Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics: Main Proceedings (HLT-NAACL), Edmonton, Canada, May 2003, pp. 173-180.

Weakly Supervised Natural Language Learning Without Redundant Views

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

Abstract

We investigate single-view algorithms as an alternative to multi-view algorithms for weakly supervised learning for natural language processing tasks without a natural feature split. In particular, we apply co-training, self-training, and EM to one such task and find that both self-training and FS-EM, a new variation of EM that incorporates feature selection, outperform co-training and are comparatively less sensitive to parameter changes.

1 Introduction

Multi-view weakly supervised learning paradigms such as co-training (Blum and Mitchell, 1998) and co-EM (Nigam and Ghani, 2000) learn a classification 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). Multi-view learning has been successfully applied to a number of tasks in natural language processing (NLP), including text classification (Blum and Mitchell, 1998; Nigam and Ghani, 2000), named entity classification (Collins and Singer, 1999), base noun phrase bracketing (Pierce and Cardie, 2001), and statistical parsing (Sarkar, 2001; Steedman et al., 2003).

The theoretical performance guarantees of multi-view weakly supervised algorithms come with two fairly strong assumptions on the views. First, each view must be *sufficient* for learning the given concept. Second, the views must be *conditionally independent* of each other given the class label. When both conditions are met, Blum and Mitchell prove that an initial weak learner can be boosted using unlabeled data.

Unfortunately, finding a set of views that satisfies both of these conditions is by no means an easy problem. In addition, recent empirical results by Muslea et al. (2002) and Nigam and Ghani (2000) have shown that multi-view algorithms are quite sensitive to the two underlying assumptions on the views. Effective view factorization in multi-view learning paradigms, therefore, remains an important issue for their successful application. In practice, views are supplied by users or domain experts, who determine a natural feature split that is *expected* to be redundant (i.e. each view is expected to be sufficient for learning the target concept) and conditionally independent given the class label.[1]

We investigate here the application of weakly supervised learning algorithms to problems for which no obvious natural feature split exists and hypothesize that, in these cases, single-view weakly supervised algorithms will perform better than their multi-view counterparts. Motivated, in part, by the results in Mueller et al. (2002), we use the task of noun phrase coreference resolution for illustration throughout the paper.[2] In our experiments, we compare the performance of the Blum and Mitchell co-training algorithm with that of two commonly used single-view algorithms, namely, self-training and Expectation-Maximization (EM). In comparison to co-training, self-training achieves substantially superior performance and is less sensitive to its input parameters. EM, on the other hand, fails to boost performance, and we attribute this phenomenon to the presence of redundant features in the underlying generative model. Consequently, we propose a wrapper-based feature selection method (John et al., 1994) for EM that results in performance improvements comparable to that observed with self-training. Overall, our results suggest that single-view

[1] Abney (2002) argues that the conditional independence assumption is remarkably strong and is rarely satisfied in real data sets, showing that a weaker independence assumption suffices.

[2] Mueller et al. (2002) explore a heuristic method for view factorization for the related problem of anaphora resolution, but find that co-training shows no performance improvements for any type of German anaphor except pronouns over a baseline classifier trained on a small set of labeled data.

weakly supervised learning algorithms are a viable alternative to multi-view algorithms for data sets where a natural feature split into separate, redundant views is not available.

The remainder of the paper is organized as follows. Section 2 presents an overview of the three weakly supervised learning algorithms mentioned previously. In section 3, we introduce noun phrase coreference resolution and describe the machine learning framework for the problem. In section 4, we evaluate the weakly supervised learning algorithms on the task of coreference resolution. Section 5 introduces a method for improving the performance of weakly supervised EM via feature selection. We conclude with future work in section 6.

2 Weakly Supervised Algorithms

In this section, we give a high-level description of our implementation of the three weakly supervised algorithms that we use in our comparison, namely, co-training, self-training, and EM.

