# Using LSI for Text Classification in the Presence of Background Text

Sarah Zelikovitz
Rutgers University
110 Frelinghuysen Road
Piscataway, NJ 08855
zelikovi@cs.rutgers.edu

Haym Hirsh
Rutgers University
110 Frelinghuysen Road
Piscataway, NJ 08855
hirsh@cs.rutgers.edu

## ABSTRACT

This paper presents work that uses Latent Semantic Indexing (LSI) for text classification. However, in addition to relying on labeled training data, we improve classification accuracy by also using unlabeled data and other forms of available "background" text in the classification process. Rather than performing LSI's singular value decomposition (SVD) process solely on the training data, we instead use an expanded term-by-document matrix that includes both the labeled data as well as any available and relevant background text. We report the performance of this approach on data sets both with and without the inclusion of the background text, and compare our work to other efforts that can incorporate unlabeled data and other background text in the classification process.

## 1. INTRODUCTION

The task of classifying textual data is both difficult and intensively studied [11, 16, 14]. Traditional machine learning programs use a training corpus of often hand-labeled data to classify new test examples. Training sets are sometimes extremely small, due to the difficult and tedious nature of labeling, and decisions can therefore be difficult to make with high confidence.

Given the huge number of unlabeled examples, articles, Web sites, and other sources of information that often exist, it would be useful to take advantage of this additional information in some automatic fashion. These sources can be looked at as background knowledge that can aid in the classification task. For example, a number of researchers have explored the use of large corpora of unlabeled data to augment smaller amounts of labeled data for classification (such as augmenting a collection of labeled Web-page titles with large amounts of unlabeled Web-page titles obtained directly from the Web). Nigam et al. [14] use such background examples by first labeling them using a classifier formed on the original labeled data, adding them to the training data to learn a new classifier on the resulting expanded data and then repeating anew the labeling of the originally unlabeled data. This approach yielded classification results that exceed those obtained without the extra unlabeled data.

Blum and Mitchell's [3] co-training algorithm also applies to cases where there is a source of unlabeled data [13], only in cases where the target concept can be described in two redundant ways (such as through two different subsets of attributes describing each example). Each view of the data is used to create a predictor, and each predictor is used to classify unlabeled data. The data labeled by one classifier can then be used to train the other classifier. Blum and Mitchell prove that under certain conditions, the use of unlabeled examples in this way is sufficient to PAC-learn a concept given only an initial weak learner.

A second example of background knowledge concerns cases where the data to which the learned classifier will be applied is available at the start of learning. For such learning problems, called *transductive learning* [12], these unlabeled examples may also prove helpful in improving the results of learning. For example, in transductive support vector machines [12, 1] the hyperplane that is chosen by the learner is based on both the labeled training data and the unlabeled test data. Joachims shows that this is a method for incorporating priors in the text classification process and performs well on such tasks.

Zelikovitz and Hirsh [18] consider an even broader range of background text for use in classification, where the background text is hopefully relevant to the text classification domain, but doesn't necessarily take the same general form of the training data. For example, a classification task given labeled Web-page titles might have access to large amounts of Web-page contents. Rather than viewing these as items to be classified or otherwise manipulated as if they were unlabeled examples, the pieces of background knowledge are used as indices into the set of labeled training examples. If a piece of background knowledge is close to both a training example and a test example, then the training example is considered close to the test example, even if they do not share any words. The background text provides a mechanism by which a similarity metric is defined, and nearest neighbor classification methods can be used. Zelikovitz and Hirsh [18] show that their approach is especially useful in cases with small amounts of training data and when each item in the data has few words.

This paper describes yet another way of using this broader range of background knowledge to aid classification. It neither classifies the background knowledge, nor does it directly compare it to any training or test examples. Instead, it exploits the fact that knowing that certain words often co-occur may be helpful in learning, and that this could be discovered from large collections of text in the domain.

To do this we use Latent Semantic Indexing (LSI) [6]. LSI is an automatic method that re-describes textual data in a new smaller semantic space. LSI assumes that there exists some inherent semantic structure between documents and terms in a corpus. The new space that is created by LSI places documents that appear related in close proximity to each other. LSI is believed to be especially useful in combating polysemy (one word can have different meanings) and synonymy (different words are used to describe the same concept), which can make classification tasks more difficult. The key idea here is to use the background text in the creation of this new re-description of the data, rather than relying solely on the training data to do so.

