Bootstrapping Coreference Classifiers with Multiple Machine Learning Algorithms

Vincent Ng and **Claire Cardie**
Department of Computer Science
Cornell University
Ithaca, NY 14853-7501
{yung,cardie}@cs.cornell.edu

Abstract

Successful application of multi-view co-training algorithms relies on the ability to factor the available features into views that are compatible and uncorrelated. This can potentially preclude their use on problems such as coreference resolution that lack an obvious feature split. To bootstrap coreference classifiers, we propose and evaluate a single-view weakly supervised algorithm that relies on two different learning algorithms in lieu of the two different views required by co-training. In addition, we investigate a method for ranking unlabeled instances to be fed back into the bootstrapping loop as labeled data, aiming to alleviate the problem of performance deterioration that is commonly observed in the course of bootstrapping.

1 Introduction

Co-training (Blum and Mitchell, 1998) is a weakly supervised paradigm that learns a task from a small set of labeled data and a large pool of unlabeled data using separate, but redundant *views* of the data (i.e. using disjoint feature subsets to represent the data). To ensure provable performance guarantees, the co-training algorithm assumes as input a set of views that satisfies two fairly strict conditions. First, each view must be sufficient for learning the target concept. Second, the views must be conditionally independent of each other given the class. Empirical results on artificial data sets by Muslea et al. (2002) and Nigam and Ghani (2000) confirm that co-training is sensitive to these assumptions. Indeed, although the algorithm has been applied successfully to natural language processing (NLP) tasks that have a natural view factorization (e.g. web page classification (Blum and Mitchell, 1998) and named entity classification (Collins and Singer, 1999)), there has been little success, and a number of reported problems, when applying co-training to NLP data sets for which no natural feature split has been found (e.g. anaphora resolution (Mueller et al., 2002)).

As a result, researchers have begun to investigate co-training procedures that do *not* require explicit view factorization. Goldman and Zhou (2000) and Steedman et al. (2003b) use *two different learning algorithms* in lieu of the multiple views required by standard co-training.[1] The intuition is that the two learning algorithms can potentially substitute for the two views: different learners have different representation and search biases and can complement each other by inducing different hypotheses from the data. Despite their similarities, the principles underlying the Goldman and Zhou and Steedman et al. co-training algorithms are fundamentally different. In particular, Goldman and Zhou rely on hypothesis testing to select new instances to add to the labeled data. On the other hand, Steedman et al. use two learning algorithms that correspond to coarsely different features, thus retaining in spirit the advantages

[1] Steedman et al. (2003b) bootstrap two parsers that use different statistical models via co-training. Hence, the two parsers can effectively be viewed as two different learning algorithms.

provided by conditionally independent feature splits in the Blum and Mitchell algorithm.

The goal of this paper is two-fold. First, we propose a single-view algorithm for bootstrapping coreference classifiers. Like anaphora resolution, noun phrase coreference resolution is a problem for which a natural feature split is not readily available. In related work (Ng and Cardie, 2003), we compare the performance of the Blum and Mitchell co-training algorithm with that of two existing single-view bootstrapping algorithms — self-training with bagging (Banko and Brill, 2001) and EM (Nigam et al., 2000) — on coreference resolution, and show that single-view weakly supervised learners are a viable alternative to co-training for the task. This paper instead focuses on developing a single-view algorithm that combines aspects of each of the Goldman and Zhou and Steedman et al. algorithms.

Second, we investigate a new method that, inspired by Steedman et al. (2003a), ranks unlabeled instances to be added to the labeled data in an attempt to alleviate a problem commonly observed in bootstrapping experiments — performance deterioration due to the degradation in the quality of the labeled data as bootstrapping progresses (Pierce and Cardie, 2001; Riloff and Jones, 1999).

In a set of baseline experiments, we first demonstrate that multi-view co-training fails to boost the performance of the coreference system under various parameter settings. We then show that our single-view weakly supervised algorithm successfully bootstraps the coreference classifiers, boosting the F-measure score by 9-12% on two standard coreference data sets. Finally, we present experimental results that suggest that our method for ranking instances is more resistant to performance deterioration in the bootstrapping process than Blum and Mitchell's "rank-by-confidence" method.

2 Noun Phrase Coreference Resolution

Noun phrase coreference resolution refers to the problem of determining which noun phrases (NPs) refer to each real-world entity mentioned in a document.[2] In this section, we give an overview of the coreference resolution system to which the bootstrapping algorithms will be applied.

