train two kinds of classifiers (naïve Bayes and decision tree) on the same set of features, and feed highly confident class labels from one classifier to another. While EM cannot outperform co-training, they also propose an FS-EM algorithm (EM

algorithm with feature selection), which achieves the best results in their experiments. FS-EM has a two-level bootstrapping structure. The outer-level performs feature selection based on the performance of semi-supervised classifiers, and the inner-level trains classifiers (with all selected features and one unselected feature) using labeled and unlabeled data.

Stoyanov and Cardie (2006) study the problem of partially supervised learning for coreference resolution. They argue that partially supervised learning is different from semi-supervised learning in that the former has partial ground truth in the training data, while the letter has full ground truth for a portion of the training data, and no ground truth for other parts of the data. They are interested in coreference resolution for aggregating opinions from the same source to the same target. They only have incomplete coreferential clusters in their training data. They modify the RIPPER algorithm to cope with their special needs.

5.3 Unsupervised Learning

Haghighi and Klein (2007) present an unsupervised, nonparametric generative Bayesian approach. They use a hierarchical Dirichlet process to capture cross-document entity sharing, and a sequential model of salience (activity) to capture within-document coreference. The unsupervised method achieves performance close to state-of-the-art supervised systems.

6 Evaluation

Unlike classification, evaluation for clustering is always controversial. Meaningful evaluation metrics are application-dependent. Besides the traditional recall, precision and F-measure, this section also introduces four other evaluation methods: the MUC Scorer, B-Cubed algorithm, ACE-value and CEAF.

6.1 Data Sets

Most public data sets are produced by MUC, ACE and LDC. As shown in Table 2, the size of corpora has grown significantly from the 1990s to the present. Sources other than newswire are also taken into consideration. Besides the standard MUC and ACE corpora, researchers have annotated coreference chains in other data sets, such as Medline, AQUAINT and Penn Treebank.

Table 2: Corpora for English coreference resolution

Data sets	**Sources**	**Traing data (word count)**	**Testing data (word count)**
MUC-6 (1995)	newswire	12.4K	13,4K
MUC-7 (1998)	newswire	19K	10K
ACE-2002	Broadcast news	60K	15K
	Newspaper	60K	15K
	Newswire	60K	15K
	EELD data	30K	20K

Table 2: Corpora for English coreference resolution

ACE-2003	Broadcast news	65K	25K
	Newspaper	65K	N/A
	Newswire	15K	25K
	EELD data	20K	N/A
ACE-2004	Broadcast news	57.5K	25K
	Newswire	57.5K	25K
	Other	35K	N/A
ACE-2005	Broadcast news	65K	10K
	Broadcast conversations	45K	7.5K
	Newswire	60K	10K
	Weblog	45K	7.5K
	Usenet	45K	7.5K
	Conversational telephone speech	45K	7.5K
ACE-2007	Broadcast news	55K	10K
	Broadcast conversations	40K	7.5K
	Newswire	50K	10K
	Weblog	40K	7.5K
	Usenet	40K	7.5K
	Conversational telephone speech	40K	7.5K

6.2 Basic Metrics

To evaluate a coreference resolution system, one simple way is to use the same measure as anaphora resolution.

$$recall = \frac{number\ of\ \ correctly\ resolved\ \ anaphors}{number\ of\ \ all\ \ anaphors}$$

$$precision = \frac{number\ of\ \ correctly\ resolved\ \ anaphors}{number\ of\ \ solved\ \ anaphors}$$

This measure is widely used because most systems view coreference resolution to be the same as anaphora resolution, and solve it by finding an antecedent for each anaphor. However, coreference resolution is intrinsically different from anaphora resolution. The goal of coreference resolution is to build clusters of noun phrases, each of which refers to an entity in the real world. Therefore, the evaluation measure should also be based on the coreference clusters.

6.3 MUC Scorer

Several sophisticated scoring schemes have been proposed. A well known one is designed by Vilain et al. (1995) for the scoring of MUC-6 task. The scheme is still link based, but the comparison is between the equivalence classes generated by the links in the coreference chain.

