Benchmarking a Reduced Multivariate Polynomial Pattern Classifier

Kar-Ann Toh, *Senior Member*, *IEEE*, Quoc-Long Tran, and Dipti Srinivasan, *Senior Member*, *IEEE*

Abstract—A novel method using a reduced multivariate polynomial model has been developed for biometric decision fusion where simplicity and ease of use could be a concern. However, much to our surprise, the reduced model was found to have good classification accuracy for several commonly used data sets from the Web. In this paper, we extend the single output model to a multiple outputs model to handle multiple class problems. The method is particularly suitable for problems with small number of features and large number of examples. Basic component of this polynomial model boils down to construction of new pattern features which are sums of the original features and combination of these new and original features using power and product terms. A linear regularized least-squares predictor is then built using these constructed features. The number of constructed feature terms varies linearly with the order of the polynomial, instead of having a power law in the case of full multivariate polynomials. The method is simple as it amounts to only a few lines of Matlab code. We perform extensive experiments on this reduced model using 42 data sets. Our results compared remarkably well with best reported results of several commonly used algorithms from the literature. Both the classification accuracy and efficiency aspects are reported for this reduced model.

Index Terms—Pattern classification, parameter estimation, pattern recognition, multivariate polynomials, and machine learning.

1 Introduction

Pattern classification is an important field of research since it encompasses a wide range of information processing problems of great application significance. These applications include human identity recognition, speech recognition, multimedia data retrieval, handwritten character recognition, bioinformatics, medical diagnosis, data fusion, data mining, process control, and many other fields of machine intelligence. The significance of such applications is very well paraphrased by H. Simon (see e.g., [1]): "The more relevant patterns at your disposal, the better your decisions will be."

While the statistical approach (see e.g., [2]) has received considerable attention, many estimators or approximators (see e.g., [3], [4]) can be used for pattern classification. The Multivariate Polynomial model (MP) provides an effective way to describe complex nonlinear input-output relationships since it is tractable for optimization, sensitivity analysis, and prediction of confidence intervals. With appropriate incorporation of certain decision criteria into the model output, MP can be used for pattern classification. However, for high-dimensional and high-order systems, multivariate polynomial regression becomes impractical due to its prohibitive number of product terms. This is especially true for the case of a full interaction model.

To circumvent this dimensionality problem, two main approaches can be identified. The first approach is by means of a compact universal basis function (e.g., perceptron and radial basis function) network which is usually nonlinear in parameters. The neural networks, radial basis function networks, Fourier series, and wavelets are good examples accounting for much success for such an approach. Since these formulations are usually nonlinear in parameters, estimation of the parameters is nontrivial and usually an iterative process (e.g., backpropagation of errors and gradient descent) is adopted to solve the problem. Although solutions in the local sense can be obtained most of the time, they are sensitive to initial estimates.

Another example for the first approach, which focuses on the use of polynomials, is seen in [5] where a ridge polynomial network was claimed to uniformly approximate any continuous function on a compact set in multidimensional input space. This network is a generalization of their earlier pi-sigma network [6] (which is not a universal approximator) proposed to circumvent the explosion of weight parameters in a High-order Processing Unit (HPU) network. Here, we note that, when using a linear activation function and adopting a single layer network architecture, HPU reduces to a multivariate polynomial expansion. The core idea of pi-sigma network uses products of sums of input component instead of sums of products as in HPU's processing units. Although the number of weight parameters is largely reduced as compared to those in HPUs, training of the network remains a nontrivial task since the formulation is nonlinear with respect to its parameter space. An iterative gradient type of search is usually applied to perform a search for some local solutions.

The second approach is by use of dimension reduction techniques compromising some approximation capability. In [7], the class of polynomial networks whose output is the weighted sum of several basis monomials is considered. Two dimension reduction methods, namely, the redundancy removal and the random dimension reduction are combined in the proposed polynomial networks to solve a speaker verification problem. Apart from the above direct dimension reduction means which sacrifice much approximation in the high frequency band, another way to tackle the problem is by piecewise fitting of data. This method splits the estimate into

- *K.-A. Toh and Q.-L. Tran are with the Institute for Infocomm Research, 21 Heng Mui Keng Terrace, Singapore 119613. E-mail: {katoh, qltran}@i2r.a-star.edu.sg.*
- *Q.-L. Tran and D. Srinivasan are with the Department of Electrical and Computer Engineering, National University of Singapore, Singapore 117576. E-mail: dipti@nus.edu.sg.*

Manuscript received 9 Sept. 2003; revised 30 Nov. 2003; accepted 5 Dec. 2003. Recommended for acceptance by L. Kuncheva.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number TPAMI-0266-0903.

several smaller dimensional pieces and performs estimation on the separate pieces. The results of these pieces are then combined to produce the final overall estimate. In [8], a nonparametric function estimate called Smoothed and Unsmoothed Piecewise-Polynomial Regressions Trees is described. The method recursively partitions the regressor space until the data in each piece are adequately fitted by a polynomial of a fixed order. The final estimate is then obtained by averaging the polynomial pieces using smooth weight functions which diminish rapidly to zero outside each associated partition. Although the approximation capability is maintained, the choice of polynomial orders and the partitioning remain a critical task in arriving at good estimates.

In view of the complexity and possibly tedious effort involved in the application of above methods, our initial proposal of a simple parametric reduced multivariate polynomial model is to circumvent this dimension explosion problem while maintaining some approximation capability mainly for multimodal biometric decision fusion applications [9]. Much to our surprise, the reduced model was found to have good performance for several classification problems from the UCI machine learning repository (e.g., Iris-Plant and Wisconsin-Breast-Cancer [10]). We are thus motivated to carry out further experiments on more data sets from the repository to understand better the empirical aspects regarding the classification accuracy of the reduced model.

In this paper, we evaluate empirically the performance of the proposed reduced multivariate polynomial model using commonly available data sets taken from the UCI machine learning repository [10]. Both the accuracy aspects and the efficiency aspects will be addressed in our experiments. Comparisons with well-used algorithms in the literature are mainly made reference to [11] and [12] which performed extensive experiments on 35 and 16 data sets respectively from UCI machine learning repository. Since there are some overlaps between these two data sets, our data sets for experimentation total up to 42. The method is particularly useful for problems with large number of training samples and relatively small number of pattern features. For the experimented data sets, the total number of samples ranges from 47 to 20,000 and the number of pattern features ranges from 4 to 64.

The paper is organized as follows: In the following section, the multivariate polynomial regression is introduced since much of the subsequent developments adopt a similar estimation process. Several existing polynomial models are discussed in this section before a reduced model is derived in Section 3. In Section 4, the reduced model is formulated to solve the pattern classification problem. The model is then extended to cater for multiple pattern class labels in the same section. With the classification algorithm in place, in Section 5, the data sets used and the existing algorithms compared are briefly accounted for. The performance evaluation criteria are then spelled out in Section 6 prior to the presentation of experimental results in Section 7. Finally, in Section 8, some concluding remarks are drawn.

2 Multivariate Polynomial Regression

In this section, we shall first recall the multivariate polynomial function model before introducing a reduced model for pattern classification. This is because the least-squares solution form for parameter estimate in the multivariate polynomial function model can be applied in a straightforward manner in subsequent development with minor modification.

2.1 Multivariate Polynomial Model

The general multivariate polynomial model can be expressed as

$$g(\boldsymbol{\alpha}, \boldsymbol{x}) = \sum_{i}^{K} \alpha_i x_1^{n_1} x_2^{n_2} \cdots x_l^{n_l}, \tag{1}$$

where the summation is taken over all nonnegative integers $n_1, n_2, \ldots, n_l$ for which $n_1 + n_2 + \cdots + n_l \leq r$ with r being the order of approximation. $\boldsymbol{\alpha} = [\alpha_1, \ldots, \alpha_K]^T$ is the parameter vector to be estimated and $\boldsymbol{x}$ denotes the regressor vector $[x_1, \ldots, x_l]^T$ containing l inputs. K is the total number of terms in $g(\boldsymbol{\alpha}, \boldsymbol{x})$.

Without loss of generality, consider a second-order bivariate polynomial model ($r = 2$ and $l = 2$) given by

$$g(\boldsymbol{\alpha}, \boldsymbol{x}) = \boldsymbol{\alpha}^T p(\boldsymbol{x}), \tag{2}$$

where

$$\boldsymbol{\alpha} = [\alpha_1 \ \alpha_2 \ \alpha_3 \ \alpha_4 \ \alpha_5 \ \alpha_6]^T, \tag{3}$$

and

$$p(\boldsymbol{x}) = [1 \ \ x_1 \ \ x_2 \ \ x_1^2 \ \ x_1 x_2 \ \ x_2^2]^T. \tag{4}$$

Given m data points with $m > K$ ($K = 6$ here) and using the least-squares error minimization objective given by

$$s(\boldsymbol{\alpha}, \boldsymbol{x}) = \sum_{i=1}^{m} [y_i - g(\boldsymbol{\alpha}, \boldsymbol{x}_i)]^2 = [\mathbf{y} - P\boldsymbol{\alpha}]^T [\mathbf{y} - P\boldsymbol{\alpha}], \tag{5}$$

the parameter vector $\boldsymbol{\alpha}$ can be estimated from

$$\boldsymbol{\alpha} = (P^T P)^{-1} P^T \mathbf{y}, \tag{6}$$

where $P \in \mathcal{R}^{m \times K}$ denotes the Jacobian matrix of $p(\boldsymbol{x})$:

$$P = \begin{bmatrix} 1 & x_{1,1} & x_{2,1} & x_{1,1}^2 & x_{1,1}x_{2,1} & x_{2,1}^2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x_{1,m} & x_{2,m} & x_{1,m}^2 & x_{1,m}x_{2,m} & x_{2,m}^2 \end{bmatrix}, \tag{7}$$

and $\mathbf{y} = [y_1, \ldots, y_m]^T$ is the known inference vector from training data. In (7), the first and second subscripts of the matrix elements $x_{j,k}$ ($j = 1, 2, \ k = 1, \ldots, m$) indicate the number of inputs and the number of instances, respectively.

It is noted here that (6) involves computation of the inverse of a matrix, the problem of multicollinearity may arise if some linear dependence among the elements of $\boldsymbol{x}$ are present. A simple approach to improve numerical stability is to perform a weight decay regularization using the following error objective:

$$\begin{aligned} s(\boldsymbol{\alpha}, \boldsymbol{x}) &= \sum_{i=1}^{m} [y_i - g(\boldsymbol{\alpha}, \boldsymbol{x}_i)]^2 + b\|\boldsymbol{\alpha}\|_2^2 \\ &= [\mathbf{y} - P\boldsymbol{\alpha}]^T [\mathbf{y} - P\boldsymbol{\alpha}] + b\,\boldsymbol{\alpha}^T \boldsymbol{\alpha}, \end{aligned} \tag{8}$$

where $\|\cdot\|_2$ denotes the l_2-norm and b is a regularization constant.

