# Co-training on Textual Documents with a Single Natural Feature Set

*Jason Chan*
School of Information Technologies
The University of Sydney
NSW 2006 Australia
*jchan3@it.usyd.edu.au*

*Irena Koprinska*
School of Information Technologies
The University of Sydney
NSW 2006 Australia
*irena@it.usyd.edu.au*

*Josiah Poon*
School of Information Technologies
The University of Sydney
NSW 2006 Australia
*josiah@it.usyd.edu*

**Abstract** *Co-training is a semi-supervised technique that allows classifiers to learn with fewer labelled documents by taking advantage of the more abundant unclassified documents. However, conventional co-training requires the dataset to be described by two disjoint and natural feature sets that are redundantly sufficient. In many practical situations datasets have a single set of features and it is not obvious how to split it into two. This paper investigates the performance of co-training with only one natural feature set in two applications: Web page classification and email filtering.*

**Keywords** Text categorization, Web page classification, spam filtering, co-training

## 1 Introduction

As the number of on-line documents increases, finding information that is really relevant to the users' needs becomes very important. This is one aspect of the information overload problem faced by a growing number of people. Research in Text Categorization [10] and Machine Learning has shown that it is possible to build effective classifiers for intelligent information retrieval given a sufficiently large set of labelled examples. However, obtaining labelled documents requires a great deal of human effort and is also a time consuming and tedious process.

Blum and Mitchell [2] introduced a new technique to overcome this problem. This method, called *co-training*, was shown to be capable of converting unlabelled Web documents into labelled Web documents by initially starting off with only a small pool of classified examples. The authors stated two main requirements on the dataset to be satisfied in order for co-training to be beneficial. Firstly, the dataset must be described by two disjoint sets of features that were sufficiently strong. That is, using only either one of the sets of attributes, a classifier can be built with reasonably high accuracy. For example, in their experiment that dealt with the problem of classifying Web pages, the two sets of features used to describe a page were the words in the body of the page and the words in hyperlinks of other documents referring to that particular page. Secondly, the two feature sets should be conditionally independent given the class.

In the great majority of practical situations, there do not exist two natural sets of features that can describe the dataset. In other cases, the data collected may only belong to one of the possible natural feature sets. In this paper, we investigate the applicability of co-training to such datasets. We compare co-training of a single natural feature set and co-training with two natural feature sets. By analysing the results, we address the question of when co-training with a random split of features is likely to be useful. The experiments are based on two applications: Web page classification and spam filtering. This paper extends our work presented in [4] by including addition experimental data and theoretical insights.

This paper is organised as follows. The next section provides important background information on co-training. In section 3, previous work on co-training is covered. Section 4 describes the experimental set-up and summarises the experimental objectives, while section 5 presents the experimental results. Section 6 gives a detailed discussion. The final conclusions and potential future work are given in section 7.

## 2 The Co-training Algorithm

In a given application, we may have a set of redundant features that can be used to classify the instances. That is, it is possible to split up all the features into two sets so that we can build two independent classifiers that can still label the instances correctly. These sets of features are said to be *redundantly sufficient*. As an example, emails can be classified accurately with just the header information (sender, subject, etc.) or just the content in the body of the message.

These two classifiers are trained with a small set of labelled instances so that we have two weak classifiers. They are then employed in a loop to classify all the unlabelled examples. Each classifier selects the most confidently predicted examples and adds these into the training set. Both classifiers then re-learn on the enlarged training set so that they take into account the newly added (and previously unlabelled) data. The loop is then repeated for a

**Proceedings of the 9th Australasian Document Computing Symposium, Melbourne, Australia, December 13, 2004. Copyright for this article remains with the authors.**

number of iterations to maximize performance on a separate validation set.

