Classifier Combination Techniques Applied to Coreference Resolution

Smita Vemulapalli[1], **Xiaoqiang Luo**[2], **John F. Pitrelli**[2] **and Imed Zitouni**[2]

[1]Center for Signal and Image Processing (CSIP)
School of ECE, Georgia Institute of Technology
Atlanta, GA 30332, USA
smita@ece.gatech.edu

[2]IBM T. J. Watson Research Center
1101 Kitchawan Road
Yorktown Heights, NY 10598, USA
{xiaoluo,pitrelli,izitouni}@us.ibm.com

Abstract

This paper examines the applicability of classifier combination approaches such as bagging and boosting for coreference resolution. To the best of our knowledge, this is the first effort that utilizes such techniques for coreference resolution. In this paper, we provide experimental evidence which indicates that the accuracy of the coreference engine can potentially be increased by use of bagging and boosting methods, without any additional features or training data. We implement and evaluate combination techniques at the mention, entity and document level, and also address issues like entity alignment, that are specific to coreference resolution.

1 Introduction

Coreference resolution is the task of partitioning a set of mentions (*i.e.* person, organization and location) into entities. A *mention* is an instance of textual reference to an object, which can be either named (*e.g.* Barack Obama), nominal (*e.g.* the president) or pronominal (*e.g.* he, his, it). An *entity* is an aggregate of all the mentions (of any level) which refer to one conceptual entity. For example, in the following sentence:

John said Mary was his sister.

there are four mentions: John, Mary, his, and sister.

John and his belong to the one entity since they refer to the same person; Mary and sister both refer to another person entity. Furthermore, John and Mary are *named* mentions, sister is a *nominal* mention and his is a *pronominal* mention.

In this paper, we present a potential approach for improving the performance of coreference resolution by using classifier combination techniques such as bagging and boosting. To the best of our knowledge, this is the first effort that utilizes classifier combination for improving coreference resolution.

Combination methods have been applied to many problems in natural-language processing (NLP). Examples include the ROVER system (Fiscus, 1997) for speech recognition, the Multi-Engine Machine Translation (MEMT) system (Jayaraman and Lavie, 2005), and part-of-speech tagging (Brill and Wu, 1998; Halteren *et al.*, 2001). Most of these techniques have shown a considerable improvement over the performance of a single classifier and, therefore, lead us to consider implementing such a multiple-classifier system for coreference resolution as well.

Using classifier combination techniques one can potentially achieve a classification accuracy that is superior to that of the single best classifier. This is based on the assumption that the errors made by each of the classifiers are not identical, and therefore if we intelligently combine multiple classifier outputs, we may be able to correct some of these errors.

The main contributions of this paper are:

- *Demonstrating the potential for improvement in the baseline* – By implementing a system that behaves like an oracle, we have shown that the output of the combination of multiple classifiers has the potential to be significantly higher in accuracy than any of the individual classifiers.
- *Adapting traditional bagging techniques* – Multiple classifiers, generated using bagging techniques, were combined using an entity-level sum

Proceedings of the NAACL HLT Student Research Workshop and Doctoral Consortium, pages 1–6,
Boulder, Colorado, June 2009.

rule and mention-level majority voting.

- *Implementing a document-level boosting algorithm* – A boosting algorithm was implemented in which a coreference resolution classifier was iteratively trained using a re-weighted training set, where the reweighting was done at the document level.
- *Addressing the problem of entity alignment* – In order to apply combination techniques to multiple classifiers, we need to address entity-alignment issues, explained later in this paper.

The baseline coreference system we use is similar to the one described by Luo *et al.* (Luo *et al.*, 2004). In such a system, mentions are processed sequentially, and at each step, a mention is either linked to one of existing entities, or used to create a new entity. At the end of this process, each possible partition of the mentions corresponds to a unique sequence of link or creation actions, each of which is scored by a statistical model. The one with the highest score is output as the final coreference result.

