Linear Discriminant Analysis in Document Classification

Kari Torkkola
Motorola Labs, 7700 South River Parkway, MD ML28, Tempe AZ 85284, USA
email: kari.torkkola@motorola.com, http://members.home.net/torkkola

Abstract

Document representation using the bag-of-words approach may require bringing the dimensionality of the representation down in order to be able to make effective use of various statistical classification methods. Latent Semantic Indexing (LSI) is one such method that is based on eigendecomposition of the covariance of the document-term matrix. Another often used approach is to select a small number of most important features out of the whole set according to some relevant criterion. This paper points out that LSI ignores discrimination while concentrating on representation. Furthermore, selection methods fail to produce a feature set that jointly optimizes class discrimination. As a remedy, we suggest supervised linear discriminative transforms, and report good classification results applying these to the Reuters-21578 database.

1 Introduction

Document classification denotes assigning an unknown document to one of predefined classes. This is a straightforward concept from supervised pattern recognition or machine learning. It implies [1)] the existence of a labeled training data set, [2)] a way to represent the documents, and [3)] a statistical classifier trained using the chosen representation of the training set.

Some classifiers are very sensitive to the representation, for example, failing to generalize to unseen data (overfitting) if the representation contains irrelevant information [2]. It would thus be advantageous to be able to extract only information pertinent to classification. However, some classifiers, such as Support Vector Machines [16], tolerate better irrelevant information. Either case, in general, it is computationally cheaper to operate the classifier in low dimensional spaces. If this can be done without sacrificing the accuracy of the classifier, the better.

We present a fairly straightforward application of Linear Discriminant Analysis (LDA) to document classification, when vector space document representations are employed. LDA is a well known method in statistical pattern recognition literature, to learn a *discriminative* transformation matrix from the original high-dimensional space to a desired dimensionality [10]. The idea is to project the documents into a low dimensional space in which the classes are well separated. This can also be viewed as extracting features that only carry information pertinent to the classification task. A subsequent classification task should then become easier.

This paper proceeds as follows. We discuss document representation methods and approaches to reduce their dimensionality, especially Latent Semantic Indexing [7]. We discuss why LSI cannot result in optimal representation for document classification. Methods to select a number of relevant features, as well as their shortcomings are then discussed. We introduce LDA, and we present how it can be applied to very high dimensional data. We describe classification experiments applying a

support vector machine classifier to the Reuters-21578 database, and we discuss the computational complexity of the approach.

The point of view of this paper is that of statistical pattern recognition. This means that we approach the problem as a classification task with exclusive classes aiming to minimize the classification error rate. Performance is evaluated by assigning a single label to each unknown document. This label can be either correct or incorrect, and the error rate is defined as the number of incorrectly assigned labels divided by the total number of documents in the test set. This is not the usual case in information retrieval where documents may carry several labels (or topics). Given a document collection, the aim is to retrieve all documents relevant to a particular topic or class. Performance is measured as precision and recall [14], that cannot be easily related to the classification error rate. Thus, in the last section of the paper we also discuss possible extensions to information retrieval.

2 Vector space document representations

The dominant vectorial document representation is based on the so called bag-of-words approach, in which each document is essentially represented as a histogram of terms, usually divided by the number of terms of the document to normalize for different document lengths.

What are the terms that are counted in the histograms? Terms (words) that occur in every document obviously do not convey much useful information for classification. Same applies to rare terms that are found only in one or two documents. These, as well as common stop words, are usually filtered out of the corpus. Furthermore the words may be stemmed. These operations leave a term dictionary that can range in size from thousands to tens of thousands. Correspondingly, this is the dimension of the space in which documents now are represented as vectors. Although the dimension may be high, a characteristic of this representation is that the vectors are sparse.

For conventional classification methods this dimensionality may be too high. Thus dimension reduction methods are called for. Two possibilities exist, either selecting a subset of the original features, or transforming the features into new ones, that is, computing new features as some functions of the old ones. We examine each in turn.

3 Feature selection

Optimal feature selection coupled with a pattern recognition system leads to a combinatorial problem since all combinations of available features need to be evaluated, by actually training and evaluating a classifier. This is called the *wrapper* configuration [20, 22]. Obviously wrapper strategy does not allow to learn feature transforms, because all possible transforms cannot be enumerated.