The framework underlying the coreference system is a standard combination of classification and clustering (see Ng and Cardie (2002) for details). Coreference resolution is first recast as a classification task, in which a pair of NPs is classified as co-referring or not based on constraints that are learned from an annotated corpus. A separate clustering mechanism then coordinates the possibly contradictory pairwise classifications and constructs a partition on the set of NPs. When the system operates within the weakly supervised setting, a weakly supervised algorithm bootstraps the coreference classifier from the given labeled and unlabeled data rather than from a much larger set of labeled instances. The clustering algorithm, however, is not manipulated by the bootstrapping procedure.

3 Learning Algorithms

We employ naive Bayes and decision list learners in our single-view, multiple-learner framework for bootstrapping coreference classifiers. This section gives an overview of the two learners.

3.1 Naive Bayes

A naive Bayes (NB) classifier is a generative classifier that assigns to a test instance i with feature values $<x_1, \ldots, x_m>$ the maximum a posteriori (MAP) label y^*, which is determined as follows:

$$\begin{aligned} y^* &= \arg\max_y P(y \mid i) \\ &= \arg\max_y P(y)P(i \mid y) \\ &= \arg\max_y P(y) \prod_{i=1}^{m} P(x_i \mid y) \end{aligned}$$

The first equality above follows from the definition of MAP, the second one from Bayes rule, and the last one from the conditional independence assumption of the feature values. We determine the class priors P(y) and the class densities P($x_i \mid y$) directly from the training data using add-one smoothing.

3.2 Decision Lists

Our decision list (DL) algorithm is based on that described in Collins and Singer (1999). For each available feature f_i and each possible value v_j of f_i in the training data, the learner induces an element of the

[2] Concrete examples of the coreference task can be found in MUC-6 (1995) and MUC-7 (1998).

Observations	Justifications
Many feature-value pairs alone can determine the class value.[3] For example, two NPs cannot be coreferent if they differ in gender or semantic class.	Decision lists draw a decision boundary based on a single feature-value pair and can take advantage of this observation directly. On the other hand, naive Bayes classifiers make a decision based on a combination of features and thus cannot take advantage of this observation directly.
The class distributions in coreference data sets are skewed. Specifically, the fact that most NP pairs in a document are not coreferent implies that the negative instances grossly outnumber the positives.	Naive Bayes classifiers are fairly resistant to class skewness, which can only exert its influence on classifier prediction via the class priors. On the other hand, decision lists suffer from skewed class distributions. Elements corresponding to the negative class tend to aggregate towards the beginning of the list, causing the classifier to perform poorly on the minority class.
Many instances contain redundant information as far as classification is concerned. For example, two NPs may differ in both gender and semantic class, but knowing one of these two differences is sufficient for determining the class value.	Both naive Bayes classifiers and decision lists can take advantage of data redundancy. Frequency counts of feature-value pairs in these classifiers are updated independently, and thus a single instance can possibly contribute to the discovery of more than one useful feature-value pair. On the other hand, some classifiers such as decision trees are not able to take advantage of this redundancy because of their intrinsic nature of recursive data partitioning.

Table 1: The justifications (shown in the right column) for using naive Bayes and decision list learner as the underlying learning algorithms for bootstrapping coreference classifiers are based on the corresponding observations on the coreference task and the features used by the coreference system in the left column.

decision list for each class y. The elements in the list are sorted in decreasing order of the *strength* associated with each element, which is defined as the conditional probability P($y \mid f_i = v_j$) and is estimated based on the training data as follows:

$$P(y \mid f_i = v_j) = \frac{N(f_i = v_j,\, y) + \alpha}{N(f_i = v_j) + k\alpha}$$

$N(x)$ is the frequency of event x in the training data, α a smoothing parameter, and k the number of classes. In this paper, k = 2 and we set α to 0.01. A test instance is assigned the class associated with the first element of the list whose predicate is satisfied by the description of the instance.

While generative classifiers estimate class densities, discriminative classifiers like decision lists focus on approximating class boundaries. Table 1 provides the justifications for choosing these two learners as components in our single-view, multi-learner bootstrapping algorithm. Based on observations of the coreference task and the features employed by our coreference system, the justifications suggest that the two learners can potentially compensate for each other's weaknesses.

4 Multi-View Co-Training

In this section, we describe the Blum and Mitchell (B&M) multi-view co-training algorithm and apply it to coreference resolution.