The co-training algorithm is summarized below:

```
Obtain a small set L of labelled examples
Obtain a large set U of unlabelled examples
Obtain two sets $F_1$ and $F_2$ of features that are
                redundantly sufficient
while U is not empty do:
    Learn classifier $C_1$ from L based on $F_1$
    Learn classifier $C_2$ from L based on $F_2$
    for each classifier $C_i$ do:
        $C_i$ labels examples from U based on $F_i$
        $C_i$ chooses the most confidently predicted
                examples E from U
        E is removed from U and added (with their
                given labels) to L
    end-for
end-while
```

The intuition for why this algorithm should work is as follows. One classifier, with its set of features, can confidently predict the class of an unlabelled example, because it is similar to the training instances. However, it may only be similar to the training instances for this classifier's set of features; the other classifier may not have been so sure about this instance's classification. Because of the confidence with which the first classifier predicts this example's class, it will be labelled accordingly and placed into the training set. Hence, the second classifier will be able to learn from this instance and adjust better in future.

For example, suppose we have two email classifiers with one classifier using the headers of emails and the other using the words in the body of the message. The first classifier has been trained to categorize any email with the word "assignment" to be placed in the folder for teaching. If another email comes along with "assignment" in its subject header, the first classifier will be very confident that this message should be put in the folder for teaching. This will allow the second classifier to learn more about those words in the body of a message that can be used to determine that an email belongs in the folder for teaching. The second classifier learns because the words used in the body of the first email will not necessarily be the same as the words used in the second email.

## 3 Previous Work

Blum and Mitchell [2] performed the first experiments on co-training. The task was to identify the home Web pages of academic courses from a large collection of Web pages collected from several Computer Science departments. Their co-training implementation used the following two natural feature sets: the words present in the Web page (page-based classifier) and the words used in another page's link that pointed to the page (hyperlink-based classifier). The results showed that the error rate of the combined classifier was reduced from 11% to 5%. It was also proved that if the feature sets used by the classifiers are *conditionally independent* given the class, and the target classification function can be approximated, then any initial weak classifier can be improved to arbitrarily high accuracy using co-training. More recent research shows that this condition can be relaxed to a certain extent [9]. For example, it was proven that for two classifiers with weak dependence, the rate of disagreement between them provides an upper-bound limit on their error rate [1].

Kiritchenko and Matwin [5] applied co-training to the domain of email classification. They found that the performance of co-training is sensitive to the learning algorithm used. In particular, co-training with Naïve Bayes (NB) worsens performance, while Support Vector Machines (SVM) improves it. The authors explained this with the inability of NB to deal with large sparse datasets. This explanation was confirmed by significantly better results after feature selection.

Nigam and Ghani [9] investigated the sensitivity of co-training to the assumptions of conditional independence and redundant sufficiency. In their first experiment, co-training was applied to the Web pages database from [2]. The results showed that co-training using NB was not better than Expectation Maximization (EM) even when there is a natural split of features. Both EM and co-training with NB improved the performance of the initial classifier by approximately 10%. The second experiment was performed on a dataset that had been created in a semi-artificial manner so that the two feature sets are truly conditionally independent. In addition, the condition of redundantly sufficient features was met, since the NB trained on each of the data sets separately was able to obtain a small error. It was found that co-training with NB well outperformed EM, and even outperformed NB trained with all instances labelled. Their third experiment involved performing co-training on a dataset whereby a natural split of feature sets is not used. The two feature sets were chosen by randomly assigning all the features of the dataset into two different groups. This was tried for two datasets: one with a clear redundancy of features, and one with an unknown level of redundancy and non-evident natural split in features. The results indicated that the presence of redundancy in the feature sets gave the co-training algorithm a bigger advantage over EM.

Together with theoretical insights, the results of these experiments led the researchers to conclude that co-training has a considerable dependence on the assumptions of conditional independence and redundant sufficiency. However, even when either or both of the assumptions are violated, the performance of co-training can still be quite useful in improving a classifier's performance. In particular, in many practical settings, co-training is likely to be beneficial.

## 4 Experimental Setup

### 4.1 Objective

In the large majority of cases, datasets consist of only a single set of features with no obvious or natural way to divide them into two separate sets. Hence, the question of whether co-training can be useful with only a single natural feature set is of great practical importance. This paper investigates the performance of co-training with only one natural feature set in comparison to the use of two natural feature sets. The main question that we address is: *How useful is co-training with a single natural feature set*?

