Improving Text Classification with LSI Using Background Knowledge

Sarah Zelikovitz and Haym Hirsh
zelikovi,hirsh@cs.rutgers.edu

Abstract

We present work in progress that uses Latent Semantic Indexing (LSI) in conjunction with background knowledge and unlabeled examples to improve text classification accuracy. The singular value decomposition (SVD) that is performed by LSI is done on an expanded term by document matrix that includes the labeled training examples as well as the unlabeled examples. We report classification accuracy on different data sets both with and without the inclusion of background knowledge and compare it to other known work.

1 Introduction

The task of classifying textual data is both difficult and intensively studied [Cohen and Hirsh, 1998; Joachims, 1998; Sebastiani, 1999; Nigam *et al.*, 2000]. Applications of various machine learning techniques that attempt to solve this problem include categorization of Web pages into sub-categories for search engines, and classification of news articles by subject. 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 based on known examples 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 exists on the Web and elsewhere, it would be extremely useful to take advantage of these additional corpora, in some automatic fashion. These sources can be looked at as background knowledge that can aid in the classification task. The background knowledge can be of various forms, all of which have been used to improve classification accuracy. Unlabeled examples, which are often abundant and easily available, are a type of background knowledge. Even the test examples themselves can be looked at as background knowledge if they are used to aid in the classification task. More generally, background knowledge is any unlabeled collection of text from any source that is related to the classification task.

Although background knowledge cannot directly help in the classification task easily, a wealth of information can be obtained from the background knowledge that can be used by the classification system. For example, words that co-occur often can be learned from the unlabeled data. In this way, associations between terms that may not be captured by the training examples alone can be learned and used.

Various researchers in machine learning and text-classification have realized that unlabeled examples might be useful. Sources external to the labeled training data have been used in many different ways, and in conjunction with many different classifiers to reduce the error rates in classification tasks. Nigam et al. [2000] have explored this using Expectation Maximization (EM) and a naive Bayes classifier. The parameters of the naive Bayes classifier are set using labeled examples. The learned model is then used by EM to probabilistically classify unlabeled documents, with the resulting collection of classified documents used to estimate a new set of parameters for naive Bayes. The EM algorithm iterates until there is no change in the naive Bayes parameters. Nigam et al. present a number of experimental results that show that error rates can be reduced significantly using unlabeled examples in this way. Other related algorithms are described by [McCallum and Nigam, 1999; Jones *et al.*,1999].

Zelikovitz and Hirsh's [2000] approach uses various sources of background knowledge in a different way. Instead of classifying these pieces of data, 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. In this way, the background set of data provides a mechanism by which the labeled examples are chosen to be used for classification of a new test example.

Blum and Mitchell's [1998] co-training algorithm also uses unlabeled data to improve learning [Nigam and Ghani, 2000]. Their algorithm applies to problems where the target concept can be described in two redundantly sufficient 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 which are then used to further train the other learner. 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. Lewis and his colleagues [Lewis and Gale, 1994; Lewis and Catlett, 1994] also make use of unlabeled data in learning, but focus on asking for labels from a human labeler for only a modest subset of the data, those whose class membership is most undecided given the result of learning on the data that has been labeled thus far.

Most recently there has been work in transductive support vector machines [Joachims, 1999; Bennet and Demiriz, 1998] that uses the test set to choose a decision function that will classify that particular test set correctly. The margins of the hyperplane that is chosen by the learner is based on both the training and test data. Joachims shows that this is a method of incorporating priors in the text classification process and is particularly well suited to the task.

This paper describes yet another way of using background knowledge to aid classification. It neither classifies the background knowledge; nor does it directly compare the it to a new test example. Instead, our method creates a model of the domain by incorporating the words, and frequencies of words, that co-occur in the large unlabeled corpus. This new model is used to represent both the training examples and the test examples, and test examples are then compared to the training examples in this new space.

To create this model of the data, we use Latent Semantic Indexing (LSI) [Deerwester *et al*, 1990]. LSI is an automatic method that redescribes 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 especially useful in combating polysemy (one word can have different meanings) and synonomy (different words are used to describe the same concept), which can make classification tasks more difficult.

In the next section we give a brief review of LSI, and describe how we use it for classification using both data and background knowledge. 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 the possible choices that we took in our research, and various directions for current and future work.