In the next section we give a brief review of LSI, and describe how we use it for traditional text classification as well as for classification in the presence of background text. We then present and describe the results of the system on four different data sets, comparing those results to other systems that incorporate unlabeled data. We conclude with a discussion of our current and ongoing work in this area.

## 2. OUR APPROACH

### 2.1 Latent Semantic Indexing

Latent Semantic Indexing [8] is based upon the assumption that there is an underlying semantic structure in textual data, and that the relationship between terms and documents can be re-described in this semantic structure form. Textual documents are represented as vectors in a vector space. Each position in a vector represents a term (typically a word), with the value of a position *i* equal to 0 if the term does not appear in the document, and having a positive value otherwise. Based upon previous research we represent the positive values as the log of the total frequency in that document [7] weighted by the entropy of the term. As a result, the corpus can be looked at as a large term-by-document ($t \times d$) matrix $X$, with each position $x_{ij}$ corresponding to the presence or absence of a term (a row $i$) in a document (a column $j$). This matrix is typically very sparse, as most documents contain only a small percentage of the total number of terms seen in the full collection of documents.

Unfortunately, in this very large space many documents that are related to each other semantically might not share any words and thus appear very distant, and occasionally documents that are not related to each other might share common words and thus appear to be closer. This is due to the nature of text, where the same concept can be represented by many different words, and words can have ambiguous meanings. LSI reduces this large space to one that hopefully captures the true relationships between documents.

The singular value decomposition (SVD) of the $t \times d$ matrix, $X$, is the product of three matrices: $TSD^T$, where $T$ and $D$ are the matrices of the left and right singular vectors and $S$ is the diagonal matrix of singular values. The diagonal elements of $S$ are ordered by magnitude, and therefore these matrices can be simplified by setting the smallest $k$ values in $S$ to zero.[1] The columns of $T$ and $D$ that correspond to the values of $S$ that were set to zero are deleted. The new product of these simplified three matrices is a matrix $\hat{X}$ that is an approximation of the term-by-document matrix. This new matrix represents the original relationships as a set of orthogonal factors. We can think of these factors as combining meanings of different terms and documents; documents are then re-expressed using these factors.

When LSI is used for retrieval, a query is represented in the same new small space that the document collection is represented in. This is done by multiplying the transpose of the term vector of the query with matrices $T$ and $S^{-1}$. Once the query is represented this way, the distance between the query and documents can be computed using the cosine metric, which represents a numerical similarity measurement between documents. LSI returns the distance between the query and all documents in the collection. Those documents that have higher cosine distance value than some cutoff point can be returned as relevant to the query.

### 2.2 LSI for Text Classification

We are using LSI for text *classification*, so we can henceforth refer to the document collection as the training examples and the query as a test example.[2] Given that a single example to be labeled may be judged by this method as similar to a range of different training examples, it is necessary to decide how the different examples "vote" on the label of the new example. To do this we take an approach used by Cohen and Hirsh [4]. We can look at the result of the LSI query as a table containing the tuples

$$\langle \textit{train-example}, \textit{train-class}, \textit{cosine-distance} \rangle$$

with one line in the table per document in the training collection. There are many lines in the table with the same *train-class* value that must be combined to arrive at one score for each class. Cohen and Hirsh [4] use the noisy-or operation to combine the similarity values that are returned by LSI to arrive at one single value per class. If the cosine values for documents of a given class are $\{s_1, \ldots, s_n\}$, the final score for that class is $1 - \prod_{i=1}^{n}(1 - s_i)$. Whichever class has the highest score is returned as the answer to the classification task. Based upon [17, 4] only the thirty closest neighbors are kept and combined.

### 2.3 Incorporating Background Knowledge

The power of LSI lies in the fact that it can place documents that do not share any words in close proximity to each other. However, when there is little data LSI can suffer drastically. With few training examples, there are many terms that occur only once, hence limiting the power of LSI to create a space that reflects interesting properties of the data.

What is most interesting to us about the singular value decomposition transformation is that it does not deal with the classes of the training examples at all. This gives us an extremely flexible learner, for which the addition of background knowledge is quite easy. Instead of simply creating the term-by-document matrix from the training examples alone, we combine the training examples with other sources of knowledge to create a much larger term-by-"document" matrix, $X_n$.