4.1 The Multi-View Co-Training Algorithm

The intuition behind the B&M co-training algorithm is to train two classifiers that can help augment each other's labeled data by exploiting two separate but redundant views of the data. Specifically, each classifier is trained using one view of the labeled data and predicts labels for all instances in the *data pool*, which consists of a randomly chosen subset of the unlabeled data. Each then selects its most confident predictions, and adds the corresponding instances with their predicted labels to the labeled data while maintaining the class distribution in the labeled data.

The number of instances to be added to the labeled data by each classifier at each iteration is limited by a pre-specified *growth size* to ensure that only the instances that have a high probability of being assigned the correct label are incorporated. The data pool is replenished with instances from the unlabeled data and the process is repeated.

During testing, each classifier makes an independent decision for a test instance. In this paper, the decision associated with the higher confidence is taken to be the final prediction for the instance.

4.2 Experimental Setup

One of the goals of the experiments is to enable a fair comparison of the multi-view algorithm with our single-view bootstrapping algorithm. Since the B&M co-training algorithm is sensitive not only to the views employed but also to other input parame-

[3] This justifies the use of a decision list as a potential classifier for bootstrapping. See Yarowsky (1995) for details.

	MUC-6						MUC-7					
	Naive Bayes			Decision List			Naive Bayes			Decision List		
Experiments	R	P	F	R	P	F	R	P	F	R	P	F
Baseline	50.7	52.6	51.6	17.9	72.0	28.7	40.1	40.2	40.1	32.4	78.3	45.8
Multi-view Co-Training	33.3	90.7	48.7	19.5	71.2	30.6	32.9	76.3	46.0	32.4	78.3	45.8
Single-view Bootstrapping	53.6	79.0	63.9	40.1	83.1	54.1	43.5	73.2	54.6	38.3	75.4	50.8
Self-Training	48.3	63.5	54.9	18.7	70.8	29.6	40.1	40.2	40.1	32.9	78.1	46.3

Table 2: Results of multi-view co-training, single-view bootstrapping, and self-training. **R**ecall, **P**recision, and **F**-measure are provided. Except for the baselines, the best results (F-measure) achieved by the algorithms are shown.

ters such as the pool size and the growth size (Pierce and Cardie, 2001), we evaluate the algorithm under different parameter settings, as described below.

Evaluation. We use the MUC-6 (1995) and MUC-7 (1998) coreference data sets for evaluation. The training set is composed of 30 "dry run" texts, from which 491659 and 482125 NP pair instances are generated for the MUC-6 and MUC-7 data sets, respectively. Unlike Ng and Cardie (2003) where we choose one of the dryrun texts (contributing approximately 3500–3700 instances) form the labeled data set, however, here we randomly select 1000 instances. The remaining instances are used as unlabeled data. Testing is performed by applying the bootstrapped coreference classifier and the clustering algorithm described in section 2 on the 20–30 "formal evaluation" texts for each of the MUC-6 and MUC-7 data sets.

Two sets of experiments are conducted, one using naive Bayes as the underlying supervised learning algorithm and the other the decision list learner. All results reported are averages across five runs.

Co-training parameters. The co-training parameters are set as follows.

Views. We used three methods to generate the views from the 25 features used by the coreference system: Mueller et al.'s (2002) greedy method, random splitting of features into views, and splitting of features according to the feature type (i.e. lexico-syntactic vs. non-lexico-syntactic features).[4]

Pool size. We tested values of 500, 1000, 5000.

Growth size. We tested values of 10, 50, 100, 200.

4.3 Results and Discussion

Results are shown in Table 2, where performance is reported in terms of recall, precision, and F-measure using the model-theoretic MUC scoring program (Vilain et al., 1995). The baseline coreference system, which is trained only on the initially labeled data using all of the features, achieves an F-measure of 51.6 (NB) and 28.7 (DL) on the MUC-6 data set and 40.1 (NB) and 45.8 (DL) on MUC-7.

