Combining Labeled and Unlabeled Data for MultiClass Text Categorization

Rayid Ghani RAYID.GHANI@ACCENTURE.COM
Accenture Technology Labs, 161 N Clark St, Chicago, IL 60601 USA

Abstract

Supervised learning techniques for text classification often require a large number of labeled examples to learn accurately. One way to reduce the amount of labeled data required is to develop algorithms that can learn effectively from a small number of labeled examples augmented with a large number of unlabeled examples. Current text learning techniques for combining labeled and unlabeled, such as EM and Co-Training, are mostly applicable for classification tasks with a small number of classes and do not scale up well for large multiclass problems. In this paper, we develop a framework to incorporate unlabeled data in the Error-Correcting Output Coding (ECOC) setup by first decomposing multiclass problems into multiple binary problems and then using Co-Training to learn the individual binary classification problems. We show that our method is especially useful for text classification tasks involving a large number of categories and outperforms other semi-supervised learning techniques such as EM and Co-Training. In addition to being highly accurate, this method utilizes the hamming distance from ECOC to provide high-precision results. We also present results with algorithms other than co-training in this framework and show that co-training is uniquely suited to work well within ECOC.

1. Introduction

A major concern with supervised learning techniques for text classification is that they often require a large number of labeled examples to learn accurately. Collecting a large number of labeled examples can be a very expensive process, thus emphasizing the need for algorithms that can provide accurate classifications after getting only a few labeled examples.

One way to reduce the amount of labeled data required is to develop algorithms that can learn from a small number of labeled examples augmented with a large number of unlabeled examples. Unlabeled examples, especially in the case of text classification problems, are much less expensive and easier to obtain than labeled examples. The Web contains a huge amount of text data that can serve as unlabeled data for many classification tasks. Collecting this text is often cheap since web spiders and crawlers can be programmed to automatically do this task.

Recently, there has been interest in supervised learning algorithms that combine labeled and unlabeled data. Examples of such research include using Expectation-Maximization to estimate maximum a posteriori parameters of a generative model for text classification (Nigam et al., 2000), using a generative model built from unlabeled data to perform discriminative classification (Jaakkola & Haussler, 1999), and using transductive inference for support vector machines to optimize performance on a specific test set (Joachims, 1999). These studies have shown that using unlabeled data can significantly improve classification performance, especially when labeled training data are sparse.

A related set of research uses labeled and unlabeled data in problem domains where the features naturally divide into two disjoint sets. Data sets whose features naturally partition into two sets, and algorithms that use this division, fall into the co-training setting (Blum & Mitchell, 1998). They present an algorithm for classifying web pages that builds two classifiers: one over the words that appear on the page, and another over the words appearing in hyperlinks pointing to that page. Blum and Mitchell (1998) also show that PAC-like guarantees on learning with labeled and unlabeled data hold under the assumptions that (1) each set of features is sufficient for classification, and (2) the two feature sets of each instance are conditionally independent given the class.

Most studies on text classification with Co-training type algorithms (Blum & Mitchell, 1998; Nigam & Ghani, 2000) have focused on small, often binary, problems and it is not clear whether their conclusions would generalize to real-world classification tasks with a large number of categories. Experimental evaluations of EM for text classification (Nigam et al., 2000) have also been conducted on data sets with a relatively small number of categories, with the largest number of classes being 20(for the 20-Newsgroups Data set). On the other hand, research in ensemble learning techniques has shown that Error-Correcting Output Codes (ECOC) are well suited for classification tasks with a large number of categories but only with labeled examples(Ghani, 2000).

In this paper, we present an approach for multiclass classification that combines labeled and unlabeled data. We decompose multiclass problems into multiple binary problems using error-correcting output codes and then use Co-Training to learn the individual binary classification problems. We show that our approach is especially useful for text classification problems involving a large number of categories and outperforms several other algorithms that are designed to combine labeled and unlabeled data. We also find that combining ECOC and Co-training results in a system that can provide a smooth tradeoff between recall and precision and can be used for high-precision classification.