Minimizing the new objective function (8) results in

$$\boldsymbol{\alpha} = (P^T P + bI)^{-1} P^T \mathbf{y}, \tag{9}$$

where $P \in \mathcal{R}^{m \times K}$, $\mathbf{y} \in \mathcal{R}^{m \times 1}$ and I is a $(K \times K)$ identity matrix. This addition of a bias term into the least-squares regression model is also termed as *ridge regression* [13].

TABLE 1
Number of Terms in Multinomials and Multivariate Polynomials

	Number of terms in MN						Number of terms in MP
$l\backslash r$	1	2	3	4	5	$\cdots$	(for $r=5$)
2	2	3	4	5	6	$\cdots$	20
3	3	6	10	15	21	$\cdots$	55
4	4	10	20	35	56	$\cdots$	125
5	5	15	35	70	126	$\cdots$	251

3 Reduced Multivariate Polynomial Model

Grounded on Weierstrass's approximation theory (see e.g., [14]), the above multivariate polynomial regression provides an effective way to describe complex nonlinear input-output relationships. However, for a rth-order model with input dimension l, the number of independent adjustable parameters would grow like l^r [4]. For medium to large sizes of data dimensions, the MP model would need a huge quantity of training data to ensure that the parameters are well determined (usually overdetermined).

In view of this problem, we resort to possible reduced models whose number of parameters do not increase exponentially and, yet, preserving the necessary classification capability.

3.1 Multinomials

A special case of multivariate polynomials is called multinomial which can be expressed as

$$(x_1+x_2+\cdots+x_l)^r=\sum\frac{r!}{n_1!n_2!\cdots n_l!}x_1^{n_1}x_2^{n_2}\cdots x_l^{n_l}, \quad (10)$$

where the summation is taken over all nonnegative integers $n_1,n_2,\ldots,n_l$ for which $n_1+n_2+\cdots+n_l=r$ with r being the order of approximation. Suppose there are a total of K terms in this multinomial model. A possible application using this multinomial model for classifier combination is to estimate the weight parameter vector $\boldsymbol{\alpha}=[\alpha_0,\alpha_1,\ldots,\alpha_K]^T$ from

$$\hat{f}_{MN}(\boldsymbol{\alpha},\boldsymbol{x})=\alpha_0+\sum_j\alpha_j(x_1^{n_1}x_2^{n_2}\cdots x_l^{n_l}),\ j=1,2,\ldots,K\ , \quad (11)$$

where the summation is taken over all nonnegative integers $n_1,n_2,\ldots,n_l$ for which $n_1+n_2+\cdots+n_l=r$. Another possibility is to lump all inputs within each power term:

$$\hat{f}_{MN}(\boldsymbol{\alpha},\boldsymbol{x})=\alpha_0+\sum_{j=1}^{r}\alpha_j(x_1+x_2+\cdots+x_l)^j. \quad (12)$$

For the general multinomial problems, the number of terms in (11) can be tabulated over the dimension l (number of inputs) and the power order r as shown in Table 1. From this table, we see that the total number of terms within a full multivariate polynomial expansion including the interacting terms, is actually the summation of the number of terms within the given multinomial order and those within all lower order multinomials, i.e., summation along the lth row up to the given order r. For instance, given a 5th-order full multivariate polynomial model, a two-inputs system contains 20 terms (see column (for $r=5$) in Table 1). Here, we see that as the number of inputs and the order increase, the number of terms in full multivariate polynomial expansion increases tremendously.

3.2 A Reduced Model

To significantly reduce the huge number of terms in multivariate polynomials, we first consider the following model:

$$\hat{f}_{MN}(\boldsymbol{\alpha},\boldsymbol{x})=\alpha_0+\sum_{j=1}^{r}(\alpha_{j1}x_1+\alpha_{j2}x_2+\cdots+\alpha_{jl}x_l)^j. \quad (13)$$

It is noted that this gives rise to a nonlinear estimation model where the weight parameters (α_{jk}, $j=1,\ldots,r$, $k=1,\ldots,l$) may not be estimated in a straight-forward manner. Although an iterative search can be formulated to obtain some solutions, there is no guarantee that these solutions are global. To circumvent this problem, a linearized model is considered.

Given two points $\boldsymbol{\alpha}$ and $\boldsymbol{\alpha}_1$ on the multinomial function which is differentiable. By the Mean Value Theorem, the multinomial function $f(\boldsymbol{\alpha})=(\alpha_{j1}x_1+\alpha_{j2}x_2+\cdots+\alpha_{jl}x_l)^j$, $j=2,\ldots,r$ (indicating only the regressor parameter to simplify the expression) about the point $\boldsymbol{\alpha}_1$ can be written as:

$$f(\boldsymbol{\alpha})=f(\boldsymbol{\alpha}_1)+(\boldsymbol{\alpha}-\boldsymbol{\alpha}_1)^T\nabla f(\bar{\boldsymbol{\alpha}}), \quad (14)$$

where $\bar{\boldsymbol{\alpha}}=(1-\beta)\boldsymbol{\alpha}_1+\beta\boldsymbol{\alpha}$ for $0\le\beta\le1$. Let $\boldsymbol{x}=[x_1,\ldots,x_l]^T$. By omitting the reference point $\boldsymbol{\alpha}_1$ and those coefficients within $f(\boldsymbol{\alpha}_1)$ and $\nabla f(\bar{\boldsymbol{\alpha}})$ and including the summation of weighted input terms, the following multivariate model can be written:

$$\hat{f}_{RM'}(\boldsymbol{\alpha},\boldsymbol{x})=\alpha_0+\sum_{j=1}^{l}\alpha_jx_j+\sum_{j=1}^{r}\alpha_{l+j}(x_1+x_2+\cdots+x_l)^j+\sum_{j=2}^{r}(\boldsymbol{\alpha}_j^T\cdot\boldsymbol{x})(x_1+x_2+\cdots+x_l)^{j-1},\ l,r\ge2, \quad (15)$$

where the number of terms is given by $K=1+r(l+1)$.

To include more individual high-order terms for (15), the following (RM) can be written:

$$\hat{f}_{RM}(\boldsymbol{\alpha},\boldsymbol{x})=\alpha_0+\sum_{k=1}^{r}\sum_{j=1}^{l}\alpha_{kj}x_j^k+\sum_{j=1}^{r}\alpha_{rl+j}(x_1+x_2+\cdots+x_l)^j+\sum_{j=2}^{r}(\boldsymbol{\alpha}_j^T\cdot\boldsymbol{x})(x_1+x_2+\cdots+x_l)^{j-1},\ l,r\ge2. \quad (16)$$

The number of terms in this model can be expressed as: $K=1+r+l(2r-1)$. It is noted that (16) has $(rl-l)$ number of terms more than that of (15). The plots for the number of terms over different model orders for each input dimension ($l=2,3,\ldots,8$) of RM is shown in Fig. 1. For comparison purpose, the same figure includes the number of terms plotted over the model orders for a full multivariate polynomial model with input dimension two ($l=2$). Due to its simplicity, the Matlab function codes for (16) is included in Appendix A for immediate use. An example function call for (16) is also included in Appendix B to show the ease of its use.

3.3 Decision Landscapes

The classification capability of the reduced model can perhaps be inferred from its decision landscape of biometrics

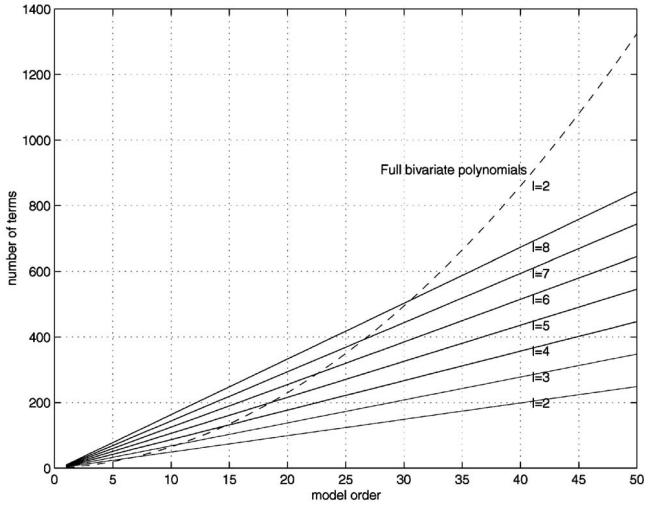

Fig. 1. Number of terms plotted over model order for different input dimensions.

data fusion (a 2-class problem) [9]. A highly localized training may end up in poor test results since the training data may not be globally representative (overfitting). Conversely, underfitting of training data may result. Fig. 2 shows the genuine and imposter classes distribution with some decision boundaries, and corresponding decision landscapes produced by a two-layer Multilayer Perceptron (MLP) with two hidden nodes, a SVM (see e.g., [15], [16]) using RBF kernel, and our proposed RM (6th-order). It is seen from this figure that the localization property of RM is somewhere between that of the selected SVM-RBF (high localization) and Neural Network (low localization).[1]

4 Pattern Classification Using the Reduced Model

The reduced model was first proposed for biometric decision fusion where the decision fusion problem was treated as a classifier fusion problem [19] with only two class labels (imposters and genuine-users) [9]. For classification problems with multiple class labels (multiclass), some modifications to the original formulation are required.

4.1 2-Class Problems

In this work on pattern classification, the regularized least-squares error objective given by (8) will be used for data training. For problems with two class labels, the target outputs can be set as "0" for class-0 and "1" for class-1. The outputs of the trained model will then be classified as $\hat{C} = 0$ if $\hat{f}_{RM} \leq 0.5$ and $\hat{C} = 1$ otherwise. The classification error rates are computed as the ratio of number of misclassified test samples over the total number of test samples using the test set which have not been used in training.

1. The localization capability depends on the size of the network and kernel units. The good application of SVM and MLP depends much on the choice of model structures with balanced localization property. The chosen size and structure of MLP and SVM are due to their good performance by combining two biometrics [9]. For fundamental model structure issues, see [17], [18] for some interesting properties on single-layer perceptron in relation to some statistical classifiers.

4.2 3-Class and Multiclass Problems

For problems with number of class labels larger than two, we shall adopt the *winner-takes-all* technique for classification as it has a uniform a priori class probability. A reduced model will be constructed for each output class with value "1" for those within the designated class and value "0" otherwise. Training is thus required to be performed N_C times when there are N_C class labels. In our implementation, the training vector can be packed into a single matrix and computation of the weight parameters can be performed in a single step. The decision for classifying a test output is then based on the highest model output value among the competing classes.

Let N_C be the total number of class labels for those multiclass problems which have more than two class labels, we have the target training vectors packed as:

$$Y = [\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_{N_C}], \tag{17}$$

where $\boldsymbol{y}_i$, $i = 1, 2, \ldots, N_C$ are defined according to each class (containing m elements which is the number of training data samples) with ones for those samples in each ith-class, and zero otherwise. In other words, each sample row of Y contains only a "1" for the corresponding class and "0" otherwise.