2 Classifier Combination Techniques

2.1 Bagging

One way to obtain multiple classifiers is via bagging or bootstrap aggregating (Breiman, 1996). These classifiers, obtained using randomly-sampled training sets, may be combined to improve classification.

We generated several classifiers by two techniques. In the first technique, we randomly sample the set of documents (training set) to generate a few classifiers. In the second technique, we need to reduce the feature set and this is not done in a random fashion. Instead, we use our understanding of the individual features and also their relation to other features to decide which features may be dropped.

2.2 Oracle

In this paper, we refer to an *oracle* system which uses knowledge of the truth. Here, truth, called the *gold standard* henceforth, refers to mention detection and coreference resolution done by a human for each document. It is possible that the gold standard may have errors and is not perfect truth, but, as in most NLP systems, it is considered the reference for evaluating computer-based coreference resolution.

To understand the oracle, consider an example in which the outputs of two classifiers for the same input document are C_1 and C_2, as shown in Figure 1.

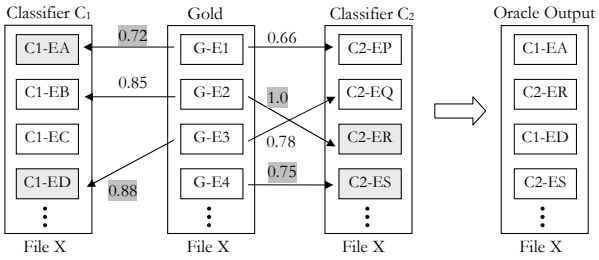

Figure 1: Working of the oracle

The number of entities in C_1 and C_2 may not be the same and even in cases where they are, the number of mentions in corresponding entities may not be the same. In fact, even finding the corresponding entity in the other classifier output or in the gold standard output G is not a trivial problem and requires us to be able to align any two classifier outputs.

The alignment between any two coreference labelings, say C_1 and G, for a document is the best one-to-one map (Luo, 2005) between the entities of C_1 and G. To align the entities of C_1 with those of G, under the assumption that an entity in C_1 may be aligned with at most only one entity in G and vice versa, we need to generate a bipartite graph between the entities of C_1 and G. Now the alignment task is a maximum bipartite matching problem. This is solved by using the Kuhn-Munkres algorithm (Kuhn, 1955; Munkres, 1957). The weights of the edges of the graph are entity-level alignment measures. The metric we use is a relative measure of the similarity between the two entities. To compute the similarity metric ϕ (Luo, 2005) for the entity pair (R, S), we use the formula shown in Equation 1, where $(\cap)$ represents the commonality with attribute-weighted partial scores. Attributes are things such as (ACE) entity type, subtype, entity class, etc.

$$\phi(R, S) = \frac{2\,|R \cap S|}{|R| + |S|} \tag{1}$$

The oracle output is a combination of the entities in C_1 and C_2 with the highest entity-pair alignment measures with the entities in G.[1] We can see in Figure 1 that the entity G-E1 is aligned with entities C1-EA and C2-EP. We pick the entity with the highest entity-pair alignment measure (highlighted in gray) which, in this case, is C1-EA. This is repeated for

[1] A mention may be repeated across multiple output entities, which is not an unwarranted advantage as the scorer insists on one-to-one entity alignment. So if there are two entities containing mention A, at most one mention A is credited and the other will hurt the score.

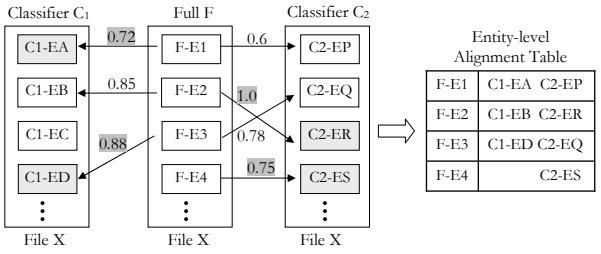

Figure 2: Entity alignment between classifier outputs

every entity in G. The oracle output can be seen in the right-hand side of Figure 1. This technique can be scaled up to work for any number of classifiers.