LSI is run on this new term-by-document matrix to obtain $\hat{X_n}$. $\hat{X_n}$ is a model of the space that was unobtainable with the training examples alone. The larger matrix contains words that did not occur in the training examples at all; it also provides us with richer and

[1]The choice of the parameter $k$ can be very important. Previous work has shown that a small number of factors (100-300) often achieves effective results. We discuss this further in our section on current work.

[2]This is in contrast to other uses of LSI for classification [10, 8, 9], in which one centroid vector is formed for each class, and a new example is labeled by those classes whose vector is sufficiently close to it.

more reliable patterns for data in the given domain. To classify a test example incorporating the background knowledge in the decision process, the test example is re-described in the new space and then compared only to the columns of $\hat{X}_n$ that correspond to the original training examples. The scores that are obtained from this comparison are combined with the noisy-or operation, to return a final class for classification. Clearly, it is important for the background knowledge to be similar in content to the original training set that it is combined with. If the background knowledge is totally unrelated to the training corpus, for example, LSI might successfully model the background knowledge, but the features would be unrelated to the actual classification task.

To give a concrete example of how LSI with background can help, we can look at one test example in the NetVet domain [4]. The training and test examples are titles of Web pages from http://netvet.wustl.-edu, and each piece of background knowledge consists of the first 100 words of the contents of Web pages that are not in the training or test set. The training data in the example below consists of 277 documents. Removing all terms that occur only once creates a $t \times d$ matrix with 109 terms. With the added 1158 entries in the background knowledge the matrix grows to $4694 \times 1435$.

For the test example

> british mule

of class *horse* the three closest training document returned were:

> livestock nutrient manag univers
> manag of the foal mare purdu univers
> avitech exot

which are of class *cow*, *horse*, and *bird*. (In this example stemming [15] is used to find the morphological roots of the words in the documents). Since LSI creates a totally new space it is not unusual to find, as in this sample, that none of the original words from the test example are found in the three closest training examples. This test example is misclassified by LSI without background knowledge. This is not surprising since the word *mule* in the test example does not occur in the training examples at all. With the addition of the background knowledge, the three closest training examples returned are:

> british columbia cattlemen
> donkei
> sicilian donkei preserv

of classes *cow*, *horse*, and *horse*. The correct class is returned. Notice that two of the closest training examples have the word donkei which is related to both *mule* and *horse*. The addition of the background knowledge allowed the learner to find this association.

## 3. EMPIRICAL RESULTS

To evaluate our approach we use four datasets previously used by work on text classification in the presence of background text [4, 5, 12, 14, 18]. We first describe the data sets, and then the results that we obtained.

### 3.1 Data Sets

**Technical papers.** One common text categorization task is assigning discipline or sub-discipline names to technical papers. We created a data set from the physics papers archive (http://xxx.lanl.gov), where we downloaded the *titles* for all technical papers in two areas in physics (astrophysics, condensed matter) for the month of March 1999 [18]. As background knowledge we downloaded the *abstracts* of all papers in these same areas from the two previous months — January and February 1999. In total there were 1530 pieces of knowledge in the background set, and 953 in the training-test set combined. Since we performed five-fold cross validation, for each run 80% of these 953 examples were used fro training and 20% were held for testing. The 1530 background knowledge abstracts were downloaded without their labels (i.e., without knowledge of what sub-discipline they were from) so that our learning program had no access to them.

**Web page titles.** As discussed above, the NetVet site (http://netvet-.wustl.edu) includes the Web page headings for pages concerning cows, horses, cats, dogs, rodents, birds and primates [4]. Each of these titles had a URL that linked the title to its associated Web page. For the labeled corpus, we chose half of these titles with their labels, in total 1789 examples. Once again, five-fold cross-validation was used on this half of the titles to divide it into training and test sets. We discarded the other half of the titles, with their labels, and simply kept the URL to the associated Web page. We used these URLs to download the first 100 words from each of these pages, to be placed into a corpus for background knowledge. Those URLs that were not reachable were ignored by the program that created the background knowledge database.