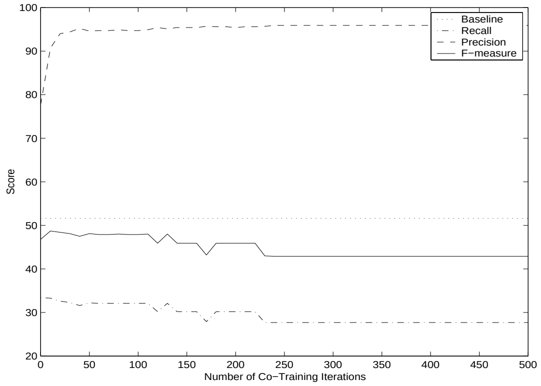

Figure 1: Learning curve for co-training (pool size = 500, growth size = 50, views formed by randomly splitting the features) for MUC-6.

The results shown in row 2 of Table 2 correspond to the best F-measure scores achieved by co-training across all of the parameter combinations described in the previous subsection. In comparison to the baseline, co-training is able to improve system performance in only two of the four classifier/data set combinations: F-measure increases by 2% and 6% for MUC-6/DL and MUC-7/NB, respectively. Nevertheless, co-training produces high-precision classifiers in all four cases (at the expense of recall). In practical applications in which precision is critical, the co-training classifiers may be preferable to the baseline classifiers despite the fact that they achieve similar F-measure scores.

Figure 1 depicts the learning curve for the co-training run that gives rise to the best F-measure for

[4] Space limitation precludes a detailed description of these methods. See Ng and Cardie (2003) for details.

the MUC-6 data set using naive Bayes. The horizontal (dotted) line shows the performance of the baseline system, as described above. As co-training progresses, F-measure rises to 48.7 at iteration ten and gradually drops to and stabilizes at 42.9. We observe similar performance trends for the other classifier/data set combinations. The drop in F-measure is potentially due to the pollution of the labeled data by mislabeled instances (Pierce and Cardie, 2001).

5 Single-View Bootstrapping

In this section, we describe and evaluate our single-view, multi-learner bootstrapping algorithm, which combines ideas from Goldman and Zhou (2000) and Steedman et al. (2003b). We will start by giving an overview of these two co-training algorithms.

5.1 Related Work

The Goldman and Zhou (G&Z) Algorithm. This single-view algorithm begins by training two classifiers on the initially labeled data using two different learning algorithms; it requires that each classifier partition the instance space into a set of equivalence classes (e.g. in a decision tree, each leaf node defines an equivalence class). Each classifier then considers each equivalence class and uses hypothesis testing to determine if adding all unlabeled instances within the equivalence class to the other classifier's labeled data will improve the performance of its counterparts. The process is then repeated until no more instances can be labeled.

The Steedman et al. (Ste) Algorithm. This algorithm is a variation of B&M applied to two diverse statistical parsers. Initially, each parser is trained on the labeled data. Each then parses and scores all sentences in the data pool, and then adds the most confidently parsed sentences *to the training data of the other parser*. The parsers are retrained, and the process is repeated for several iterations.

The algorithm differs from B&M in three main respects. First, the training data of the two parsers diverge after the first co-training iteration. Second, the data pool is *flushed* and refilled entirely with instances from the unlabeled data after each iteration. This reduces the possibility of having unreliably labeled sentences accumulating in the pool. Finally, the two parsers, each of which is assumed to hold a unique "view" of the data, are effectively two different learning algorithms.

5.2 Our Single-View Bootstrapping Algorithm

As mentioned before, our algorithm uses two different learning algorithms to train two classifiers on the *same* set of features (i.e. the full feature set). At each bootstrapping iteration, each classifier labels and scores all instances in the data pool. The highest scored instances labeled by one classifier are added to the training data of the other classifier and vice versa. Since the two classifiers are trained on the same view, it is important to maintain a separate training set for each classifier: this reduces the probability that the two classifiers converge to the same hypothesis at an early stage and hence implicitly increases the ability to bootstrap. Like Ste, the entire data pool is replenished with instances drawn from the unlabeled data after each iteration, and the process is repeated. So our algorithm is effectively Ste applied to coreference resolution — instead of two parsing algorithms that correspond to different features, we use two learning algorithms, each of which relies on the same set of features as in G&Z. The similarities and differences among B&M, G&Z, Ste, and our algorithm are summarized in Table 3.

5.3 Results and Discussion

We tested different pool sizes and growth sizes as specified in section 4.2 to determine the best parameter setting for our algorithm. For both data sets, the best F-measure score is achieved using a pool size of 5000 and a growth size of 50. The results under this parameter setting are given in row 3 of Table 2. In comparison to the baseline, we see dramatic improvement in F-measure for both classifiers and both data sets. In addition, we see simultaneous gains in recall and precision in all cases except MUC-7/DL. Furthermore, single-view bootstrapping beats co-training (in terms of F-measure scores) by a large margin in all four cases. These results provide suggestive evidence that single-view, multi-learner bootstrapping might be a better alternative to its multi-view, single-learner counterparts for coreference resolution.