2 LSI

2.1 LSI for Text Classification

Latent Semantic Indexing [Dumais *et al.*, 1995] 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 redescribed in this semantic structure form. Textual documents are represented as vectors in a vector space. Each position in a vector represents a term, 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 [Dumais, 1993] weighted by the entropy or noise. The corpus can therefore be looked at as a large term-by-document ($t \times d$) matrix. This matrix is very sparse, as most documents contain only a small percentage of the total number of terms in the matrix.

This $t \times d$ matrix represents the relationships between terms and documents. However, in this very large space, many documents that are related to each other semantically might not share any words, and occasionally documents that are not related to each other might share common words. 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, where T and D are the matrices of the left and right singular values 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. The columns of T and and the rows of 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 reexpressed using these factors.

The choice of the parameter k is very important. In general, previous work in LSI has shown that a small number of factors is sufficient (usually between 100 and 300) although this is not small enough to visualize in a geometric sense. In our section on current work we discuss the classification accuracy using a range of values for k.

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 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 the statistical similarity 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 are returned as relevant to the query. *Precision/recall* curves are easily obtained by choosing various cutoff distance values, and for each cutoff returning the documents whose cosine distance is closer than that value. For each cutoff value, the percent of those documents above the cutoff that are relevant can be plotted on the *precision* axis, and the percent of relevant documents that are above the cutoff can be plotted on the *recall* axis.

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. As we wish to compare this system to others, we needed a method to combine the similarity metrics across the many examples that are returned by LSI to arrive at one single classification per test example. For multi-class problems, 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. For this we borrowed from [Cohen and Hirsh, 1998] the noisy-or operation which can be used 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 [Yang and Chute, 1994; Cohen and Hirsh, 1998] only the thirty closest neighbors in the newest space are kept and combined. With these results we also report the class that would be chosen if the single closest nearest neighbor (i.e. the class of the document with the largest score) is returned.

2.2 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 new factor 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. In most cases, when the set of training data is small, *both* dimensions of this new $t \times d$ matrix are substantially larger than the original matrix.

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 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.

To give a concrete example of how LSI with background can help, we can look at one test example in the NetVet domain [Cohen and Hirsh, 1998]. The training and test examples are titles of Web pages from http://netvet.wustl.edu, and each piece of background knowledge consists 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. Running LSI creates a $t \times d$ matrix with 109 terms. With the added 1158 entries in the background knowledge the matrix grows to 4694×1435.

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 is used [Porter, 1980; Cohen and Hirsh, 1998]). 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

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 the two areas in physics (astrophysics, condensed matter) for the month of March 1999 [Zelikovitz and Hirsh, 2000]. 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. These 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 its pages concerning cows, horses, cats, dogs, rodents, birds and primates. [Cohen and Hirsh, 1998] used this set to determine usefulness of their WHIRL nearest neighbor classification algorithm. 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. 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 [Craven *et al.*, 1998] contains a collection of web pages from computer science departments. As in [Nigam *et al.*, 2000; Joachims, 1999] 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 [Nigam *et al.*, 2000]; 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 is identical to that of [Nigam *et al.*, 2000].

20 Newsgroups The 20 Newsgroups data set, collected by Ken Lang 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 [Nigam *et al*, 2000].

3.2 Results

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 [Zelikovitz and Hirsh, 2000]. 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 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 10 to 7, 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 the unlabeled examples are used for background knowledge and come from the same dis-

