Under consideration for publication in Knowledge and Information Systems

Using Discriminant Analysis for Multi-class Classification: An Experimental Investigation

Tao Li[1], Shenghuo Zhu[2] and Mitsunori Ogihara[3]

[1]School of Computer Science, Florida International University, Miami, FL;
[2]NEC Labs America, Inc., Cupertino, CA;
[3]Department of Computer Science, University of Rochester, Rochester, NY.

Abstract. Many supervised machine learning tasks can be cast as multi-class classification problems. Support vector machines (SVMs) excel at binary classification problems, but the elegant theory behind large-margin hyperplane cannot be easily extended to their multi-class counterparts. On the other hand, it was shown that the decision hyperplanes for binary classification obtained by SVMs are equivalent to the solutions obtained by Fisher's linear discriminant on the set of support vectors. Discriminant analysis approaches are well known to learn discriminative feature transformations in the statistical pattern recognition literature and can be easily extend to multi-class cases. The use of discriminant analysis, however, has not been fully experimented in the data mining literature. In this paper, we explore the use of discriminant analysis for multi-class classification problems. We evaluate the performance of discriminant analysis on a large collection of benchmark datasets and investigate its usage in text categorization. Our experiments suggest that discriminant analysis provides a fast, efficient yet accurate alternative for general multi-class classification problems.

Keywords: multi-class classification, discriminant analysis

1. Introduction

Classification tasks aim to assign a predefined class to each instance. It can help to understand existing data and be used to predict how new instances will behave. The problem can be treated as a regression problem which be formally defined as follows: given a set of training samples, in the form of $\langle \mathbf{x}_i, y_i \rangle$, the goal

Received xxx
Revised xxx
Accepted xxx

is to learn an approximate function $\hat{f}$ of the underlying function f, such that $y_i = f(\mathbf{x}_i) + \epsilon_i$, where ϵ_i is the error. The error here is defined as the difference between the real targets, where numeric values, 1 and 0, represent logic values, true and false, respectively, and the estimated values, which is the score of being true. Though the distribution of the errors is neither normal nor individually independent, we treat them as noise for simplicity. Generally each sample is represented as a multi-dimensional vector and the function f takes values from a discrete set of "class labels": $\{c_1, c_2, \cdots, c_n\}$. In cases when $n = 2$, *i.e.*, there are only two possible values for f, the classification problems are referred as binary classification problems. We can use the n estimated scores to decide which one is real true case. Most machine learning algorithms were devised first for binary classification problems. On the other hand, many real-word problems have more than two classes to deal with. Typical examples include optical character recognition (OCR), object and gesture recognition, part-of-speech tagging, text categorization and microarray data analysis.

Support vector machines (SVMs) (Vapnik, 1998) have shown superb performance at binary classification tasks. They are accurate, robust and quick to apply to test instances. However, the elegant theory behind the use of large-margin hyperplanes cannot be easily extended to multi-class classification problems. A number of reduction approaches such as one-versus-the-rest method (Bottou, et al., 1994), pairwise comparison (Hastie & Tibshirani, 1998), direct graph traversal (Platt, et al., 2000), error-correcting output coding (Dietterich & Bakiri, 1995; Allwein, et al., 2000), and multi-class objective functions (Weston & Watkins, 1998) have been proposed to first reduce a multi-class problem to a collection of binary-class problems and then combine their predictions in various ways. In practice, the choice of reduction method from multi-class to binary is problem-dependent and not a trivial task since each reduction method has its own limitations (Allwein et al., 2000). In addition, regardless of specific details, these reduction techniques are not well suited for classification problems with a large number of categories because SVMs, while accurate and fast to apply, require $O(n^\alpha)$ time to train, where usually $\alpha \in [1.7, 2.1]$ (Joachims, 2001). The prediction time of a new instance also increases significantly when the number of classes becomes larger. Hence, despite the theoretical elegance and superiority of SVMs, the training/prediction time requirement and scaling are great concerns.

Discriminant analysis approaches are well known to learn discriminative feature transformations in the statistical pattern recognition literature and have been successfully used in many recognition tasks (Fukunaga, 1990). Fisher discriminant analysis (Fisher, 1936) finds a discriminative feature transform as eigenvectors of matrix $T = \hat{\Sigma}_w^{-1}\hat{\Sigma}_b$ where $\hat{\Sigma}_w$ is the intra-class covariance matrix and $\hat{\Sigma}_b$ is the inter-class covariance matrix. Basically T captures both compactness of each class and separations between classes and hence eigenvectors corresponding to largest eigenvalues of T would constitute a discriminative feature transform. Shashua (1999) showed that the decision hyperplanes for binary classification obtained by SVMs is equivalent to the solution obtained by Fisher's linear discriminant on the set of support vectors. For the multi-class cases, such equivalence is not clear. Liner discriminant analysis approach is similar to Gaussian Processes (Barber & Williams, 1997; Gibbs & MacKay, 2000) in the way of inference from the covariance matrix of training data. Also, Gallinari, et al. (1991) showed that neural network classifiers are equivalent to discriminant analysis. Fisher's discriminant analysis was first described for two-class

cases (Fisher, 1936), and can be easily extended to multi-class cases via multiple discriminant analysis (Johnson & Wichern, 1988). In fact, discriminant analysis has been widely used in face recognition (Fukunaga, 1990). These observations hint that discriminant analysis could be very promising for multi-class classification tasks. However, there is little investigation on the benchmark datasets in machine learning and data mining.

In this paper, we present a comprehensive experimental study of the use of discriminant analysis for multi-class classification. We evaluate the performance of discriminant analysis on a large collection of benchmark datasets and investigate its usage in text categorization. Our study shows that discriminant analysis has several favorable properties: first, it is simple and can be easily implemented; second, it is efficient and most of our experiments only took a few seconds; last but not the least, it also has comparable accuracy in performance on most of the datasets we experimented. The rest of the paper is organized as follows: Section 2 reviews the related work on multi-class classification. Section 3 gives a brief overview of Linear Discriminant Analysis (LDA). Section 4 discusses some of the issues in discriminant analysis. Section 5 shows our experimental results on a variety of benchmark data sets. Section 6 presents the case study of using LDA for text categorization, and finally Section 7 provides our conclusions.

2. Related Work

Generally speaking, multi-class classification approaches can be roughly partitioned into two groups. The first group consists of those algorithms that can be naturally extended to handle multi-class cases. This group contains such algorithm as nearest neighborhoods (Hastie, et al., 2001), regression and decision trees including C4.5 (Quinlan, 1993) and CART (Breiman, et al., 1993). The second group consists of methods that involve reduction of multi-class classification problems to binary ones. Depending on the reduction technique that is used, the group can be further divided into one-versus-the-rest method (Schölkopf & Smola, 2002; Bottou et al., 1994), pairwise comparison (Kreeel, 1999; Hastie & Tibshirani, 1998; Friedman, 1996), direct graph traversal (Platt et al., 2000), error-correcting output coding (Dietterich & Bakiri, 1995; Allwein et al., 2000), multi-class objective functions (Weston & Watkins, 1998).