**WebKB.** The WebKB dataset [5] contains a collection of Web pages from computer science departments. As in [14, 12] we use only those of the categories student, faculty, course, and project. For this data set the background knowledge is simply unlabeled examples. Using information-gain as the criterion, only the top 300 words were kept. This value was used to be consistent with the data sets used in [14]; it was optimized for their EM code, and is not clear that it was the best value to use for LSI. Four test sets, from four different universities were used, and training was performed on pages from the other three universities, and results that are reported are averages across all these sets. Our divisions of the data into training and test sets are identical to that of [14], with the test data set, depending on the specific university, ranging from 225 to 307 examples and the training sets ranging from only four examples, one per class, to two hundred examples, fifty per class. The size of the background text remains steady at 2500 examples.

**20 Newsgroups.** The 20 Newsgroups data set consists of articles from 20 different newsgroups. The latest 4000 articles are used for testing, and a random 10000 are used for the background text. As in the WebKB data, training and test set divisions are the same as in [14]. Results that are reported are averages across ten runs, and, as described shortly, the training set sizes that we report range from twenty examples, with one per class, to four hundred examples, with twenty per class.

### 3.2 Results

#### *3.2.1 The Utility of Background Knowledge with LSI*

We obtained the Latent Semantic Indexing Package from Telcordia Technologies, Inc. (http://lsi.research.telcordia.com/) and all results are with use of this LSI package. We report the classification accuracy for text classification using LSI both with and without background knowledge for the physics data in Figure 1 and for the NetVet data in Figure 2. We label LSI with background knowledge as LSI-bg. In each case we report error rates as we vary the number of training examples given to the learner. Each point represents an average of five cross-validated runs. For each cross-validated run, four-fifths of the data is used as the training set and one-fifth is used as the test set. Holding this test set steady, the number of examples in the training set was varied. Each data set was tested with both LSI and LSI-bg using 20, 40, 60, 80, and 100 percent of the data [18]. For both of these domains the incorporation of background knowledge aided the classification task for training sets of all sizes. In each case the reduction of error increased as the training size decreased. Also, although accuracy for both LSI and LSI-bg decreased as the training set size decreased, the accuracy when using LSI-bg changed very little, as can be seen by the flatness of the lines. This leads us to believe that the utility of background knowledge is that it compensates for the limited training data.

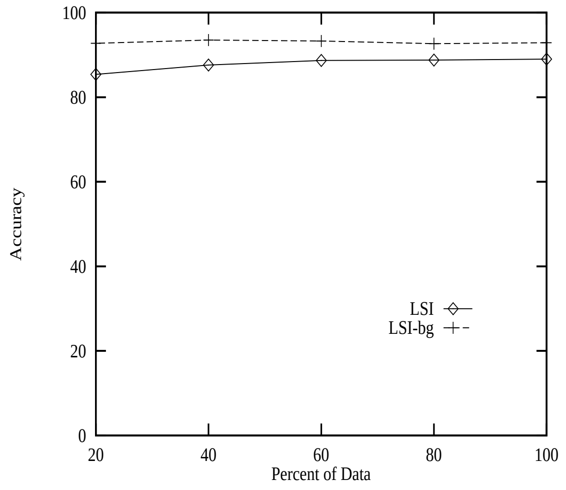

**Figure 1: LSI and LSI-bg for the two class paper title problem**

Figure 3 presents the results on the WebKB classification task. We report the classification accuracy of nine different size training sets ranging from 4 examples (1 per class) to 200 examples (50 per class). Each accuracy number is an average across multiple runs, ranging from 7 to 10, depending upon training set size. The horizontal axis represents this on a log scale. In this domain LSI-bg only outperforms LSI on small training sets. As training class size grows, LSI-bg degrades and actually hurts learning. Since unlabeled examples are used as background knowledge, coming from the same distribution as the training examples, we are not quite sure of why this is so. It would seem that the model of the data should be more accurate with LSI-bg than with LSI. This is a question that we are currently exploring in our ongoing work.

The results for the 20 Newsgroups data are graphed in Figure 4. The horizontal axis is once again a log scale, and results are graphed for training set size varying from 20 (1 per class) to 400 (20 per class). Each of these numbers are averages across ten unique runs. Even with much larger training sets (150 per class) the addition of background knowledge does not cause degradation in accuracy. However, once again, the unlabeled examples are most useful with small training sets. Interestingly, on this data set LSI without the addition of the unlabeled examples performed *extremely* poorly.