2.1 Co-Training

Co-training (Blum and Mitchell, 1998) is a multi-view weakly supervised algorithm that trains two classifiers that can help augment each other's labeled data using two separate but redundant views of the data. Each classifier is trained using one view of the data and predicts the 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 from the pool 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 refilled with instances drawn from the unlabeled data and the process is repeated for several iterations. During testing, each classifier makes an independent decision for a test instance and the decision associated with the higher confidence is taken to be the final prediction for the instance.

2.2 Self-Training

Self-training is a single-view weakly supervised algorithm that has appeared in various forms in the literature. The version of the algorithm that we consider here is a variation of the one presented in Banko and Brill (2001).

Initially, we use *bagging* (Breiman, 1996) to train a committee of classifiers using the labeled data. Specifically, each classifier is trained on a *bootstrap sample* created by randomly sampling instances with replacement from the labeled data until the size of the bootstrap sample is equal to that of the labeled data. Then each member of the committee (or bag) predicts the labels of all unlabeled data. The algorithm selects an unlabeled instance for adding to the labeled data if and only if all bags agree upon its label. This ensures that only the unlabeled instances that have a high probability of being assigned the correct label will be incorporated into the labeled set. The above steps are repeated until all unlabeled data is labeled or a fixed point is reached. Following Breiman (1996), we perform simple majority voting using the committee to predict the label of a test instance.

2.3 EM

The use of EM as a single-view weakly supervised classification algorithm is introduced in Nigam et al. (2000). Like the classic unsupervised EM algorithm (Dempster et al., 1977), weakly supervised EM assumes a parametric model of data generation. The labels of the unlabeled data are treated as missing data. The goal is to find a model such that the posterior probability of its parameters is locally maximized given both the labeled data and the unlabeled data.

Initially, the algorithm estimates the model parameters by training a probabilistic classifier on the labeled instances. Then, in the *E-step*, all unlabeled data is probabilistically labeled by the classifier. In the *M-step*, the parameters of the generative model are re-estimated using both the initially labeled data and the probabilistically labeled data to obtain a *maximum a posteriori* (MAP) hypothesis. The E-step and the M-step are repeated for several iterations. The resulting model is then used to make predictions for the test instances.

3 The Machine Learning Framework for 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. In this section, we give an overview of the coreference resolution system to which the weakly supervised algorithms described in the previous section are applied.

The framework underlying the system is a standard combination of classification and clustering employed by supervised learning approaches (e.g. Ng and Cardie (2002); Soon et al. (2001)). Specifically, coreference resolution is 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. Training instances are generated by pairing each NP with each of its preceding NPs in the document. The classification associated with a training instance is one of COREFERENT or NOT COREFERENT depending on whether the NPs

Feature Type	Feature	Description
Lexical	PRO_STR	C if both NPs are pronominal and are the same string; else I.
	PN_STR	C if both NPs are proper names and are the same string; else I.
	SOON_STR_NONPRO	C if both NPs are non-pronominal and the string of NP_{id} matches that of NP_{jd}; else I.
Grammatical	PRONOUN_1	Y if NP_{id} is a pronoun; else N.
	PRONOUN_2	Y if NP_{jd} is a pronoun; else N.
	DEMONSTRATIVE_2	Y if NP_{jd} starts with a demonstrative such as "this," "that," "these," or "those;" else N.
	BOTH_PROPER_NOUNS	C if both NPs are proper names; NA if exactly one NP is a proper name; else I.
	NUMBER	C if the NP pair agree in number; I if they disagree; NA if number information for one or both NPs cannot be determined.
	GENDER	C if the NP pair agree in gender; I if they disagree; NA if gender information for one or both NPs cannot be determined.
	ANIMACY	C if the NPs match in animacy; else I.
	APPOSITIVE	C if the NPs are in an appositive relationship; else I.
	PREDNOM	C if the NPs form a predicate nominal construction; else I.
	BINDING	I if the NPs violate conditions B or C of the Binding Theory; else C.
	CONTRAINDICES	I if the NPs cannot be co-indexed based on simple heuristics; else C. For instance, two non-pronominal NPs separated by a preposition cannot be co-indexed.
	SPAN	I if one NP spans the other; else C.
	MAXIMALNP	I if both NPs have the same maximal NP projection; else C.
	SYNTAX	I if the NPs have incompatible values for the BINDING, CONTRAINDICES, SPAN or MAXIMALNP constraints; else C.
	INDEFINITE	I if NP_{jd} is an indefinite and not appositive; else C.
	PRONOUN	I if NP_{id} is a pronoun and NP_{jd} is not; else C.
	EMBEDDED_1	Y if NP_{id} is an embedded noun; else N.
	TITLE	I if one or both of the NPs is a title; else C.
Semantic	WNCLASS	C if the NPs have the same WordNet semantic class; I if they don't; NA if the semantic class information for one or both NPs cannot be determined.
	ALIAS	C if one NP is an alias of the other; else I.
Positional	SENTNUM	Distance between the NPs in terms of the number of sentences.
Others	PRO_RESOLVE	C if NP_{jd} is a pronoun and NP_{id} is its antecedent according to a naive pronoun resolution algorithm; else I.