The Jacobian matrices P_i of the reduced model for each class $i = 1, 2, \ldots, N_C$ are the same since they all take the same inputs and, hence, no packing is necessary. Let

$$P = P_1 = P_2 = \cdots = P_{N_C}, \tag{18}$$

where

$$P_i = \left(\frac{\partial \hat{\boldsymbol{f}}_{RM}(\boldsymbol{\alpha}, \boldsymbol{x}_j)}{\partial \boldsymbol{\alpha}^T} \right)_i, \quad j = 1, 2, \ldots, m;\ i = 1, 2, \ldots, N_C;\ \ P_i \in \mathcal{R}^{m \times K}. \tag{19}$$

Then, the regularized solution from (9) can be modified as

$$\Theta = (P^T P + bI)^{-1} P^T Y, \tag{20}$$

to solve for the packed polynomial weight parameter matrix $\Theta = [\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2, \ldots, \boldsymbol{\alpha}_{N_C}]$ $(\boldsymbol{\alpha}_i \in \mathcal{R}^{K \times 1},\ i = 1, 2, \ldots, N_C)$ in a single step.

Having learned Θ, the multiclass model outputs for test can finally be computed as

$$\hat{F} = [\hat{\mathbf{f}}_{RM}(\boldsymbol{\alpha}_1, \boldsymbol{x}), \ldots, \hat{\mathbf{f}}_{RM}(\boldsymbol{\alpha}_{N_C}, \boldsymbol{x})] = P\,\Theta \tag{21}$$

using P generated from test set. For each data sample, the largest element of $\hat{F}$ will determine the output pattern class.

4.3 Model Order Selection and Regularization Parameter Setting

The neural network has been recognized to be a universal approximator (see e.g., [20], [21]). The generic nature comes from its wide span of complexity wherein at least two model structure parameters need to be determined (number of layers and number of nodes in each layer which come with many possible combinations). There would be more parameters if momentum, learning rate, and type of activation functions are considered. It is noted that determination of these model structure parameters is a nontrivial task where much research is ongoing. Unlike the neural networks, the RM has only two model structure parameters (model order and regularization parameter) and from our experience it

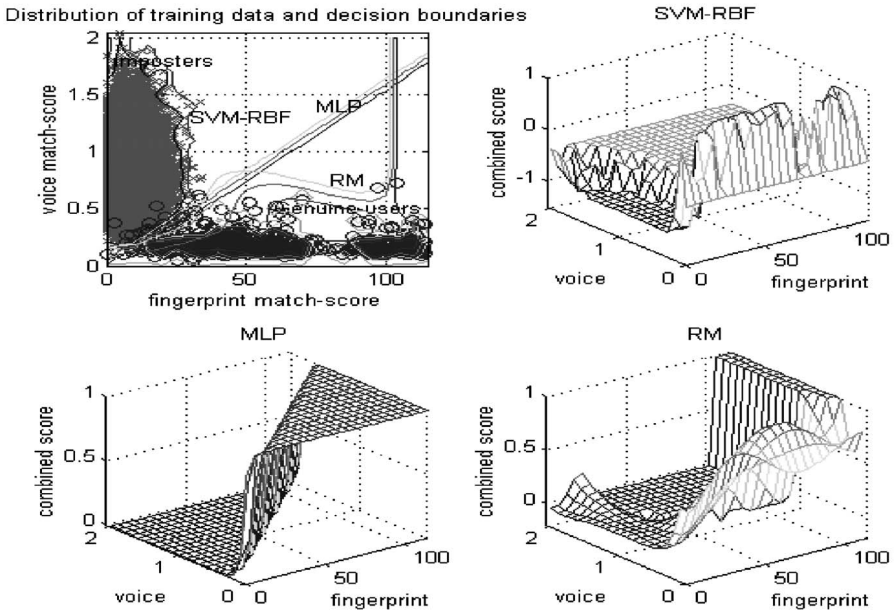

Fig. 2. Decision landscapes for SVM-RBF, MLP, and RM.

operates under reasonably narrow numerical range. The reason being that the regularization parameter (with small values) is found to be rather inert which does not affect significantly the classification error rate while maintaining its role in stabilizing the solution. The main model structure parameter is the model order which is to be chosen such that the number of weight parameters is less than the number of data samples to yield an overdetermined system. Empirically, we found that a good starting point would be $r = 6$ and $b = 10^{-4}$. While maintaining this value of b, the model order can be varied within $[1, 10]$ using cross-validations for best result. Upon finding the best model order, the b can be varied within $[10^{-6}, 1]$ for possible improvement.

5 Data Sets and Classification Algorithms

Before moving on to the experiments, a brief account on the data sets used and the classification algorithms compared is presented in this section.

5.1 Data Sets

The data sets used in our experiments are all taken from the UCI Machine Learning Repository [10] except for the StatLog-DNA data set which was obtained from [22] maintained by P. Brazdil and the Attitudes Towards Workplace Smoking Restrictions which was obtained from StatLib [23].[2] Our choice of these data sets is mainly according to [11] and [12] where several important classification algorithms and their variants were compared. A total of 42 data sets covering a wide range of applications are used in our experiments. It is noted that this data set covers 30 out of the 35 listed in [11], and all the 16 data sets listed in [12]. Those data sets not included either have a continuous nature of output where the number of classes used is not known, or have too many missing data from a single class which renders the training to be nonrepresentative.

We organize the data sets according to the number of pattern labels into three groups, namely, 2-class problems (16 sets), 3-class problems (12 sets), and multiclass problems (14 sets). The purpose is to observe possible trends related to number of pattern classes. Since many details can be found from the Web sites, we shall provide only a brief account on the data sets and summarize the size of data set and the number of pattern attributes used in Table 2.

1. *Two-class problems.* The data sets for 2-class problems include: Shuttle Landing Control, BUPA Liver-Disorders donated by R.S. Forsyth, Monk's Problems donated by S. Thrun, Pima Indians Diabetes donated by P. Turney, Tic-Tac-Toe Endgame donated by D.W. Aha, Wisconsin Breast Cancer Databases donated by O. Mangasarian and N. Street, StatLog Project data set of Heart donated by R. King, Credit Card Application Approval, Congressional Voting Records, Mushrooms Characteristics from Audobon Society Field Guide, Ionosphere radar returns from V. Sigillito, and Sonar data set.
2. *Three-class problems.* The data sets for 3-class problems include: Iris Plant from Fisher (1936), Balance Scale donated by T. Hume, Teaching Assistant Evaluation donated by W.-Y. Loh and T.-S. Lim, Thyroid Disease data sets from Garavan Institute and Stefan Aeberhard (New), Abalone Age Prediction donated by S. Waugh, Contraceptive Method Choice Prediction donated by T.-S. Lim, Housing Prices in Suburbs of Boston from CMU StatLib Library, Wine Recognition donated by S. Aeberhard, Attitude Towards Workplace Smoking Restrictions from StatLib, Waveform Data Generator from classification and regression trees book, and StatLog data set of DNA donated by R. King.
3. *Multiclass problems.* The data sets for multiclass problems include: Car Evaluation donated by M. Bohanec and B. Zupan, StatLog Project data sets

2. Data from the Attitudes Towards Smoking Legislation Survey-Metropolitan Toronto 1988, which was funded by NHRDP (Health and Welfare Canada), were collected by the Institute for Social Research at York University for Dr. Linda Pederson and Dr. Shelley Bull.

TABLE 2
Summary of Data Sets

	Database name	(i) #cases	(ii) #feat	(iii) #class	(iv) #miss
1.	Shuttle-l-control	279(15)	6	2	no
2.	BUPA-liver-disorder	345	6	2	no
3.	Monks-1	124(432)	6	2	no
4.	Monks-2	169(432)	6	2	no
5.	Monks-3	122(432)	6	2	no
6.	Pima-diabetes	768	8	2	no
7.	Tic-tac-toe	958	9	2	no
8.	Breast-cancer-Wiscn	683(699)	9(10)	2	16
9.	StatLog-heart	270	13	2	no
10.	Credit-app	653(690)	15	2	37
11.	Votes	435	16	2	yes
12.	Mushroom	5644(8124)	22	2	attr#11
13.	Wdbc	569	30	2	no
14.	Wpbc	194(198)	33	2	4
15.	Ionosphere	351	34	2	no
16.	Sonar	208	60	2	no
17.	Iris	150	4	3	no
18.	Balance-scale	625	4	3	no
19.	Teaching-assistant	151	5	3	no
20.	New-thyroid	215	5	3	no
21.	Abalone	4177	8	3(29)	no
22.	Contraceptive-methd	1473	9	3	no
23.	Boston-housing	506	12(13)	3(cont)	no
24.	Wine	178	13	3	no
25.	Attitude-smoking	2855	13	3	no
26.	Waveform	3600	21	3	no
27.	Thyroid	7200	21	3	no
28.	StatLog-DNA	3186	60	3	no
29.	Car	2782	6	4	no
30.	StatLog-vehicle	846	18	4	no
31.	Soybean-small	47	35	4	no
32.	Nursery	12960	8	4(5)	no
33.	StatLog-satimage	6435	36	6	no
34.	Glass	214	9(10)	7	no
35.	Zoo	101	17(18)	7	no
36.	StatLog-image-seg	2310	19	7	no
37.	Ecoli	336	7	8	no
38.	LED-display	6000	7	10	no
39.	Yeast	1484	8(9)	10	no
40.	Pendigit	10992	16	10	no
41.	Optdigit	5620	64	10	no
42.	Letter	20000	16	26	no

(i) Total number of instances, i.e. examples, data points, observations (given number of instances). Note: the number of instances used is larger than the given number of instances when we expand those "don't care" kind of attributes in some data sets;
(ii) Number of features used, i.e. dimensions, attributes (total number of features given);
(iii) Number of classes (assuming a discrete class variable);
(iv) Missing features.

(Vehicle Silhouettes, Landsat Satellite Image and Image Segmentation) donated by R. King, Soynean Small data set donated by Michalski, Nursery Rank Applications donated by M. Bohanec and B. Zupan, Glass Identification from USA Forensic Science Service, Zoo Data set from R. Forsyth, Ecoli Cellular Localization Sites of Protein Prediction donated by P. Horton, LED display from classification and regression trees book, Yeast Cellular Localization Sites of Protein Prediction donated by P. Horton, Pen-Based Recognition of Handwritten Digits from E. Alpaydin, F. Alimoglu, Optical Recognition of Handwritten Digits from E. Alpaydin, C. Kayna, and Letter Recognition from D. Slate.

5.2 Classification Algorithms Compared

To show the effectiveness of our proposed reduced model as compared to nonreduced ones, the classification performance of full multivariate third-order and sixth-order polynomial models (abbreviated as MP3 and MP6) are included. As for the performances of existing classification algorithms in the literature, the classification results are directly taken from [11], [12], and [24] since much similarity among the experimental conditions (including ours) can be identified, besides the fact that the best known tuning of their proposed methods have been obtained by the originators themselves. As details of experiments can be found in the references, these comparative works are only briefly described below for immediate reference.