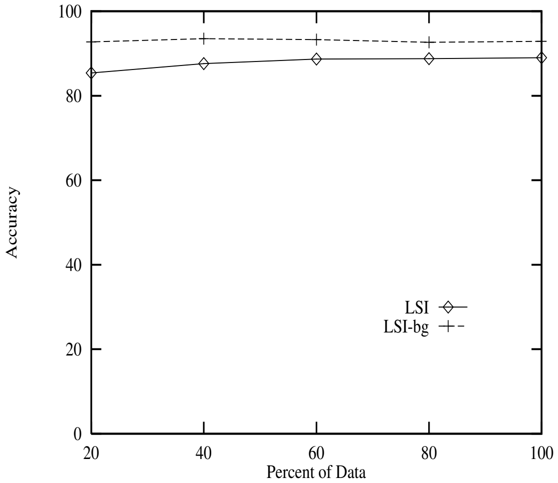

Figure 1: LSI and LSI-bg for the two class paper title problem

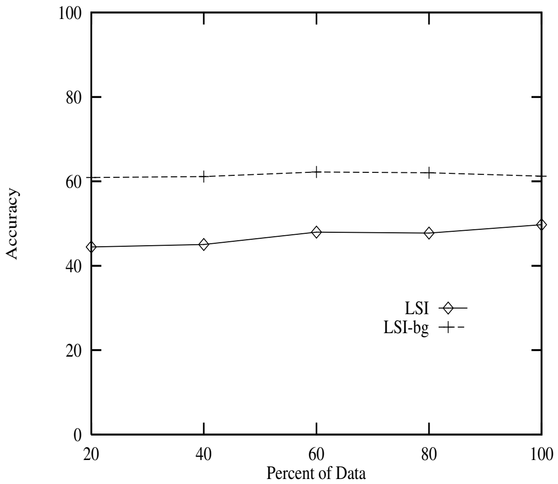

Figure 2: LSI and LSI-bg for the NetVet problem

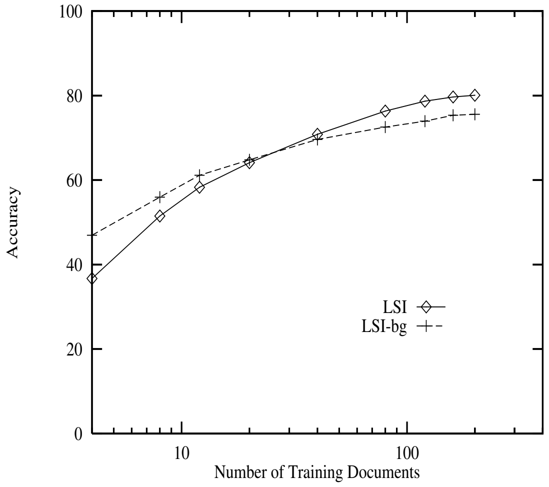

Figure 3: LSI and LSI-bg for WebKB four class problem

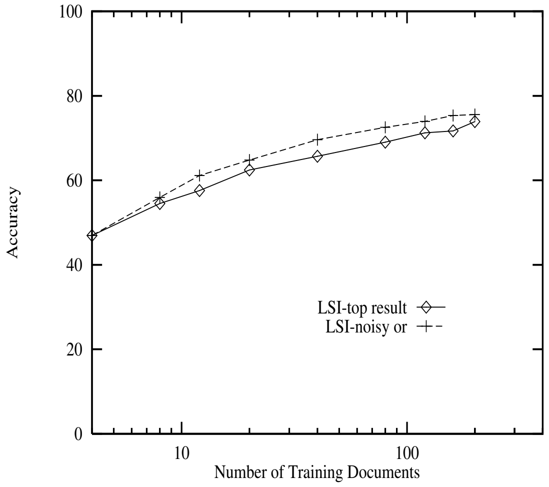

Figure 4: LSI-bg with top result and LSI-bg with noisy-or combination rule for WebKB four class problem

tribution 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. However, since the comparison for the final classification takes place only with the original training examples, if there are enough training examples to model the space moderately well, the training examples are better represented in the space that is created without any additional background knowledge. This is an area that we are currently exploring.

The noisy-or operation that we use to combine the various scores that are returned by LSI helps reduce error slightly, above choosing the highest ranking neighbor. In Figure 4 we show these results for the WebKB data set. This same pattern is true across all four data sets that are described in this paper.

The results for the 20 Newsgroups data are graphed in Figure 5. The horizontal axis is once again a log scale, and results are graphed for training set size varying from 20 (1 per class)

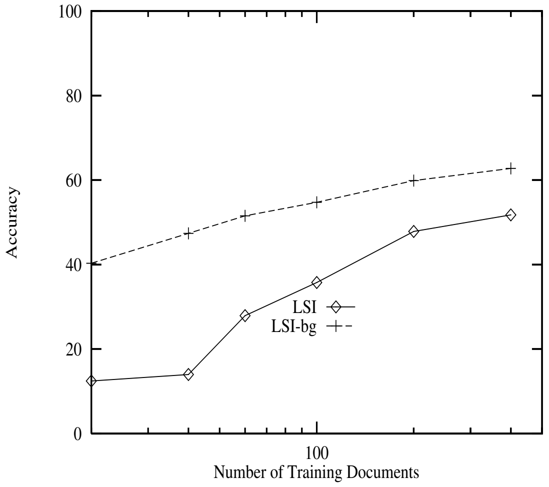

Figure 5: LSI and LSI-bg for the 20 Newsgroups problem

Table 1: Accuracy rates on physics data

Percent of Data	NB	EM	LSI-bg
20	87.62	**95.49**	92.8
40	91.39	**95.49**	93.5
60	92.86	**95.8**	93.3
80	93.49	**96**	92.7
100	94.33	**96.3**	92.9

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.