Some terms used in the scheme are defined as follows: an *entity* refers to a previously mentioned NP, an *equivalence class* of a coreference chain is the cluster of all entities in the chain, *key links* refer to the manually annotated coreference chains, and *response links* are the coreference chains output by a system.

The recall error is calculated by the least number of links that need to be added to the response with respect to the key in order to have the classes aligned, divided by the minimum number of links needed to form the equivalence classes in the key. The precision error is computed by reversing the roles of the key and response. The calculation details are as follows.

First, let S be an equivalence set generated by the key, and let $R_1 ... R_m$ be equivalent classes generated by the response. p(S) is a partition of S relative to the response. Each subset of S in the partition is formed by intersecting S and those response sets Ri that overlap S. Singleton sets are formed for the entities mentioned in the key but not in the response.

For example, if S = {A,B,C,D} and the response is <A-B>, then p(S) is {A B} {C} and {D}. c(S) is the minimum number of links necessary to generate the equivalence class S and therefore, c(S) = (|S|-1). m(S) is the number of missing links in the response relative to the key set S. It is also the number of links necessary to fully reunite any components in p(S). Thus m(S) = (|p(S)|-1)

Looking in isolation at a single equivalence class in the key, the recall for that class is

$$\frac{c(S) - m(S)}{c(S)} = \frac{(|S| - 1) - (|p(S)| - 1)}{|S| - 1}$$

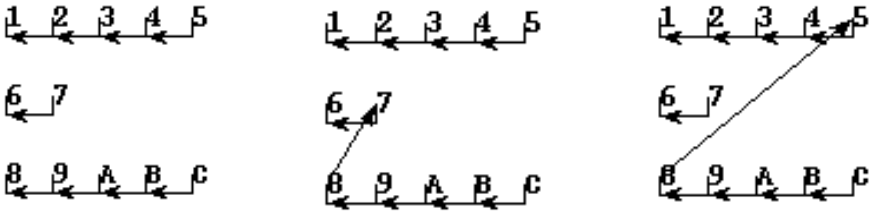
The recall for the entire test set is defined as the sum over all the equivalence classes of the key.

$$recall = \frac{\sum (|S_i| - |p(S_i)|)}{\sum (|S_i| - 1)}$$

Precision is calculated by reversing the role of response and key.

6.4 B-Cubed Algorithm

There are two known shortcomings associated with Vilain's scoring scheme. Firstly, it ignores single-entity coreference clusters, since no link can be found in these entities. Secondly, all type of errors are considered to be equal. Figure 1 shows an example to illustrate this shortcoming:



(a) Key (b) Response 1 (c) Response 2

Figure 1. Shortcomings of the MUC scorer (Bagga and Baldwin, 1998)

For both response 1 and response 2, recall is 100% and precision is 90%. The scoring algorithm does not differentiate between the errors in the first and second responses.

Bagga and Baldwin (1998) proposed another scoring algorithm (B-Cubed) to overcome the two shortcomings of the MUC scorer. The B-Cubed algorithm is completely based on clusters. The precision and recall for each entity are calculated and then combined to produce the final precision and recall for the entire output.

For an entity i, the precision and recall are defined as follows.

$$recall_i = \frac{\text{number of correct elements in the output chain containing entity i}}{\text{number of elements in the true chain containing entity i}}$$

$$precision_i = \frac{\text{number of correct elements in the output chain containing entity i}}{\text{number of elements in the output chain containing entity i}}$$

The final precision and recall for all the entities are:

$$recall = \sum_{i=1}^{N} w_i \cdot recall_i$$

$$precision = \sum_{i=1}^{N} w_i \cdot precision_i$$

where N is the number of entities in the document, and w_i is the weight assigned to entity i in the document. The choice of weighting scheme is determined by the task for which the algorithm is going to be used. Two weighting schemes are provided in the paper.

(1) When coreference is used for an information extraction task, where information about every entity is important, the weighting scheme assigns equal weights to every entity i. In the previous example, the weight assigned to each entity in the key and the two responses is 1/12. The recall for each of the two responses is 100%. The precision for the first response (Fig. 1(b)) is 76%, and the precision for the second response (Fig. 1(c)) is 58%.