Best-I: In the recent work by [11], the proposed best tuned algorithm ICPL (Integrated Concept Prototype Learner, which integrates instance filtering and abstraction techniques) was compared with four other algorithms namely, RT3 (an instance pruning technique), kNN (k-Nearest Neighbors), C4.5 (decision tree), and SVM-Poly (Support Vector Machine using polynomial kernel). Their experiments used a single run of the 10-fold stratified cross-validation on 35 data sets from UCI machine learning repository. Only the average classification accuracy and the data retention rate (defined as ratio of *number of prototypes learned* over *number of training instances*) are reported and no CPU times recorded. Application of the above five algorithms in the reported 35 data sets resulted in the following ranking in terms of average classification accuracy: SVM-Poly (0.878), kNN (0.875), ICPL (0.863), RT3 (0.861), and C4.5(0.842). We will compare our two settings (RM-Fixed and RM-Tuned) of the reduced model with these five algorithms using those 30 data sets with known and comparable settings. In addition to the accuracy from individual algorithm, the set of best accurate results containing individual best out of those five compared algorithms in each data set as seen in [11] is abbreviated as Best-I (Best from reference-I) and tabulated in our subsequent presentation for immediate reference. The reference [11] will be denoted as Ref-I for convenient reading.

Best-II: In [12], a total of 33 old and new classification algorithms (22 belonged to the decision trees type including the C4.5, nine belonged to the statistical type including the Nearest Neighbor, and two belonged to the neural networks type including the RBF) were compared using 16 data sets from the UCI machine learning repository. Extensive experiments were performed on these data sets and a comprehensive analysis was presented regarding the *error rates*, *ranks*, *training time*, *size of trees*, and *scalability* aspects for the compared algorithms. For the reported results, most data sets used the average 10-fold validation error rates (a single run) except for those six data sets listed in Section 6.1.2 that used the given test set to compute the error rates. Their results placed a statistical spline-based algorithm (acronymed as POL) at the top in terms of average classification accuracy even though it was ranked third last in terms of training time. It is noted that the mean rank of POL among the 33 algorithms on these data sets is found to be 8.3. This shows that no algorithm in this study is close to being uniformly most accurate across the data sets. The interested reader is referred to [12] for detailed ranking of other algorithms. The set of best accurate results containing individual best out of those 33 compared algorithms in each data set as seen in [12] is abbreviated as Best-II (Best from reference-II) in our subsequent presentation. The reference [12] will be denoted as Ref-II for convenient reading.

Best-III: In the most recent reference [24], the authors compared their proposed CLEF (a Constructive Learning method) algorithm with other five existing algorithms including C4.5, SVM-RBF (using RBF kernel), and DNC(Dynamic Node Creation, a constructive neural network). Except for the Monk-2 data set that used the training and test sets provided, all accuracy results on the totaling 20 data sets are reported using 10-fold stratified cross validation (a single run). In terms of average classification accuracy, the following ranking was established: CLEF (76.25 percent), SVM-RBF (75.81 percent), C4.5 (70.03 percent), DNC (69.39 percent), Φ-RT (67.51 percent), and Φ-DNC (65.45 percent). As some of the data sets are either different from that, in UCI but with same name or different class grouping being used, only nine data sets are found to be common to those in [11], [12] and they are also listed for our comparison. The set of best accurate results containing individual best out of those six compared algorithms in each of those nine data sets as seen in [24] is abbreviated as Best-III (Best from reference-III) in our subsequent presentation. The reference [24] will be denoted as Ref-III for convenient reading.

6 Performance Evaluation Criteria

We shall evaluate the accuracy and efficiency of the proposed algorithm empirically. The following measures will be adopted in our performance evaluation.

6.1 Accuracy

6.1.1 Ten Runs of 10-Fold Validations

In all the experiments except for six cases following that of [12], the classification errors are estimated using 10-fold stratified cross validation and this cross validation is *repeated 10 times using different random reordering of the samples* in the data set. The same set of reorderings have been used for all 10-fold experiments on RM and MP models. The minimum (min), average (ave), maximum (max), and standard deviation (std) of the classification error rates of these 10 runs of 10-fold validations are recorded and the average error rates will be used as basis for comparison for our proposed reduced model. We believe that this average value provides a less biased representation of classifier performance as compared to that obtained from a single run.

6.1.2 Training and Test Sets

According to [12], the following six data sets are partitioned into two sets, namely, the training set and the test set for experimentation: Attitude-Smoking, Waveform, Thyroid, StatLog-DNA, StatLog-sat-image, and LED-display. These data sets are considered to be large [12] as their sizes are much larger than 1,000 and the test set sizes are all at least 1,000. In these six cases, the error rates are estimated from the test sets and these error rates are compared with those in [12].

6.1.3 Accuracy Rankings

Apart form the classification error rates, the accuracy rankings of our reduced model are also tabulated for each data set with reference to those compared algorithms in [11], [12], and [24]. Similar to that of [12], for each data set, the algorithm with the lowest error rate is assigned as rank 1 and the second lowest error rate assigned as rank 2, and this continues for the rest of data sets. In cases of ties, an average rank will be assigned for those algorithms which share a similar rank. These rankings provide information regarding the relative performances of the reduced model with reference to those compared algorithms in individual data sets and it reveals whether a top ranked algorithm is close to uniformly most accurate (average rank across data sets approaches one) across the data sets.

6.2 Efficiency

6.2.1 Computational Effort

The computing effort is recorded for the training time of the proposed reduced model in terms of standard CPU time unit where each standard time unit is the CPU time taken to evaluate 1,000 times the Shekel-5 function at the point (4,4,4,4) [25]. In our experimental setup on a Pentium IV-1.8GHz computer, each standard CPU time unit is equivalent to 0.0569 seconds. Although the standard CPU time unit is supposed to be machine independence, it nevertheless depends on efficiency of implementation and computer architecture.[3] The purpose of the standard CPU time unit is to provide some hints about the computing effort for our nonoptimized Matlab implementation under the commonly used Windows environment, since according to [12], the training CPU times for different algorithms can have large difference (seconds versus days) and this cannot be attributed to implementation alone.

6.2.2 Memory Storage Requirement

The number of learning parameters to be stored for future pattern classification tasks can be an important issue especially for stand-alone applications where only limited memory is available. For model-based algorithms like in our case, the number of weight parameters to be estimated for the reduced polynomial expansion is tabulated for each data set. For decision tree algorithms, the size of the tree is directly related to storage requirement. For those decision tree algorithms as seen in [12], the reported number of leaves will be directly used as a comparison quantity.

6.2.3 Initialization and Model Structure Parameters

For many iterative algorithms especially formulated in a nonlinear fashion, initialization of estimate is a nontrivial matter since it could result in different local solutions [26]. Our proposed reduced model does not inherit this problem since it's training is a single step task and no initialization is required.

Many model-based algorithms require some model structure parameters to be selected before training can begin. For example, the neural networks require the number of layers and the number of nodes within each layer to be selected. For radial basis function networks, additional parameters like the centers and width parameters are chosen before a single-step computation of weights can be performed. In our reduced model, only two model structure parameters, namely, the model order (r) and the regularization parameter (b) are required to be preselected. From our experience, the choice of $r \in [1, 10]$ and $b \in [10^{-6}, 1]$ can produce good results in many applications. This reduces the training task to just a few trials of settings where one can simply begin with these values and then tune for better results based on validation.

3. Computing resource with vectorization can create much difference among different implementations of matrix multiplications.

TABLE 3
Matlab CPU Benchmarks (Time in Seconds)

Computers	ODE	LU	SPARSE	3-D	2-D	AVE
This computer (Pentium-IV-1.8GHz)	0.15	0.08	0.11	1.04	0.74	0.42
DEC Alpha, 600	0.71	0.31	0.43	0.94	0.73	0.62
Pentium II, NT, 400	0.76	0.46	0.44	1.61	1.19	0.89
Pentium II, Linux, 400	0.65	0.43	0.52	1.72	1.19	0.90
SGI Octane, 195	1.10	0.43	0.60	1.63	1.19	0.99
Pentium II, Win98, 350	0.84	0.51	0.50	1.34	1.85	1.01
Sparc Ultra 2, 300	0.82	0.60	0.66	1.73	1.31	1.02
Pentium II Laptop, NT, 266	1.02	0.67	0.64	1.78	2.50	1.32
Pentium Pro, Linux, 200	1.21	0.83	1.05	2.45	1.57	1.42
HP 780, 180	1.69	0.46	1.13	2.83	2.24	1.67
IBM RS6000, 167	1.42	0.50	0.77	3.39	2.98	1.81
Sparc 10, Dual 160	2.12	1.07	1.29	4.50	3.08	2.41
SGI O2, 180	2.52	1.73	1.60	3.99	2.62	2.49
Sparc 2 (circa 1992)	10.00	10.00	10.00	10.00	10.00	10.00

TABLE 4
Number of Leaves and Number of Parameters

Data-set Index	2	6	8	9	11	19	22	23	25	26	27	28	30	33	36	38
QL0	6	5	3	2	2	10	24	6	1	5	13	7	16	30	39	31
QL1	4	2	2	2	2	6	11	3	1	5	6	5	8	11	21	15
FTL	2	3	3	3	2	1	3	4	1	3	13	5	22	49	18	12
C4T	26	18	11	23	10	79	143	36	1	54	12	97	65	216	42	29
IBO	26862	23889	27829	35238	46695	5548	26153	26278	9884	24295	129	7513	9683	3174	13759	415
IMO	31247	30375	23939	34839	19838	11653	12325	25500	4849	18641	290	3582	1390	1381	2964	207
OCL	5	8	4	3	2	2	12	10	4	4	11	10	14	22	16	14
RM-Fixed	73	95	106	150	183	186	318	417	450	714	714	2001	820	2418	1512	840
RM-Tuned	21	10	49	42	51	285	261	267	45	69	327	912	820	2418	1512	90

7 Experiments

7.1 Preprocessing and Settings for the Reduced Model

We shall test the proposed reduced model using two settings: *RM-Fixed* and *RM-Tuned*. The fixed setting uses a 6th-order model with $b = 10^{-4}$. The tuned setting is the result of model order selection using 10-fold validation search for $r \in [1, 10]$ with $b = 10^{-4}$ based on the training set. The selected model order r was then used to compute the errors for all the 10 runs of 10-fold tests. As tuning for good generalization could easily open up much research issues, our purpose of this study is to show how simple the model can be and reasonably good results can be achieved from such simple model and tuning.

For all problems using the fixed setting, all the input features are normalized to values within $[0, 1]$. This input normalization is carried forward to the tuned case for all data sets except in Monk-1 problem where the first two inputs are scaled to within the range $[0, 10]$ with the rest normalized as above.

7.2 Results

7.2.1 Standard CPU-Time

Table 5 lists the model settings (model order r and regularization parameter b) and the resulted total number of weight parameters to be estimated for RM-Fixed and RM-Tuned. The standard CPU time for training a single fold of the 10-fold partitioning is tabulated to provide some hints regarding the computing speed for training each data set. For RM-Tuned, the total training CPU time (in standard unit) including the model order search using cross-validation process (labeled as CPU-CV) is also tabulated to reflect the additional tuning effort incurred. It is noted that the "Optdigit" problem took the longest training CPU-CV time which is about 24 minutes (25,140.60 standard units as seen in Table 5) running in our Pentium IV computer.