Another approach is to evaluate some criterion related to the final classification error that would reflect the "importance" of a feature or a number of features jointly. This is called the *filter* configuration in feature selection [20, 22]. What would be an optimal criterion for this purpose? Such a criterion would naturally reflect the classification error rate. Approximations to the Bayes error rate can be used, based on Bhattacharyya bound or an interclass divergence criterion. However, these joint criteria are usually accompanied by a parametric estimation, such as Gaussian, of the multivariate densities at hand [12, 26], and are characterized by heavy computational demands.

In document classification problems, the dominant approach has been sequential greedy selection using various criteria [30, 4, 23]. This is dictated by the sheer dimensionality of the document-term representation. However, greedy algorithms based on sequential feature selection using any criterion are suboptimal because they fail to find a feature set that would *jointly* optimize the criterion. For example, two features might both be very highly ranked by the criterion, but they may carry the same information about class discrimination, and are thus redundant.

Thus, feature selection through any joint criteria such as the actual classification error, leads to a combinatorial explosion in computation. For this very reason finding a *transform* to lower dimensions might be easier than selecting features, given an appropriate objective function.

4 Latent Semantic Indexing

One well known dimension reducing transform is the principal component analysis (PCA), also called Karhunen-Loeve transform. PCA seeks to optimally *represent* the data in a lower dimensional space in the mean squared error sense. The transform is derived from the eigenvectors corresponding to the largest eigenvalues of the covariance matrix of training data.

In the information retrieval community this method has been named Latent Semantic Indexing (or LSI) [7]. The covariance matrix of data in PCA corresponds now to the document-term matrix multiplied by its transpose. Entries in the covariance matrix represent co-occurring terms in the documents. Eigenvectors of this matrix corresponding to the dominant eigenvalues are now directions related to dominant combinations of terms occurring in the corpus ("topics", "semantic concepts"). A transformation matrix constructed from these eigenvectors projects a document onto these "latent semantic concepts", and the new low dimensional representation consists of the magnitudes of these projections. The eigenanalysis can be computed efficiently by a sparse variant of singular value decomposition of the document-term matrix [1].

Although LSI has been proven to be extremely useful in various information retrieval tasks, it is not an optimal representation for classification. LSI/PCA are completely unsupervised, that is, they pay no attention to the class labels of the existing training data. LSI aims at optimal *representation* of the original data in the lower dimensional space in the mean squared error sense. This representation has nothing to do with the optimal *discrimination* of the document classes.

Independent component analysis (ICA) has also been proposed as a tool to find "interesting" projections of the data [11, 29, 19]. Girolami et al. maximize negentropy to find a subspace on which the data has the least Gaussian projection [11]. The criterion corresponds to finding a clustered structure in the data, and bears a close relationship to projection pursuit methods [9]. This appears to be a very useful tool revealing non-Gaussian structures in the data. However, as PCA, the method is completely unsupervised with regard to the class labels of the data, and it is not able to enhance class separability.

Thus, *supervised* feature extraction schemes are called for. We describe one such method, Linear Discriminant Analysis.

5 Linear Discriminant Analysis

The term linear discriminant analysis (LDA) refers to two distinct but related methods. The first is classifier design. Given a number of variables as the data representation, each class is modeled as Gaussian (with a covariance matrix and a mean vector). Observations are now classified to the class of the nearest mean vector according to Mahalanobis distance. The decision surfaces between classes become linear if the classes have a shared covariance matrix. In this case the decision surfaces are called Fisher discriminants, and the procedure of constructing them is called Linear Discriminant Analysis [10, 2].

The second use of the term LDA refers to a discriminative feature transform that is optimal for certain cases [10]. This is what we denote by LDA throughout this paper. In the basic formulation, LDA finds eigenvectors of matrix $\mathbf{T} = \mathbf{S}_w^{-1}\mathbf{S}_b$. Here $\mathbf{S}_b$ is the between-class covariance matrix, that is, the covariance matrix of class means. $\mathbf{S}_w$ denotes the within-class covariance matrix, that is equal to the sum of covariance matrices computed for each class separately. $\mathbf{S}_w^{-1}$ captures the compactness of each class, and $\mathbf{S}_b$ represents the separation of the class means. Thus $\mathbf{T}$ captures both. The eigenvectors corresponding to largest eigenvalues of $\mathbf{T}$ form the rows of the transform matrix $\mathbf{W}$, and new discriminative features $\mathbf{y}$ are derived from the original ones $\mathbf{x}$ simply by $\mathbf{y} = \mathbf{W}\mathbf{x}$.