2.3 Preliminary Combination Approaches

Imitating the oracle. Making use of the existing framework of the oracle, we implement a combination technique that imitates the oracle except that in this case, we do not have the gold standard. If we have N classifiers C_i, $i = 1$ to N, then we replace the gold standard by each of the N classifiers in succession, to get N outputs $Comb_i$, $i = 1$ to N.

The task of generating multiple classifier combination outputs that have a higher accuracy than the original classifiers is often considered to be easier than the task of determining the best of these outputs. We used the formulas in Equations 2, 3 and 4 to assign a score S_i to each of the N combination outputs $Comb_i$, and then we pick the one with the highest score. The function Sc (which corresponds to the function ϕ in Equation 1) gives the similarity between the entities in the pair (R, S).

$$S_i = \frac{1}{N-1} \sum_{\substack{j=1 \; to \; N \\ j \neq i}} Sc(Comb_i, C_j) \quad (2)$$

$$S_i = Sc(Comb_i, C_i) \quad (3)$$

$$S_i = \frac{1}{N-1} \sum_{\substack{j=1 \; to \; N \\ j \neq i}} Sc(Comb_i, Comb_j) \quad (4)$$

Entity-level sum-rule. We implemented a basic sum-rule at the entity level, where we generate only one combination classifier output by aligning the entities in the N classifiers and picking only one entity at each level of alignment. In the oracle, the reference for entity-alignment was the gold standard. Here, we use the baseline/full system (generated using the entire training and feature set) to do this. The entity-level alignment is represented as a table in Figure 2.

Let A_i, $i = 1$ to M be the aligned entities in one row of the table in Figure 2. Here, $M \leq N$ if

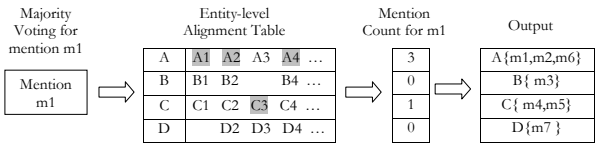

Figure 3: Mention-level majority voting

we exclude the baseline from the combination and $M \leq N + 1$ if we include it. To pick one entity out of these M entities, we use the traditional sum rule (Tulyakov *et al.*, 2008), shown in Equation 5, to compute the $S(A_i)$ for each A_i and pick the entity with the highest $S(A_i)$ value.

$$S(A_i) = \sum_{\substack{j=1 \; to \; N \\ j \neq i}} Sc(A_i, A_j) \quad (5)$$

2.4 Mention-level Majority Voting

In the previous techniques, entities are either picked or rejected as a whole but never broken down further. In the mention-level majority voting technique, we work at the mention level, so the entities created after combination may be different from the entities of all the classifiers that are being combined.

In the entity-level alignment table (shown in Figure 3), A, B, C and D refer to the entities in the baseline system and A1, A2, ..., D4 represent the entities of the input classifiers that are aligned with each of the baseline classifier entities. Majority voting is done by counting the number of times a mention is found in a set of aligned entities. So for every row in the table, we have a mention count. The row with the highest mention count is assigned the mention in the output. This is repeated for each mention in the document. In Figure 3, we are voting for the mention m1, which is found to have a voting count of 3 (the majority vote) at the entity-level A and a count of 1 at the entity-level C, so the mention is assigned to the entity A. It is important to note that some classifier entities may not align with any baseline classifier entity as we allow only a one-to-one mapping during alignment. Such entities will not be a part of the alignment table. If this number is large, it may have a considerable effect on the combination.