As different machines running different types of floating operations have different speeds since each machine could be optimized for a certain specific operation, we nevertheless perform a Matlab's benchmarking [27] in Table 3 and, hopefully, some clues on the relative machines' speeds can be observed. Following the approach in Ref-II, we convert the RM-Tuned's CPU time into "SPARC-2-equivalent" times (seconds).[4] The "conversion" of our CPU times for those sixteen data sets used in Ref-II [12] into "SPARC-2-equivalent" times is done using the average values for all the benchmarking functions. The converted SPARC-2 CPU seconds are listed in Table 5 for immediate reference.

The median and mean SPARC-2 times for RM-Tuned are found to be respectively 0.17 seconds and 5.14 seconds for the 16 data sets from [12]. These results show that the CPU time for training the reduced model is "either fastest or comparable" (i.e., we do not expect more than a few hundred times difference in computing time given nondrastic differences in CPU architecture) to that of the fastest reported algorithms in

4. In Ref-II, the CPU times recorded on a SPARCstation 5 and a SPARCstation 20 are converted into DEC 3000-equivalent seconds using a factor of 1.4 and 0.8, respectively. In our CPU benchmarking, the closest compatible machine from Table 3 is SPARC-2 and, hence, it is used for timing reference.

TABLE 5
Algorithm Settings and CPU Time (Standard CPU Unit: 1,000 Evaluations of Shekel-5 at (4,4,4,4))

	Date sets	RM-Fixed ($b = 0.0001$)			RM-Tuned ($b = 0.0001$)				RM-Tuned:SPARC-2
No	Name	order	no. params	CPU(std)	order	no. params	CPU(std)	CPU-CV(std)	equiv. CPU seconds
1	Shuttle-l-contr	6	73	0.0916	2	21	0.0228	25.87	0.0309
2	BUPA-liver	6	73	0.1074	2	21	0.0264	31.34	0.0358
3	Monk-1	6	73	0.0439	4	47	0.0510	14.60	0.0691
4	Monk-2	6	73	0.0439	7	86	0.0968	17.79	0.1311
5	Monk-3	6	73	0.0351	2	21	0.0123	14.78	0.0167
6	Pima-diabetes	6	95	0.3995	1	10	0.0494	105.43	0.0669
7	Tic-tac-toe	6	106	0.7091	2	30	0.7091	154.52	0.9607
8	Breast-cancer-W	6	106	0.5248	3	49	0.0443	111.58	0.0600
9	StatLog-heart	6	150	0.3924	2	42	0.0265	81.67	0.0359
10	Credit-app	6	172	1.0736	1	17	0.0967	242.00	0.1310
11	Votes	6	183	0.9253	2	51	0.0650	169.67	0.0881
12	Mushroom	6	249	21.2091	2	69	6.5961	4581.09	8.9361
13	Wdbc	6	337	2.9336	3	154	0.2837	642.57	0.3843
14	Wpbc	6	370	1.8325	1	35	0.1250	305.54	0.1693
15	Ionosphere	6	381*	2.7074	3	174	0.2269	481.18	0.3074
16	Sonar	6	667	6.5334	1	62	0.0511	1609.53	0.0692
17	Iris	6	153	0.0369	4	99	0.0334	15.83	0.0452
18	Balance-scale	6	153	0.1248	2	45	0.0283	58.61	0.0383
19	Teaching-assist	6	186	0.0404	9	285	0.1585	19.53	0.2147
20	New-thyroid	6	186	0.0511	4	120	0.0316	25.87	0.0428
21	Abalone	6	285	2.7262	7	336	3.5938	682.71	4.8687
22	Contraceptive-mthd	6	318	0.9580	5	261	0.7228	246.05	0.9792
23	Boston-housing	6	417	0.5531	4	267	0.2337	129.17	0.3166
24	Wine	6	450	0.3606	1	45	0.0088	68.30	0.0119
25	Attitude-smoking	6	450	2.2847	1	45	0.0879	531.34	0.1191
26	Waveform	6	714	3.5149	1	69	0.0351	392.30	0.0476
27	Thyroid	6	714	11.9680	3	327	3.7135	2393.60	5.0309
28	StatLog-DNA	6	2001	40.8260	3	912	8.7118	7686.27	11.8024
29	Car	6	292	1.2160	3	136	0.5109	324.89	0.6921
30	StatLog-vehicle	6	820	1.8942	6	820	1.8942	373.83	2.5662
31	Soyabean-small	6	1568*	1.4167	1	148	0.0088	199.58	1.9193
32	Nusery	6	380	11.7654	4	244	11.7654	2518.93	15.9393
33	StatLog-satimage	6	2418	39.5958	6	2418	39.5958	7392.18	53.6429
34	Glass	6	742	0.1515	2	210	0.0176	51.39	0.0238
35	Zoo	6	1358*	0.2974	1	133	0.0035	84.66	0.0047
36	StatLog-image-seg	6	1512	5.2984	6	1512	5.2984	1076.77	7.1781
37	Ecoli	6	672	0.1480	2	192	0.0193	52.44	0.0261
38	Led-display	6	840	1.0086	1	90	0.0059	329.30	0.0080
39	Yeast	6	950	0.9956	6	950	0.9956	279.49	1.3488
40	Pendigit	6	1830	32.1750	6	1830	32.1750	7600.63	43.5895
41	Optdigit	6	7110*	79.9768	3	3240	36.9114	25140.60	50.0062
42	Letter	6	4758	53.1004	6	4758	53.1004	9788.07	71.9384

* Underdetermined system, but we managed to obtain solutions from regularization.

Ref-II [12] since, among the 33 studied algorithms, the median training times was reported to range from 5 seconds (for C4.5) to 11.3 hours (for RBF) operating in a faster DEC 3000 machine [12].

7.2.2 Number of Memory Parameters

The reduced model, as compared to radial basis functions and neural networks, would use more weight parameters (memory storage required) for pattern classification since it is linear in parameters and probably noncompact. For highly nonlinear pattern classification problems, we expect to have more weight parameters in the reduced model than that in radial basis functions and neural networks. The gain from paying such price of larger number of weight parameters is its single step training that is also least-squares optimal. In view of the low memory cost nowadays, the gain in obtaining possible "good" solutions in a single step could be a significant achievement since nonlinear formulations have yet to have their global optimality characterized like those for local minima [28]. We shall show in the following that the reduced model can achieve accurate classification solutions.

As we do not have the necessary and sufficient number of weights in neural networks and radial basis function networks for pattern classification from the literature,[5] in Table 4,

5. Determining the minimum number of neurons for pattern classification or function approximation is still an unsolved problem to the best of theauthors' knowledge. These parameters related to memory storage are thus left out in our tabulation though we generally know that neural networks are highly nonlinear and they may use only a small set of weight parameters in many applications.

TABLE 6
Classification Accuracy Statistics for Two Settings of RM Using 10 Runs of 10-fold Cross Validation

No	Name	RM-Fixed				RM-Tuned			
		min	ave	max	std	min	ave	max	std
1	Shuttle-l-contr	0.9481	0.9552	0.9630	0.0045	0.9519	0.9578	0.9630	0.0038
2	BUPA-liver	0.7029	0.7132	0.7441	0.0120	0.7088	0.7274	0.7412	0.0098
3	Monk-1	0.7000	0.7208	0.7500	0.0176	0.9750	0.9867	0.9917	0.0067
4	Monk-2	0.7500	0.7694	0.7875	0.0132	0.7250	0.7669	0.7937	0.0190
5	Monk-3	0.8833	0.8950	0.9083	0.0085	0.9000	0.9150	0.9333	0.0090
6	Pima-diabetes	0.7566	0.7636	0.7697	0.0039	0.7671	0.7755	0.7829	0.0047
7	Tic-tac-toe	0.9832	0.9835	0.9842	0.0005	0.9832	0.9835	0.9842	0.0005
8	Breast-cancer-W	0.9657	0.9685	0.9716	0.0024	0.9657	0.9700	0.9731	0.0022
9	StatLog-heart	0.7667	0.7863	0.7963	0.0104	0.8296	0.8441	0.8630	0.0105
10	Credit-app	0.8516	0.8611	0.8672	0.0042	0.8625	0.8642	0.8672	0.0020
11	Votes	0.9238	0.9338	0.9429	0.0054	0.9524	0.9543	0.9571	0.0021
12	Mushroom	0.9956	0.9960	0.9963	0.0002	1.0000	1.0000	1.0000	0.0000
13	Wdbc	0.9625	0.9668	0.9732	0.0034	0.9554	0.9609	0.9679	0.0040
14	Wpbc	0.7389	0.7589	0.7722	0.0010	0.8056	0.8189	0.8278	0.0083
15	Ionosphere	0.8794	0.8879	0.8971	0.0061	0.8794	0.8882	0.8971	0.0057
16	Sonar	0.6650	0.7135	0.7550	0.0256	0.7350	0.7475	0.7750	0.0115
17	Iris	0.9667	0.9673	0.9733	0.0020	0.9733	0.9760	0.9800	0.0033
18	Balance-scale	0.9183	0.9198	0.9217	0.0014	0.9267	0.9290	0.9317	0.0013
19	Teaching-assist	0.5500	0.5957	0.6286	0.0229	0.5643	0.5807	0.5929	0.0096
20	New-thyroid	0.9190	0.9343	0.9476	0.0085	0.9143	0.9367	0.9476	0.0102
21	Abalone	0.6637	0.6658	0.6683	0.0014	0.6642	0.6660	0.6687	0.0014
22	Contraceptive-mthd	0.5411	0.5460	0.5534	0.0040	0.5425	0.5481	0.5534	0.0039
23	Boston-housing	0.7633	0.7729	0.7816	0.0065	0.7776	0.7835	0.7898	0.0040
24	Wine	0.9313	0.9494	0.9688	0.0120	0.9812	0.9875	0.9938	0.0028
25	Attitude-smoking+	–	0.6860	–	–	–	0.6950	–	–
26	Waveform+	–	0.7797	–	–	–	0.8330	–	–
27	Thyroid+	–	0.9370	–	–	–	0.9399	–	–
28	StatLog-DNA+	–	0.9325	–	–	–	0.9452	–	–
29	Car	0.8563	0.8593	0.8614	0.0015	0.8693	0.8717	0.8747	0.0018
30	StatLog-vehicle	0.8122	0.8229	0.8354	0.0069	0.8122	0.8229	0.8354	0.0069
31	Soyabean-small	0.9500	0.9500	0.9500	0.0000	0.9500	0.9500	0.9500	0.0000
32	Nusery	0.9076	0.9085	0.9090	0.0004	0.9084	0.9093	0.9101	0.0004
33	StatLog-satimage+	–	0.8815	–	–	–	0.8815	–	–
34	Glass	0.5004	0.6210	0.7012	0.0701	0.5111	0.6498	0.7131	0.0577
35	Zoo	0.8625	0.9506	0.9900	0.0344	0.8750	0.9691	1.0000	0.0327
36	StatLog-image-seg	0.9403	0.9411	0.9429	0.0010	0.9403	0.9411	0.9429	0.0010
37	Ecoli	0.7910	0.8590	0.9646	0.0473	0.7910	0.8743	0.9646	0.0480
38	Led-display+	–	0.7140	–	–	–	0.7275	–	–
39	Yeast	0.4487	0.6101	0.6980	0.0713	0.4487	0.6101	0.6980	0.0713
40	Pendigit	0.9568	0.9573	0.9580	0.0005	0.9568	0.9573	0.9580	0.0005
41	Optdigit	0.9493	0.9510	0.9523	0.0008	0.9521	0.9532	0.9549	0.0008
42	Letter	0.7404	0.7414	0.7423	0.0005	0.7404	0.7414	0.7423	0.0005
	mean(I)	0.8448	**0.8669**	0.8854	0.0112	0.8635	**0.8858**	0.9006	0.0099
	mean(II)	0.7908	**0.7984**	0.8061	0.0047	0.8052	**0.8106**	0.8159	0.0034
	mean(III)	0.8650	**0.8815**	0.8953	0.0089	0.8806	**0.8992**	0.9089	0.0079
	mean(All)	0.8184	0.8364	0.8514	0.0093	0.8361	0.8534	0.8653	0.0080