The idea of the one-versus-the-rest method is as follows: to get a K-class classifier, first construct a set of binary classifiers $C_1, C_2, \cdots, C_K$. Each binary classifier is first trained to separate one class from the rest and then the multi-class classification is carried out according to the maximal output of the binary classifiers. Since the binary classifiers are obtained by training on different binary classification problems, it is unclear whether their real-valued outputs (before thresholding) are on comparable scales (Schölkopf & Smola, 2002). In practice, however, situations often arise where several binary classifiers assign the same instance to their respective class (or where none does). In addition, binary one-versus-the-rest classifiers has been criticized for dealing with rather asymmetric problems (Schölkopf & Smola, 2002).

In pairwise comparison, a classifier is trained for each possible pair of classes. For K classes, this results in $(K-1)K/2$ binary classifiers. Given a new instance, the multi-class classification is then executed by evaluating all $(K-1)K/2$ individual classifiers and assigning the instance to the class which gets the highest number of votes. Basically the individual classifiers used in pairwise comparison

have smaller training sets comparing with the one-versus-the-rest method. Also the individual classifiers are usually easier to be learned since the classes have less overlap. However, pairwise comparison implies a large number of individual classifiers, especially for datasets with lots of classes.

Direct graph traversal method is an extension of pairwise comparison. The training phase of the direct graph traversal is the same as that of pairwise comparison by building $(K-1)K/2$ individual classifiers. To classify new instances, however, direct graph traversal method uses a rooted binary directed acyclic graph which has $(K-1)K/2$ internal nodes and K leaves. Each classification run then corresponds to a directed traversal of the graph and classification can be much faster. Hsu & Lin (2002) compared the performance of three methods for multi-class support vector machine: one-versus-the-rest, pairwise comparison and direct graph traversal. Their experiments indicated that the performance of three methods are very similar and no one method is statistically better the others.

Error-correcting output coding (ECOC) was developed by Dietterich & Bakiri (1995). In a nutshell, the idea here is to generate a number of binary classification problems by smartly splitting the original set of classes into two sets. In other words, each class is assigned a unique binary strings of length l (these strings are regarded to codewords). Then l classifiers are trained to predict each bit of the string. For new instances, the predicted class is the one whose codeword is the closest (in Hamming distance) to the codeword produced by the classifiers. Allwein et al. (2000) extended ECOC and presented a general framework that unifies methods of reducing multi-class to binary including one-versus-the-rest, pairwise comparison and ECOC. Allwein et al. (2000) also gave the loss-based coding scheme which takes margins into consideration and is more sophisticated and efficient than Hamming coding. In addition, it presented experimental results with a variety of multi-to-binary-class reductions and demonstrated that although loss-based coding is better than Hamming coding in most cases, the best method seems to be problem-dependent.

Weston & Watkins (1998) and Vapnik (1998) proposed approaches for multi-class classification by solving one single optimization problem. The idea is to directly modify the objective function of support vector machine (SVM) in such a way that it allows simultaneous computation of a multi-class classifier. In terms of accuracy, the results obtained by this approach are comparable to those obtained by the widely used one-versus-the-rest method. But, the multi-class objective function has to deal with all the support vectors at the same time and hence lead to long training time.

In practice, the choice of reduction method from multi-class to binary is problem-dependent and not a trivial task. Crammer & Singer (2000) discussed the problem of designing output codes for multi-class classification problems. Discriminant analysis is a direct method for multi-class classification and it does not require reducing multi-class to binary.

Next we review some other multi-class classification approaches that were developed recently. Park, et al. (2001) presented an algorithm for dimensional reduction for text representation based on cluster structure preserving projection using generalized singular value decomposition (GSVD). Godbole, et al. (2002) presented a new technique for multi-class classification by exploiting the accuracy of SVMs and the speed of Naive Bayes (NB) classifiers. The new technique first utilized a NB classifier to quickly compute a confusion matrix which is used to reduce the number and complexity of the two-class SVMs that are built in the

second stage. More literature on multi-class classification and its applications can be found in (Lee, et al., 2001; Ghani, 2001; Crammer & Singer, 2001; Zadrozny, 2001; Guruswami & Sahai, 1999; Roth, 2001; Rennie, 2001).

In summary, as pointed out in (Schölkopf & Smola, 2002), it is fair to say that there is probably no multi-class approach generally outperforms the others. For practical problems, the choice of approach will depend on constraints on hand such as required accuracy, the time available for development and training and the nature of the classification problem. The simple, efficient and accurate discriminant analysis provides a good choice for practical multi-class classification problems.

3. Linear Discriminant Analysis (LDA)

In this paper, we focus on linear transformation since linear discriminant analysis frequently achieves good performances in the tasks of face and object recognition, even though the assumptions of common covariance matrix among groups and normality are often violated (Duda, et al., 2001). In addition, kernel tricks can be used with linear discriminant analysis for non-linear transformation (Mika, et al., 1999). The basic idea of LDA is to find a linear transformation that best discriminate among classes and the classification is then performed in the transformed space based on some metric such as Euclidean distance. Mathematically a typical LDA implementation is carried out via scatter matrix analysis (Fukunaga, 1990).

3.1. Two-class LDA

Fisher (1936) first introduced LDA for two classes and its idea was to transform the multivariate observations $\mathbf{x}$ to univariate observations $\mathbf{y}$ such that the $\mathbf{y}$'s derived from the two classes were separated as much as possible. Suppose that we have a set of m p-dimensional samples $\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_m$ (where $\mathbf{x}_i = (x_{i1}, \cdots, x_{ip})$) belonging to two different classes, namely c_1 and c_2. For the two classes, the scatter matrices are given as

$$S_i = \sum_{\mathbf{x} \in c_i} (\mathbf{x} - \bar{\mathbf{x}}_i)(\mathbf{x} - \bar{\mathbf{x}}_i)',$$

where $\bar{\mathbf{x}}_i = \frac{1}{m_i} \sum_{\mathbf{x} \in c_i} \mathbf{x}$ and m_i is the number of samples in c_i. Hence the total intra-class scatter matrix is given by

$$\hat{\Sigma}_w = S_1 + S_2 = \sum_i \sum_{\mathbf{x} \in c_i} (\mathbf{x} - \bar{\mathbf{x}}_i)(\mathbf{x} - \bar{\mathbf{x}}_i)'. \quad (1)$$

The inter-class scatter matrix is given by

$$\hat{\Sigma}_b = (\bar{\mathbf{x}}_1 - \bar{\mathbf{x}}_2)(\bar{\mathbf{x}}_1 - \bar{\mathbf{x}}_2)'. \quad (2)$$

Fisher's criterion suggested the linear transformation Φ to maximize the so-called *Rayleigh coefficient*, that is, the ratio of the determinant of the inter-class scatter matrix of the projected samples to the intra-class scatter matrix of the projected

samples:

$$\mathcal{J}(\Phi) = \frac{|\Phi^T \hat{\Sigma}_b \Phi|}{|\Phi^T \hat{\Sigma}_w \Phi|}. \tag{3}$$

If $\hat{\Sigma}_w$ is non-singular, Equation (3) can be solved as a conventional eigenvalue problem and Φ is given by the eigenvectors of matrix $\hat{\Sigma}_w^{-1}\hat{\Sigma}_b$.