2. Error Correcting Output Coding

Error Correcting Output Coding (ECOC) (Dietterich & Bakiri, 1995) has been shown to perform extraordinarily well for text classification (Berger, 1999; Ghani, 2000; ?). ECOC converts a m-class supervised learning problem into n binary supervised learning problems. Any learning algorithm that can handle two-class learning problems can then be applied to learn each of these n problems. The algorithm works by assigning each class a unique binary string of length n, referred to as codewords (Dietterich & Bakiri, 1995). Then we train n classifiers to predict each bit of the string. The predicted class is the one whose codeword is closest to the codeword produced by the classifiers. The distance metric we use in our experiments is the Hamming distance which counts the number of bits that the two codewords differ by. This process of mapping the output string to the nearest codeword is identical to the decoding step for error-correcting codes (Bose & Ray-Chaudhri, 1960; Hocuenghem, 1959).

- Training Phase
  1. Given a problem with m classes, create an m x n binary matrix M (where n can be less than m).[1]
  2. Each class is assigned one row of M (Each column divides the entire class space into two parts).
  3. Train the base classifier to learn the n binary functions (one for each column since each column divides the data set into two groups).
- Test Phase
  1. Apply each of the n single-bit classifiers to the test example.
  2. Combine the predictions to form a binary string of length n.
  3. Classify to the class with the nearest codeword (In this paper, we use hamming distance as the distance measure).

An interesting fact about using ECOC for classification is that an m-class problem can be decomposed into fewer than m binary problems. This results in a classification algorithm that scales up sublinearly with the number of classes and can be significantly more efficient than Naive Bayes or SVMs which only scale up linearly.

3. The Co-Training Setting

The co-training setting applies when a data set has a natural division of its features. For example, web pages can be described by either the text on the web page, or the text on hyperlinks pointing to the web page. Traditional algorithms that learn over these domains ignore this division and pool all features together. An algorithm that uses the co-training setting may learn separate classifiers over each of the feature sets, and combine their predictions to decrease classification error. Co-training algorithms using labeled and unlabeled data explicitly leverage this split during learning.

Blum and Mitchell (1998) formalize the co-training setting and provide theoretical learning guarantees subject to certain assumptions. In the formalization, each instance is described by two sets of features.

[1]More information about creating the matrix and experiments with various types of matrices/codes can be found in (Ghani, 2000; Ghani, 2001)

Blum and Mitchell (1998) prove under certain assumptions that co-training algorithms can learn from unlabeled data starting from only a weak predictor. The first assumption is that the instance distribution is compatible with the target function; that is, for most examples, the target functions over each feature set predict the same label. For example, in the web page domain, the class of the instance should be identifiable using either the hyperlink text or the page text alone. The second assumption is that the features in one set of an instance are conditionally independent of the features in the second set, given the class of the instance. This assumes that the words on a web page are not related to the words on its incoming hyperlinks, except through the class of the web page, a somewhat unrealistic assumption in practice.

They argue that a weak initial hypothesis over one feature set can be used to label instances. These instances seem randomly distributed to the other classifier (by the conditional independence assumption), but have classification noise from the weak hypothesis. Thus, an algorithm that can learn in the presence of classification noise will succeed at learning from these labeled instances.

4. Combining ECOC and Co-Training

We propose a new algorithm that aims at combining the advantages that ECOC offers for supervised classification with a large number of categories and that of Co-Training for combining labeled and unlabeled data. Since ECOC works by decomposing a multiclass problem into multiple binary problems, we can incorporate unlabeled data into this framework by learning each of these binary problems using Co-training.

The algorithm we propose is as follows:

- Training Phase
    1. Given a problem with m classes, create an m x n binary matrix M.
    2. Each class is assigned one row of M.
    3. Train n Co-trained classifiers to learn the n binary functions (one for each column since each column divides the data set into two groups) using both labeled and unlabeled data.
- Test Phase
    1. Apply each of the n single-bit Co-trained classifiers to the test example.
    2. Combine the predictions to form a binary string of length n.
    3. Classify to the class with the nearest codeword

Of course, an n-class problem can be decomposed naively into n binary problems and co-training can then be used to learn each binary problem, but our approach results in a more accurate and efficient classifier since by using ECOC we reduce the number of models that our classifier constructs and our approach scales up sublinearly with the number of classes (More details about using ECOC for efficient text classification using ECOC can be found in (Ghani, 2001). We also believe that our approach will perform better than the naive approach under the conditions that:

1. ECOC can outperform Naive Bayes on a multiclass problem (which actually learns one model for every class)
2. Co-Training can improve a single Naive Bayes classifier on a binary problem by using unlabeled data

The complication that arises from fulfilling condition 2 is that unlike normal binary classification problems where Co-Training has been shown to work well, our case involves binary problems in which each class consists of multiple classes. This is the case becuase the two classes in each binary problem are created artificially by ECOC by arbitrarily grouping together a number of original classes. Each "new" class consists of multiple "real" classes. In this scenario, there is no guarantee that the underlying classifier used by Co-Training will be able to learn these arbitrary binary functions.