Table 1: Feature set for the coreference system. The feature set contains relational and non-relational features that are used to generate an instance representing two NPs, NP_{id} and NP_{jd}, in document d, where NP_{id} precedes NP_{jd}. Non-relational features test some property P of one of the NPs under consideration and take on a value of **Y**ES or **N**O depending on whether P holds. Relational features test whether some property P holds for the NP pair under consideration and indicate whether the NPs are **C**OMPATIBLE or **I**NCOMPATIBLE w.r.t. P; a value of **N**OT **A**PPLICABLE is used when property P does not apply.

co-refer in the text. A separate clustering mechanism then coordinates the possibly contradictory pairwise classifications and constructs a partition on the set of NPs.

We perform the experiments in this paper using our coreference resolution system (see Ng and Cardie (2002)). For the sake of completeness, we include the descriptions of the 25 features employed by the system in Table 1. Linguistically, the features can be divided into five groups: lexical, grammatical, semantic, positional, and others. However, we use naive Bayes rather than decision tree induction as the underlying learning algorithm to train a coreference classifier, simply because (1) it provides a generative model assumed by EM and hence facilitates comparison between different approaches and (2) it is more robust to the skewed class distributions inherent in coreference data sets than decision tree learners. When the coreference system is used 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.

We conclude this section by noting that view factorization is a non-trivial task for coreference resolution. For many lexical tagging problems such as part-of-speech tagging, views can be drawn naturally from the left-hand and right-hand context. For other tasks such as named entity classification, views can be derived from features inside and outside the phrase under consideration (Collins and Singer, 1999). Unfortunately, neither of these options is possible for coreference resolution. We will explore several heuristic methods for view factorization in the next section.

4 Evaluation

In this section, we empirically test our hypothesis that single-view weakly supervised algorithms can potentially outperform their multi-view counterparts for problems without a natural feature split.

4.1 Experimental Setup

To ensure a fair comparison of the weakly supervised algorithms, the experiments are designed to determine the best parameter setting of each algorithm (in terms of its effectiveness to improve performance) for the data sets we investigate. Specifically, we keep the parameters common to all three weakly supervised algorithms (i.e. the labeled and unlabeled data) constant and vary the algorithm-specific parameters, 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, 1 of which is selected to be the annotated text and the remaining 29 texts are used as unannotated data. For MUC-6, 3486 training instances are generated from 84 NPs in the annotated text. For MUC-7, 3741 training instances are generated from 87 NPs. The unlabeled data is composed of 488173 instances and 478384 instances for the MUC-6 and MUC-7 data sets, respectively. Testing is performed by applying the bootstrapped coreference classifier and the clustering algorithm described in section 3 on the 20–30 "formal evaluation" texts for each of the MUC-6 and MUC-7 data sets.

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

Views. We tested three pairs of views. Table 2 reproduces the 25 features of the coreference system and shows the views we employ. Specifically, the three view pairs are generated by the following methods.