2.5 Document-level Boosting

Boosting techniques (Schapire, 1999) combine multiple classifiers, built iteratively and trained on re-weighted data, to improve classification accuracy. Since coreference resolution is done for a whole document, we can not split a document fur-

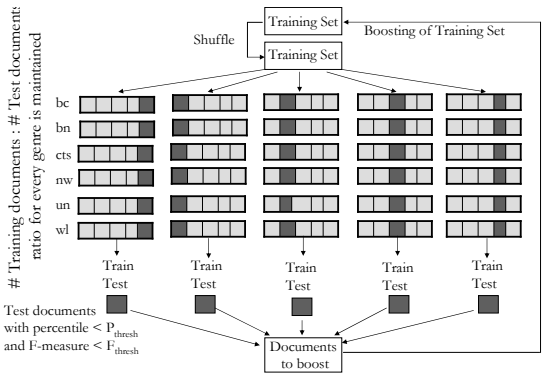

Figure 4: Document-level boosting

ther. So when we re-weight the training set, we are actually re-weighting the documents (hence the name document-level boosting). Figure 4 shows an overview of this technique.

The decision of which documents to boost is made using two thresholds: percentile threshold P_{thresh} and the F-measure threshold F_{thresh}. Documents in the test set that are in the lowest P_{thresh} percentile and that have a document F-measure less than F_{thresh} will be boosted in the training set for the next iteration. We shuffle the training set to create some randomness and then divide it into groups of training and test sets in a round-robin fashion such that a predetermined ratio of the number of training documents to the number of test documents is maintained. In Figure 4, the light gray regions refer to training documents and the dark gray regions refer to test documents. Another important consideration is that it is difficult to achieve good coreference resolution performance on documents of some genres compared to others, even if they are boosted significantly. In an iterative process, it is likely that documents of such genres will get repeatedly boosted. Also our training set has more documents of some genres and fewer of others. So we try to maintain, to some extent, the ratio of documents from different genres in the training set while splitting this training set further into groups of training and test sets.

3 Evaluation

This section describes the general setup used to conduct the experiments and presents an evaluation of the combination techniques that were implemented.

Experimental setup. The coreference resolution system used in our experiments makes use of a Maximum Entropy model which has lexical, syntactical, semantic and discourse features (Luo *et al.*, 2004). Experiments are conducted on ACE 2005 data (NIST, 2005), which consists of 599 documents from rich and diversified sources. We reserve the last 16% documents of each source as the test set, and use the rest of the documents as the training set. The ACE 2005 data split is tabulated in Table 1.

Table 1: Statistics of ACE 2005 data

DataSet	#Docs	#Words	#Mentions	#Entities
Training	499	253771	46646	16102
Test	100	45659	8178	2709
Total	599	299430	54824	18811

Table 2: Accuracy of generated and baseline classifiers

Classifier	Accuracy (%)
$C_1 - C_{15}$ Average	77.52
Highest	79.16
Lowest	75.81
C_0 Baseline	78.53

Bagging A total of 15 classifiers (C_1 to C_{15}) were generated, 12 of which were obtained by sampling the training set and the remaining 3 by sampling the feature set. We also make use of the baseline classifier C_0. The accuracy of C_0 to C_{15} has been summarized in Table 2. The agreement between the classifiers' output was found to be in the range of 93% to 95%. In this paper, the metric used to compute the accuracy of the coreference resolution is the Constrained Entity-Alignment F-Measure (CEAF) (Luo, 2005) with the entity-pair similarity measure in Equation 1.

Oracle. To conduct the oracle experiment, we train 1 to 15 classifiers and align their output to the gold standard. For all entities aligned with a gold entity, we pick the one with the highest score as the output. We measure the performance for varying number of classifiers, and the result is plotted in Figure 5.

First, we observe a steady and significant increase in CEAF for every additional classifier, because additional classifiers can only improve the alignment score. Second, we note that the oracle accuracy is 87.58% for a single input classifier C_1, *i.e.* an absolute gain of 9% compared to C_0. This is because the availability of gold entities makes it possible to remove many false-alarm entities. Finally, the oracle accuracy when all 15 classifiers are used as input is 94.59%, a 16.06% absolute improvement.

This experiment helps us to understand the performance bound of combining multiple classifiers and the contribution of every additional classifier.