Let's take a sample classification task consisting of classes C1 through C10 where one of the ECOC bits partitions the data such that classes C1 through C5 are in one class (B0) and C6 through C10 are in the other class (B1). The actual classes C1 through C10 contain different number of training examples and it is possible that the distribution is very skewed. If we pick our initial labeled examples randomly from the two classes B0 and B1, there is no guarantee that we will have at least one example from all of the original classes C1 through C10. If the set of labeled examples that Co-training is provided does notcontain at least one labeled example from one of the original classes, the underlying classifier will never be confident about labeling any unlabeled example from that class. Under the conditions that :

1. the initial labeled examples cover every "original" class,

2. the target function for the binary partition is learnable by the underlying classifier,

3. the two feature sets are "redundantly sufficient" so that the co-training algorithm can utilize unlabeled data,

theoretically, our combination of ECOC and Co-Training should result in improved performance by using unlabeled data.

5. Descriptions of Algorithms Used

We use Naive Bayes as the base classifier in our experiments to learn each of the binary problems in ECOC and also as the classifier within Co-Training. We also use Expectation-Maximization (EM) algorithm to compare with our proposed approach.

5.1 Naive Bayes

Naive Bayes is a simple but effective text classification algorithm for learning from labeled data alone (McCallum & Nigam, 1998; Lewis, 1998). We use the multinomial model as defined in (McCallum & Nigam, 1998) where each word in a document is assumed to be generated independently of the others given the class and use Laplace smoothing to calculate word probabilities.

5.2 Expectation-Maximization

We use the EM algorithm to compare with our approach when learning with both labeled and unlabeled data. It has been shown by Nigam et al. (2000) that this technique can significantly increase text classification accuracy when given limited amounts of labeled data and large amounts of unlabeled data. However, on data sets where the assumption correlating the classes with a single multinomial component is badly violated, basic EM performance suffers.

EM is an iterative statistical technique for maximum likelihood estimation in problems with incomplete data (Dempster et al., 1977). Given a model of data generation, and data with some missing values, EM will locally maximize the likelihood of the parameters and give estimates for the missing values. The naive Bayes generative model allows for the application of EM for parameter estimation. In our scenario, the class labels of the unlabeled data are treated as the missing values. We use the same implementation as used in (Nigam et al., 2000).

6. Datasets

To test our approach and compare it with other methods, we used two data sets that both reflect real-world classification problems and are obtained from the Web.

6.1 Hoovers Data set

This corpus of web pages was collected by the WebKB Group at CMU using the Hoovers Online Web resource (www.hoovers.com) by crawling 4285 companies on the web and examining just over 108,000 Web pages. The set of categories consists of 255 classes and label each company with the industry sector it belongs to. Each web-site is classified into one category only for each classification scheme. The most populous (majority) class contains 2% of the documents. This data set has previously been used to compare hypertext classification algorithms (Ghani et al., 2001). Since there is no natural feature split available in this data set, we randomly divide the vocabulary in two equal parts and apply Co-Training to the two feature sets. We have previously (Nigam & Ghani, 2000) shown that random partitioning can work reasonably well in the absence of a natural feature split for text classification problems. Note that this will only be true for data sets with sufficient redundncy where a random half of the feature set is sufficient to approximate the learning task.

6.2 Jobs Data set

We also use a data set obtained from WhizBang! Labs consisting of Job Titles and Job Descriptions organized in a two level hierarchy with 15 first level categories and 65 leaf categories. In all, there are 132,000 examples and each example consists of a Job Title and a corresponding Job Description. We consider the Job title and Job Description as two separate feature sets for Co-Training.

7. Experimental Results

All the codes used in the following experiments are BCH codes (31-bit codes for the Jobs data set and 63-bit codes for the Hoovers Data set) and are the same as those used in (Ghani, 2000). BCH codes have been commonly used in coding theory and are known to have good error-correcting properties.