3.2. Multi-class LDA

If the number of classes are more than two, then a natural extension of Fisher Linear discriminant exists using multiple discriminant analysis (Johnson & Wichern, 1988). As in two-class case, the projection is from high dimensional space to a low dimensional space and the transformation suggested still maximize the ratio of intra-class scatter to the inter-class scatter. But unlike the two-class case, the maximization should be done among several competing classes.

Suppose that now there are n classes. The intra-class matrix is calculated similar to Equation (1):

$$\hat{\Sigma}_w = S_1 + \cdots + S_n = \sum_{i=1}^{n} \sum_{\mathbf{x} \in c_i} (\mathbf{x} - \bar{\mathbf{x}}_i)(\mathbf{x} - \bar{\mathbf{x}}_i)'.$$

The inter-class scatter matrix slightly differs in computation and is given by

$$\hat{\Sigma}_b = \sum_{i=1}^{n} m_i (\bar{\mathbf{x}}_i - \bar{\mathbf{x}})(\bar{\mathbf{x}}_i - \bar{\mathbf{x}})'$$

where m_i is the number of training samples for each class, $\bar{\mathbf{x}}_i$ is the mean for each class and $\bar{\mathbf{x}}$ is total mean vector given by $\bar{\mathbf{x}} = \frac{1}{m}\sum_{i=1}^{n} m_i \bar{\mathbf{x}}_i$. After obtaining $\hat{\Sigma}_b$ and $\hat{\Sigma}_w$, the linear transformation Φ we want should still maximize equation 3. It can be shown that the transformation Φ can be obtained by solving the generalized eigenvalue problem:

$$\hat{\Sigma}_b \Phi = \lambda \hat{\Sigma}_w \Phi \tag{4}$$

It is easy to prove that the upper bounds of the rank of $\hat{\Sigma}_w$ and $\hat{\Sigma}_b$ are respectively $m - n$ and $n - 1$. Multiple discriminant analysis provides an elegant way for classification using discriminant features.

3.3. Classification

Once the transformation Φ is given, the classification is then performed in the transformed space based on some distance metric, such as Euclidean distance $d(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_i (x_i - y_i)^2}$ and cosine measure $d(\mathbf{x}, \mathbf{y}) = 1 - \frac{\sum_i x_i y_i}{\sqrt{\sum_i x_i^2}\sqrt{\sum_i y_i^2}}$. Then upon the arrival of the new instance $\mathbf{z}$, it is classified to

$$\arg\min_k d(\mathbf{z}\Phi, \bar{\mathbf{x}}_k \Phi),$$

where $\bar{\mathbf{x}}_k$ is the centroid of k-th class.

4. Discussions on LDA

4.1. Fisher Criterion and its non-optimality

Although practical evidences have been shown that discriminant analysis is effective (and also will be demonstrated in our experimental study in Section 5), it should be pointed out that a significant separation does not necessarily imply a good classification. Multi-class discriminant analysis is concerned with the search for a linear transformation that reduces the dimension of a given p-dimensional statistical model, consisting of n classes, to $n-1$ dimensions while preserving a maximum amount of discriminant information in the lower-dimensional model. It is in general too complicated to use the Bayes error directly as a criterion and Fisher's criterion is just a suboptimal criterion and easy to optimize. The solution of optimizing Fisher's criterion is obtained by an eigenvalue decomposition of $\hat{\Sigma}_w^{-1}\hat{\Sigma}_b$ and taking the rows of the transformation matrix to equal to the $n-1$ eigenvectors corresponding the $n-1$ largest eigenvalues. It has been shown that, however, for multi-class problem, the fisher's criterion is actually maximizing the mean squared distance between the classes in the lower-dimensional space and is clearly different from minimizing classification error (Loog, et al., 2001). In maximizing the squared distances, pairs of classes, between which there are large distances, completely dominate the eigenvalue decomposition. The resulting transformation preserves the distances of already well separated classes. As a consequence, there is a large overlap among the remaining classes, leading to an overall low and suboptimal classification rate.

4.2. When the inner scatter is singular

There are at most $n-1$ nonzero generalized eigenvector of $\hat{\Sigma}_w^{-1}\hat{\Sigma}_b$, and so an upper bound of dimensions d in the transformed space is $n-1$. At least $p+n$ samples are required to guarantee that $\hat{\Sigma}_w$ does not become singular. In practice, especially in text categorization and pattern recognition, where the number of samples m is small and/or the dimensionality p is large, $\hat{\Sigma}_w$ is usually singular. To deal with the singularity of $\hat{\Sigma}_w$, several methods have been proposed:

– Regularization. In regularization method (McLachlan, 1992; Zhao, et al., 1999), the matrix $\hat{\Sigma}_w$ is regularized by biasing the diagonal components by

$$\hat{\Sigma}'_w = \hat{\Sigma}_w + \delta I$$

where δ is a relatively small parameter such that $\hat{\Sigma}'_w$ is positive definite. In practice, δ can be chosen as the average of the diagonal elements of $\hat{\Sigma}_w$ multiplied by a small constant k.

– Subspace. The subspace method uses a non-singular intermediate space of $\hat{\Sigma}_w$ obtained by removing the null space of $\hat{\Sigma}_w$ to compute the transformation. In other words, the subspace method usually employs the eigenanalysis for dimension reduction and projects the original samples onto a lower-dimensional space to make the resulting intra-class scatter matrix full-rank. Typical subspace methods are presented in (Swets & Weng, 1996; Martinez & Kak, 2001; Yang, et al., 2000).

– Null space method. Instead of discarding the null space of $\hat{\Sigma}_w$, null space method actually makes use of the null space (Chen, et al., 2000; Huang, et al., 2002). The removal of the null space of $\hat{\Sigma}_w$ may potentially lose useful information since some discriminant dimensions are potentially lost by removing the null space of $\hat{\Sigma}_w$. In fact, the null space of $\hat{\Sigma}_w$ contains considerable discriminant information when the projection of $\hat{\Sigma}_b$ is not zero along that direction. In null space method, all the samples are first projected onto the null space of $\hat{\Sigma}_w$, where the intra-class scatter is zero and then use the traditional method to find the optimal discriminant vectors. Some methods also have been proposed to first remove the null space of $\hat{\Sigma}_w + \hat{\Sigma}_b$ method since those vectors in the null space of $\hat{\Sigma}_w + \hat{\Sigma}_b$ contains no useful information.