Comparison of LSI and EM

To place the graphs described above in context with other research in this area, we present tables that list the accuracy results on the same sets of data using naive Bayes and EM [Nigam *et al*]. We used the rainbow package (http://www.cs.cmu.edu/~mccallum/bow/rainbow/) to run naive Bayes and 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. 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 [Nigam *et al*]. 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. On larger data sets 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

Table 2: Accuracy rates on NetVet data

Percent of Data	Naive Bayes	EM	LSI-bg
20	51.49	49.58	**60.90**
40	55.67	56.12	**61.12**
60	58.30	57.86	**62.24**
80	59.14	59.48	**62.02**
100	59.75	**61.43**	61.23

Table 3: Accuracy rates on WebKB

Training Documents	Naive Bayes	EM	LSI-bg
4	39.62	**55.14**	46.91
8	47.86	**56.99**	55.93
12	53.13	**62.03**	61.12
20	60.31	**67.22**	64.77
40	69.12	**74.58**	69.60
80	75.99	**76.39**	72.55
120	78.51	**79.14**	73.93
160	**79.36**	78.73	75.32
200	**80.77**	78.8	75.56

the physics data, EM is far superior. It is not surprising that EM does so well on the physics data. An algorithm that uses WHIRL [Cohen, 1997] to classify the background text in just one iteration, and simply augments that training set with this new data, and then classifies the test instances using WHIRL as well gets an extremely high accuracy as well. Although the background knowledge is not of the same type as the training examples, they are from the same source, and abstract and titles overlap in many words.

4 Final Remarks

We have presented a method of incorporating background knowledge with LSI to aid in text classification. 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 this new space. We have shown empirically that this reduces the error rates in classification in a variety of problems.

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.

Table 4: Accuracy rates on 20 Newsgroups

Training Documents	Naive Bayes	EM	LSI-bg
20	20.31	35.34	**40.29**
40	26.77	43.08	**47.39**
60	30.60	49.23	**51.55**
100	37.05	**55.4**	54.75
200	46.30	**62.96**	59.86
400	55.48	**68.21**	62.78

The nature and type of background knowledge that is used to improve learning is an interesting topic for study. 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 well defined, but for other sources of data this is an open issue. For LSI, if the background knowledge is divided into smaller chunks, word co-occurrences change and the SVD will be different. These are issues that we are exploring.

We are also currently studying issues specific to LSI with background knowledge. One important question is: How does the choice of the number of factors used when reducing the SVD matrix affect 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%. 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.

An interesting point to note is that if the training set is small compared to the background text, it does not need to be included in X_n. Instead SVD can be performed on the background test alone; both the training and test examples can be re-described in terms of the new space using $\hat{X}_n$. Although this SVD will not be identical to the one of the training and background examples combined, empirical test have shown that classification accuracy does not significantly change. This provides a mechanism by which new known training examples can be compared to a test example in the context of background knowledge without having to re-run LSI.

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 Michael Littman for helpful discussions on LSI.

References

[Bennet and Demiriz, 1998] K. Bennet and A. Demiriz. Semi-Supervised Support Vector Machines *Advances in Neural Information Processing Systems*, 12, M. S. Kearns, S. A. Solla, D. A. Cohn, editors, MIT Press, Cambridge, MA, pp 368-374, 1998.

[Blum and Mitchell, 1998] A. Blum and T. Mitchell. Combining labeled and unlabeled data with co-training. *Proceedings of the Eleventh Annual Conference on Computational Learning Theory* (pp. 92–100). New York: ACM Press, 1998.

[Cohen, 1998a] W. Cohen. Integration on heterogeneous databases without common domains using queries based on textual similarity. *Proceedings of the 1998 ACM-SIGMOD International Conference on the Management of Data* (pp. 201-212). New York: ACM Press, 1998.

[Cohen, 1998b] W. Cohen. A web-based information system that reasons with the structured collections of text. *Proceedings of the Second International ACM Conference on Autonomous Agents* (pp. 400-407). New York: ACM Press, 1998.