Preliminary combination approaches. While the oracle results are encouraging, a natural question is

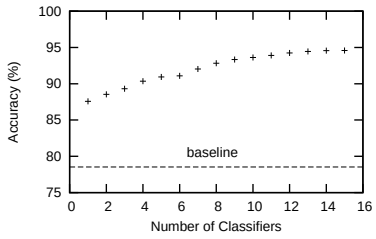

Figure 5: Oracle performance vs. number of classifiers

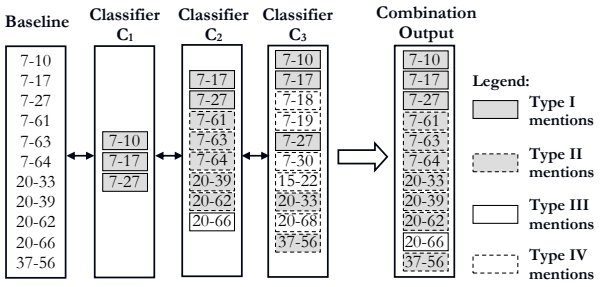

Figure 6: A real example showing the working of mention-level majority voting

how much performance gain can be attained if the gold standard is not available. To answer this question, we replace the gold standard with one of the classifiers C_1 to C_{15}, and align the classifiers. This is done in a round robin fashion as described in Section 2.3. The best performance of this procedure is 77.93%. The sum-rule combination output had an accuracy of 78.65% with a slightly different baseline of 78.81%. These techniques do not yield a statistically significant increase in CEAF but this is not surprising as C_1 to C_{15} are highly correlated.

Mention-level majority voting. This experiment is conducted to evaluate the mention-level majority voting technique. The results are not statistically better than the baseline, but they give us valuable insight into the working of the combination technique. The example in Figure 6 shows a single entity-alignment level for the baseline C_0 and 3 classifiers C_1, C_2, and C_3 and the combination output by mention-level majority voting. The mentions are denoted by the notation *'EntityID - MentionID'*, for example *7-10* is the mention with *EntityID=7* and *MentionID=10*. Here, we use the *EntityID* in the gold file. The mentions with *EntityID=7* are "correct" *i.e.* they belong in this entity, and the others are "wrong" *i.e.* they do not belong in this entity.

The aligned mentions are of four types:

- *Type I mentions* – These mentions have a highest voting count of 2 or more at the same entity-level alignment and hence appear in the output.
- *Type II mentions* – These mentions have a highest voting count of 1. But they are present in more than one input classifier and there is a tie between the mention counts at different entity-level alignments. The rule to break the tie is that mentions are included if they are also seen in the full system C_0. As can been seen, this rule brings in correct mentions such as *7-61, 7-63, 7-64*, but it also admits *20-33,20-39* and *20-62*. In the oracle, the gold standard helps to remove entities with false-alarm mentions, whereas the full system output is noisy and it is not strong enough to reliably remove undesired mentions.
- *Type III mentions* – There is only one mention *20-66* which is of this type. It is selected in the combination output since it is present in C_2 and the baseline C_0, although it has been rejected as a false-alarm in C_1 and C_3.
- *Type IV mentions* – These false-alarm mentions (relative to C_0) are rejected in the output. As can be seen, this correctly rejects mentions such as *15-22* and *20-68*, but it also rejects correct mentions *7-18, 7-19* and *7-30*.

In summary, the current implementation of this technique has a limited ability to distinguish correct mentions from wrong ones due to the noisy nature of C_0 which is used for alignment. We also observe that mentions spread across different alignments often have low-count and they are often tied in count. Therefore, it is important to set a minimum threshold for accepting these low-count majority votes and also investigate better tie-breaking techniques.

Document-level Boosting This experiment is conducted to evaluate the document-level boosting technique. Table 3 shows the results with the ratio of the number of training documents to the number of test documents equal to 80:20, F-measure threshold F_{thresh} = 74% and percentile threshold P_{thresh} = 25%. The accuracy increases by 0.7%, relative to the baseline. Due to computational complexity considerations, we used fixed values for the parameters. Therefore, these values may be suboptimal and may not correspond to the best possible increase in accuracy.