- Mueller et al.'s heuristic method. Starting from two empty views, the iterative algorithm selects for each view the feature whose addition maximizes the performance of the respective view on the labeled data at each iteration. [3] This method produces the view pair V1 and V2 in Table 2 for the MUC-6 data set. A different view pair is produced for MUC-7.
- Random splitting of features into views. Starting from two empty views, an iterative algorithm that randomly chooses a feature for each view at each step is used to split the feature set. The resulting view pair V3 and V4 is used for both the MUC-6 and MUC-7 data sets.
- Splitting of features according to the feature type. Specifically, one view comprises the lexico-syntactic features and the other the remaining ones. This approach produces the view pair V5 and V6, which is used for both data sets.

Pool size. We tested pool sizes of 500, 1000, 5000.

Growth size. We tested values of 10, 50, 100, 200, 250.

[3]Space limitation precludes a detailed description of this method. See Mueller et al. (2002) for details.

Feature	V1	V2	V3	V4	V5	V6
PRO_STR		X		X	X	
PN_STR		X		X	X	
SOON_STR_NONPRO		X	X		X	
PRONOUN_1		X		X	X	
PRONOUN_2	X			X	X	
DEMONSTRATIVE_2		X	X		X	
BOTH_PROPER_NOUNS	X		X		X	
NUMBER	X		X		X	
GENDER		X	X		X	
ANIMACY	X			X	X	
APPOSITIVE	X		X		X	
PREDNOM		X	X		X	
BINDING		X		X	X	
CONTRAINDICES	X			X	X	
SPAN	X		X		X	
MAXIMALNP	X		X		X	
SYNTAX		X		X	X	
INDEFINITE	X		X		X	
PRONOUN	X		X		X	
EMBEDDED_1		X		X	X	
TITLE		X	X		X	
WNCLASS	X			X		X
ALIAS	X		X			X
SENTNUM		X		X		X
PRO_RESOLVE	X			X		X

Table 2: Co-training view pairs employed by the coreference system. Column 1 lists the 25 features shown in Table 1. Columns 2-7 show three different pairs of views that we have attempted for co-training coreference classifiers.

Number of co-training iterations. We monitored performance on the test data at every 10 iterations of co-training and ran the algorithm until performance stabilized.

Self-training parameters. Given the labeled and unlabeled data, self-training requires only the specification of the number of bags. We tested all odd number of bags between 1 and 25.

EM parameters. Given the labeled and unlabeled data, EM has only one parameter — the number of iterations. We ran EM to convergence and kept track of its test set performance at every iteration.

4.2 Results and Discussion

Results are shown in Table 4.1, 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 labeled document using naive Bayes, achieves an F-measure of 55.5 and 43.8 on the MUC-6 and MUC-7 data sets, respectively.

The results shown in row 2 of Table 4.1 correspond to the best F-measure scores achieved by co-training across all of the parameter combinations described in the previous subsection. The parameter settings with which the best results are obtained are also shown in the table. To

Experiments	MUC-6				MUC-7			
	Best Parameter Setting	R	P	F	Best Parameter Setting	R	P	F
Baseline	—	58.3	52.9	55.5	—	52.8	37.4	43.8
Co-Training	v=V5/V6,g=50,p=5000,i=220	47.5	81.9	60.1	v=V5/V6,g=100,p=500,i=260	40.6	77.6	53.3
Self-Training	b=7	54.1	78.6	64.1	b=9	54.6	62.6	58.3
EM	i=20	64.8	51.8	57.6	i=2	54.1	40.7	46.4
FS-EM	—	64.2	66.6	65.4	—	53.3	70.3	60.5

Table 3: Comparative results of co-training, self-training, EM, and FS-EM (to be described in section 5). **R**ecall, **P**recision, and **F**-measure are provided. For co-training, self-training, and EM, the best results (F-measure) achieved by the algorithms and the corresponding parameter settings (with views v, growth size g, pool size p, number of iterations i, and number of bags b) are shown.