4.3. Computational Complexity

The computational complexity of training time consists of evaluating the inner and between covariance matrices, eigenvalue decomposition, and selecting discriminating features. The computational complexity of evaluating the covariance matrices is mp^2. If we use SVD to decompose the matrices, we can directly use the feature vectors and avoid the multiplications in evaluating covariance matrices. The computational complexity of eigenvector decomposition is $mp \min(m, p)$, where m and p are the number of rows and the number of columns of the given matrix. In our case, m and p are the number of instances and the number of features respectively. The time of selecting discriminating features is almost linear to p. Therefore, the total time is about $O(mp \min(m, p))$. Since we only need the largest $n - 1$ eigenvalues and corresponding eigenvectors, the complexity could be even lower if the feature vectors are sparse.

5. Experiments on Benchmark Datasets

5.1. Data Description

We used a wide range of data sets in our experiments as summarized in Table 1. The number of classes ranges from 3 to 100, the number of samples ranges from 72 to 581012 and the number of attributes ranges from 8 to 12558. In addition, theses data sets represent applications from different domains such as image processing, gene expression data and pattern recognition. We anticipated that these data sets provide us enough insights on the behavior of LDA.

ALL-AML (acute lymphoblastic leukemia - acute myeloblastic leukemia) dataset contains measurements corresponding to ALL (B-cell and T-cell) and AML samples from Bone Marrow and Peripheral Bloodis Blood[1]. It was first studied in (Golub, et al., 1999) for binary class classification. ALL (acute lymphoblastic leukemia) dataset is used to classify subtypes of pediatric acute lymphoblastic leukemia and the data has been divided into six diagnostic groups[2]. DNA dataset is used to recognize, given a sequence of DNA, the boundaries between exons (the parts of the DNA sequence retained after splicing) and introns

[1] `http://www-genome.wi.mit.edu`.
[2] `http://www.stjuderesearch.org/data/ALL1/`.

(the parts of the DNA sequence that are spliced out)[3]. Coil-100 dataset consists of color images of 100 objects where the images of objects that were taken at a pose interval of 5°, *i.e.*, 72 poses per object[4]. All the other datasets are from UCI repository (Blake & Merz, 1998).

When available, we used the original partition of the datasets into training and test sets. For the other datasets, if not specified, we used ten-fold cross validation method to evaluate results.

We implemented regularization, subspace and null space methods to deal with the singularity of inner scatter matrix. For regularization method, we set δ equal to 2 times the average of the diagonal components of the inner scatter matrix (Kawatani, 2002). When the inner matrix is singular, the results reported to be the best one obtained by the three methods.

5.2. Data Preprocessing

There are four types of data values, continuous, binary, ordered, and categorical. Discriminant analysis algorithms are originally designed for continuous valued data sets. However, with simple preprocessing, it can be used on any type of data sets too. Values of a binary valued attribute can be translated into 0 and 1. Values of an ordered valued attribute can be translated into natural numbers according to their order.

A categorical valued attribute can be replaced with the same number of binary attributes as its cardinality, each of which represents whether a value belongs to the corresponding category of the original attribute. For example, suppose that an attribute takes values from set $\{A, B, C\}$. It is replaced with three binary attributes, named A, B, and C. Attribute A takes value 1 if the original attribute takes value A and 0 otherwise. This translation scheme was used in our experiments on Audiology dataset. It is possible to use a conversion with one dimension less than this conversion. We prefer this conversion because this conversion is symmetrical to all categorical values, and it is not as efficient to represent all categorical values in linear transformations without the additional dimension as in logic combination.

Many datasets contain missing values, such Dermatology and Water datasets. Discriminant analysis can be easily extended to handle such datasets. For our experiments, in the training phase, a missing value was imputed with the mean of the existent attribute values in the same class. In predicting phase, the attributes with missing values were ignored. In other words, an attribute with missing value was imputed with the attribute mean value of each class when comparing with the class.

5.3. Results Analysis

In this section, we present and discuss our experimental results. All of our experiments were performed on a P4 2GHz machine with 512M memory running Linux 2.4.9-31.

[3] `http://www.niaad.liacc.up.pt/statlog/datasets.html`.
[4] `ftp://ftp.cs.columbia.edu/pub/CAVE/SLAM_coil-20_coil-100/coil-100/coil-100.zip`.

Datasets	# training	# test	# attributes	# class
ALL-AML	72	-	7129	3
DNA	2000	1186	180	3
Iris	150	-	4	3
Waveform	300	5000	21	3
Wine	178	-	13	3
car	1728	-	6	4
Vehicle	846	-	18	4
Heart(Hungarian)	294	-	76	5
Page-blocks	5473	-	10	5
Dermatology	366	-	34	6
ALL	215	112	12558	6
Satimage	4435	2000	36	6
Glass	214	-	9	7
Shuttle	3866	1934	9	7
Segmentation	2310	-	19	7
Zoo	101	-	18	7
Covertype	581012	-	54	8
Ecoli	336	-	8	8
Optdigits	3823	1797	64	10
Pendigits	7494	3498	16	10
Yeast	1484	-	8	10
Vowel	528	462	10	11
Water Treatment	527	-	38	13
Soybean	307	376	35	19
Audiology	226	-	69	24
Isolet	6238	1559	617	26
Letter	16000	4000	16	26
Coil-100	800	6400	160	100

Table 1. The Description of Datasets.

Whenever possible, we compared our experimental results with those presented in (Allwein et al., 2000) since most of them were regarded to be state-of-the-art. If we can not find corresponding results from (Allwein et al., 2000), we compared our results with the documented usage or other reported results. Finally, if both means are not available, we then compared the results on LDA with those obtained using our available implementations of classification methods.

For Dermatology, Satimage, Glass, Ecoli, Pendigits, Yeast, Vowel and Soy-

Datasets	Past Usage	Past Results	LDA
ALL-AML	-	0.944	0.914
DNA	(Noordewier, et al., 1991)	0.937	0.947
Iris	(Dzeroski & Zenko, 2002)	0.948	0.980
Waveform	(Dzeroski & Zenko, 2002)	0.8652	0.858
Wine	(Dzeroski & Zenko, 2002)	0.9798	0.994
Car	(Dzeroski & Zenko, 2002)	0.9868	0.81
Vehicle	(SGI, 2000)	0.85/0.448	0.785
Heart	(Dzeroski & Zenko, 2002)	0.8415	0.521
Page-blocks	-	0.930	0.931
Dermatology	(Allwein et al., 2000)	0.969/0.967	0.970
ALL	-	0.941	0.970
Satimage	(Allwein et al., 2000)	0.867/0.591	0.828
Glass	(Allwein et al., 2000)	0.676/0.624	0.593
Shuttle	(SGI, 2000)	0.9999/0.9328	0.946
Segmentation	(Allwein et al., 2000)	0.993/1.00	0.916
Zoo	-	0.970	0.930
Covertype	(Blake & Merz, 1998)	0.70	0.581
Ecoli	(Allwein et al., 2000)	0.852/0.849	0.827
Optdigits	(Blake & Merz, 1998)	0.9755/0.98	0.938
Pendigits	(Allwein et al., 2000)	0.973/0.975	0.829
Yeast	(Allwein et al., 2000)	0.528/0.271	0.517
Vowel	(Allwein et al., 2000)	0.530/0.491	0.447
Water	-	0.970	0.970
Soybean	(Allwein et al., 2000)	0.910/0.790	0.89
Audiology	(Allwein et al., 2000)	0.669/0.808	0.818
Isolet	(Allwein et al., 2000)	0.902/0.947	0.941
Letter	(Allwein et al., 2000)	0.734/0.854	0.681
Coil-100	(Roth, et al., 2000)	0.8923	0.963