4 Related Work

A large body of literature related to statistical methods for coreference resolution is available (Ng and Cardie, 2003; Yang *et al.*, 2003; Ng, 2008; Poon and

Table 3: Results of document-level boosting

Iteration	Accuracy (%)
1	78.53
2	78.82
3	79.08
4	78.37

Domingos, 2008; McCallum and Wellner, 2003). Poon and Domingos (Poon and Domingos, 2008) use an unsupervised technique based on joint inference across mentions and Markov logic as a representation language for their system on both MUC and ACE data. Ng (Ng, 2008) proposed a generative model for unsupervised coreference resolution that views coreference as an EM clustering process. In this paper, we make use of a coreference engine similar to the one described by Luo *et al.* (Luo *et al.*, 2004), where a Bell tree representation and a Maximum entropy framework are used to provide a naturally incremental framework for coreference resolution. To the best of our knowledge, this is the first effort that utilizes classifier combination techniques to improve coreference resolution. Combination techniques have earlier been applied to various applications including machine translation (Jayaraman and Lavie, 2005), part-of-speech tagging (Brill and Wu, 1998) and base noun phrase identification (Sang *et al.*, 2000). However, the use of these techniques for coreference resolution presents a unique set of challenges, such as the issue of entity alignment between the multiple classifier outputs.

5 Conclusions and Future Work

In this paper, we examined and evaluated the applicability of bagging and boosting techniques to coreference resolution. We also provided empirical evidence that coreference resolution accuracy can potentially be improved by using multiple classifiers. In future, we plan to improve (1) the entity-alignment strategy, (2) the majority voting technique by setting a minimum threshold for the majority-vote and better tie-breaking, and (3) the boosting algorithm to automatically optimize the parameters that have been manually set in this paper. Another possible avenue for future work would be to test these combination techniques with other coreference resolution systems.

Acknowledgments

The authors would like to acknowledge Ganesh N. Ramaswamy for his guidance and support in conducting the research presented in this paper.

References

L. Breiman. 1996. Bagging predictors. In *Machine Learning*.

E. Brill and J. Wu. 1998. Classifier combination for improved lexical disambiguation. In *Proc. of COLING*.

J. Fiscus. 1997. A post-processing system to yield reduced word error rates: Recogniser output voting error reduction (rover). In *Proc. of ASRU*.

H. V. Halteren et al. 2001. Improving accuracy in word class tagging through the combination of machine learning systems. *Computational Linguistics*, 27.

S. Jayaraman and A. Lavie. 2005. Multi-engine machine translation guided by explicit word matching. In *Proc. of ACL*.

H. W. Kuhn. 1955. The hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2.

X. Luo et al. 2004. A mention-synchronous coreference resolution algorithm based on the bell tree. In *Proc. of ACL*.

X. Luo. 2005. On coreference resolution performance metrics. In *Proc. of EMNLP*.

A. McCallum and B. Wellner. 2003. Toward conditional models of identity uncertainty with application to proper noun coreference. In *Proc. of IJCAI/IIWeb*.

J. Munkres. 1957. Algorithms for the assignment and transportation problems. *Journal of the Society of Industrial and Applied Mathematics*, 5(1).

V. Ng and C. Cardie. 2003. Bootstrapping coreference classifiers with multiple machine learning algorithms. In *Proc. of EMNLP*.

V. Ng. 2008. Unsupervised models for coreference resolution. In *Proc. of EMNLP*.

NIST. 2005. ACE'05 evaluation. `www.nist.gov/speech/tests/ace/ace05/index.html`.

H. Poon and P. Domingos. 2008. Joint unsupervised coreference resolution with Markov Logic. In *Proc. of EMNLP*.

E. F. T. K. Sang et al. 2000. Applying system combination to base noun phrase identification. In *Proc. of COLING 2000*.

R.E. Schapire. 1999. A brief introduction to boosting. In *Proc. of IJCAI*.

S. Tulyakov et al. 2008. Review of classifier combination methods. In *Machine Learning in Document Analysis and Recognition*.

X. Yang et al. 2003. Coreference resolution using competition learning approach. In *Proc. of ACL*.