(2) When coreference is used for information retrieval tasks, where a user is presented with classes of documents, the weighting scheme assigns equal weights to each equivalence class. The weight for each entity within a class is the same. In the previous example, in Fig. 2, the weight for each entity in the first equivalence class is 1/10, and the weight for each entity in the second class is 1/14. The recall for each of the two responses is still 100%. The precision is 79.6% for the first response and 75% for the second response.

6.5 ACE-value

Another evaluation measure called ACE-value is designed for the ACE task (NIST, 2003). In the ACE task, coreference resolution is called entity detection, and the definition of an entity is different from the one in MUC task. An entity in ACE task refers to the coreference cluster, and the noun phrases are called mentions. The ACE-value is characterized in terms of misses and false alarms. A miss occurs when an entity of a target type is mentioned but not output. A false alarm occurs when a spurious entity is output. Each error is associated with a cost factor that depends on the entity type (e.g., "location", "person") and a mention level (e.g., "name", "nominal", and "pronoun"). The ACE-value is computed by summing over the cost of each type with some normalization. The ACE-value measures the cost of a system, and is not very intuitive.

6.6 CEAF

Luo (2005) introduced a scoring scheme, called Constrained Entity-Aligned F-measure (CEAF). CEAF is computed based on the best one-to-one map between true coreference classes and output classes. The details are as follows:

Let true classes be $R = \{r_i : i = 1,2,...,|R|\}$ and the output classes be $S = \{s_i : i = 1,2,...,|S|\}$

Let $m = \min\{|R|,|S|\}$, and R_m *and* S_m be any subset of R and S with m classes.

Let $G(R_m, S_m)$ be the set of one-to-one class maps from R_m to S_m, and let G_m be the set of all possible one-to-one maps between the size-m subsets of R and S.

$$G(R_m, S_m) = \{g : R_m \to S_m)$$

The requirement of one-to-one mapping means that for any $g \in G(R_m, S_m)$, and any $r \in R_m$ and r'$\in R_m$, we have that $r \neq r'$ implies that $g(r) \neq g(r')$, and vise versa.

Let $\phi(r,s)$ be the similarity measure between two classes R and S. Zero value means that R and S have nothing in common. Given the above notations, we have

$$precision = \frac{\max_{g \in G_m} \sum_{r \in R_m} \phi(r, g(r))}{\sum_i \phi(s_i, s_i)}$$

$$recall = \frac{\max_{g \in G_m} \sum_{r \in R_m} \phi(r, g(r))}{\sum_i \phi(r_i, r_i)}$$

$$F - measure = \frac{2\, precision \cdot recall}{precision + recall}$$

Finding the best one-to-one mapping is a maximum bipartite matching problem and can be solved by the Kuhn-Munkres algorithm (Kuhn, 1955; Munkres 1957), which needs polynomial running time.

Variations of this algorithm have different similarity measure $\phi(r, s)$. Similar to the B-Cubed algorithm, CEAF also defines two versions of the scoring scheme, one is entity-based, the other is class-based. The similarity scores for these two versions are defined as follows.

$$\phi_{entity}(r, s) = | r \cap s |$$

$$\phi_{class}(r, s) = \frac{2 | r \cap s |}{| r | + | s |}$$

CEAF claimed to be able to overcome the problem in B-Cubed, which is that the same classes can be used more than once in the scoring. For example, a response that clusters all of the entities in one class will yield a recall of 100%, but since the set of the key classes is not a subset of the response class, this is counter-intuitive.

6.7 Empirical Comparison

Different systems use different corpora, different evaluation metrics and different types of NPs (pronouns, proper nouns and common nouns). Although it is hard to make fair comparison across different systems, we still summarize some evaluation results here. We use the two classic systems (Soon et al., 2001; Ng and Cardie, 2002b) as baselines, and compare system performance on standard data sets. Table 3 compares different sets of features. Adding corpus-based semantic features improves performance. Table 4 compares various distance measures. Single-candidate methods using simple decision trees work surprisingly well. Table 5 compares different clustering algorithms. Global clustering methods have not shown significant improvement over local clustering. NP type detection (e.g., person, organization) is very helpful for the resolution. Table 6 compares three learning styles. The performance of semi-supervised and unsupervised algorithms is comparable to that of supervised ones.