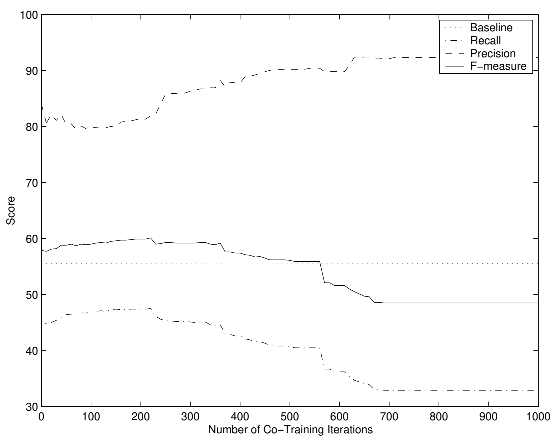

Figure 1: Learning curve for co-training (pool size = 5000, growth size = 50) for the MUC-6 data set.

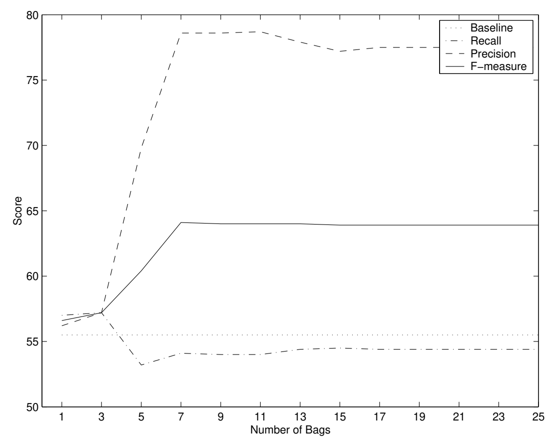

Figure 2: Effect of the number of bags on the performance of self-training for the MUC-6 data set.

get a better picture of the behavior of co-training, we present the learning curve for the co-training run that gives rise to the best F-measure for the MUC-6 data set in Figure 1. The horizontal (dotted) line shows the performance of the baseline system, which achieves an F-measure of 55.5, as described above. As co-training progresses, F-measure peaks at iteration 220 and then gradually drops below that of the baseline after iteration 570.

Although co-training produces substantial improvements over the baseline at its best parameter settings, a closer examination of our results reveals that they corroborate previous findings: the algorithm is sensitive not only to the number of iterations, but to other input parameters such as the pool size and the growth size as well (Nigam and Ghani, 2000; Pierce and Cardie, 2001). The lack of a principled method for determining these parameters in a weakly supervised setting where labeled data is scarce remains a serious disadvantage for co-training.

Self-training results are shown in row 3 of Table 4.1: self-training performs substantially better than both the baseline and co-training for both data sets. In contrast to co-training, however, self-training is relatively insensitive to its input parameter. Figure 2 shows the fairly consistent performance of self-training with seven or more bags for the MUC-6 data set. We observe similar trends for the MUC-7 data set. These results are consistent with empirical studies of bagging across a variety of classification tasks where seven to 25 bags are deemed sufficient (Breiman, 1996).

To gain a deeper insight into the behavior of self-training, we plot the learning curve for self-training using 7 bags in Figure 3, again for the MUC-6 data set. At iteration 0 (i.e. before any unlabeled data is incorporated), the F-measure score achieved by self-training is higher than that of the baseline system (58.5 vs. 55.5). The observed difference is due to voting within the self-training algorithm. Voting has proved to be an effective technique for improving the accuracy of a classifier when training data is scarce by reducing the variance of a particular training corpus (Breiman, 1996). After the first iteration, there is a rapid increase in F-measure, which is accompanied by large gains in precision and smaller drops in recall. These results are consistent with our intuition regarding self-training: at each iteration the algorithm incorporates only instances whose label it is most confident about into the labeled data, thereby ensuring that precision will in-

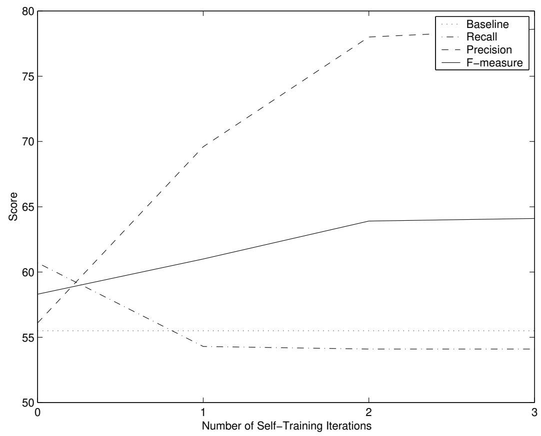

Figure 3: Learning curve for self-training using 7 bags for the MUC-6 data set.