Table 2. Accuracy Comparison Table. Results in the form of nnn/mmm are two typical results from past usages. Results from (Allwein et al., 2000) are sparse code/one-versus-the-rest accuracies. Results from (SGI, 2000) are the highest/the lowest accuracies. For Covertype, the result of 0.70 from (Blake & Merz, 1998) is obtained using neural network. For Optdigits, 0.9755/0.98 is the lowest/the highest accuracies.

bean datasets, we compared our experimental results with those presented in (Allwein et al., 2000) using the support vector machine algorithm as the base binary learner. There are two decoding schemes used: Hamming decoding and loss-based decoding. The authors pointed out, and it actually demonstrated by their experiments, that loss-based decoding is almost always better than Hamming coding. So our comparison was against the results of loss-based decoding. For loss-based decoding, the paper gave the results of five different types of output codes: one-versus-the-rest, pairwise comparison, complete code (in which there is one column for every possible non-trivial split of the classes) and two types of random codes (dense code and sparse code). Although one-versus-the-rest is generally not as good as other four codes, it is the most widely used method. For all other four codes, there is no clear winners. However, because complete and pairwise comparison codes are not suitable for large datasets and in fact, the paper did not provide the results on these two codes for some datasets. Hence, for presentation purpose, we include only the results of sparse code and one-versus-the-rest for comparison. For more details on these codes, refer to (Allwein et al., 2000). For Isolet, Letter, Audiology and Segmentation, (Allwein et al., 2000) did not have the results using SVM as binary classifier since their SVM implementation could not handle those datasets. However, they gave the results on these datasets using AdaBoost as base binary learner. We compared our results on these datasets with those using AdaBoost and loss-based decoding.

For Iris, Waveform, Wine, Car and Heart datasets, Dzeroski & Zenko (2002) evaluated several state-of-the-art methods for constructing ensembles of classifiers with stacking. Stacking with Multi-response Model Tree (SMM5) achieved the best performance in their reported experiments. Therefore, we referred their results with SMM5 for comparison.

For Vehicle and Shuttle datasets, we compared our results with those recorded results in (SGI, 2000). For Covertype and Optdigits datasets, we compared our results with those documented results in (Blake & Merz, 1998). For DNA, we compared our results with that from (Noordewier et al., 1991). For Coil-100 dataset, we used the same experiment strategy as (Roth et al., 2000). In this paper, we only present the result of an experiment using 1/9 of all the images for training, the rest for testing. For ALL-AML, Page-blocks, All, Zoo and water datasets, we compared the results on LDA with those obtained using our available implementations of SVM methods with one-against-all reductions. The detailed results are presented in Table 2.

The results of LDA on DNA, Iris, Wine, Dermatology, Ecoli, Audiology and Coil-100, outperform their counterparts. For Car, Heart, Segmentation, Covertype, Pendigits and Letter, LDA results are inferior to their counterparts. The main reason is that the number of attributes of these datasets is relatively low with respect to the size of large training sets. One possible solution is to increase dimensionality of datasets, such as using kernel functions. We will discuss some related issues in Section 7. Other results are generally comparable.

Discriminant analysis is very efficient and most experiments only took less than a second. Table 3 gave the running time for experiments on the large datasets. In summary, the extensive experimental results on benchmark datasets have clearly demonstrated the efficiency and effectiveness of LDA.

Datasets	Training	Prediction
ALL	26	1
Satimage	1	1
Covertype	85	10
Isolet	157	22
Letter	1	1
Pendigits	1	1
Yeast	1	1
DNA	3	1
Waveform	1	1
Segmentation	1	1
Optdigits	1	1
ALL-AML	12	1
Page-blocks	1	1
Coil-100	2	19

Table 3. Running Time Table. Each number is training/prediction time in seconds (rounded up to 1s). Several numbers are actually less than 1s.

6. LDA in Text Categorization

Automated text categorization is a multi-class classification problem, defined as assigning pre-defined category labels to new documents based on the likelihood suggested by the training set of labeled documents. Naive Bayes has been a very successful practical learning method in text categorization despite its impractical and simplified conditional independence assumptions (Mitchell, 1997). Fisher linear discriminant analysis can be derived by starting with the theoretically optimal Bayes classifiers and assuming normal distribution for classes (Hastie et al., 2001). This suggests the applicability of LDA in text categorization.

Little work on discriminant analysis has been reported in the document analysis domain. One reason is the extremely high dimensions of the document representation. Usually, a document collection would have thousands of terms. The intra-class covariance matrix is thus usually singular. In addition, the large document-term matrices incur considerable computation cost (eigen-analysis, inverse computation etc.) and hence restrict the usage of discriminant. Fortunately, a simple remedy exists: *Feature Selection*. It has been shown that feature selection via information gain can remove up to 90% or more of the unique terms without significant performance degrade (Yang & Pederson, 1997). With the feature selection, we then could explore the use of LDA in text categorization.

6.1. Text Data Description

We used a wide range of text datasets in our experiments. Most of them are well-known in information retrieval literature. The number of classes ranges from 4 to 105 and the number of documents ranges from 476 to 20,000.

20Newsgroups: The 20Newsgroups (20NG) dataset contains about 20,000 articles evenly divided among 20 Usenet newsgroups. The raw text takes up 26 MB. All words were stemmed using a porter stemmer, all HTML tags were skipped and all header fields except subject and organization of the posted article were ignored.

WebKB: The WebKB dataset [5] contains webpages gathered from university computer science departments. There are about 8300 documents and they are divided into seven categories: student, faculty, staff, course, project, department and other. The raw text is about 27MB. Among these seven categories, student, faculty, course and project are four most populous entity-representing categories. The associated subset is typically called **WebKB4**. In this paper, we did experiments on both seven-category classification and four-category classification. In either case, we did not use stemming or a stoplist.

Industry Sector: The Industry Sector dataset, based on data made available by Market Guide Inc. consists of company homepages classified in a hierarchy of industry sectors[6]. In our experiments, we did not take the hierarchy into account and used a flattened version of the dataset. There are 9637 documents in the dataset divided into 105 classes. In tokening the data, we skipped all MIME and HTML headers, used a standard stoplist and did not perform stemming.