[Cohen and Hirsh, 1998] W. Cohen and H. Hirsh. Joins that generalize: Text categorization using WHIRL. *Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining* (pp. 169-173). Menlo Park, California: AAAI Press, 1998.

[Craven *et al*, 1998] Mark Craven, Dan DiPasquo, Dayne Freitag, Andrew McCallum, Tom Mitchell, Kamal Nigam and Sean Slattery. Learning to Extract Symbolic Knowledge from the World Wide Web. *Proceedings of the 15th National Conference on Artificial Intelligence (AAAI-98) and of the 10th Conference on Innovative Applications of Artificial Intelligence (IAAI-98)*, pp. 509-516, AAAI Press, 1998.

[Deerwester *et al*, 1990] S. Deerwester, S. Dumais, G. Furnas and T. Landauer. Indexing by Latent Semantic Analysis. *Journal of the American Society for Information Science*, 41(6), pp. 391-407, 1990.

[Dumais, 1991] S. Dumais. Improving the retrieval of information from external sources. *Behavior Research Methods, Instruments, & Computers* 23(2) pp. 229-236, 1991.

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

[Dumais, 1995] S. Dumais. Using LSI for information filtering: TREC-3 experiments. In: D. Harman (Ed.), *The Third Text REtrieval Conference (TREC3) National Institute of Standards and Technology Special Publication* , 1995.

[Joachims, 1998] T. Joachims. Text categorization with support vector machines: learning with many relevant features. *Proceedings of the Tenth European Conference on Machine Learning* (pp. 137–142). Berlin: Springer, 1998.

[Joachims, 1999] T. Joachims. Transductive inference for text classification using support vector machines. *Proceedings of the Sixteenth International Conference on Machine Learning* (pp. 200-209). San Francisco: Morgan Kaufmann, 1999.

[Jones *et al*, 1999] R. Jones, A. McCallum, K. Nigam, and E. Riloff, E. (1999). Bootstrapping for text learning tasks. *Working Notes of the IJCAI-99 Workshop on Text Mining: Foundations, Techniques and Applications* (pp. 52–63), 1999.

[Lewis and Catlett, 1994] D. Lewis and J. Catlett. Heterogeneous uncertainty sampling for supervised learning. *Proceedings of the Eleventh International Conference on Machine Learning* (pp. 148–156). San Francisco: Morgan Kaufmann, 1994.

[Lewis and Gale, 1994] D. D. Lewis and W. Gale. A sequential algorithm for training text classifiers. *Proceedings of the Seventeenth Annual International ACM-SIGIR Conference on Research Development in Information Retrieval* (pp. 3–12). New York: ACM Press, 1994.

[McCallum and Nigam, 1999] A. McCallum and K. Nigam. Text classification by bootstrapping with keywords, EM and shrinkage. *Working Notes of ACL 1999 Workshop for the Unsupervised Learning in Natural Language Processing* (pp. 52-58), 1999.

[Mitchell, 1999] T. Mitchell. The role of unlabeled data in supervised learning. *Proceedings of the Sixth International Colloquium on Cognitive Science*, 1999.

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

[Nigam and Ghani, 2000] Kamal Nigam and Rayid Ghani. Analyzing the Effectiveness and Applicability of Co-training. *Proceedings of the Ninth International Conference on Information and Knowledge Management*, 2000.

[Porter, 1980] M. F. Porter. An algorithm for suffix stripping. *Program,* 14, 130-137, 1980.

[Salton 1989] G. Salton Automatic text processing. Reading, MA: Addison-Welsley, 1989.

[Sebastiani, 1999] Fabrizio Sebastiani. Machine Learning in Automated Text Categorisation. Technical Report IEI-B4-31-1999, Istituto di Elaborazione dell'Informazione, Consiglio Nazionale delle Ricerche, Pisa, IT, 1999.

[Yang and Chute, 1994] Y. Yang and C. Chute. An example-based mapping method for text classification and retrieval. *ACM Transactions on Information Systems*, 12, 252-277, 1994.

[Zelikovitz and Hirsh, 2000] S. Zelikovitz and H. Hirsh. Improving Short Text Classification Using Unlabeled Background Knowledge to Assess Document Similarity. *Proceedings of the Seventeenth International Conference on Machine Learning* pp. 1183-1190, San Francisco: Morgan Kaufmann Publishers, 2000.