crease. [4]

As we can see from Table 4.1, the recall level achieved by co-training is much lower than that of self-training. This is an indication that each co-training view is insufficient for learning the concept: the feature split limits any interaction of features in different views that might produce better recall. Overall, these results provide evidence that self-training is a better alternative to co-training for weakly supervised learning for problems such as coreference resolution where no natural feature split exists.

On the other hand, EM only gives rise to modest performance gains over the baseline system, as we can see from row 4 of Table 4.1. The performance of EM depends in part on the correctness of the underlying generative model (Nigam et al., 2000), which in our case is naive Bayes. In this model, an instance with m feature values $<x_1, \ldots, x_m>$ and class y is created by first choosing the class with prior probability $P(y)$ and then generating each available feature x_j with probability $P(x_j|y)$ independently, under the assumption that the feature values are conditionally independent given the class. As a result, model correctness is adversely affected by redundant features, which clearly invalidate the conditional independence assumption. In fact, naive Bayes is known to be bad at handling redundant features (Langley and Sage, 1994).

We hypothesize that the presence of redundant features causes the generative model and hence EM to perform poorly. Although self-training depends on the same model, it only makes use of the binary decisions returned by the model and is therefore more robust to the naive Bayes assumptions, as reflected in its fairly impressive empirical performance.[5] In contrast, the fact that EM relies on the probability estimates of the model makes it more sensitive to the correctness of the model.

5 Meta-Bootstrapping with Feature Selection

If our hypothesis regarding the presence of redundant features were correct, then feature selection could result in an improved generative model, which could in turn improve the performance of weakly supervised EM. This section discusses a wrapper-based feature selection method for EM.

5.1 A Two-Tiered Bootstrapping Algorithm

We now describe the FS-EM algorithm for boosting the performance of weakly supervised algorithms via feature selection. Although named after EM, the algorithm as described is potentially applicable to all single-view weakly supervised algorithms. FS-EM takes as input a supervised learner, a single-view weakly supervised learner, a labeled data set L, and an unlabeled data set U. In addition, it assumes knowledge of the positive class prior (i.e. the true percentage of positive instances in the data) like co-training and requires a deviation threshold that we will explain shortly.

FS-EM, which has a two-level bootstrapping structure, is reminiscent of the meta-bootstrapping algorithm introduced in Riloff and Jones (1999). The outer-level bootstrapping task is feature selection, whereas the inner-level task is to learn a bootstrapped classifier from labeled and unlabeled data as described in section 4. At a high level, FS-EM uses a forward feature selection algorithm to impose a total ordering on the features based on the order in which the features are selected. Specifically, FS-EM performs the three steps below for each feature f_i that has not been selected. First, it uses the weakly supervised learner to train a classifier C from the labeled and unlabeled data ($L \cup U$) using only the feature f_i as well as the features selected thus far. Second, the algorithm uses C to classify all of the instances in $L \cup U$. Finally, FS-EM trains a new model on just U, which is now labeled by C. At the end of the three steps, exactly one model is trained for each feature that has not been selected. The forward selection algorithm then selects the feature with which the corresponding model achieves the best performance on L (w.r.t. the true labels of the instances in L) for addi-

[4]When tackling the task of confusion set disambiguation, Banko and Brill (2001) observe only modest gains from self-training by bootstrapping from a seed corpus of one million words. We speculate that a labeled data set of this size can possibly enable them to train a reasonably good classifier with which self-training can only offer marginal benefits, but the relationship between the behavior of self-training and the size of the seed (labeled) corpus remains to be shown.

[5]It is possible for naive Bayes classifiers to return optimal classifications even if the conditional independence assumption is violated. See Domingos and Pazzani (1997) for an analysis.

tion to F_{new} (the set of features selected thus far).[6] The process is repeated until all features have been selected.