Reuters: The Reuters-21578 Text Categorization Test collection contains documents collected from the Reuters newswire in 1987. It is a standard text categorization benchmark and contains 135 categories. In our experiments, we used two subsets of the data collection. The first one include the 10 most frequent categories among the 135 topics and we call it Reuters-top10. the second one contains the documents which have unique topic, i.e, the documents that have multiple class assignments were ignored, and we call it Reuters-2. There are about 9000 documents and 50 categories.

TDT2: We also used the NIST Topic Detection and Tracking (TDT2) text corpus version 3.2 released in December 6, 1999 (TDT2, 1998). The TDT2 corpus contains news data collected daily from 9 news sources in two languages (American English and Mandarin Chinese), over a period of six months (January - June, 1998). In our experiment, we used only the English news which were collected from New York Times Newswire Service, Associated Press Worldstream Service, Cable News Network, Voice of America, American Broadcasting Company and Public Radio International. The documents were judged relevant to any of 96 target topics by manual annotation. We selected the documents having annotated topics and removed the brief news data. We finally got $7,980$ documents.

K-dataset: The K-dataset was from WebACE project (Han, et al., 1998) and it was used in (Boley, et al., 1999) for document clustering. The K-dataset contains 2340 documents consisting news articles from Reuters new service via the Web in October 1997. These documents were divided into 20 classes. The

[5] Both 20Newsgroups and WebKB are available from `http://www-2.cs.cmu.edu/afs/cs/project/theo-4/text-learning/www/datasets.html`.
[6] `http://www.cs.cmu.edu/~TextLearning/datasets.html`.

Datasets	# documents	# class
20Newsgroups	20,000	20
WebKB4	4,199	4
WebKB	8,280	7
Industry Sector	9,637	105
Reuters-top 10	2,900	10
Reuters-2	9,000	50
CSTR	476	4
K-dataset	2,340	20
TDT2	7,980	96

Table 4. The Description of Text Datasets.

documents were processed by eliminating stop words and HTML tags, stemming the remaining words using Porter's suffix-stripping algorithm.

CSTR: CSTR dataset is the collection of abstracts of technical reports published by the computer science department at university of Rochester from year 1991 to 2002 [7]. The dataset contains 476 abstracts that are divided into 4 different research areas: Symbolic-AI, Spatial-AI, Systems, and Theory. We processed the abstracts by removing stop words and applying stemming operations on the remaining words.

The text datasets and their characteristics are summarized in Table 4.

6.2. Text Data Preprocessing

In all our experiments, we randomly chose 70% of the documents for training and the remaining 30% for testing. As suggested in (Yang & Pederson, 1997), information gain can be effective in term removal and it can remove up to 90% or more of the unique terms without performance degrade. Hence, we first selected the top 1000 words by information gain with class labels [8]. The feature selection is done with the rainbow package (McCallum, 1996).

In our experiments we used classification accuracy as the evaluation measure. Different measures such as precision-recall graphs and F_1 measure (Yang & Liu, 1999) have been used in literature. However, since the document datasets used in the experiments are relatively balanced and our goal of text categorization is to achieve a low misclassification error and high separation between different classes on a test set, accuracy is a good performance for our purposes (Schapire & Singer, 2000). All of our experiments were performed on a P4 2GHz machine with 512M memory running Linux 2.4.9-31.

[7] The technical reports are available from *http://www.cs.rochester.edu/trs*.

[8] One of our future work would be to explore how the performance correlates with different feature selection methods and the number of words selected.

Data Sets	LDA	NB	SVM
20Newsgroups	**93.90**	85.60	91.07
WebKB4	90.72	85.13	**92.04**
WebKB	77.35	61.01	**78.89**
Industry Sector	**66.49**	56.32	65.96
Reuters-top10	71.46	**81.65**	81.13
Reuters-2	88.65	87.88	**92.43**
CSTR	78.21	**90.85**	88.71
K-dataset	77.69	86.14	**91.90**
TDT2	88.41	91.59	**93.85**

Table 5. Performance Comparison Table on Text Datasets. NB stands for Naive Bayes, LDA for Linear Discriminant Analysis, SVM for Support Vector Machine.

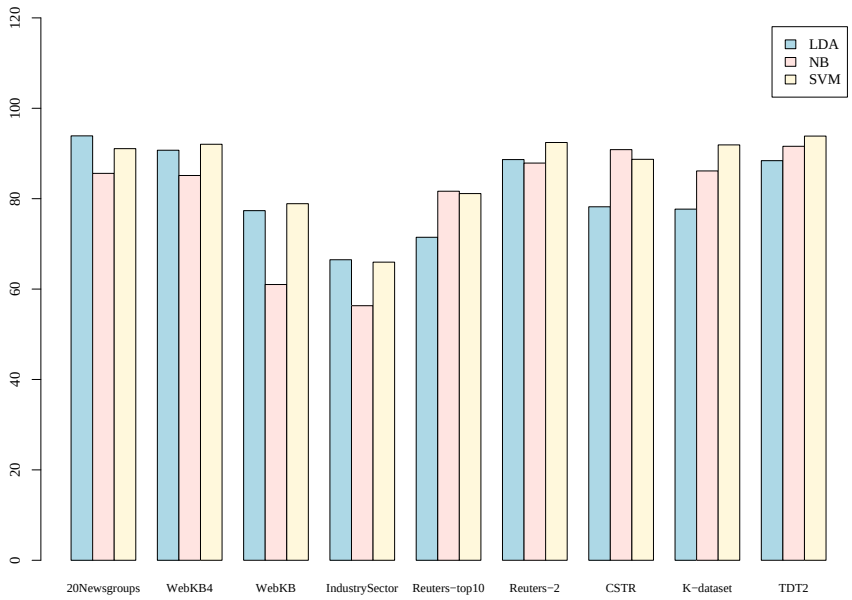

Fig. 1. Performance Comparison on Text Datasets.

6.3. Experimental Results on Text Datasets

We compared LDA with Naive Bayes and Support Vector Machine on the exactly same datasets with the same training and test settings. SVMTorch (Collobert & Bengio, 2001) [9] is used for experiments involving SVMs. SVMTorch handles multi-class classification using one-versus-the-rest decomposition. The Naive Bayes classifier is built on the Bow (A Toolkit for Statistical Language Modeling,

[9] `http://old-www.idiap.ch/learning/SVMTorch.html`.

Data Sets	LDA Training	LDA Prediction	SVM Training	SVM Prediction
20Newsgroups	172	7	270	64
WebKB4	63	1	115	55
WebKB	95	1	1108	103
Industry Sector	88	6	424	80
Reuters-top 10	61	1	94	19
Reuters-2	96	1	567	85
CSTR	4	1	8	3
K-dataset	63	1	85	48
TDT2	22	5	90	27

Table 6. Time Table on Text Datasets. Each number is training/prediction time in seconds.