The relation to LDA as a clasifier design is that these eigenvectors span the same space as directions orthogonal to the decision surfaces of Fisher discriminants.

The straightforward algebraic way of deriving the LDA transform matrix is both a strength and a weakness of the method. Since LDA makes use of only second-order statistical information, covariances, it is optimal for data where each class has a unimodal Gaussian density with well

separated means and similar covariances. Large deviations from these assumptions may result in sub-optimal features. Also the maximum rank of $\mathbf{S}_b$ in this formulation is $N_c - 1$, where N_c is the number of different classes. Thus basic LDA cannot produce more than $N_c - 1$ features. This is, however, simple to remedy by projecting the data onto a subspace orthogonal to the computed eigenvectors, and repeating the LDA analysis in this space [24].

Further extensions to LDA exist. For example, Heteroscedastic Discriminant Analysis (HDA) allows the classes have different covariances [21]. However, simple linear algebra is no longer sufficient to compute the solution. One must resort to iterative optimization methods. Same applies to methods that further relax the Gaussianity assumptions of the classes [26, 28].

The following section discusses why LDA and the assumptions behind it are well suited for document-term data, and how LDA can be made feasible with high-dimensional data.

6 Linear Discriminant Analysis for document-term data

To our knowledge, LDA feature transforms have not been applied earlier to document classification tasks, although LDA has been used before in the sense of designing a linear classifier [15, 27]. In contrast, we suggest LDA as a means of deriving efficient features, which can be classified by any, possibly nonlinear, classifier.

If LDA is such a well known and well-behaved method why is it not in wider use in the document analysis community? LDA is, of course, only applicable when labeled training data exists, and this represents only a part of possible document analysis tasks. Unsupervised methods, such as clustering are thus excluded. However, nothing prevents applying LDA *after* clustering to find features that best separate the clusters, then repeating the clustering using new features, and iterating this a few times.

One high barrier in applying LDA directly to document-term data of tens of thousands dimensions is the following. Unlike the document-term matrix, the criterion matrix $\mathbf{T}$ in LDA is no longer sparse. Thus efficient methods for inversion, SVD, or eigenanalysis of sparse matrices cannot be used [1].

Luckily, a simple remedy exists. Random projections have been shown to be very useful in various dimension reduction tasks where the source data has had extremely high dimensionality [3, 25, 6]. Random projections tend to Gaussianize the data, since each resulting component is a sum of a large number of original components with random weights. In this respect, data after a random projection conforms well to the assumptions behind LDA. A straightforward method is thus to generate a random matrix with, say, normally distributed entries, to transform the original dimension down by an order of magnitude, thereafter followed by a normal LDA transformation with one more (or two) order(s) of magnitude further dimension reduction.

Another option is to perform a conventional LSI-based dimension reduction by an order of magnitude starting from the original document-term matrix. Since this is a sparse matrix, it is a feasible task. This can again be followed by LDA.

7 Document classification experiments

Experiments were performed using the Reuters-21578 database [2]. We used the "ModLewis" split of the database into training and testing parts. Since the aim is classification, in which each document has an exclusive category, we discarded documents with no label or with multiple labels. Furthermore, those rare classes were discarded that did not occur at least once in both training and testing set. The resulting training set has 6535 documents, and the test set 2570 documents with 52 document classes.

[1] For example, if the number of terms $D = 20.000$, $\mathbf{T}$ requires memory of 3.2GB. Since inversion of a matrix requires computation $O(D^3)$, only the inversion of $\mathbf{S}_w$ would take approximately 12 hours of CPU time on a 750MHz machine (on which the inversion of a 1000x1000 matrix takes 5.6 seconds).

[2] http://www.research.att.com/ lewis/reuters21578.html

Granted, from the information retrieval standpoint this is an artificial task, which makes it difficult to compare our results to those of others. However, this facilitated a straightforward application of statistical pattern classification tools to the task. Possible improvements in class discrimination within this classification task are very likely to carry over to a retrieval task.