Unfortunately, since L can be small, selecting a feature for incorporation into F_{new} by measuring the performance of the corresponding model on L may not accurately reflect the actual model performance. To handle this problem, FS-EM has a preference for adding features whose inclusion results in a classification in which the positive class prior (i.e. the probability that an instance is labeled as positive), p_i, does not deviate from the true positive class prior, p, by more than a pre-specified threshold value, δ. A large deviation from the true prior is an indication that the resulting classification of the data does not correspond closely to the actual classification. This algorithmic bias is particularly useful for weakly supervised learners (such as EM) that optimize an objective function other than classification accuracy and can potentially produce a classification that is substantially different from the actual one. Specifically, FS-EM attempts to ensure that the classification produced by the weakly supervised learner *weakly agrees* with the actual classification, where the weak disagreement rate between two classifications is defined as the difference between their positive class priors. Note that weak agreement is a necessary but not sufficient condition for two classifications to be identical.[7]

Nevertheless, if the addition of any of the features to F_{new} does not produce a classification that weakly agrees with the true one, FS-EM picks the feature whose inclusion results in a positive class prior that has the least deviation instead. This step can be viewed as introducing "pseudo-random" noise into the feature selection process. The hope is that the deviation of the high-scoring, "high-deviation" features can be lowered by first incorporating those with "low deviation", thus continuing to strive for weak agreement while potentially achieving better performance on L.

The final set of features, F_{final}, is composed of the first n features chosen by the feature selection algorithm, where n is the largest number of features that can achieve the best performance on L subject to the condition that the corresponding classification produced by the weakly supervised algorithm weakly disagrees with the true one by at most δ. The output of FS-EM is a classifier that the weakly supervised learner learns from L and U using only the features in F_{final}. The pseudo-code describing FS-EM is shown in Figure 4.

[6]The reason for using only L (instead of L and U) in the validation step is primarily to preclude the possibility of getting a poor estimation of model performance as a result of the presence of potentially inaccurately labeled data from U.

[7]In other words, $p = p_i$ does *not* imply that the corresponding classifications are identical.

```
Input: S (a supervised learning algorithm)
       W (a single-view weakly supervised learning algorithm)
       L (labeled data)
       U (unlabeled data)
       F (original feature set)
       p (true positive class prior)
       δ (deviation threshold)
Initialize: K := |F|, F_new := ∅
for k = 1, ..., K:
   foreach f_i in F:
      use W to learn a classifier C from L and U using only
         F_new ∪ {f_i} with S as the underlying supervised learner
      use C to classify all instances in L ∪ U
      p_i := probability that an instance in L ∪ U is labeled
         as positive by C
      use S to train a classifier C' on U using F_new ∪ {f_i} only
      s_i := classification accuracy of C' on L
   Scores := { s_i | |p_i − p| ≤ δ }
   if Scores ≠ ∅
      i' := argmax_i Scores
   else
      i' := argmin_i { |p_i − p| }
   F'(k) := f_i'
   S'(k) := s_i'
   P'(k) := p_i'
   F_new := F_new ∪ {f_i'}
   F := F \ {f_i'}
T := { j | (|P'(j) − p| ≤ δ) }
t' := max { t | S'(t) > S'(j) ∀j, t ∈ T }
F_final := { F'(t) | t ≤ t' }
use W to learn a classifier C_final from L and U using only
   F_final with S as the underlying supervised learner
return C_final
```

Figure 4: The FS-EM algorithm.

5.2 Results and Discussion

We instantiate FS-EM with naive Bayes as the supervised learner and EM as the weakly supervised learner, providing it with the same amount of labeled and unlabeled data as in previous experiments and setting δ to 0.01. EM is run for 7 iterations whenever it is invoked.[8] Results using FS-EM are shown in row 5 of Table 4.1. In comparison to EM, F-measure increases from 57.6 to 65.4 for MUC-6, and from 46.4 to 60.5 for MUC-7, allowing FS-EM to even surpass the performance of self-training. These results are consistent with our hypothesis that the performance of EM can be boosted by improving the underlying generative model using feature selection.