Table 3: Performance of different feature sets

(Measured in F1, unless specified. A: ACE score, B: B-cubed, M: MUC score)

System	MUC-6	MUC-7	ACE'03
Soon et al., 2001 (12 features)	62.6	60.4	
Ng and Cardie, 2002b (53 features)	70.4	63.4	
Zhou and Su, 2004 (multi-agent)	73.9	66.5	
Bergsma and Lin, 2006 (+dependency patterns)		Accuracy: 71.6	
Ponzetto and Strube, 2006 (+semantic)			M:69.5(BNews) 71.7(NWire)
Yang and Su, 2007 (+semantic)			M:62.7(BNews) 65.0(NPaper) 67.1(NWire)
Ng 2007a (+semantic)			M: 64.2

Table 4: Performance of distance measures

(Measured in F1, unless specified. A: ACE score, B: B-cubed, M: MUC score)

System	MUC-6	MUC-7	ACE'04
Soon et al., 2001 (decision tree)	62.6	60.4	
Ng and Cardie, 2002b (decision tree)	70.4	63.4	
Culotta et al., 2007 (log-linear)			B: 79.3
McCallum and Wellner, 2004 (log-linear)	73.4		
Yang et al., 2003 (twin-candidate)	71.3	60.2	
Denis and Baldridge, 2007 (multi-candidate)			Accuracy: 72.9(BNEWS) 76.4(NPAPER) 72.4(NWIRE)

Table 5: Performance of clustering algorithms

(Measured in F1, unless specified. A: ACE score, B: B-cubed, M: MUC score)

System	MUC-6	MUC-7	ACE'03	ACE'04
Soon et al., 2001 (local, first-match)	62.6	60.4		

Table 5: Performance of clustering algorithms

(Measured in F1, unless specified. A: ACE score, B: B-cubed, M: MUC score)

System	MUC-6	MUC-7	ACE'03	ACE'04
Ng and Cardie, 2002b (local, best-match)	70.4	63.4		
Culotta et al., 2007 (agglomerative)				B: 79.3
McCallum and Wellner, 2004 (global)	73.4			
Daume and Marcu, 2005 (global)				A: 79.4
Nicolar and Nicolae, 2006 (global)			63.8	
Nicolar and Nicolae, 2006 (global + NP type detection)			81.2	
Ng 2005a (meta)				64.9(BNews) 69.3(NPaper) 54.7(NWire)
Luo et al., 2004 (search)			55.1	
Luo et al., 2004 (search + NP type detection)			81.3	
Florian et al., 2004 (search)			A:73.4	

Table 6: Performance of learning algorithms

(Measured in F1, unless specified. A: ACE score, B: B-cubed, M: MUC score)

System	MUC-6	MUC-7	ACE'04
Ng and Cardie, 2002b (supervised)	70.4	63.4	
McCallum and Wellner, 2004 (supervised)	73.4		
Ng and Cardie, 2003a (semi-supervised)	65.4	60.5	
Ng and Cardie, 2003b (semi-supervised)	63.9	54.6	
Denis and Baldridge, 2007 (supervised)			67.1(NWire) 69.2(BNews)
Haghighi and Klein, 2007 (unsupervised)	70.3		64.2(NWiew) 62.3(BNews)

7 Conclusions

This paper reviews recent success of applying machine learning methods to coreference resolution. Although we mainly focus on English coreference, a lot of work has been done on coreference resolution for other languages (e.g., Arabic, Chinese, German and Japanese). Most techniques developed for English can also be transferred to other languages.

Coreference resolution depends on many features, including positional, morphological, syntactic and semantic. Some of the features (e.g., string matching) can be reliably acquired. Generating other features (e.g., parsing) are still current research topics. Semantic features provided by WordNet are useful, but have limited coverage. Large corpora such as Wikipedia and the Web are good sources for extracting semantic and syntactic knowledge.