The term dictionary was constructed by discarding stop words, too frequent words, too infrequent words, and words shorter than three letters. Porter stemmer was used. This resulted in a term dictionary of 5718 terms. Normalized term histograms of this dimension were thus produced from the corpus.

As the classifier, we used a Support Vector Machine implementation called SVMTorch[3], which is well suited for large scale and/or sparse problems [5]. In all experiments we used a Gaussian kernel ($width = 10$), and $C = 100$ as the trade-off between training error and margin. A binary classifier was trained for each of the 52 classes by using each class at a time as positive examples with the rest of the data as negative examples. The output of an SVM is positive when it decides that an unknown test vector belongs to the class it was trained for. Since there may be several simultaneous such claims, and since the setting is exclusive classification, a simple maximum selector is applied to the SVMs to make the final decision for an exclusive class. The LSI analysis was done using SVDPAKC/las2[4] [1].

Results are depicted in a single chart in Fig. 1. The horizontal axis denotes the dimension of the document representation. Instead of precision/recall we are reporting just a single number, the error rate, which is common in pattern recognition literature. This is defined as the number of misclassified documents divided by the total number of documents. This would translate to one minus micro-averaged recall in the information retrieval terms. Calculating corresponding precision requires computation of the full confusion matrices, which the used SVM package does not directly do. The error rate is represented by the vertical axis of the chart. Naturally, the classifier is trained using only the training set, and the error rate is evaluated using only the testing set.

The single blue diamond represents the error rate using the original 5718-dimensional normalized term histograms as the document representation (11.2%). Results on features generated using completely random transform matrices to various output dimensions are plotted as black triangles. Each point is an average of results from five random trials. Variance is high (not plotted) at low dimensions depending whether and how pertinent information happened to be included in the features. Results with LSI are plotted as magenta squares ranging from an order of magnitude dimension reduction (513) down to one. LSI with dimension 513 is very close to the original error rate (11.4%), but deteriorates approximately logarithmically: halving the dimension increases the error rate by 4-5 percentage points, which resembles the behavior of a random projection [3], and shows that LSI does not really provide good features for discrimination.

In contrast, LDA exhibits much lower error rates (yellow triangles). The starting point of LDA was the 513-dimensional representation produced by LSI, which was transformed down to 1-64 dimensions. It is remarkable that a document representation of only twelve features achieves an error rate as low as 8.9%. The optimal dimension appears to be somewhat higher. With dimension 64 we achieved an error rate of 7.8%. LDA computed from a 513-dimensional random projection behaved very similarly (black crosses). These figures are again averages of five random trials. LSI appears to be able to provide somewhat more pertinent information to the 513-dimensional data for classification purposes, than a mere random transformation matrix, which LDA takes advantage of.

A transform to a lower dimension can only retain or reduce information of the original data. If an SVM was completely immune to irrelevant information, the error rate should only increase as the dimension is reduced. As lower error rates were achieved, this is not obviously the case. Furthermore, this difference should be much more pronounced when classifiers, such as neural networks or decision trees are used that are more susceptible to overfitting than SVMs.