The bootstrapping run that corresponds to this parameter setting for the MUC-6 data set using naive Bayes is shown in Figure 2. Again, we see a "typi-

	Blum and Mitchell	Goldman and Zhou	Steedman et al.	Ours
Bootstrapping basis	Use different views	Use different learners	Use different parsers	Use different learners
Number of instances added per iteration	Fixed	Variable	Fixed	Fixed
Training sets for the two learners/parsers	Same	Different	Different	Different
Data pool flushed after each iteration	No	N/A (No data pool is used)	Yes	Yes
Example selection method	Highest scored instances	Instances in all equivalance classes that are expected to improve a classifier	Highest scored sentences	Highest scored instances

Table 3: Summary of the major similarities and differences among four bootstrapping schemes: Blum and Mitchell, Goldman and Zhou, Steedman et al., and ours. Only the relevant dimensions are discussed here.

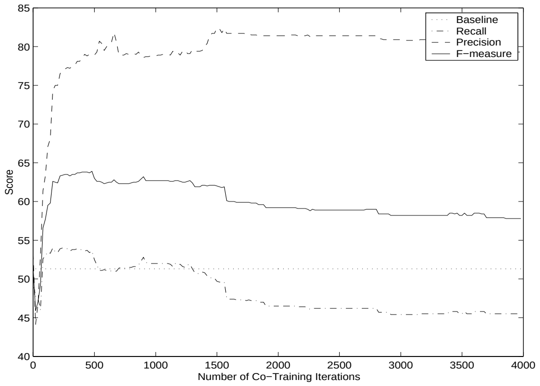

Figure 2: Learning curve for our single-view bootstrapping algorithm (pool size = 5000, growth size = 50) for MUC-6.

cal" bootstrapping curve: an initial rise in F-measure followed by a gradual deterioration. In comparison to Figure 1, the recall level achieved by co-training is much lower than that of single-view bootstrapping. This appears to indicate that each co-training view is insufficient for learning the target concept: the feature split limits any interaction of features that can produce better recall.

Finally, Figure 2 shows that performance increases most rapidly in the first 200 iterations. This provides indirect evidence that the two classifiers have acquired different hypotheses from the initial data and are exchanging information with each other. To ensure that the classifiers are indeed benefiting from each other, we conducted a self-training experiment for each classifier separately: at each self-training iteration, each classifier labels all 5000 instances in the data pool using all available features and selects the most confidently labeled 50 instances for addition to its labeled data.[5] The best F-measure scores achieved by self-training are shown in the last row of Table 2. Overall, self-training only yields marginal performance gains over the baseline.

Nevertheless, self-training outperforms co-training in both cases where naive Bayes is used. While these results seem to suggest that co-training is inherently handicapped for coreference resolution, there are two plausible explanations against this conclusion. First, the fact that self-training has access to all of the available features may account for its superior performance to co-training. This is again partially supported by the fact that the recall level achieved by co-training is lower than that of self-training in both cases in which self-training outperforms co-training. Second, 1000 instances may simply not be sufficient for co-training to be effective for this task: in related work (Ng and Cardie, 2003), we find that starting with 3500–3700 labeled instances instead of 1000 allows co-training to improve the baseline by 4.6% and 9.5% in F-measure using naive Bayes classifiers for the MUC-6 and MUC-7 data sets, respectively.

6 An Alternative Ranking Method

As we have seen before, F-measure scores ultimately decrease as bootstrapping progresses. If the drop were caused by the degradation in the quality of the bootstrapped data, then a more "conservative" instance selection method than that of B&M would help alleviate this problem. Our hypothesis is that selection methods that are based solely on the confidence assigned to an instance by a *single* classifier

[5] Note that this is self-training *without* bagging, unlike the self-training algorithm discussed in Ng and Cardie (2003).