Many coreference resolution systems use classifiers to measuring the coreference distance between NPs. Commonly used classifiers include decision trees, log-linear models and SVMs. Single-candidate classification looks at one possible antecedent at a time. Twin-candidate classification compares each pair of possible antecedents and then aggregates the results. Multi-candidate ranking compares all possible antecedents at the same time.

Based on the distance measure, clustering algorithms partition NPs into coreferent sets. Local clustering connects individual NPs. Incremental clustering connects an NP to the closest cluster. Agglomerative clustering runs bottom-up and connects similar clusters to form larger clusters. Global clustering searches for the best partitioning of all NPs. Sequence ranking methods also belong to global clustering. They search for global optimal solution in the sequence space instead of the clustering space. Meta-clustering ranks the output of different resolution systems and picks the best one.

Over the last decade, MUC and ACE have produced several high quality data sets for evaluating coreference resolution systems. Sophisticated metrics are developed to provide meaningful evaluation results. Empirical results show that current coreference resolution systems are far from being perfect. There are plenty room for improvement, but making improvement is hard.

Different types of NPs have different levels of difficulty for coreference resolution. Generally speaking, proper nouns are the easiest (with F1 measure around 90%), pronouns are harder, and common nouns are the hardest. Proper nouns resolution mostly relies on string matching and other morphological features. Pronoun resolution needs more syntactic features, while common noun resolution requires both syntactic and semantic features (world knowledge).

There are several research questions to be answered by future research.
(1) How to extract more syntactic and semantic knowledge from large corpora? This is a common question for many language related research areas.
(2) How to define better global features? Current “global features” are mostly simple compositions (e.g., max, min, mean) of local features. Is there better ways to measure NP-to-cluster and cluster-to-cluster distances?
(3) How to do better global clustering? Global clustering methods usually take greedy strategies and approximation algorithms to search in large hypothesis spaces. How to improve their effectiveness and efficiency?
(4) How to make good use of unlabeled data? How to leverage semi-supervised and unsupervised learning algorithms?
(5) Does it help to integrate coreference resolution with other NLP systems? Daume and Marcu (2005) combine coreference resolution with named entity recognition, and achieve good performance in both tasks. Is it possible to connect coreference resolution with other NLP modules (e.g., semantic parsing)?

Acknowledgement

We thank Carol Sisson, Einat Minkov and Noah Smith for many insightful comments.

References

(Bagga and Baldwin, 1998) Amit Bagga and Breck Baldwin, Algorithms for Scoring Coreference Chains. MUC-7 1998.

(Bean and Riloff, 2004) David Bean and Ellen Riloff. Unsupervised Learning of Contextual Role Knowledge for Coreference. HLT/NAACL 2004.

(Bergsma and Lin, 2006) Shane Bergsma and Dekang Lin. Bootstrapping Path-Based Pronoun Resolution. COLING/ACL 2006

(Bhattacharya and Getoor, 2006) Indrajit Bhattacharya and Lise Getoor, A Latent Dirichlet Model for Unsupervised Entity Resolution. SIAM-SDM 2006

(Blum and Mitchell, 1998) Avrim Blum and Tom Mitchell, Combining Labeled and Unlabeled Data with Co-training, Proceedings of the Workshop on Computational Learning Theory, 1998.

(Bos, 2003) Johan Bos. Implementing the Binding and Accommodation Theory for Anaphora Resolution and Presupposition Projection. CL 2003

(Byron, 2002) Donna K. Byron. Resolving pronominal reference to abstract entities. ACL 2002

(Culotta et al., 2007) Aron Culotta, Michael Wick, Robert Hall, Andrew McCallum. First-Order Probabilistic Models for Coreference Resolution. HLT/NAACL, 2007.

(Daume and Marcu, 2005) Hal Daume, III and Daniel Marcu. A large-scale exploration of effective global features for a joint entity detection and tracking model. HLT/EMNLP 2005

(Denis and Baldridge, 2007) Pascal Denis and Jason Baldridge. A Ranking Approach to Pronoun Resolution. IJCAI 2007.