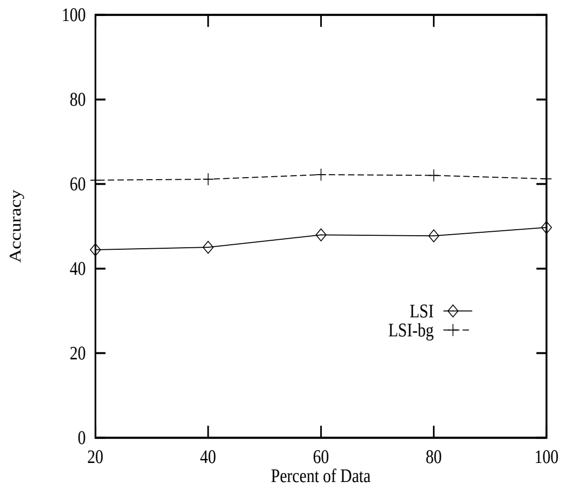

**Figure 2: LSI and LSI-bg for the NetVet problem**

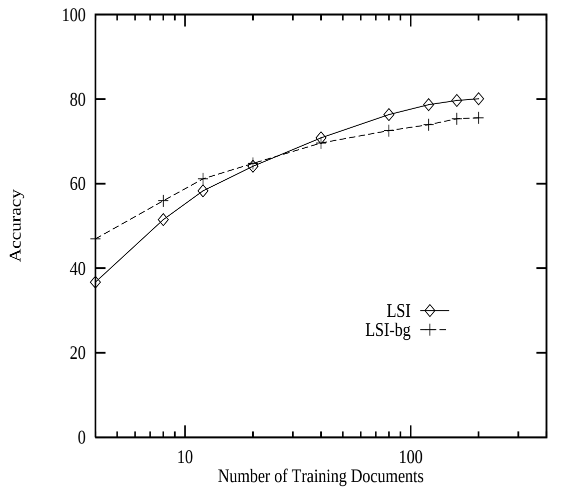

**Figure 3: LSI and LSI-bg for WebKB four class problem**

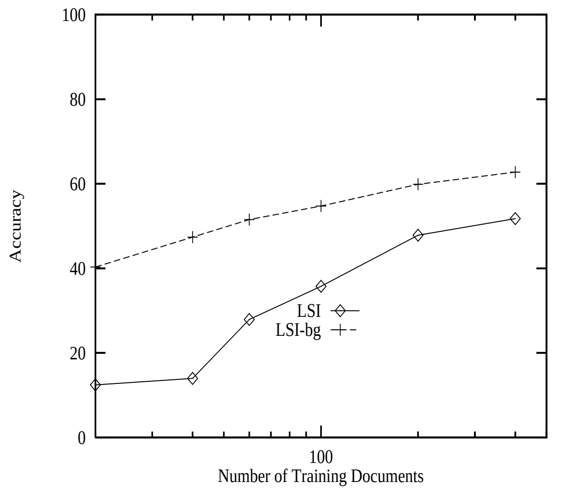

**Figure 4: LSI and LSI-bg for the 20 Newsgroups problem**

**Table 1: Accuracy rates on physics data**

| Percent of Data | EM | LSI-bg |
|---|---|---|
| 20 | **95.49** | 92.8 |
| 40 | **95.49** | 93.5 |
| 60 | **95.8** | 93.3 |
| 80 | **96** | 92.7 |
| 100 | **96.3** | 92.9 |

**Table 2: Accuracy rates on NetVet data**

| Percent of Data | EM | LSI-bg |
|---|---|---|
| 20 | 49.58 | **60.90** |
| 40 | 56.12 | **61.12** |
| 60 | 57.86 | **62.24** |
| 80 | 59.48 | **62.02** |
| 100 | **61.43** | 61.23 |