7.1 Does Combining ECOC and Co-Training Work?

Table 1 shows the results of the experiments comparing our proposed algorithm with EM and Co-Training.

Table 1. Classification accuracies for the two data sets. Naive Bayes and ECOC do not use any unlabeled data. All the other algorithms have access to the same amount of data. We run EM until convergence and the Co-Training iterates until it runs out of unlabeled data.

Data set	Naive Bayes		ECOC		EM	Co-Training	ECOC + Co-Training
	10% Labeled	100% Labeled	10% Labeled	100% Labeled	10% Labeled	10% Labeled	10% Labeled
Jobs-65	50.1	68.2	59.3	71.2	58.2	54.1	64.5
Hoovers-255	15.2	32.0	24.8	36.5	9.1	10.2	27.6

The baseline results with Naive Bayes and ECOC using no unlabeled data are also given, as well as those when all the labels are known. The latter serve as an upper bound for the performance of our algorithm. The results labeled ECOC are using Naive Bayes as the binary classifier for each bit of the code.

From results reported in recent papers (Blum & Mitchell, 1998; Nigam & Ghani, 2000), it is not clear whether co-training will be able to learn effectively by using unlabeled data on a data set consisting of a large number of classes. The results in Table 1 show that Co-Training does not improve the classification accuracy by using unlabeled data on the Hoovers-255 data set; rather it has a negative effect and results in performance worse than that of Naive Bayes (which only uses the labeled examples). EM behaves similarly and is not able to improve classification accuracy by using unlabeled data.

On the other hand, our proposed combination of ECOC and Co-Training does indeed take advantage of the unlabeled data much better than EM and Co-Training and outperforms both of those algorithms on both data sets. It is also worth noting that ECOC outperforms Naive Bayes for both data sets and this is more pronounced when the number of labeled examples is small. This effect has also been previously observed in (Dietterich & Bakiri, 1995; ?)

We find that given the same amount of labeled data, our approach, with 64.5% accuracy, outperforms both Co-Training (54.1%) and EM (58.2%) on the Jobs-65 data set. This trend is even more pronounced for the Hoovers-255 where combining ECOC and Co-Training results in 27.6% accuracy compared to 10.2% and 9.1% accuracies for Co-Training and EM respectively.

Figure 1 shows the performance of our algorithm in terms of precision-recall tradeoff. Precision and Recall are both standard evaluation measures in text classification and Information Retrieval literature. As we can see from the figure, both Naive Bayes and EM are not very good at giving high-precision results. This is not surprising for Naive Bayes as for text classification problems, Naive Bayes gives very skewed probability estimates. Since the resulting classifier after learning with EM is also a Naive Bayes classifier, EM also inherits this deficiency and provides poor (and skewed) probabilities. Interestingly, Naive Bayes used within ECOC results in high-precision classification at reasonable levels of recall. [2]. This result is very encouraging and of enormous value in applications which require high-precision results such as search engines and hypertext classification systems.

8. Can algorithms other than Co-Training be used?

The framework to incorporate unlabeled data into ECOC requires an algorithm that is able to combine labeled and unlabeled data for binary classification problems. We use Co-Training to learn the individual binary functions as just one example. The next question that arises is whether Co-Training is especially suited to this framework or can other algorithms perform as well? In this section, we use two more algorithms that learn from labeled and unlabeled data, namely EM and Co-EM (which is a hybrid of EM and Co-Training) to replace Co-Training in the ECOC framework. First, we give a description of Co-EM, which is a hybrid of EM and Co-Training.

8.1 Co-EM

Co-EM is a hybrid algorithm, proposed by Nigam & Ghani (Nigam & Ghani, 2000), combining features from both co-training and EM. Co-EM is iterative, like EM , but uses the feature split present in the data, like Co-Training.Given a feature split with two feature sets A and B, it trains two classifiers (one for each feature set). It proceeds by initializing the A-feature-set naive Bayes classifier from the labeled data only. Then, A probabilistically labels all the unlabeled data. The B-feature-set classifier then trains using the labeled data

[2]More details can be found in (Ghani, 2001)