$i_1 > i_2$ if any of the following is true:

$$[\mu(C_1(i_1)) = \mu(C_2(i_1))] \wedge [\mu(C_1(i_2)) \neq \mu(C_2(i_2))]$$

$$[\mu(C_1(i_1)) = \mu(C_2(i_1))] \wedge [\mu(C_1(i_2)) = \mu(C_2(i_2))] \wedge [|C_1(i_1) - C_2(i_1)| > |C_1(i_2) - C_2(i_2)|]$$

$$[\mu(C_1(i_1)) \neq \mu(C_2(i_1))] \wedge [\mu(C_1(i_2)) \neq \mu(C_2(i_2))] \wedge [\text{max}(C_1(i_1), 1 - C_1(i_1)) > \text{max}(C_1(i_2), 1 - C_1(i_2))]$$

Figure 3: The ranking method that a binary classifier C_1 uses to impose a partial ordering on the instances to be selected and added to the training set of binary classifier C_2. i_1 and i_2 are arbitrary instances, and μ is a function that rounds a number to its closest integer.

may be too liberal. In particular, these methods allow the addition of instances with opposing labels to the labeled data; this can potentially result in increased incompatibility between the classifiers.

Consequently, we develop a new procedure for ranking instances in the data pool. The bootstrapping algorithm then selects the highest ranked instances to add to the labeled data in each iteration. The method favors instances whose label is agreed upon by *both* classifiers (**Preference 1**). However, incorporating instances that are confidently labeled by both classifiers may reduce the probability of acquiring new information from the data. Therefore, the method imposes an additional preference for instances that are confidently labeled by one but not both (**Preference 2**). If none of the instances receives the same label from the classifiers, the method resorts to the "rank-by-confidence" method used by B&M (**Preference 3**).

More formally, define a binary classifier as a function that maps an instance to a value that indicates the probability that it is labeled as positive. Now, let μ be a function that rounds a number to its nearest integer. Given two binary classifiers C_1 and C_2 and instances i_1 and i_2, the ranking method shown in Figure 3 uses the three preferences described above to impose a partial ordering on the given instances for incorporation into C_2's labeled data. The method similarly ranks instances to be added to C_1's labeled data, with the roles of C_1 and C_2 reversed.

Steedman et al. (2003a) also investigate instance selection methods for co-training, but their goal is primarily to use selection methods as a means to explore the trade-off between maximizing coverage and maximizing accuracy.[6] In contrast, our focus here is on examining whether a more conservative ranking method can alleviate the problem of performance deterioration. Nevertheless, **Preference 2** is inspired by their $S_{\textit{int-n}}$ selection method, which selects an instance if it belongs to the intersection of the set of the n percent highest scoring instances of one classifier and the set of the n percent lowest scoring instances of the other. To our knowledge, no previous work has examined a ranking method that combines the three preferences described above.

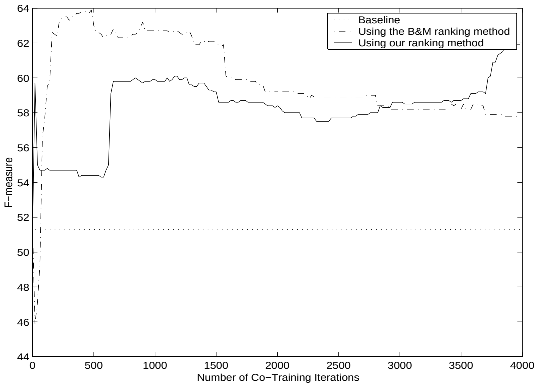

Figure 4: F-measure curves for our single-view bootstrapping algorithm with different ranking methods (pool size = 5000, growth size = 50) for MUC-6.

To compare our ranking procedure with B&M's rank-by-confidence method, we repeat the bootstrapping experiment shown in Figure 2 except that we replace B&M's ranking method with ours. The learning curves generated using the two ranking methods with naive Bayes for the MUC-6 data set are shown in Figure 4. The results are consistent with our intuition regarding the two ranking meth-

[6] McCallum and Nigam (1998) tackle this idea of balancing accuracy and coverage by combining EM and active learning.

ods. The B&M ranking method is more liberal. In particular, each classifier always selects the most confidently labeled instances to add to the other's labeled data at each iteration. If the underlying learners have indeed induced two different hypotheses from the data, then each classifier can potentially acquire informative instances from the other and yield performance improvements very rapidly.