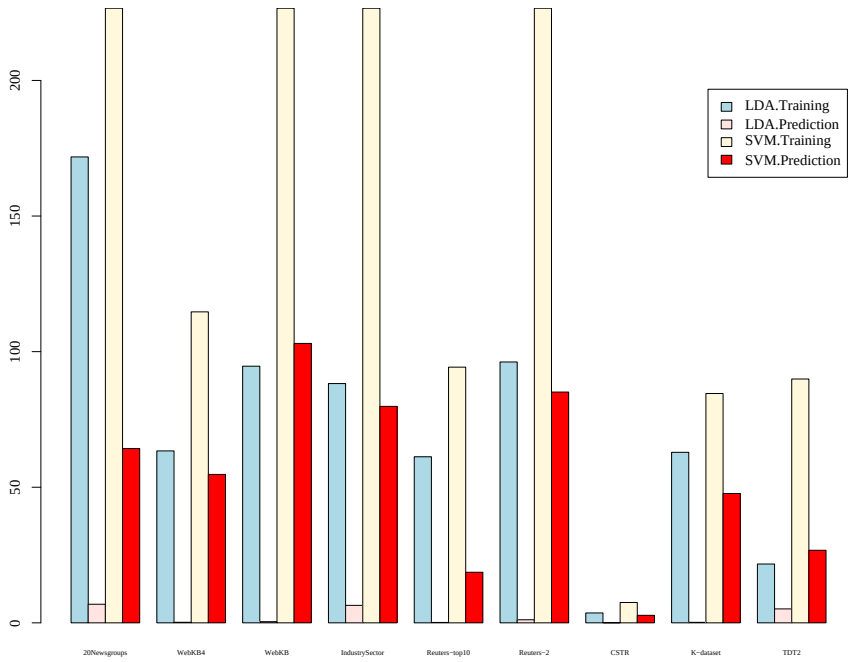

Fig. 2. Time Comparison on Text Datasets.

Text Retrieval, Classification and Clustering) University [10]. For SVM, we used the linear kernel.

Table 5 gives performances comparisons. SVM achieves the highest performance on WebKB4, WebKB, Reuters-2, K-dataset and TDT2. LDA achieves the best performance on 20Newsgroups and Industry Sector while NB has the highest performance on On Reuters-top 10 and CSTR. On 20Newsgroups, the performance of LDA is 93.90%, which is about 8% higher than that of Naive Bayes and 2% higher than SVM. The results of LDA and SVM are quite close to each other

[10] The tool can be downloaded at *http://www-2.cs.cmu.edu/∼mccallum/bow/*.

on WebKB4, WebKB, Industry Sector, Reuters-2 and TDT2. On Reuters-top 10 and K-dataset, LDA is beaten by Naive Bayes and SVM by about 10%. The comparisons show that, although there is no single winner on all the datasets, LDA is a viable and competitive algorithm in text categorization domain.

LDA is very efficient and most experiments are done in several seconds. Table 6 summarizes the running time for all the experiments of LDA and SVM. The time saving of LDA is quite obvious.

7. Conclusions and Future Work

In this paper, we investigate the use discriminant analysis for multi-class classification. Our experiments have shown that LDA is a simple efficient yet accurate approach for multi-class classification problems. The precision of LDA approach is comparable to other approaches such as SVM, however, the time consumption of LDA approach is much less than other approaches.

There are several avenues for future research directions. First, as demonstrated in the experiments, LDA performed poorly on several datasets, such as Letter, owing to the low dimensionality of the datasets. One possible solution to increase the VC-dimension is introducing kernel functions.

Second, recently people have used Generalized Singular Value Decomposition (GSVD) (Loan, 1976) to solve generalized eigenvalue problems. A *Generalized Singular Value Decomposition* (GSVD) is an SVD of a sequence of matrices and it brings several favorable computation properties such as stability for computation (Bai, 1992). One promising direction is to explore the use of GSVD to improve the performance of LDA.

Third, LDA can be seen as an efficient dimension reduction method since it transforms the original data into a low dimensional space determined by the $n-1$ different eigenvectors where n is the number of different classes. Understanding the interpretations of the LDA basis vectors could potentially lead to efficient dimension reduction. For example, it would be very interesting to study what are the original genes that the discriminating features correspond to in ALL-AML and ALL datasets.

Fourth, as we mentioned in Section 6, it would be interesting to explore how the performance correlates with different feature selection methods and the number of words selected when applying LDA for text categorization.

Finally, there are also some other possible extensions such as using random projection to reduce the dimensionality before applying discriminant analysis (Papadimitriou, et al., 1998).

Acknowledgment

The authors would like to thank the anonymous reviewers for their invaluabale comments.

References

E. L. Allwein, et al. (2000). 'Reducing Multiclass to Binary: A Unifying Approach for Margin Classifiers'. *JMLR* **1**:113–141.

Z. Bai (1992). 'The CSD, GSVD, Their Applications and Computations'. Tech. Rep. IMA Preprint Series 958, Minneapolis, MN.

D. Barber & C. K. I. Williams (1997). 'Gaussian Processes for Bayesian Classification via Hybrid Monte Carlo'. In M. C. Mozer, M. I. Jordan, & T. Petsche (eds.), *Advances in Neural Information Processing Systems*, vol. 9, p. 340. The MIT Press.

C. Blake & C. Merz (1998). 'UCI Repository of machine learning databases'.

D. Boley, et al. (1999). 'Document Categorization and Query Generation on the World Wide Web Using WebACE'. *AI Review* **13**(5-6):365–391.

L. Bottou, et al. (1994). 'Comparison of classifier methods: A case study in handwriting digit recognition'. In *International Conference on Pattern Recognition*, pp. 77–87.

L. Breiman, et al. (1993). *Classification and Regression Trees*. Chapman & Hall, New York.

L. Chen, et al. (2000). 'A new LDA-based face recognition system which can solve the small sample size problem'. *Pattern Recognition* **33**(10):1713–1726.

R. Collobert & S. Bengio (2001). 'SVMTorch: Support Vector Machines for Large-Scale Regression Problems'. *Journal of Machine Learning Research* **1**:143–160.

K. Crammer & Y. Singer (2000). 'On the Learnability and Design of Output Codes for Multiclass Problems'. In *Computational Learing Theory*, pp. 35–46.

K. Crammer & Y. Singer (2001). 'Ultraconservative Online Algorithm for Multiclass Problems'. In *COLT 2001*, pp. 99–115.

T. G. Dietterich & G. Bakiri (1995). 'Solving Multiclass Learning Problems via Error-Correcting Output Codes'. *Journal of Artificial Intelligence Research* **2**:263–286.

R. O. Duda, et al. (2001). *Pattern Classification*. John Wiley & Sons, Inc.

S. Dzeroski & B. Zenko (2002). 'Stacking with Multi-response Model Trees'. In *Proceedings of The Third International Workshop on Multiple Classifier Systems, MCS*, pp. 201–211. Springer-Verlag.