To tackle this question, we performed two sets of experiments in the two domains of Web page classification and spam filtering. Table 1 summarizes the four different experiments that we conducted, and the names that we have assigned to them.

| Experiment | Web Page Classification | Spam Filtering |
|---|---|---|
| **Supervised Learning** | *WebSL* | *SpamSL* |
| **Co-Training** | *WebCT* | *SpamCT* |

**Table 1.** Experiment names

The Supervised Learning Experiment deals with supervised learning. Here, we gain some insights into the different classifiers and feature sets, and in particular, determine how redundantly sufficient the different feature sets are. Good classification performance on a given feature subset implies that it is redundantly sufficient.

In the Co-Training Experiment, we perform co-training and compare the performance between using the natural split against using a random split of all the features. We also compare these performances against traditional supervised learning on the initial labelled set to see what improvement the co-training process gives a learner.

### 4.2 Pre-processing and Feature Selection

In both the Web page classification and spam filtering experiments, the documents were initially pre-processed by applying a stop-list[1], with the Web pages being further processed by removing certain html tags.

The standard bag-of-words representation was used and feature selection was performed with Information Gain [12]. Feature selection is a common method to reduce running time by using only the most important attributes, and has been shown previously to improve performance, such as in [12]. Upon inspection of the word lists, it was decided that the top 100 words was a suitable cut-off for experiments in both domains, resulting in a dimensionality reduction of about 98%. Hence, for each feature set, each document was represented using the term frequencies of the selected 100 features. Note that feature selection was applied individually to each feature subset.

### 4.3 Web Page Classification

Four topics with approximately 90 Web pages on each topic were retrieved and rated by four users. The Web pages thus formed 16 datasets. The phrases "nuclear fusion", "circulatory system", "food pyramid" and "greenhouse effect ozone layer" were used as queries in the Google[2] search engine to obtain Web documents on four topics.

For each topic, the users were given an objective for which they had to attempt to achieve with the given Web pages. A rating of either "good" or "bad" was assigned, depending on how useful a given user found that particular page for the task assigned to that topic. Each user's rating for the different pages was saved separately. Table 2 summarizes the feature sets used in the Web page classification experiments.

| Feature set | Description |
|---|---|
| *Headers* | all words that appear in either titles, headings or hyperlinks |
| *Words* | all words that appear in the Web page without counting occurrences in the titles, headings, or hyperlinks |
| *Half1* | a random selection of half of the feature set *Words* |
| *Half2* | the other half of the words not found in *Half1* |

**Table 2.** Feature sets used in Web page classification experiments

The two feature sets *Half1* and *Half2* are created to test the hypothesis that it is possible to randomly split the feature set into two smaller feature sets to obtain useful results in co-training.

The *Words* and *Headers* feature sets in the domain of Web page classification will hereon be referred to as the *natural feature sets*, while the other feature sets that contain a random selection of words will be referred to as the *random selection feature sets*. Our experiment with supervised learning applied to Web page classification is given the name *WebSL*, while co-training with Web page classification is called *WebCT*.

[1] http://alt-usage-english.org/excerpts/fxcommon.html

[2] http://www.google.com

## 4.4 Spam Filtering

To test the domain of email classification, we used the LingSpam[3] corpus. This dataset consists of 2883 emails of which 479 are spam and 2404 are genuine emails. Each email is broken up into two sections: the text found in the subject header of the email and the words found in the main body of the message. A distinction was made between the words that appeared in the subject header and those that appeared in the body. A summary of the feature sets used in the experiments is given in Table 3.

| Feature set | Description |
|---|---|
| *Body* | all words that appear in the body of an email |
| *Subject* | all words that appear in subject of an email |
| *Half1* | a random selection of half of the feature set consisting of the combination of *Subject* and *Body* |
| *Half2* | the other half of the features not found in *Half1* |

**Table 3.** Feature sets used in email filtering experiments

In similar style to Web page classification, the *Body* and *Subject* feature sets in the domain of spam filtering will hereon be referred to as the *natural feature sets*, while the other feature sets that contain a random selection of words will be referred to as the *random selection feature sets*. The experiment with supervised learning applied to spam filtering is given the name *SpamSL*, while the co-training experiment in spam filtering is called *SpamCT*.