In contrast, our ranking method is more conservative in that it places more emphasis on maintaining labeled data accuracy than the B&M method. As a result, the classifier learns at a slower rate when compared to that in the B&M case: it is not until iteration 600 that we see a sharp rise in F-measure. Due to the "liberal" nature of the B&M method, however, its performance drops dramatically as bootstrapping progresses, whereas ours just dips temporarily. This can potentially be attributed to the more rapid injection of mislabeled instances into the labeled data in the B&M case. At iteration 2800, our method starts to outperform B&M's. Overall, our ranking method does not exhibit the performance trend observed with the B&M method: except for the spike between iterations 0 and 100, F-measure does *not* deteriorate as bootstrapping progresses. Since it is hard to determine a "good" stopping point for bootstrapping due to the paucity of labeled data in a weakly supervised setting, our ranking method can potentially serve as an alternative to the B&M method.

7 Conclusions

We have proposed a single-view, multi-learner bootstrapping algorithm for coreference resolution and shown empirically that the algorithm is a better alternative to the Blum and Mitchell co-training algorithm for this task for which no natural feature split has been found. In addition, we have investigated an example ranking method for bootstrapping that, unlike Blum and Mitchell's rank-by-confidence method, can potentially alleviate the problem of performance deterioration due to the pollution of the labeled data in the course of bootstrapping.

Acknowledgments

We thank the anonymous reviewers for their invaluable and insightful comments. This work was supported in part by NSF Grant IIS–0208028.

References

Michele Banko and Eric Brill. 2001. Scaling to very very large corpora for natural language disambiguation. In *Proceedings of the ACL/EACL*, pages 26–33.

Avrim Blum and Tom Mitchell. 1998. Combining labeled and unlabeled data with co-training. In *Proceedings of COLT*, pages 92–100.

Michael Collins and Yoram Singer. 1999. Unsupervised models for named entity classification. In *Proceedings of EMNLP/VLC*, pages 100–110.

Sally Goldman and Yan Zhou. 2000. Enhancing supervised learning with unlabeled data. In *Proceedings of ICML*, pages 327–334.

Andrew McCallum and Kamal Nigam. 1998. Employing EM and pool-based active learning for text classification. In *Proceedings of ICML*, pages 359–367.

MUC-6. 1995. *Proceedings of the Sixth Message Understanding Conference (MUC-6).*

MUC-7. 1998. *Proceedings of the Seventh Message Understanding Conference (MUC-7).*

Christoph Mueller, Stefan Rapp, and Michael Strube. 2002. Applying co-training to reference resolution. In *Proceedings of the ACL*, pages 352–359.

Ion Muslea, Steven Minton, and Craig Knoblock. 2002. Active + Semi-Supervised Learning = Robust Multi-View Learning. In *Proceedings of ICML*.

Vincent Ng and Claire Cardie. 2002. Combining sample selection and error-driven pruning for machine learning of coreference rules. In *Proceedings of EMNLP*, pages 55–62.

Vincent Ng and Claire Cardie. 2003. Weakly supervised natural language learning without redundant views. In *Proceedings of HLT-NAACL*.

Kamal Nigam and Rayid Ghani. 2000. Analyzing the effectiveness and applicability of co-training. In *Proceedings of CIKM*, pages 86–93.

Kamal Nigam, Andrew McCallum, Sabastian Thrun, and Tom Mitchell. 2000. Text classification from labeled and unlabeled documents using EM. *Machine Learning*, 39(2/3):103–134.

David Pierce and Claire Cardie. 2001. Limitations of co-training for natural language learning from large datasets. In *Proceedings of EMNLP*, pages 1–9.

Ellen Riloff and Rosie Jones. 1999. Learning dictionaries for information extraction by multi-level bootstrapping. In *Proceedings of AAAI*, pages 474–479.

M. Steedman, R. Hwa, S. Clark, M. Osborne, A. Sarkar, J. Hockenmaier, P. Ruhlen, S. Baker, and J. Crim. 2003a. Example selection for bootstrapping statistical parsers. In *Proceedings of HLT-NAACL*.

M. Steedman, M. Osborne, A. Sarkar, S. Clark, R. Hwa, J. Hockenmaier, P. Ruhlen, S. Baker, and J. Crim. 2003b. Bootstrapping statistical parsers from small datasets. In *Proceedings of the EACL*.

M. Vilain, J. Burger, J. Aberdeen, D. Connolly, and L. Hirschman. 1995. A model-theoretic coreference scoring scheme. In *Proceedings of the Sixth Message Understanding Conference (MUC-6)*, pages 45–52.

David Yarowsky. 1995. Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of the ACL*, pages 189–196.