(Elango, 2005) Pradheep Elango. Coreference Resolution: A Survey. Technical Report, University of Wisconsin Madison, 2005. PDF

(Finley and Joachims, 2005) Thomas Finley and Thorsten Joachims. Supervised Clustering with Support Vector Machines. ICML 2005.

(Florian et al., 2004) Radu Florian, Hany Hassan, Abraham Ittycheriah, Hongyan Jing, Nanda Kambhatla, Xiaoqiang Luo, Nicolas Nicolov and Salim Roukos: A Statistical Model for Multilingual Entity Detection and Tracking. HLT/NAACL 2004.

(Haghighi and Klein, 2007) Aria Haghighi and Dan Klein. Unsupervised Coreference Resolution in a Nonparametric Bayesian Model. ACL 2007.

(Iida et al., 2006) Ryu Iida, Kentaro Inui and Yuji Matsumoto. Exploiting Syntactic Patterns as Clues in Zero-Anaphora Resolution. COLING/ACL 2006

(Kuhn 1955) Harold Kuhn. 1955. The hungarian method for the assignment problem. Naval Research Logistics Quarterly, 2(83).

(Luo et al., 2004) Xiaoqiang Luo; Abe Ittycheriah; Hongyan Jing; Nanda Kambhatla; Salim Roukos. A Mention-Synchronous Coreference Resolution Algorithm Based On the Bell Tree. ACL 2004

(Luo, 2005) Xiaoqiang Luo. On Coreference Resolution Performance Metrics. EMNLP 2005.

(Markert and Nissim, 2005) Katja Markert and Malvina Nissim. Comparing Knowledge Sources for Nominal Anaphora Resolution. CL 2005

(McCallum and Wellner, 2004) Andrew McCallum and Ben Wellner. Conditional Models of Identity Uncertainty with Application to Noun Coreference. NIPS 2004.

(Mitkov et al., 2002) Ruslan Mitkov, Richard Evans, and Constantin Orasan. A new, fully automatic version of Mitkov's knowledge-poor pronoun resolution method. Computational Linguistics and Intelligent Text Processing, pp. 169–187. Springer.

(Modjeska et al., 2003) Natalia Modjeska, Katja Markert and Malvina Nissim. Using the Web in Machine Learning for Other-Anaphora Resolution. EMNLP 2003.

(Muller et al., 2002) Christoph Muller, Stefan Rapp and Michael Strube. Applying Co-Training to Reference Resolution. ACL 2002

(Munkres, 1957) James Munkres, Algorithms for the Assignment and Transportation Problems, Journal of the Society of Industrial and Applied Mathematics, 5(1):32-38, 1957

(Nicolare and Nicolare, 2006) Cristina Nicolae and Gabriel Nicolae. BESTCUT: A Graph Algorithm for Coreference Resolution. EMNLP 2006

(NIST, 2002/2003/2004/2005/2007) ACE Evaluation Plan. 2002 / 2003 / 2004 / 2005 / 2007.

(Ng, 2002) Vincent Ng. Machine Learning for Coreference Resolution: Recent Successes and Future Challenges. PhD thesis proposal, Cornell University, 2002. PS

(Ng and Cardie, 2002a) Vincent Ng and Claire Cardie. Combining Sample Selection and Error-Driven Pruning for Machine Learning of Coreference Rules. EMNLP 2002

(Ng and Cardie, 2002b) Vincent Ng and Claire Cardie. Improving Machine Learning Approaches to Coreference Resolution. ACL 2002.