R. Fisher (1936). 'The use of multiple measurements in taxonomic problems'. *Annals of Eugenics* (7):179–188.

J. Friedman (1996). 'Another approach to polychotomous classification'. Tech. rep., Department of Statistics, Stanford.

K. Fukunaga (1990). *Introduction to statistical pattern recognition*. Academic Press.

P. Gallinari, et al. (1991). 'On the relations between discriminant analysis and multilayer perceptrons'. *Neural Networks* **4**(3):349–360.

R. Ghani (2001). 'Combining Labeled and Unlabeled data for Text Classification with a Large Number of Categories'. In *ICDM-01*.

M. N. Gibbs & D. J. C. MacKay (2000). 'Variational Gaussian Process Classifiers'. *IEEE Transactions on Neural Networks* **11**(6):1458.

S. Godbole, et al. (2002). 'Scaling multi-class support vector machine using inter-class confusion'. In *KDD-02*.

T. R. Golub, et al. (1999). 'Molecular Classification of Cancer: Class Discovery and Class Prediction by Gene Expression monitoring'. *Science* **286**:531–536.

V. Guruswami & A. Sahai (1999). 'Multiclass learning, boosting, and error-correcting codes'. In *Proceedings of the twelfth annual conference on Computational learning theory*, pp. 145–155. ACM Press.

E.-H. Han, et al. (1998). 'WebACE: A Web Agent for Document Categorization and Exploration'. In *Agents-98*.

T. Hastie & R. Tibshirani (1998). 'Classification by Pairwise Coupling'. In M. I. Jordan, M. J. Kearns, & S. A. Solla (eds.), *Advances in Neural Information Processing Systems*, vol. 10. The MIT Press.

T. Hastie, et al. (2001). *The Elemetns of Statistical Learning: Data Mining, Inference, Prediction*. Springer.

C.-W. Hsu & C.-J. Lin (2002). 'A comparison of methods for multi-class support vector machines'. *IEEE Transactions on Neural Networks* (13):415–425.

R. Huang, et al. (2002). 'Solving the Small Size Problem of LDA'. In *ICPR 2002*.

T. Joachims (2001). 'A Statistical Learning Model of Text Classification with Support Vector Machines'. In *SIGIR-01*.

R. A. Johnson & D. W. Wichern (1988). *Applied Multivariate Statistical Analysis*. Prentice Hall.

T. Kawatani (2002). 'Topic Difference Factor Extraction Between Two Document Sets of its Application to Text Categorization'. In *SIGIR 2002*, pp. 137–144.

U. H.-G. Kreeel (1999). 'Pairwise Classification and Support Vector Machines'. In *Advances in Kernel Methods*. MIT Press.

Y. Lee, et al. (2001). 'Multicategory support vector machines'. In *Proceedings of the 33rd Symposium on the Interface*.
C. V. Loan (1976). 'Generalizing the singular value decomposition'. *SIAM J. Num. Anal.* **13**:76–83.
M. Loog, et al. (2001). 'Multiclass linear dimension reduction by weighted pairwise fisher criteria'. *IEEE Transaction on Pattern Analysis and Machine Intelligence* **23**(7):762–766.
A. M. Martinez & A. C. Kak (2001). 'PCA versus LDA'. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **23**(2):228–233.
A. K. McCallum (1996). 'Bow: A toolkit for statistical language modeling, text retrieval, classification and clustering'. http://www.cs.cmu.edu/ mccallum/bow.
G. J. McLachlan (1992). *Discriminant Analysis and Statistical Pattern Recognition*. John Wiley & Sons, Inc.
S. Mika, et al. (1999). 'Fisher Discriminant Analysis with Kernels'. In Y.-H. Hu, J. Larsen, E. Wilson, & S. Douglas (eds.), *Neural Networks for Signal Processing IX*, pp. 41–48. IEEE.
T. M. Mitchell (1997). *Machine Learning*. The McGraw-Hill Companies,Inc.
M. O. Noordewier, et al. (1991). 'Training Knowledge-Based Neural Networks to Recognize Genes'. In *Advances in Neural Information Processing Systems*.
C. H. Papadimitriou, et al. (1998). 'Latent Semantic Indexing: A Probabilistic Analysis'. pp. 159–168.
H. Park, et al. (2001). 'Dimension Reduction for Text Data Representation Based on Cluster Structure Preserving Projection'. Tech. Rep. 01-013, Department of Computer Science, University of Minnesota.
J. Platt, et al. (2000). 'Large Margin DAGs for Multiclass Classification'. In S. Solla, T. Leen, & K.-R. Muller (eds.), *Advances in Neural Information Processing Systems*, vol. 12. MIT Press.
J. Quinlan (1993). *C4.5: Programs for machine learning*. Morgan Kaufmann.
J. D. M. Rennie (2001). 'Improving Multi-class Text Classification with Naive Bayes'. Master's thesis, Massachusetts Institute of Technology.
D. Roth, et al. (2000). 'Learning to Recognize Objects'. In *CVPR*, pp. 724–731.
V. Roth (2001). 'Probabilistic Discriminative Kernel Classifiers for Multi-class Problems'. *Lecture Notes in Computer Science* **2191**:246–253.
R. E. Schapire & Y. Singer (2000). 'BoosTexter: A Boosting-based System for Text Categorization'. *Machine Learning* **39**(2/3):135–168.
B. Schölkopf & A. J. Smola (2002). *Learning with Kernels*. MIT Press.
SGI (2000). 'MLC++: Datasets from UCI'.
A. Shashua (1999). 'On the equivalence between the support vector machine for classification and sparsified Fisher's linear discriminant'. *Neural Processing Letters* **9**(2):129–139.
D. L. Swets & J. Weng (1996). 'Using Discriminant Eigenfeatures for Image Retrieval'. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **18**(8):831–836.
TDT2 (1998). 'Nist Topic detection and tracking corpus'. http://www.nist.gove/speech/tests/tdt/tdt98/index.htm.
V. N. Vapnik (1998). *Statistical Learning Theory*. Wiley, New York.
J. Weston & C. Watkins (1998). 'Multi-class support vector machines'. Tech. rep., Department of Computer Science, University of London.
C.-H. Yang, et al. (2000). 'Efficient routability check algorithms for segmented channel routing'. *ACM Transactions on Design Automation of Electronic Systems.* **5**(3):735–747.
Y. Yang & X. Liu (1999). 'A re-examination of text categorization methods'. In *SIGIR-99*.
Y. Yang & J. O. Pederson (1997). 'A Comparative study on Feature selection in text categorization'. In *ICML-97*.
B. Zadrozny (2001). 'Reducing multiclass to binary by coupling probability estimates'. In *NIPS*.
W. Zhao, et al. (1999). 'Subspace Linear Discriminant Analysis for Face Recognition'. Tech. Rep. CAR-TR-914., University of Maryland, College Park.