+ : Accuracy measured from the given training and test set instead of 10-fold validation.

we list only the number of leaves for some algorithms as seen in [12] along side with the number of weight parameters used in our RM-Fixed and RM-Tuned algorithms which we think could be comparable though not so directly. The first row of the table lists the data set indices and the first column lists some algorithms as seen in [12]. The algorithms with smallest number of leaves (QL1, FTL, and OCL) are listed together with those with largest number of leaves (IBO and IMO). The most accurate among the decision tree algorithms is QL0 with average accuracy of 0.792 is also included in Table 4 along side with an implementation of the well-known C4.5 algorithm with average accuracy of 0.780 for immediate reference. The storage sizes for RM-Fixed and RM-Tuned are seen to be of medium requirement from this tabling. The number of weight parameters is seen to be relatively large for high-dimensional multiclass problems. It is noted here that, for the given medium number of parameters used, the RM-Tuned algorithm scores an average accuracy (for 10 runs of 10-folds) of 0.811 which is above the most accurate POL (a statistical algorithm) with average accuracy of 0.805 (for a single run of 10-folds) [12]. The RM-Fixed scores an average accuracy of 0.798 which ranks right after POL (see Table 9).

TABLE 7
Classification Accuracy Statistics for Two Full-Order Multivariate Polynomial Models Using 10 Runs of 10-Fold Cross Validation

No	Name	MP3 (full 3-rd order)				MP6 (full 6-th order)			
		min	ave	max	std	min	ave	max	std
1	Shuttle-l-contr	0.9815	0.9885	0.9926	0.0026	0.9889	0.9941	0.9963	0.0030
2	BUPA-liver	0.6735	0.6950	0.7176	0.0123	0.6794	0.6979	0.7206	0.0114
3	Monk-1	0.7250	0.7567	0.8250	0.0278	0.7250	0.7633	0.8083	0.0248
4	Monk-2	0.6062	0.6506	0.6813	0.0233	0.5875	0.6262	0.6937	0.0412
5	Monk-3	0.8750	0.9025	0.9417	0.0167	0.6417	0.6792	0.7167	0.0230
6	Pima-diabetes	0.7237	0.7321	0.7421	0.0070	*	*	*	*
7	Tic-tac-toe	0.9832	0.9835	0.9842	0.0005	*	*	*	*
8	Breast-cancer-W	0.8776	0.8930	0.9075	0.0095	*	*	*	*
9	StatLog-heart	0.6407	0.6689	0.7000	0.0185	*	*	*	*
10	Credit-app	*	*	*	*	*	*	*	*
11	Votes	*	*	*	*	*	*	*	*
12	Mushroom	*	*	*	*	*	*	*	*
13	Wdbc	*	*	*	*	*	*	*	*
14	Wpbc	*	*	*	*	*	*	*	*
15	Ionosphere	*	*	*	*	*	*	*	*
16	Sonar	*	*	*	*	*	*	*	*
17	Iris	0.9667	0.9713	0.9800	0.0052	0.9533	0.9613	0.9733	0.0050
18	Balance-scale	0.9150	0.9172	0.9233	0.0026	0.8617	0.8670	0.8750	0.0046
19	Teaching-assist	0.4929	0.5614	0.6286	0.0382	0.4929	0.5414	0.5714	0.0216
20	New-thyroid	0.9429	0.9500	0.9571	0.0049	0.9381	0.9448	0.9667	0.0086
21	Abalone	0.6620	0.6657	0.6685	0.0019	*	*	*	*
22	Contraceptive-mthd	0.5212	0.5353	0.5534	0.0095	*	*	*	*
23	Boston-housing	0.7347	0.7555	0.7694	0.0101	*	*	*	*
24	Wine	0.9187	0.9406	0.9563	0.0106	*	*	*	*
25	Attitude-smoking+	*	*	*	*	*	*	*	*
26	Waveform+	*	*	*	*	*	*	*	*
27	Thyroid+	*	*	*	*	*	*	*	*
28	StatLog-DNA+	*	*	*	*	*	*	*	*
29	Car	0.9347	0.9363	0.9383	0.0012	0.9852	0.9866	0.9877	0.0009
30	StatLog-vehicle	*	*	*	*	*	*	*	*
31	Soyabean-small	*	*	*	*	*	*	*	*
32	Nusery	0.9591	0.9597	0.9605	0.0004	*	*	*	*
33	StatLog-satimage+	*	*	*	*	*	*	*	*
34	Glass	0.5463	0.6643	0.7678	0.0701	*	*	*	*
35	Zoo	*	*	*	*	*	*	*	*
36	StatLog-image-seg	*	*	*	*	*	*	*	*
37	Ecoli	0.8067	0.8486	0.8956	0.0267	*	*	*	*
38	Led-display+	0.7297	0.7346	0.7390	0.0029	*	*	*	*
39	Yeast	0.4487	0.6075	0.7006	0.0806	*	*	*	*
40	Pendigit	*	*	*	*	*	*	*	*
41	Optdigit	*	*	*	*	*	*	*	*
42	Letter	*	*	*	*	*	*	*	*

* : Either matrix size too large to be computed or singularity of matrix encountered.

+ : Accuracy measured from the given training and test set instead of 10-fold validation.

7.2.3 Accuracy Statistics for the Proposed Reduced Model Using Fixed and Tuned Settings

The classification error rates using the fixed and the tuned settings of the reduced model are presented in Table 6 in terms of the accuracy statistics (minimum, average, maximum, and standard deviation of one minus error rates) across the 10 runs of 10-fold cross validation for each data set. In the last four rows of the table, we present also the means taken with reference to those data sets used in Ref-I, Ref-II, Ref-III, and that for all the 42 data sets. In this table, we see that the accuracy differences between the fixed and the tuned reduced model are quite uniformly close to 0.02 across different data set groupings (mean(I), mean(II), mean(III), and mean(All)) for min, ave, and max values. This suggests that the tuning provides an average of 2 percent accuracy improvement over the fixed setting.

To show the effectiveness of the reduced model in handling those high-dimensional problems, two full order multivariate polynomial models with similar weight decay regularization are experimented using the same validation data sets from the 10 runs. The accuracy results for the third-order and sixth-order models (MP3 and MP6) are shown in Table 7. It is seen from this table that, for 2-class problems, MP3 and MP6 broke down at input dimensions 15 and 8, respectively, in our experimental setup. This is also generally true for 3-class and multiclass problems with higher possibility of matrix singularity due to packing of multiple outputs.

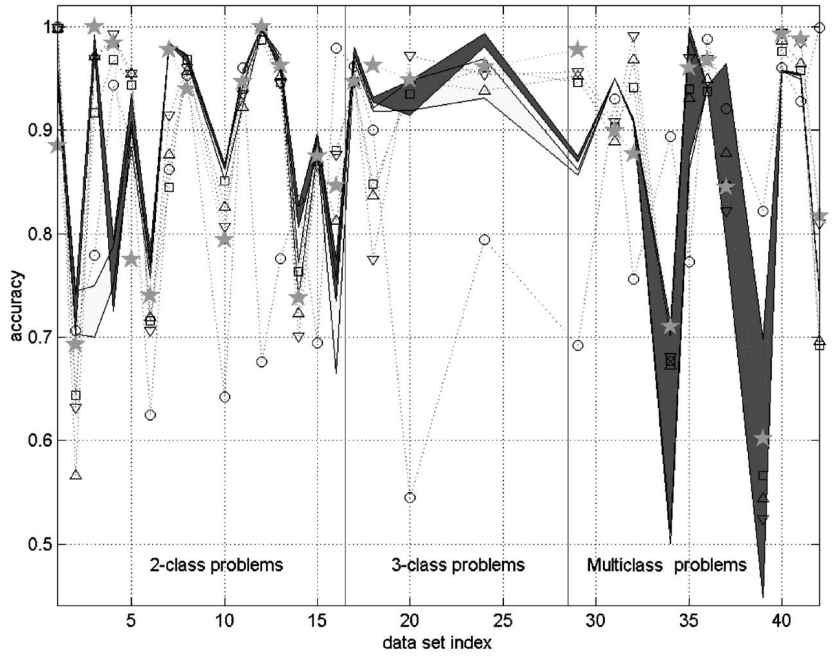

Fig. 3. Accuracy plotted over data sets with reference to those in Ref-I (shaded-dark-tone: RM-Tuned, shaded-light-tone: RM-Fixed, ⋆: SVM-Poly, □: ICPL, △: RT3, ▽: kNN, ○: C4.5).

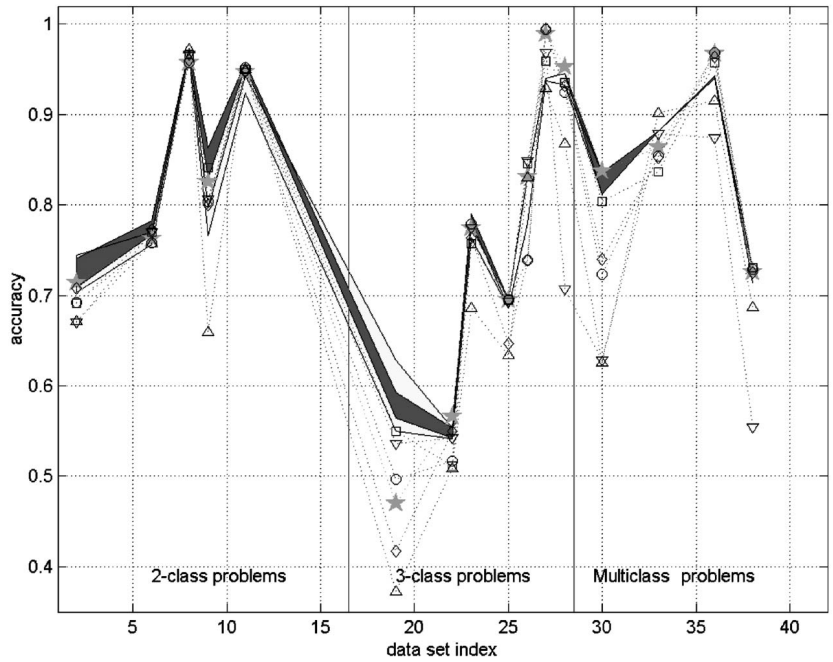

Fig. 4. Accuracy plotted over data sets with reference to six algorithms from Ref-II (shaded-dark-tone: RM-Tuned, shaded-light-tone: RM-Fixed, ⋆: POL, □: LOG, △: LVQ, ▽: RBF, ○: C4T, ◇: C4R).