## 4.5 Classifiers

Four types of classifiers were tested: Decision Tree (DT), Random Forest (RF) [3], Naïve Bayes (NB) and Support Vector Machine (SVM). In previous work on co-training [5], NB has often been used as a benchmark. The SVM was used in text categorization and email classification [5] with great success. Implementations of these classifiers were obtained from WEKA [11].

## 4.6 Evaluation

In this paper, f-measure will refer to macro-averaged f1-measure, which is given by the formula: f1 = 2pr / (p+r) where *p* is the macro-averaged precision and *r* is the macro-averaged recall. The macro-averaged precision is the average of the precision of each of the two classes; similarly for recall.

[3] http://www.mlnet.org/cgi-bin/mlnetois.pl/?File=dataset-details.html&Id=963839410Ling-Spam

## 5 Experimental Results

## 5.1 Experiment *WebSL*

In order to focus on general trends, our results were averaged over all users and topics. We analysed the results for each user and topic and found them to be consistent with our general findings. Table 4 summarizes the classification performance using 10-fold cross validation. Note that in each case, the classifiers have access to the top 100 features in the respective feature sets since feature selection was performed on each feature set individually.

| Classifier | Words | Headers | Random halves | All features |
|---|---|---|---|---|
| DT | 74.0 | 65.6 | 73.2 | 73.4 |
| RF | 79.1 | 75.9 | 78.6 | 78.5 |
| NB | 80.5 | 75.1 | 80.5 | 80.5 |
| SVM | 81.6 | 75.7 | 81.2 | 81.6 |

**Table 4.** F-measures (%) using different feature sets in *WebSL*

## 5.2 Experiment *WebCT*

Table 5 summarises the results of the co-training experiments over all users and topics, after labelling all initially unlabelled instances. Shown are the maximum classification performance (max) that is achieved and the performance improvement (increase) achieved over the base classifier trained with only the initial labelled set of 10%.

| Classifier | Natural Feature Split | | Random Feature Split | |
|---|---|---|---|---|
| | max | increase | max | increase |
| DT | 67.5 | 0.0 | 67.9 | 0.5 |
| RF | 72.3 | 3.3 | 72.0 | 3.6 |
| NB | 73.6 | 3.0 | 71.9 | 1.1 |
| SVM | 40.2 | 0.5 | 75.0 | 1.9 |

**Table 5.** F-measures (%) with natural vs random split in *WebCT*

Another 10% of the instances were set aside for the test set, and the remaining data was used as initially unlabelled data. Stratification was performed to ensure an even distribution of instances from the two classes. Ten different combinations of labelled, unlabelled and test sets were used, and the results were averaged over these 10 runs. This somewhat resembles 10-fold cross validation.

The proportion of positive instances to negative instances varied between each user and topic. However, in most cases, this ratio was approximately 2 negative instances to 1 positive instance. For the results shown here, the number of positive and negative instances to be transferred from the unlabelled set to the labelled set was set to 1 and 2 respectively.

## 5.3 Experiment *SpamSL*

Table 6 summarizes the results. 10-fold cross validation was performed, with 2595 instances used in the training set and 288 instances in the test set. The results of RF are not reported here because we experienced memory constraints on our system.

| Classifier | Body | Subject | Random halves | All features |
|---|---|---|---|---|
| DT | 92.7 | 45.5 | 89.2 | 92.7 |
| NB | 86.9 | 45.5 | 85.0 | 86.9 |
| SVM | 88.9 | 63.0 | 87.3 | 88.9 |

**Table 6.** F-measures (%) using various feature sets in *SpamSL*

## 5.4 Experiment *SpamCT*

Table 7 summarises the results. Again, the best classification performance (max) that is achieved and the performance improvement (increase) achieved over the base classifier is given. Figure 1 illustrates the difference in performance over the number of co-training iterations completed for NB.

| Classifier | Natural Feature Split | | Random Feature Split | |
|---|---|---|---|---|
| | max | increase | max | increase |
| DT | 45.5 | 0.0 | 45.5 | 0.0 |
| NB | 84.9 | 8.2 | 86.8 | 10.1 |
| SVM | 78.0 | 0.0 | 79.5 | 1.5 |