[3] http://www.idiap.ch/learning/SVMTorch.html

[4] http://www.netlib.org/svdpack/

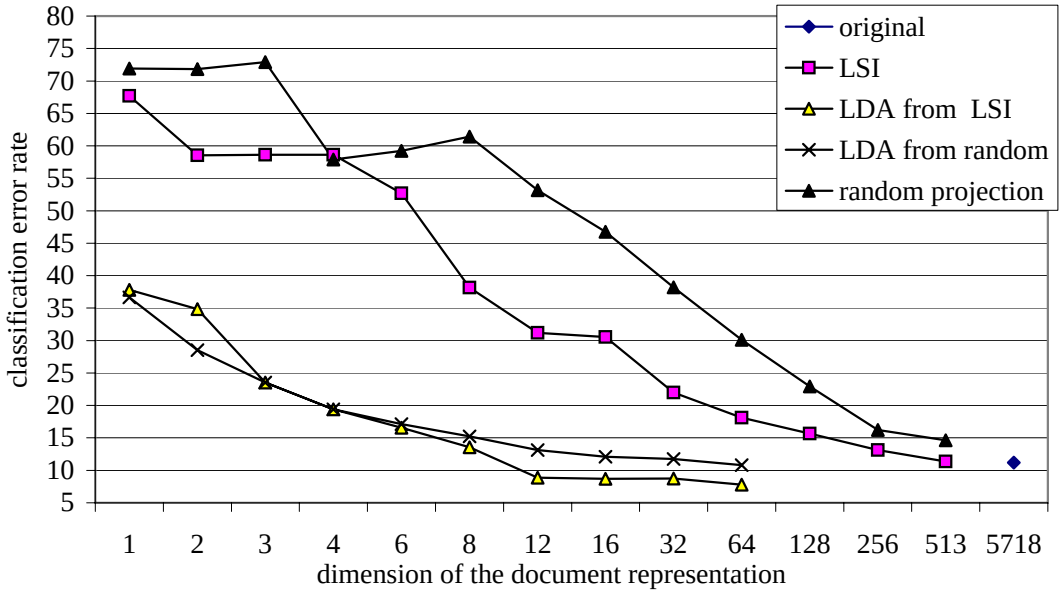

Figure 1: Classification error rates on Reuters-21578 test data using LDA, LSI, and random projections.

Figure 2 depicts an example projection of the training data onto a discriminative subspace of dimension two (which resulted in an error rate of 34.9%). Figure also lists all the categories present in this experiment. We can see that the two largest categories, "acq" and "earn" are not really well separated, although both have quite compact clusters. In this projection, the class distributions appear to be Gaussian-like, with unequal covariances, though. Thus there might be hope that projections based on HDA, or joint maximum mutual information [28], might provide even better results.

8 Discussion

Computational Complexity. With an SVM, the classification process consists of evaluating the kernel between an unknown vector and all the support vectors. When a kernel that makes use of an inner product is used, the bulk of computations consists of evaluating the inner products. The document-term data is sparse by nature, and since support vectors are actual samples of the data, the inner products are between two sparse vectors. In the Reuters-21578 experiments in this paper the dimension of the sparse vectors is 5718, but on the average, each vector has only 36 non-zero components. The inner product between two such vectors requires on the average 72 indexing operations of which some may lead to actual multiply-accumulate operations. We count this as 72 operations/vector. Direct application of the SVM to the document-term data and to the 52 binary classification problems produced 10496 support vectors altogether in the 52 classifiers. Roughly, the total number of operations in classifying one unknown vector is thus $10496 \times 72 = 755712$ operations.

Assuming an LDA transform to dimension 12, the transform matrix size is 12×5718. Computing the transform consists of evaluating the 12 inner products between the sparse input vector and each row of the dense transform matrix. This takes on the average 36 indexing operations and as many multiply/accumulates. We count this as 72 operations/vector, too, which results in only $12 \times 72 = 864$ operations. This amount is independent of whether LSI or a random projection is applied prior to LDA, because the final transform matrix is a product of the two matrices, and it is computed beforehand. Assuming the same number of support vectors, we have now $10496 \times 12 = 125952$ operations. Thus even with an efficient sparse implementation of the SVM, using LDA features cuts the computation down to one fifth of the original. Comparing to the direct application of an SVM

to the data, the breakeven point appears to be at the LDA transform dimension of about twice the average number of nonzero entries in the original document vectors. The exact point depends, of course, on the implementation. For example, inner products with LDA vectors, as they are dense, can be made using the vector instructions of the processor.

A Gaussian kernel is somewhat less favorable to the direct sparse implementation, as computing the difference between two sparse vectors effectively doubles the number of nonzero elements in the result. The norm of the difference is then computed as an inner product with itself. This doubling of the size does not happen with dense and low-dimensional LDA vectors.

Computation in SVM training appears to retain the same proportions as the testing phase. Computing the LSI and LDA are extra, but the LSI takes only a few minutes for the Reuters collection, and the LDA no more than 30 seconds (on a 750 MHz Pentium III), and these only need to be computed once. Of course, the training data needs to be transformed, but this is again an insignificant amount of computation and it only needs to be done once.

Application to retrieval tasks. The biggest difference between a classification and a retrieval task is the use of multiple labels per document in retrieval. There are two possible ways to incorporate them into the LDA. The first is what has been done in this paper: Assume that the documents possessing multiple labels do not carry information about class separation and ignore them in the computation of the LDA basis vectors.