Finally, although FS-EM is only applicable to two-class problems, it can be generalized fairly easily to handle multi-class problems, where the true label distribution

[8]Seven is used because we follow the choice of previous work (Muslea et al., 2002; Nigam and Ghani, 2000). Additional experiments in which EM is run for 5 and 9 iterations give similar results.

is assumed to be available and the weak agreement rate can be measured based on the similarity of two distributions.

6 Conclusions and Future Work

We have investigated single-view algorithms (self-training and EM) as an alternative to multi-view algorithms (co-training) for weakly supervised learning for problems that do not appear to have a natural feature split. Experimental results on two coreference data sets indicate that self-training outperforms co-training under various parameter settings and is comparatively less sensitive to parameter changes. While weakly supervised EM is not able to outperform co-training, we introduce a variation of EM, FS-EM, for boosting the performance of EM via feature selection. Like self-training, FS-EM easily outperforms co-training.

Co-training algorithms such as CoBoost (Collins and Singer, 1999) and Greedy Agreement (Abney, 2002) that explicitly trade classifier agreement on unlabeled data against error on labeled data may be more robust to the underlying assumptions of co-training and can conceivably perform better than the Blum and Mitchell algorithm for problems without a natural feature split.[9] Other less studied single-view weakly supervised algorithms in the NLP community such as co-training with different learning algorithms (Goldman and Zhou, 2000) and graph mincuts (Blum and Chawla, 2001) can be similarly applied to these problems to further test our original hypothesis. We plan to explore these possibilities in future research.

Acknowledgments

We thank Lillian Lee, Thorsten Joachims, and the Cornell NLP group including Regina Barzilay, Eric Breck, Bo Pang, and Steven Baker for many helpful comments. We also thank three anonymous reviewers for their feedback and Ted Pedersen for encouraging us to apply ensemble methods to coreference resolution. This work was supported in part by NSF Grant IIS–0208028.

References

S. Abney. 2002. Bootstrapping. In *Proceedings of the ACL*, pages 360–367.

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

A. Blum and S. Chawla. 2001. Learning from labeled and unlabeled data using graph mincuts. In *Proceedings of ICML*, pages 19–26.

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

L. Breiman. 1996. Bagging predictors. *Machine Learning*, 24:123–140.

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

S. Dasgupta, M. Littman, and D. McAllester. 2001. PAC generalization bounds for co-training. In *Advances in NIPS*.

A. Dempster, N. Laird, and D. Rubin. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1):1–38.

P. Domingos and M. J. Pazzani. 1997. On the optimality of the simple Bayesian classifier under zero-one loss. *Machine Learning*, 29:103–130.

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

G. John, R. Kohavi, and K. Pfleger. 1994. Irrelevant features and the subset selection problem. In *Proceedings of ICML*.

P. Langley and S. Sage. 1994. Induction of selective Bayesian classifiers. In *Proceedings of UAI*, pages 399–406.

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

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

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

I. Muslea, S. Minton, and C. Knoblock. 2002. Active + Semi-Supervised Learning = Robust Multi-View Learning. In *Proceedings of ICML*.

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

K. Nigam and R. Ghani. 2000. Analyzing the effectiveness and applicability of co-training. In *Proceedings of CIKM*.

K. Nigam, A. McCallum, S. Thrun, and T. Mitchell. 2000. Text classification from labeled and unlabeled documents using EM. *Machine Learning*, 39(2/3):103–134.

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

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

A. Sarkar. 2001. Applying co-training methods to statistical parsing. In *Proceedings of the NAACL*, pages 175–182.

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

M. Steedman, M. Osborne, A. Sarkar, S. Clark, R. Hwa, J. Hockenmaier, P. Ruhlen, S. Baker, and J. Crim. 2003. 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*, pages 45–52.

[9] Indeed, Dasgupta et al. (2001) show that, when the conditional independence assumption of the views is satisfied, view classifiers whose agreement on unlabeled data is explicitly maximized will have low generalization error.