7.2.4 Comparison with Accuracy Results in the Literature

The accuracies for our RM-Fixed and RM-Tuned algorithms are plotted in Figs. 3, 4, and 5 together with results of those algorithms from Ref-I, Ref-II, and Ref-III. The accuracies as shown in these figures for the proposed RM-Fixed and RM-Tuned algorithms are presented as shaded areas using the minimum and maximum values from the 10 runs of 10-fold cross validation process (see Table 6). In all the three figures, the darker tone and the lighter tone represent those results from RM-Tuned and RM-Fixed, respectively. The stars in the figures are those best accurate algorithms reported in Ref-I, Ref-II, and Ref-III. It is clear from these figures that the RM-Tuned scores many top accuracies in many instances while the RM-Fixed followed by closely. For those data sets compared, Table 9 summarizes the average accuracies in ranking order.

To summarize, the RM-Tuned scores the highest average accuracy among the compared algorithms for the three groups of data sets compared and the RM-Fixed is comparable to those top algorithms.

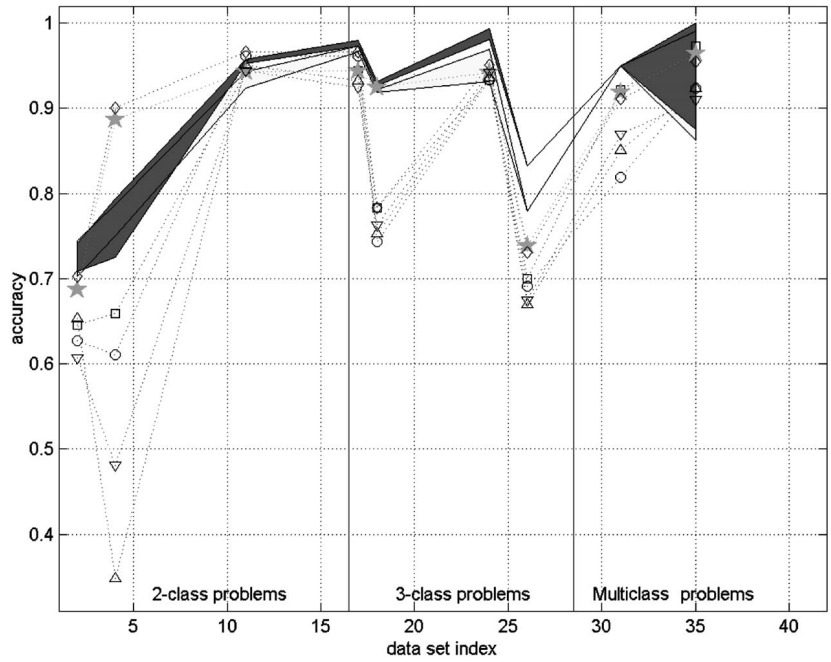

Fig. 5. Accuracy plotted over data sets with reference to those in Ref-III (shaded-dark-toner: RM-Tuned, shaded-light-tone: RM-Fixed, ⋆: CLEF, □: C4.5, △: Φ-RT, ▽: Φ-DNC, ○: DNC, ◇: SVM-RBF).

7.2.5 Ranking

Table 8 lists the best classification results for each data set from the references (Ref-I, Ref-II, and Ref-III) together with the average accuracies of the fixed and tuned reduced models. The best accurate results are taken from different algorithms in each reference for each data set (see columns Best-I, Best-II, and Best-III). The average of these top ranked accuracies is shown in the table as mean (Best-I:III). Here, we see that the RM-Tuned algorithm has an average accuracy higher than mean(Best-III) and it is found to be close to mean(Best-I) and mean(Best-II).

The average rankings for the fixed and tuned reduced models are also compared with those in Ref-I, Ref-II, and Ref-III for the experimented data sets (see Table 8). Here, we found that the ranking (average rank value in bracket inclusive of RM-Tuned) in descending order of accuracy for the algorithms compared with reference to Ref-I is: RM-Tuned(3.0),[6] SVM-Poly(3.1), kNN(3.4), C4.5(3.8), RT3(3.9), and IPCL(3.9). Here, we note that the rank value for RM-Fixed is 3.4 (out of six algorithms) (See Table 8) excluding RM-Tuned.

Comparing with those algorithms in Ref-II, we have the following ranking order (average rank value in bracket): RM-Tuned(8.1) (See Table 8 last row), POL(8.3), LOG(12.1), FM1(12.2), FM2(12.2), QLO(12.6), LDA(13.7), QU0(13.9), C4R(14.0), IMO(14.0), MDA(14.3), PDA(14.5), C4T(14.5), IBO(14.7), ..., NN(25.5), T1(27.5)[12].[7] The rank value for RM-Fixed is found to be 14.8 (out of 34 algorithms) (See Table 8) excluding that of RM-Tuned.

For those algorithms experimented in Ref-III, we found that the top two ranked algorithms are SVM-RBF(2.4) and CLEF(3.1). The fixed and tuned RM scores 2.4 (See Table 8) and 1.6 (out of 7 algorithms), respectively, in the average rank (See Table 8).

6. See Table 8, last row. The rest of the average ranks are computed from the published accuracies in Ref-I.

7. The actual ranking values for many algorithms in [12] should add about 1 since these algorithms do not have RM-Tuned included in rank count.

To summarize, the average ranks for the proposed RM-Tuned and RM-Fixed, respectively, score the lowest (best) and above-average among the compared algorithms on three data groups based on Ref-I, Ref-II, and Ref-III.

7.3 Summary of Results

The results in terms of accuracy and efficiency are summarized in Table 10 where we include only those top and bottom ranked algorithms. It can be seen from this summary that RM, SVM, and kNN are among the top accurate classifiers with relatively good training CPU speed. The C4.5 has good CPU speed but not classification accuracy. Combining implementation simplicity, memory storage requirement, number of prior model structure settings to be decided, and training task, the RM appears to be a good candidate for pattern classification.

8 Conclusion

In this paper, extensive experiments were performed on a reduced multivariate polynomial model based on 42 data sets from UCI machine learning repository. Ten runs of 10-fold stratified cross validation were performed on these data sets to present a good picture of performance statistics. The resulted average classification accuracy were compared with those results in the literature which used only a single run of 10-fold validation. The empirical results show that the reduced model is either better than or comparable to top ranking algorithms in the literature in terms of average classification accuracy despite its simplicity in implementation. The computing time needed for training is also observed to be among the fastest in those compared algorithms. Main reason for the fast computing speed is that no initialization is needed and the solution can be obtained in a single-step that is also least squares optimal. While awaiting for possibly optimal solutions or tuning to universal approximators or classifiers like Neural Networks, RBF and SVM to appear in

TABLE 8
RM-Fixed and RM-Tuned Compared to Best Reported Results in the Literature

No	Name	RM-Fixed				RM-Tuned				Best-I	Best-II	Best-III
		(ave)	Rank-I	Rank-II	Rank-III	(ave)	Rank-I	Rank-II	Rank-III	(ave)	(ave)	(ave)
1	Shuttle-l-contr	0.955	5	–	–	0.958	5	–	–	1.000	–	–
2	BUPA-liver	0.713	1	4	1	0.727	1	1	1	0.706	0.721	0.703
3	Monk-1	0.721	6	–	–	0.987	2	–	–	1.000	–	–
4	Monk-2	0.769	6	–	3	0.767	6	–	3	0.993	–	0.900+
5	Monk-3	0.895	4	–	–	0.915	4	–	–	0.995	–	–
6	Pima-diabetes	0.764	1	15	–	0.776	1	5	–	0.740	0.779	–
7	Tic-tac-toe	0.984	1	–	–	0.984	1	–	–	0.978	–	–
8	Breast-cancer-W	0.969	1	4	–	0.970	1	2	–	0.968	0.972	–
9	StatLog-heart	0.786	–	19	–	0.844	–	6	–	–	0.859	–
10	Credit-app	0.861	1	–	–	0.864	1	–	–	0.851	–	–
11	Votes	0.934	5	34	7	0.954	2	15	3	0.960	0.964	0.966
12	Mushroom	0.996	4	–	–	1.000	2.5	–	–	1.000	–	–
13	Wdbc	0.967	1	–	–	0.961	2	–	–	0.963	–	–
14	Wpbc	0.759	3	–	–	0.819	2	–	–	0.832	–	–
15	Ionosphere	0.888	1	–	–	0.888	1	–	–	0.878	–	–
16	Sonar	0.714	6	–	–	0.748	6	–	–	0.979	–	–
17	Iris	0.967	1	–	1	0.976	1	–	1	0.962	–	0.966
18	Balance-scale	0.920	2	–	2	0.929	2	–	1	0.963	–	0.925
19	Teaching-assist	0.596	–	6	–	0.581	–	8	–	–	0.675	–
20	New-thyroid	0.934	5	–	–	0.937	4	–	–	0.972	–	–
21	Abalone	0.666	–	–	–	0.666	–	–	–	–	–	–
22	Contraceptive-mthd	0.546	–	9	–	0.548	–	9	–	–	0.566	–
23	Boston-housing	0.773	–	5	–	0.784	–	1	–	–	0.779	–
24	Wine	0.948	4	–	2	0.988	1	–	1	0.961	–	0.951
25	Attitude-smoking+	0.686	–	24	–	0.695	–	10	–	–	0.696	–
26	Waveform+	0.780	–	16	1	0.833	–	5	1	–	0.849	0.739
27	Thyroid+	0.937	–	30	–	0.940	–	28	–	–	0.995	–
28	StatLog-DNA+	0.933	–	17	–	0.945	–	5	–	–	0.953	–
29	Car	0.859	5	–	–	0.872	5	–	–	0.978	–	–
30	StatLog-vehicle	0.823	–	3	–	0.823	–	3	–	–	0.855	–
31	Soyabean-small	0.950	1	–	1	0.950	1	–	1	0.930	–	0.922
32	Nusery	0.909	4	–	–	0.909	4	–	–	0.991	–	–
33	StatLog-satimage+	0.882	–	2	–	0.882	–	2	–	–	0.902	–
34	Glass	0.621	6	–	–	0.650	6	–	–	0.894	–	–
35	Zoo	0.951	3	–	4	0.969	2	–	2	0.970	–	0.973
36	StatLog-image-seg	0.941	5	23	–	0.941	5	23	–	0.988	0.978	–
37	Ecoli	0.859	3	–	–	0.874	3	–	–	0.921	–	–
38	Led-display+	0.714	–	26.5	–	0.728	–	7	–	–	0.732	–
39	Yeast	0.610	2	–	–	0.610	2	–	–	0.822	–	–
40	Pendigit	0.957	6	–	–	0.957	6	–	–	0.994	–	–
41	Optdigit	0.951	5	–	–	0.953	5	–	–	0.988	–	–
42	Letter	0.741	4	–	–	0.741	4	–	–	0.999	–	–
	mean(Best-I:III)	–	–	–	–	–	–	–	–	0.939	0.830	0.894
	mean(RM-Fixed)	–	–	–	–	–	–	–	–	0.867	0.798	0.882
	mean(RM-Tuned)	–	–	–	–	–	–	–	–	0.886	0.811	**0.899**
	mean(Rank-I:III)	–	3.4	14.8	2.4	–	3.0	8.1	1.6	–	–	–

+ : Accuracy measured from the given training and test set instead of 10-fold validation.

the research literature, we hope that this simple reduced model can provide a benchmark considering both accuracy and efficiency for good classification algorithms design.