The second option is to account for multiply labeled documents multiple times, once for each label, wighted by the reciprocal of the number of labels. For example, when computing the class means and covariance matrices, a document carrying three labels is added to the mean vectors of all those three classes, but by a weight of one third each. This may have an effect of smearing class distinctions, and needs to be experimentally evaluated.

The actual SVM approach is straightforward to modify so that it can make use of training data with multiple labels. Since a multiclass SVM can consist of a number of binary classifiers where one class is set against all others, all documents carrying a particular label, independent of whether they also have other labels or not, are counted as positive examples of that class, and the rest as negative. Setting the decision thresholds for desired precision/recall is then another matter that may require a validation data set.

Interpretation of the LDA basis vectors. It would be interesting to study to what original terms these discriminating features correspond. However, these 12 basis vectors are dense and have both positive and negative entries, which makes the interpretation hard. Now, as the transform, we can have any 12 vectors that span the same subspace as the original basis vectors. It is possible to find a rotation of this 12-dimensional subspace such that the basis vectors become positive and sparse. This has been suggested by Kabán and Girolami as a means to make LSI more amenable to interpretation [17]. They propose projection pursuit with a *skewness* projection index. Same method can now be applied after LDA if interpretability by the original terms is desired.

Other work on feature selection and feature transforms. Comparing this work to others on both feature selection and feature transforms using the same database, Joachims reports SVM experiments with 86.4% precision using 9962 features [16]. Yang and Pedersen report an average precision of about 90% with 2000 selected features on a same task [30]. They used a k-NN classifier, but kept multiply labeled documents in the train and test sets. A few papers report classification error rate (or accuracy which is one minus error rate) on Reuters. Han et al report a classification accuracy of 90% also by using 2000 selected features, and a weighted k-NN approach [13]. Karypis et al describe a method that determines the columns of the projection matrix as the differences of the means between clusters or classes in the data [18]. This is a similar but slightly more heuristic criterion than using only the between-class covariance matrix in LDA and paying no attention to class compactness. This criterion has been used earlier in data visualization [8]. Their results are 82-85% classification accuracy on small subsets of Reuters using 50 transformed features.

9 Conclusion

This paper shows how Linear Discriminant Analysis can be used to reduce drastically the dimension of document representation in classification tasks without sacrificing the accuracy. In fact, the classification error rate decreased from 11.2% to 8.9% when reducing the original 5718 dimensional document representation into mere 12 features.

Although these results can not be directly compared to previous work on the same database due to different methods of selecting the train/test data and different scoring methods, the trend is visible: Previous work shows that a high accuracy is possible with a large subset of features. This work points out that a high accuracy in document classification is possible with a small number of discriminative features. This offers some computational advantages even with Support Vector Machines that take advantage of the sparse nature of the data.

References

[1] Michael W. Berry. Large scale singular value computations. *International Journal of Supercomputer Applications*, 6(1), 1992.

[2] Christopher M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, Oxford, New York, 1995.

[3] William Campbell, Kari Torkkola, and Sree Balakrishnan. Dimension reduction techniques for training polynomial networks. In *Proceedings of the 17th International Conference on Machine Learning*, pages 119–126, Stanford, CA, USA, June 29 - July 2 2000.

[4] Soumen Chakrabarti, Byron Dom, Rakesh Agrawal, and Prabhakar Raghavan. Scalable feature selection, classification and signature generation for organizing large text databases into hierarchical topic taxonomies. *VLDB Journal: Very Large Data Bases*, 7(3):163–178, 1998.

[5] Ronan Collobert and Samy Bengio. SVMTorch: Support Vector Machines for Large-Scale Regression Problems. *Journal of Machine Learning Research*, 1:143–160, 2001.

[6] Sanjoy Dasgupta. Experiments with random projection. In *Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence*, pages 143–151, Stanford, CA, June30 - July 3 2000.

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

[8] I.S. Dhillon, D.S. Modha, and W.S. Spangler. Visualizing class structure of multidimensional data. In *Proceedings of the 30th Symposium of on the Interface, Computing Science, And Statistics*, pages 488–493, Minneapolis, MN, USA, May 1998. Interface Foundation of North America.