### *3.2.2 Comparison of LSI and EM*

One successful method for incorporating unlabeled data in the classification task is the Expectation Maximization (EM) approach [14]. Using naive Bayes, a classifier is trained with only the labeled training examples. This classifier is in turn used to probabilistically label the unlabeled examples; these newly labeled examples are then used to retrain the classifier, and obtain more accurate parameters for the learner. This process iterates until it converges. Although in some of the problems that we present, the background text is not really of the same form as the data (such as Web-page titles for data and Web-page contents for background), methods such as EM can still be applied, since they treat every text item as a "bag" of the terms that occur in the item. We therefore present results that compare the accuracy of our LSI approach for using background text to the approach for using naive Bayes and EM to learn from labeled and unlabeled data [14]. We used the rainbow package (http://www.cs-.cmu.edu/~mccallum/bow/rainbow/) to run EM on both the physics set and the NetVet data. EM was run with 7 iterations and since this number of iterations was not maximized for these data sets we report the highest results out of the seven iterations.[3] Although this skews the results slightly in favor of EM, we can still get a fair picture of the comparative values of these programs. These results can be seen in Table 1 and Table 2. The results reported on WebKB and 20 Newsgroups in Table 3 and Table 4 were obtained directly from [14]. On all tables the highest accuracy rate is shown in bold.

To summarize the results in the tables, on small data sets in both the NetVet domain and the 20 Newsgroups domain, LSI-bg outperforms EM. As more training examples are added to the these problems LSI-bg does not perform as well. This has been a phenomenon that has occurred across all data sets, and is a focus of our current work. On the physics data, EM is far superior in all cases. It is not surprising that EM does so well on the physics data. Zelikovitz and Hirsh [18] showed that in this domain the simple process of labeling the background text using the training data and then adding the resulting data to the training data, without any further processing, gets extremely high accuracy as well. Although the background knowledge is not of the same type as the training examples (paper abstracts versus paper titles), they are from the same source, and abstract and titles overlap in many words.

[3]We chose the number 7 based on discussions with Nigam (personal communication).

**Table 3: Accuracy rates on WebKB**

| Training Documents | EM | LSI-bg |
|---|---|---|
| 4 | **55.14** | 46.91 |
| 8 | **56.99** | 55.93 |
| 12 | **62.03** | 61.12 |
| 20 | **67.22** | 64.77 |
| 40 | **74.58** | 69.60 |
| 80 | **76.39** | 72.55 |
| 120 | **79.14** | 73.93 |
| 160 | **78.73** | 75.32 |
| 200 | **78.8** | 75.56 |

**Table 4: Accuracy rates on 20 Newsgroups**

| Training Documents | EM | LSI-bg |
|---|---|---|
| 20 | 35.34 | **40.29** |
| 40 | 43.08 | **47.39** |
| 60 | 49.23 | **51.55** |
| 100 | **55.4** | 54.75 |
| 200 | **62.96** | 59.86 |
| 400 | **68.21** | 62.78 |

## 4. FINAL REMARKS

We have presented a method for incorporating background knowledge in text classification using LSI. The singular value decomposition is performed on a term-by-document matrix that includes both the training examples and background knowledge. This allows test examples to be compared to the training examples in a new space that reflects patterns in the text in the domain that may not be found when confronted solely with training data. We have shown empirically that this increases the accuracy rates in classification on a range of benchmark problems used in previous work.

There are many different dimensions along which a variety of choices may be made to explore the use of LSI with background knowledge. Our empirical results have shown what other researchers have already discovered: background knowledge is most useful when the training set is small. However there are a number of open questions, both in the exploration of the use of background knowledge and in the use of LSI for this purpose.

For example, we are currently studying issues on how the choice of the number of dimensions to use with SVD affects the usefulness of unlabeled data. Some preliminary results show that running SVD with many factors limits the usefulness of background knowledge. For example, without background knowledge, if we use 300 factors instead of 100 factors on 20% of the NetVet data, accuracy rises from 50% to 55%. However, accuracy for LSI-bg essentially remains the same. This same phenomenon is observable in larger and other data sets as well. We are doing further studies in this area.

Another issue relevant to LSI specifically is that if the training set is small compared to the background text, it may be sufficient to use only the background text, without training data, in $X_n$. SVD could be performed on the background text alone; both the training and test examples can be re-described in terms of the new space using $\hat{X}_n$. This method of updating SVDs by "folding-in" new documents, has been studied by the LSI community [2]. Although this SVD will not be identical to the one of the training and background examples combined, our initial tests have shown that classification accuracy does not significantly change. This might then provide a mechanism by which incremental learning is achievable — where a new example can be added without requiring a new SVD calculation. In this paper we have primarily focused on the information value of background text using LSI, without concern for the potential run-times involved. Such a method for avoiding the often costly SVD calculation would be one way to manage the otherwise costly work that would be necessary in obtaining new training data in incremental learning scenarios.