**Table 7.** F-measures (%) with natural vs random split in *SpamCT*

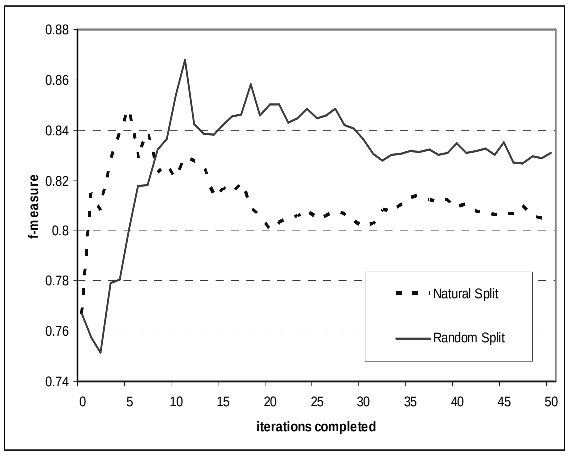

**Figure 1.** Performance of NB in *SpamCT*

We started off with a labelled set of 1 spam and 1 genuine email. 10% of the dataset were retained for testing, while the rest were used as initially unlabelled data. The ratio of spam to genuine emails in LingSpam is 1:5. Following this distribution, 1 newly-labelled spam and 5 newly-labelled genuine emails were transferred from the unlabelled set to the labelled set on each iteration of co-training. We repeated each trial 5 times, obtaining a different sample of labelled, unlabelled and test set each time. The above results are averaged over these 5 trials.

As shown in Figure 1, co-training with a random split of features improves the performance of a classifier in comparison to only using the initial labelled set. We found this to be the case in most settings.

## 6 Discussion

### 6.1 Supervised Learning Experiments

Table 4 and 6 show that the best performing feature sets (for all classifiers) were *Words* in Web page classification and *Body* in spam email filtering, respectively, and the worst were *Headers* and *Subject*. The performance of *Words* is better than *Headers* by 3-9%. (WebST) and *Body* outperforms *Subject* by 25-37% (EmailST). In fact, the performance of *Words* and *Body* alone is the same as using all features. That is, combining *Headers* with Words and *Subject* with *Body* does improve performance.

However, it is very interesting to notice that the performance does not suffer any significant degradation when a random half of the features was selected (compare *Random halves* with *All features*). For example, using half of the words randomly selected from all the available words only decreases the classification performance by no more than 4% in both applications. Hence, the supervised learning results indicate that there is redundant sufficiency in the feature sets from which the random halves were drawn. This suggests that the two domains have satisfied the necessary requirements for co-training.

For Web page classification, we even went one step further by performing a more aggressive sampling and randomly selecting only one-fifth of the feature set. The results (not shown here) indicated that the performance was reduced by a maximum of only 3.3% and 4.2% in comparison to using half and all of the features, respectively. This indicated a significant redundancy among the words in the body of the Web documents.

An interesting question is why the *Headers* and *Subject* feature sets do not perform well. The words found in the *Header* and *Subject* feature sets tend to be more meaningful than other words found in the main body of the document, since they summarise the main topics and their words are usually more selectively chosen than the words found in the main body. The most likely reason for the poor performance of the *Header* and *Subject* features is the significantly lower number of word tokens present in them in comparison to the entire email message or Web page, which makes them poor indicators of the document class. In [2], Blum and Mitchell also reported that their hyperlink-based classifier has inferior performance due to similar reasoning. Another reason is the increased sensitivity to noise when using a fewer word tokens. This means that non-discriminating words are more likely to be treated as significant for the classification when the total number of word tokens is small.

## 6.2 Co-training Experiments

The *increase* columns in Table 5 (WebCT) and Table 7 (SpamCT) show that co-training does not degrade the performance of the initial classifier trained with the small number of positive and negative examples. This holds for both natural-split and random-split co-training.