[9] J.H. Friedman and J.W. Tukey. A projection pursuit algorithm for exploratory data analysis. *IEEE Trans. on Computers*, C-23:881, 1974.

[10] K. Fukunaga. *Introduction to statistical pattern recognition (2nd edition)*. Academic Press, New York, 1990.

[11] Mark Girolami, Andrzej Cichocki, and Shun-Ichi Amari. A common neural network model for unsupervised exploratory data analysis and independent component analysis. *IEEE Transactions on Neural Networks*, 9(6):1495 – 1501, November 1998.

[12] Xuan Guorong, Chai Peiqi, and Wu Minhui. Bhattacharyya distance feature selection. In *Proceedings of the 13th International Conference on Pattern Recognition*, volume 2, pages 195 – 199. IEEE, 25-29 Aug. 1996.

[13] Eui-Hong (Sam) Han, George Karypis, and Vipin Kumar. Text categorization using weight adjusted k-nearest neighbor classification. In *Proc. PAKDD*, 2001.

[14] David Hull. Using statistical testing in the evaluation of retrieval performance. In *Proc. of the 16th ACM/SIGIR Conference*, pages 329–338, 1993.

[15] David Hull. Improving text retrieval for the routing problem using latent semantic indexing. In *Proc. SIGIR'94*, pages 282–291, Dublin, Ireland, July 3-6 1994.

[16] Thorsten Joachims. Text categorization with support vector machines: Learning with many relevant features. In Claire Nédellec and Céline Rouveirol, editors, *Proceedings of ECML-98, 10th European Conference on Machine Learning*, pages 137–142, Chemnitz, DE, 1998. Springer Verlag, Heidelberg, DE.

[17] Ata Kabán and Mark Girolami. Fast extraction of semantic features from a latent semantic indexed corpus. *Neural Processing Letters*, 15(1), 2002.

[18] G. Karypis and E. Sam. Concept indexing: A fast dimensionality reduction algorithm with applications to document retrieval and categorization. Technical Report TR-00-0016, University of Minnesota, Department of Computer Science and Engineering, 2000.

[19] Thomas Kolenda, Lars Kai Hansen, and Sigurdur Sigurdsson. Indepedent components in text. In M. Girolami, editor, *Advances in Independent Component Analysis*. Springer-Verlag, 2000.

[20] Daphne Koller and Mehran Sahami. Toward optimal feature selection. In *Proceedings of ICML-96, 13th International Conference on Machine Learning*, pages 284–292, Bari, Italy, 1996.

[21] N. Kumar and A. G. Andreou. Heteroscedastic discriminant analysis and reduced rank HMMs for improved speech recognition. *Speech Communcation*, 26:283–297, 1998.

[22] Huan Liu and Hiroshi Motoda. *Feature Selection for Knowledge Discovery and Data Mining*. Kluwer Academic Publishers, 1998.

[23] Dunja Mladenic. Feature subset selection in text-learning. In *European Conference on Machine Learning*, pages 95–100, 1998.

[24] T. Okada and S. Tomita. An optimal orthonormal system for discriminant analysis. *Pattern Recognition*, 18(2):139–144, 1985.

[25] Christos H. Papadimitriou, Prabhakar Raghavan, Hisao Tamaki, and Santosh Vempala. Latent semantic indexing: A probabilistic analysis. *Journal of Computer and System Sciences*, 61(2):217–235, 2000.

[26] George Saon and Mukund Padmanabhan. Minimum bayes error feature selection for continuous speech recognition. In Todd K. Leen, Thomas G. Dietterich, and Volker Tresp, editors, *Advances in Neural Information Processing Systems 13*, pages 800–806. MIT Press, 2001.

[27] Hinrich Schütze, David Hull, and Jan O. Pedersen. A comparison of classifiers and document representations for the routing problem. In *Proc. SIGIR'95*, 1995.

[28] Kari Torkkola and William Campbell. Mutual information in learning feature transformations. In *Proceedings of the 17th International Conference on Machine Learning*, pages 1015–1022, Stanford, CA, USA, June 29 - July 2 2000.

[29] H. Yang and J. Moody. Data visualization and feature selection: New algorithms for nongaussian data. In *Proceedings NIPS'99*, Denver, CO, USA, November 29 - December 2 1999.