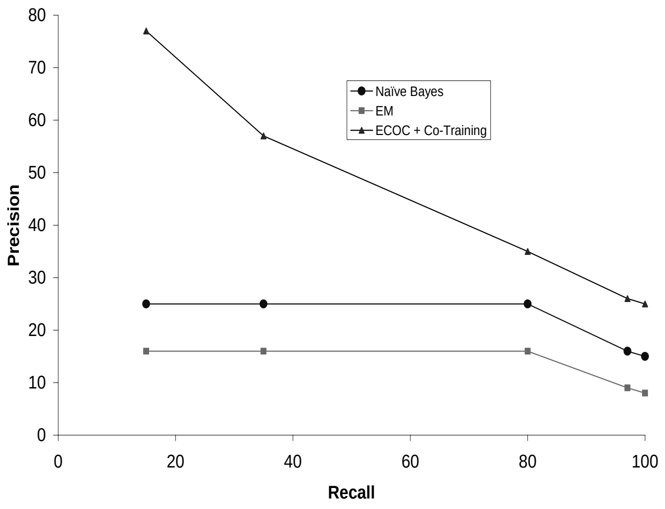

Figure 1. Precision Recall Graph for the Jobs-65 Data set

and the unlabeled data with A's labels. B then relabels the data for use by A, and this process iterates until the classifiers converge. A and B predictions are combined together as co-training embedded classifiers are. In practice, co-EM converges as quickly as EM does, and experimentally we run co-EM for 10 iterations. The co-EM algorithm can be thought of as a closer match to the theoretical argument of (Blum & Mitchell, 1998) than the co-training algorithm. The essence of their argument is that an initial A classifier can be used to generate a large sample of noisily-labeled data to train a B classifier. The co-EM algorithm does exactly this using one learner to assign labels to all the unlabeled data, from which the second classifier learns. In contrast, the co-training algorithm learns from only a single example at a time.

Muslea et al. (2001) extend the co-EM algorithm to incorporate active learning and show its robust behavior on a large spectrum of problems because of its ability to ask for the labels of the most ambiguous examples.

8.2 Results with EM and Co-EM

We repeat the experiments described in section 5, replacing Co-Training with EM and Co-EM. The implementations for these two algorithms are the same as used in (Nigam & Ghani, 2000). The results for these experiments are shown in Table 2.

Using either EM or Co-EM instead of Co-Training with ECOC results in worse performances for both data sets. This result leads us to ask why Co-Training is seemingly uniquely suited to work with ECOC? If ECOC decomposes a multiclass problem into multiple binary problems, why can't EM and Co-EM be used to learn these binary classification problems?

This performance can be explained by a closer analysis of the differences in the three algorithms. One possible explanation for EM's low accuracy is the underlying assumption that the data is generated by a mixture model and that there is one-to-one correspondence between mixture components and classes. When EM is trying to learn the binary classes created by ECOC, this assumption is severely violated. EM tries to model each class with one mixture component whereas in reality, each of these "artificial" classes is generated by a large number of classes contained in the original class space. This severely limits the modeling performance of EM and results in low classification accuracy.

Co-EM also has the same underlying assumption, but performs slightly better than EM because it utilizes the feature split available in the data. In previous work, we have shown that using the feature split can be extremely beneficial and results in improved performance.

Table 2. Classification accuracies for the two data sets comparing Co-Training, EM and Co-EM combined with ECOC.

Data set	ECOC Combined with		
	Co-Training	EM	Co-EM
Jobs-65	64.5	34.5	58.5
Hoovers-255	27.6	5.5	10.0

9. Discussion and Future Work

We noted while running our experiments that our approach was very sensitive to the initial documents that are provided as labeled examples. This leads us to believe that some form of active learning combined with this method to pick the initial documents should perform better than picking random documents.

As mentioned in Section 4, there is no guarantee that Co-Training can learn these arbitrary binary functions where the two classes are created artificially. If Co-Training does not have at least one labeled example from each of the original classes, it will never be confident about labeling any unlabeled example from that class. We ran some experiments using training examples that did not cover all 'original' classes and as expected, the results were much worse than the ones reported in the previous section where a certain number of examples were chosen initially from every "original" class.

One potential drawback of any approach using Co-Training type algorithms is the need for redundant and independent feature sets. In the experiments reported in this paper, we split our feature sets in a random fashion (for the Hoovers data set). In previous work (Nigam & Ghani, 2000), we have shown that random partitions of the feature set can result in reasonable performance and some preliminary work has also been done for developing algorithms that can partition a standard feature set into two redundantly sufficient feature sets. This would extend the applicability of our proposed approach to regular data sets.