The same tables also show that co-training with a random split of the features produces results that are comparable with using the natural feature sets. In many cases, the f-measure is higher for the random feature split in comparison to the natural feature split. Why is co-training with a random split of the features so comparable, and in some cases, even better than using the natural feature sets?

There are two reasons for this. Firstly, one of the classifiers used in co-training with the natural feature sets is even weaker than either of the two classifiers using randomly generated feature sets. As shown in the supervised learning experiments, the *Header* and *Subject* feature set are much weaker than any of the random feature sets that were produced. As a result, the classifier using such a feature set is incorrectly labelling many instances in comparison with classifiers built using the random selection feature sets, hence transferring many incorrectly labelled instances into the labelled set.

Secondly, the supervised learning experiments also found that there is redundant sufficiency in the *All* feature set. That is, using a random selection of half of the features from all the features resulted in classifiers that only perform slightly worse than a classifier using all the attributes available. As a result, when performing co-training, both classifiers using their respective half of the features are able to improve the training set because they label unlabelled instances with a sufficiently high classification accuracy.

These two reasons combined suggest that co-training with a random split of a redundantly sufficient feature set can be just as competitive as and even better than co-training with two natural feature sets. This is especially the case when there exists a considerable difference between the classification performances of the two classifiers using the natural feature sets separately. As shown in the supervised learning experiment, a natural feature set consisting of fewer words, such as using the hyperlinks of a Web page, or using the subject header of an email, may produce significantly poorer results. In this event, co-training without the use of this lower quality feature set is likely to be more beneficial.

## 6.3 Comparison between the Classifiers

From Table 5 and 7, it is very clear that the DT classifier performs much worse than any of the other classifiers. Only in the *SpamSL* experiment does the DT do well. This classifier's poor performance is expected, since the branching of a DT is not very effective when there exists a large number of weak features in the dataset.

In the supervised learning experiments, there was little observed difference in performance between the NB and SVM classifiers (see Table 4 and 6). Previous research [5] showed that the SVM classifier was superior to NB in the application of email classification. Our experiment does not obtain this result because we perform drastic feature selection and keep only the best 100 features in each feature set. Hence, the SVM, which performs in high-dimensional feature spaces, does not get to illustrate its advantage over NB in such a setting.

It should be noted that while some classifiers perform best in the supervised classification experiments, others perform better in the co-training experiments, both in terms of highest f-measure obtained and greater improvement. This is the case because the supervised classification experiments are performed using 10-fold cross validation of the entire dataset, which means that the training set is 90% of the dataset. But with co-training, the base classifier starts off with only a few labelled instances, so a classifier's performance before co-training begins will not be similar to the 10-fold cross validated supervised classification results.

Hence, improvement with co-training is obtained if the base classifier is sufficiently strong to make good classifications for the unlabelled data. In particular, the classifier needs to be accurate with those instances in which it has high confidence in, since these are the instances that are selected and transferred from the unlabelled set to the labelled set during co-training. At the same time, we require the classifier to be weak enough so that extra labelled data will help it learn something new.

From Table 7, it can be seen that for spam filtering, the SVM classifier did not improve much with co-training in comparison to the NB classifier. In the case of Web page classification, it is not so clear from Table 5 which classifier improved more. These results occur because in the spam filtering case, the initial labelled set is just 2 instances, compared with 8 in the Web page classification experiment. Hence, in the spam filtering case, a very poor base classifier is built with the SVM, whereas NB can better classify the most confidently labelled instances. In the Web page classification experiment, the SVM base classifier has more training data.

Table 5 shows that in Web page classification, the classifier that improved most was the random forest. The hypothesized reason for this is that the RF classifier is more accurate in selecting the most confident prediction. This is because the RF classifier uses bagging of individual trees [3], which is more noise resilient than using a single classifier. The instance that is considered to be the most confident to label by the RF classifier is supported by a large number of decision trees that are individually confident with the classification of the same instance.

Thus, the performance of co-training depends on the choice of classifier. The encouraging results discussed in sections 6.1 and 6.2 are shown to be valid provided that a suitable classifier is used for co-training.