[30] Yiming Yang and Jan O. Pedersen. A comparative study on feature selection in text categorization. In *Proc. 14th International Conference on Machine Learning*, pages 412–420. Morgan Kaufmann, 1997.

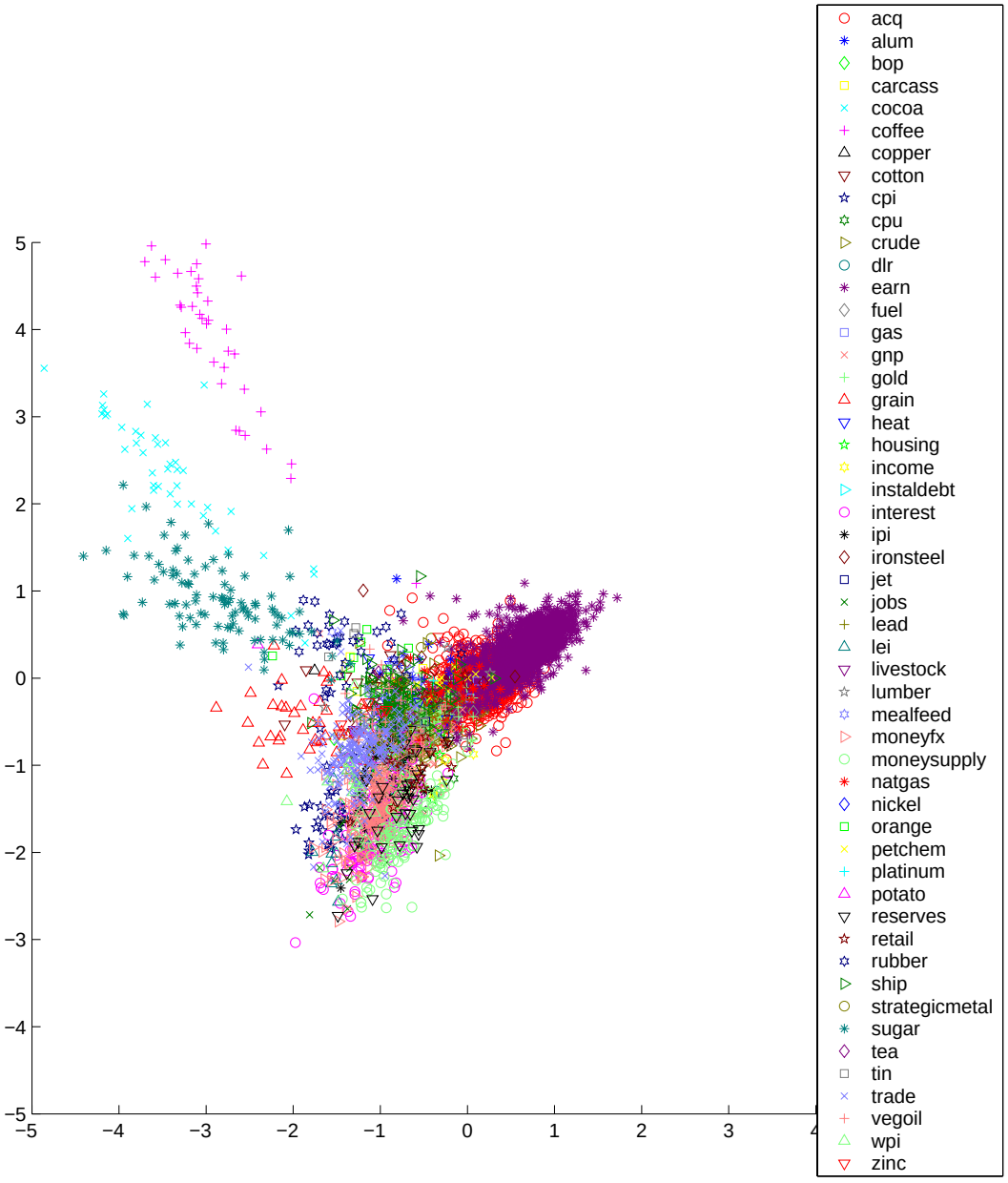

Figure 2: LDA projection of the Reuters-21578 training set onto a two-dimensional discriminative subspace.