Finally, the nature and type of background knowledge that is used to improve learning is of central interest to us. The data sets that we used had background knowledge of different types. Are unlabeled examples more helpful than background knowledge that comes from a different source? For unlabeled examples the size of each piece of background knowledge is generally well-defined (being similar to that of the training and test data), but for other sources of data this is an open issue. The "cleanliness" of background text can also vary greatly, from encyclopedia entries at one end of the spectrum to ad hoc collections obtained from uncoordinated Web sites or text obtained through speech recognition technology. These are some of the topics we are currently exploring.

## 5. ACKNOWLEDGMENTS

We would like to thank Kamal Nigam for his help with rainbow and the WebKB and 20 Newsgroups data sets. We would also like to thank Sue Dumais and Michael Littman for helpful discussions and comments. Portions of this work were supported by the Binational Science Foundation, NASA, and the New Jersey Commission on Science and Technology.

## 6. REFERENCES

[1] K. Bennet and A. Demiriz. Semi-supervised support vector machines. *Advances in Neural Information Processing Systems*, 12:368–374, 1998.

[2] M. Berry, S. Dumais, and G. W. O'Brien. Using linear algebra for intelligent information retrieval. *SIAM Review*, 37(4):573–595, 1995.

[3] A. Blum and T. Mitchell. Combining labeled and unlabeled data with co-training. In *Proceedings of the 11th Annual Conference on Computational Learning Theory*, pages 92–100, 1998.

[4] W. Cohen and H. Hirsh. Joins that generalize: Text categorization using WHIRL. In *Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining*, pages 169–173, 1998.

[5] M. Craven, D. DiPasquo, D. Freitag, A. McCallum, T. Mitchell, K. Nigam, and S. Slattery. Learning to extract symbolic knowledge from the World Wide Web. In *Proceedings of the 15th National Conference on Artificial Intelligence (AAAI-98) and of the 10th Conference on Innovative Applications of Artificial Intelligence (IAAI-98)*, pages 509–516, 1998.

[6] S. Deerwester, S. Dumais, G. Furnas, and T. Landauer. Indexing by latent semantic analysis. *Journal for the American Society for Information Science*, 41(6):391–407, 1990.

[7] S. Dumais. LSI meets TREC: A status report. In D. Hartman, editor, *The first Text REtrieval Conference: NIST special publication 500-215*, pages 105–116, 1993.

[8] S. Dumais. Latent semantic indexing (LSI): TREC-3 report. In D. Hartman, editor, *The Third Text REtrieval Conference, NIST special publication 500-225*, pages 219–230, 1995.

[9] S. Dumais. Combining evidence for effective information filtering. In *AAAI Spring Symposium on Machine Learning and Information Retrieval, Tech Report SS-96-07*, 1996.

[10] P. W. Foltz and S. Dumais. Personalized information delivery: An analysis of information filtering methods. *Communications of the ACM*, 35(12):51–60, 1992.

[11] T. Joachims. Text categorization with support vector machines: Learning with many relevant features. In *Machine Learning: ECML-98, Tenth European Conference on Machine Learning*, pages 137–142, 1998.

[12] T. Joachims. Transductive inference for text classification using support vector machines. In *Proceedings of the Sixteenth International Conference on Machine Learning*, pages 200–209, 1999.

[13] K. Nigam and R. Ghani. Analyzing the effectiveness and applicability of co-training. In *Proceedings of the Ninth International Conference on Information and Knowledge Management*, 2000.

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

[15] M. F. Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, 1980.

[16] F. Sebastiani. Machine learning in automated text categorization. *Technical Report IEI-B4-31-1999*, 1999.

[17] Y. Yang and C. Chute. An example-based mapping method for text classification and retrieval. *ACM Transactions on Information Systems*, 12, 1994.

[18] S. Zelikovitz and H. Hirsh. Improving short text classification using unlabeled background knowledge to assess document similarity. In *Proceedings of the Seventeenth International Conference on Machine Learning*, pages 1183–1190, 2000.