Although the combination of ECOC and Co-Training does indeed improve classification accuracy using unlabeled data, there is a lot of room for improvement. In Table 1, the cells for Naive Bayes and ECOC with 100% labeled data give an upper bound on the classification accuracy if correct labels for all the unlabeled data were known. Our approach does not reach this level of performance and leaves room for improved algorithms based on this approach.

There are several other ways in which ECOC and Co-Training can be combined, e.g. training two ECOC classifiers on the two feature sets separately and combining them using Co-Training. Another approach to obtain better results would be explicitly trying to model the classes that are contained in every "binary" problem created by ECOC. One way would be to require the co-training algorithm to label and add at least n examples from each original class at every iteration. Another interesting area of work would be to relax the assumption of one-to-one correspondence between mixture components and classes (similar to (Nigam et al., 2000)) in EM and/or Co-EM and then use them instead of Co-Training.

10. Conclusions

The results described in this paper lead us to believe that the combination of ECOC and Co-Training algorithms is indeed useful for learning with labeled and unlabeled data. We have shown that our approach outperforms both Co-Training and EM algorithms, which have previously been shown to work well on several text classification tasks. Our approach not only performs well in terms of accuracy but also provides a smooth precision-recall tradeoff which is useful in applications requiring high-precision results. Our results also show that Co-Training can perform reasonably well when there is no natural split of the features present in the data set and a random split is created.

References

Berger, A. (1999). Error-correcting output coding for text classification. *IJCAI-99: Workshop on machine learning for information filtering.*

Blum, A., & Mitchell, T. (1998). Combining labeled and unlabeled data with co-training. *Proceedings of the 11th Annual Conference on Computational Learning Theory* (pp. 92–100).

Bose, R., & Ray-Chaudhri, D. (1960). On a class of error-correcting binary group codes. *Information andControl*, *3*, 68–69.

Dempster, A. P., Laird, N. M., & Rubin, D. B. (1977). Maximum likelihood from incomplete data via the

EM algorithm. *Journal of the Royal Statistical Society, Series B*, *39*, 1–38.

Dietterich, T., & Bakiri, G. (1995). Solving multiclass learning problems via error-correcting output codes. *Journal of Artificial Intelligence Research*, *2*, 263–286.

Ghani, R. (2000). Using error-correcting codes for text classification. *Proceedings of ICML-00, 17th International Conference on Machine Learning* (pp. 303–310). Stanford, US: Morgan Kaufmann Publishers, San Francisco, US.

Ghani, R. (2001). *Using error-correcting codes for efficient text classification with a large number of categories. masters thesis* (Technical Report). Center for Automated Learning and Discovery, Carnegie Mellon University.

Ghani, R., Slattery, S., & Yang, Y. (2001). Hypertext categorization using hyperlink patterns and meta data. *Proceedings of ICML-01, 18th International Conference on Machine Learning*. Williams College, US: Morgan Kaufmann Publishers, San Francisco, US.

Hocuenghem, A. (1959). Codes corecteurs d'erreurs. *Chiffres*, *2*, 147–156.

Jaakkola, T., & Haussler, D. (1999). Exploiting generative models in discriminative classifiers. *Advances in Neural Information Processing Systems 11* (pp. 487–493).

Joachims, T. (1999). Transductive inference for text classification using support vector machines. *Proceedings of ICML-99, 16th International Conference on Machine Learning* (pp. 200–209). Bled, SL: Morgan Kaufmann Publishers, San Francisco, US.

Lewis, D. D. (1998). Naive (Bayes) at forty: The independence assumption in information retrieval. *Machine Learning: ECML-98, Tenth European Conference on Machine Learning* (pp. 4–15).

McCallum, A., & Nigam, K. (1998). A comparison of event models for naive bayes text classification. *AAAI-98 Workshop on Learning for Text Categorization*.

Muslea, I., Minton, S., & Knoblock, C. A. (2001). Selective sampling + semi-supervised learning = robust multi-view learning. *IJCAI-01 Workshop on Text Learning: Beyond Supervision*.

Nigam, K., & Ghani, R. (2000). Analyzing the applicability and effectiveness of co-training. *Proceedings of CIKM-00, 9th ACM International Conference on Information and Knowledge Management* (pp. 86–93). McLean, US: ACM Press, New York, US.

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