## 7 Conclusion

Document filtering is becoming increasingly important as excessive quantities of data become available in the Information Age. Classifiers can be built to filter out unwanted information but typically require many labelled examples. Co-training has been shown to be a beneficial tool in improving the performance of a classifier that is given only a small training set. However, conventional co-training requires the documents to be able to be described by two natural sets of features, which is not always possible.

The primary objective of the supervised learning experiments was to determine whether the two corpora were redundantly sufficient. It was found that classifiers could be built using a random selection of only half of all available features and still obtain very good classification performance. This implies that the datasets are redundantly sufficient.

In the co-training experiments, the first natural feature set contained the words used in the main body of the Web pages or emails messages, while the second feature set consisted of the words occurring in the subject header for emails and header information for Web pages. It was found that co-training using a random split of all the features was just as competitive as, and often outperformed, co-training with the natural feature sets. Also, classification performance generally improved over the initial classifier trained on the small initial labelled set with random-split co-training.

An important element that is needed in a feature set for co-training with a random split to work well is a dataset with high redundancy. When this condition is met, a random split of the feature set will produce two subsets, each of which can still be used on its own by a classifier to achieve a sufficiently high classification performance.

Also, co-training with a random split of a single natural feature set should be more preferable than co-training with two natural feature sets if one of the natural feature sets is considerably weaker than the other. This is particularly the case with Web pages and emails, where feature sets other than the words in the main body will typically be weak because of their small size. In such cases, co-training with a random split of all the features should produce comparable and possibly better results.

Further research is needed in other domains to determine just how successful this approach will be in comparison to co-training with natural feature sets currently used. Another possible path for future work is to compare the performance of co-training on two sets of features that are randomly split from a single natural feature set, with self-training [9] on this single natural feature set. A deeper investigation into the relation between choice of classifier and performance of co-training would also be beneficial.

Finally, rather than using a random split with reasonable results, developing a view-factorization algorithm capable of obtaining the optimal (or a near-optimal) split of the features presents a future challenge of considerable importance. There has already been a view-validation algorithm proposed [7] that can predict whether a given pair of views for a task will allow for a multi-view algorithm to outperform its single-view counterpart. This is the first step towards implementing a view-factorization algorithm. Also, a greedy heuristic method that gradually builds two sets of classifiers, adding strong features to each classifier one at a time, has been introduced in [6]. However, this method has been used in applications [8] where using the natural feature split has been reported to perform better.

## 8 References

[1] S. Abney, Bootstrapping, *Proceedings of the 40th Annual Meeting of the Association for Computational Linguisitics*, 2002.

[2] A. Blum, T. Mitchell, Combining Labeled and Unlabeled Data with Co-Training, *Proceedings of the Workshop on Computational Learning Theory,* 1998.

[3] L. Breiman, Random Forests, *Technical Report, Department of Statistics, University of California, Berkeley,* 1999.

[4] J. Chan, I. Koprinska, J. Poon, Co-training with a Single Natural Feature Set Applied to Email Classification, *Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence (WI'04)*, 2004.

[5] S. Kiritchenko, S. Matwin, Email Classification with Co-Training, *Proceedings of CASCON*, 2001.

[6] C. Muller, S. Rapp, M. Strube, Applying Co-training to Reference Resolution, *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*, 2002.

[7] I. Muslea, S. Minton, C. A. Knoblock, Adaptive View Validation: A First Step towards Automatic View Detection, *Proceedings of International Conference on Machine Learning*, 2002.

[8] V. Ng, C. Cardie, Weakly Supervised Natural Language Learning without Redundant Views, *Proceedings of HLT-NAACL*, 2003.

[9] K. Nigam, R. Ghani, Analyzing the Effectiveness and Applicability of Co-Training, *Proceedings of the 9th International Conference on Information and Knowledge Management,* 2000.

[10] F. Sebastiani, Machine Learning in Automated Text categorization, *ACM Comuting. Surveys*, 2002.

[11] I. H. Witten, E. Frank, *Data Mining: Practical machine learning tools with Java implementations*. Morgan Kaufmann, 1999.

[12] Y. Yang, J. O. Pedersen, A Comparative Study on Feature Selection in Text Categorization, *Proceedings of the 14th International Conference on Machine Learning*, 1997.