(Ng and Cardie, 2002c) Vincent Ng and Claire Cardie. Identifying Anaphoric and Non-Anaphoric Noun Phrases to Improve Coreference Resolution. COLING 2002

(Ng and Cardie, 2003a) Vincent Ng and Claire Cardie. Weakly Supervised Natural Language Learning Without Redundant Views. HLT-NAACL 2003

(Ng and Cardie, 2003b) Vincent Ng and Claire Cardie. Bootstrapping Coreference Classifiers with Multiple Machine Learning Algorithms. EMNLP 2003 (extended from 2003a)

(Ng, 2004a) Vincent Ng. Learning Noun Phrase Anaphoricity to Improve Conference Resolution: Issues in Representation and Optimization. ACL 2004

(Ng, 2004b) Vincent Ng. Improving Machine Learning Approaches to Noun Phrase Coreference Resolution. PhD Thesis. Cornell University. 2004

(Ng, 2005a) Vincent Ng. Machine Learning for Coreference Resolution: From Local Classification to Global Ranking. ACL 2005

(Ng, 2007a) Vincent Ng. Shallow Semantics for Coreference Resolution. IJCAI 2007

(Ng, 2007b) Vincent Ng. Semantic Class Induction and Coreference Resolution. ACL 2007

(Chomsky, 1981) Noam Chomsky. Lectures on Government and Binding: The Pisa Lectures. published by Mouton de Gruyter, 1981

(Poesion et al., 2004a) Massimo Poesio, Rahul Mehta, Axel Maroudas and Janet Hitzeman. Learning to Resolve Bridging References. ACL, 2004.

(Poesion et al., 2004b) Massimo Poesio, Rosemary Stevenson, Barbara Di Eugenio and Janet Hitzeman. Centering: A Parametric Theory and its Instantiations. Computational Linguistics, 2004

(Ponzetto and Strube, 2006) Simone Paolo Ponzetto and Michael Strube. Exploiting Semantic Role Labeling, WordNet and Wikipedia for Coreference Resolution. HLT 2006

(Schiehlen, 2004) Michael Schiehlen. Optimizing Algorithms for Pronoun Resolution. COLING 2004

(Soon et al., 2001) Wee Meng Soon, Hwee Tou Ng, and Daniel Chung Yong Lim. A Machine Learning Approach to Coreference Resolution of Noun Phrases. Computational Linguistics, 2001.

(Strube et al., 2002) Michael Strube, Stefan Rapp and Christoph Müller. The Influence of Minimum Edit Distance on Reference Resolution. EMNLP 2002.

(Stoyanov and Cardie, 2006) Veselin Stoyanov and Claire Cardie. Partially Supervised Coreference Resolution for Opinion Summarization through Structured Rule Learning. EMNLP 2006

(Strube and Muller, 2003) Michael Strube and Christoph Müller. A Machine Learning Approach to Pronoun Resolution in Spoken Dialogue. ACL 2003.

(Versley, 2007) Yannick Versley. Antecedent Selection Techniques for High-Recall Coreference Resolution. EMNLP 2007

(Vilain et al. 1995) Marc Vilain, John Burger, John Aberdeen, Dennis Connolly and Lynette Hirschman. A Model-Theoretic Coreference Scoring Scheme. MUC-6

(Yang et al., 2003) Xiaofeng Yang, Guodong Zhou, Jian Su and Chew Lim Tan. Coreference Resolution Using Competitive Learning Approach. ACL 2003.

(Yang et al., 2004a) Xiaofeng Yang, Jian Su, Guodong Zhou and Chew Lim Tan. Improving Pronoun Resolution by Incorporating Coreferential Information of Candidates. ACL 2004

(Yang et al., 2004b) Xiaofeng Yang; Jian Su; GuoDong Zhou and Chew Lim Tan. An NP-Cluster Based Approach to Coreference Resolution. COLING 2004

(Yang et al., 2005) Xiaofeng Yang, Jian Su and Chew Lim Tan. Improving Pronoun Resolution Using Statistics-Based Semantic Compatibility Information. ACL 2005

(Yang et al., 2006) Xiaofeng Yang, Jian Su, and Chew Lim Tan. Kernel-Based Pronoun Resolution with Structured Syntactic Knowledge. COLING/ACL 2006

(Yang and Su, 2007) Xiaofeng Yang and Jian Su. Coreference Resolution Using Semantic Relatedness Information from Automatically Discovered Patterns. ACL 2007

(Zhou and Su, 2004) GuoDong Zhou and Jian Su. A High-Performance Coreference Resolution System using a Constraint-based Multi-Agent Strategy. COLING 2004