APPENDIX A

THE REDUCED MULTIVARIATE POLYNOMIAL MODEL CODED AS MATLAB FUNCTION

```
%            Beginning of Function                %
function P = RMmodel(order,X)
% Build regressor matrix P (mxK):
%     order = desired order of approximation,
%     X = input matrix (mxl), K = number of
              parameters to be est.
%     m = number of data samples, l = input
              dimension.
[m,l] = size(X); MM1=[]; MM3=[];
Msum=sum(X,2);
for i=1:order
  for k=1:l
```

TABLE 9
Summary of Average Accuracy in Rank Order with Reference to Ref-I, Ref-II, and Ref-III

RM-Fixed			RM-Tuned		
Ref-I	Ref-II	Ref-III	Ref-I	Ref-II	Ref-III
SVM-Poly(0.883)	POL(0.805)	CLEF(0.884)	**RM-Tuned(0.886)**	**RM-Tuned(0.811)**	**RM-Tuned(0.899)**
kNN(0.880)	**RM-Fixed(0.798)**	**RM-Fixed(0.882)**	SVM-Poly(0.883)	POL(0.805)	CLEF(0.884)
ICPL(0.869)	LOG(0.796)	SVM-RBF(0.874)	kNN(0.880)	LOG(0.796)	SVM-RBF(0.874)
RT3(0.868)	MDA(0.793)	C4.5(0.834)	ICPL(0.869)	MDA(0.793)	C4.5(0.834)
RM-Fixed(0.867)	QL0(0.792)	DNC(0.808)	RT3(0.868)	QL0(0.792)	DNC(0.808)
C4.5(0.833)	LDA(0.792)	Φ-DNC(0.791)	C4.5(0.833)	LDA(0.792)	Φ-DNC(0.791)
–	QL1(0.789)	Φ-RT(0.780)	–	QL1(0.789)	Φ-RT(0.780)
–	PDA(0.787)	–	–	PDA(0.787)	–
–	⋮	–	–	⋮	–
–	QDA(0.699)	–	–	QDA(0.699)	–
–	T1(0.646)	–	–	T1(0.646)	–

TABLE 10
Summary of Algorithm Properties (Tabulation Solely Based on Compared References)

	Accuracy		Efficiency				Matlab
Algorithm	class-accuracy	breakdown	CPU-time	storage requirement	training	model-settings	implementation
RM	+++	no	+++	++	single-step	2	simple
SVM	+++	no	?	++	iterative	kernel settings	tough
kNN	+++	no	P	P	no training	nil	simple
CLEF	+++	no	?	?	feature construction	?	?
POL	+++	no	+	?	?	model selection	?
LOG	+++	no	+	?	?	?	?
ICPL	++	no	?	?	abstraction & filtering	rule-based	?
C4.5	+	no	+++	+++	rule-based	?	?
RBF	O/?	no	+++	+++	single-step*	4*	tough
DNC(neural)	O/?	no	+	+++	iterative	?	tough
MP3,6	O	yes	+++	exponential	single-step	order and terms	simple

+++ : HIGH in ranking (e.g. top 3 algorithms in respective references).
++ : MEDIUM in ranking.
+ : LOW in ranking (e.g bottom few algorithms in respective references).
* : Single step training when the width and center parameters are pre-selected. The 4 prior-settings in RBF include: width, centers, number of layers and number of nodes.
? : Not known from the cited source alone/depends.
O : Good approximation capability since it is compact, classification accuracy depends on specific implementation and tuning.
P : Proportional to data size.

```
    M1(:,k)=X(:,k).^i;
    if (i>1)
       M3(:,k)=X(:,k).*Msum.^(i-1);
    end
  end
  M2(:,i)=Msum.^i;
  MM1=[MM1,M1];
  if (i>1)
     MM3=[MM3,M3];
  end
end
P = [ones(m,1),MM1,M2,MM3];
return;
%              End of Function                  %
```

APPENDIX B

AN EXAMPLE MATLAB FUNCTION CALL FOR TRAINING AND TEST FOR USE IN 2-CLASS PROBLEMS

```
%                  Training                     %
% setup input matrix X(mxl) and target output
   vector y(mx1)
% m = number of data samples; l = input dimension
% X = input matrix, y = target output vector,
load TrainFILENAME X,y; %need a prestored file
   containing X and y
% generate regressor matrix P and identity
   matrix for training
P = RMmodel(6,X);     %6th-order-Reduced-Model
I = eye(size(P'*P)); %identity matrix
b = 1e-4;             %regularization constant
% solve for the weight coefficients
alpha = inv(P'*P + b*I)*P'*y;
```

```
% generate trained output
y_train = P*alpha;
%                  Test                          %
% Note: do not clear the variable "alpha" here
% setup input matrix for test: Xt(mxl)
% m = number of data samples; l = input dimension
load TestFILENAME Xt; %need a prestored file
    containing Xt
% generate regressor matrix Pt for test
Pt = RMmodel(6,Xt); %6th-order Reduced model

% generate test output
y_test = Pt*alpha;
%                  End Test                      %
```

ACKNOWLEDGMENTS

The authors would like to thank Dr. Ludmila Kuncheva and the reviewers for very constructive comments to improve the paper. The authors are also thankful to Dr. Junxian Wang, Dr. How-Lung Eng, Dr. Wei-Yun Yau, and Dr. Xudong Jiang for useful discussions.

REFERENCES

[1] A.K. Jain, R.P.W. Duin, and J. Mao, "Statistical Pattern Recognition: A Review," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 22, no. 1, pp. 4-37, Jan. 2000.
[2] R.O. Duda, P.E. Hart, and D.G. Stork, *Pattern Classification,* second ed. New York: John Wiley & Sons, Inc., 2001.
[3] T. Poggio and F. Girosi, "Networks for Approximation and Learning," *Proc. IEEE,* vol. 78, no. 9, pp. 1481-1497, 1990.
[4] C.M. Bishop, *Neural Networks for Pattern Recognition.* New York: Oxford Univ. Press Inc., 1995.
[5] Y. Shin and J. Ghosh, "Ridge Polynomial Networks," *IEEE Trans. Neural Networks,* vol. 6, no. 3, pp. 610-622, 1995.
[6] Y. Shin and J. Ghosh, "The Pi-Sigma Network: An Efficient Higher-Order Neural Network for Pattern Classification and Function Approximation," *Proc. Int'l Joint Conf. Neural Networks,* vol. 1, pp. 13-18, July 1991.
[7] W. Campbell, K. Torkkola, and S. Balakrishnan, "Dimension Reduction Techniques for Training Polynomial Networks," *Proc. Int'l Conf. Machine Learning,* June 2000.
[8] P. Chaudhuri, M.-C. Huang, W.-Y. Loh, and R. Yao, "Piecewise-Polynomial Regression Trees," *Statistica Sinica,* vol. 4, pp. 143-167, 1994.
[9] K.-A. Toh, W.-Y. Yau, and X. Jiang, "A Reduced Multivariate Polynomials Model for Multi-Modal Biometrics and Classifiers Fusion," *IEEE Trans. Circuits and Systems for Video Technology,* pending publication, 2004.
[10] C.L. Blake and C.J. Merz, "UCI Repository of Machine Learning Databases," Dept. of Information and Computer Sciences, Univ. of Calif., Irvine, http://www.ics.uci.edu/mlearn/MLRepository.html, 1998.
[11] W. Lam, C.-K. Keung, and D. Liu, "Discovering Useful Concept Prototypes for Classification Based on Filtering and Abstraction," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 24, no. 8, pp. 1075-1090, Aug. 2002.
[12] T.-S. Lim, W.-Y. Loh, and Y.-S. Shil, "A Comparison of Prediction Accuracy, Complexity, and Training Time of Thirty-Three Old and New Classification Algorithms," *Machine Learning,* vol. 40, no. 3, pp. 203-228, 2000.
[13] J. Neter, M.H. Kutner, C.J. Nachtsheim, and W. Wasserman, *Applied Linear Regression Models,* third ed. Chicago: Irwin, 1996.
[14] W.R. Wade, *An Introduction to Analysis,* second ed. Upper Saddle River, N.J.: Prentice Hall, 2000.
[15] B.E. Boser, I.M. Guyon, and V.N. Vapnik, "A Training Algorithm for Optimal Margin Classifiers," *Proc. Fifth Ann. Workshop Computational Learning Theory,* pp. 144-152, 1992.
[16] B. Schölkopf and A.J. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond.* Cambridge, Mass.: MIT Press, 2002.
[17] S. Raudys, "Evolution and Generalization of a Single Neurone: I. Single-Layer Perceptron as Seven Statistical Classifiers," *Neural Networks,* vol. 11, pp. 283-296, 1998.
[18] S. Raudys, "Evolution and Generalization of a Single Neurone: II. Complexity of Statistical Classifiers and Sample Size Considerations," *Neural Networks,* vol. 11, pp. 297-313, 1998.
[19] L. I. Kuncheva, "A Theoretical Study on Six Classifier Fusion Strategies," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 24, no. 2, pp. 281-286, Feb. 2002.
[20] K. Hornik, M. Stinchcombe, and H. White, "Multi-Layer Feedforward Networks Are Universal Approximators," *Neural Networks,* vol. 2, no. 5, pp. 359-366, 1989.
[21] G. Cybenko, "Approximations by Superpositions of a Sigmoidal Function," *Math. Cont. Signal & Systems,* vol. 2, pp. 303-314, 1989.
[22] P. Brazdil, "Statlog Datasets," Inst. for Social Research at York Univ., http://www.liacc.up.pt/ML/statlog/datasets. html, 1999.
[23] L. Pederson and S. Bull, "StatLib—Case Studies in Biometry," Inst. for Social Research at York Univ., http://lib.stat.cmu.edu/datasets/csb/, 1988.
[24] D. Precup and P.E. Utgoff, "Classification Using Φ-Machines and Constructive Function Approximation," *Machine Learning,* 2003.
[25] A. Torn and A. Zilinskas, "Global Optimization," *Lecture Notes in Computer Science,* Berlin: Springer-Verlag, 1989.
[26] K.-A. Toh, "Deterministic Global Optimization for FNN Training," *IEEE Trans. Systems, Man, and Cybernetics, Part B,* vol. 33, no. 6, pp. 977-983, 2003.
[27] The MathWorks, "Matlab and Simulink," http://www.mathworks.com/, 2003.
[28] K.-A. Toh, "Global Optimization by Monotonic Transformation," *Computational Optimization and Applications,* vol. 23, pp. 77-99, Oct. 2002.

Kar-Ann Toh (M'92-SM'03) received the PhD degree from Nanyang Technological University (NTU Singapore) in 1999. Prior to his postdoctoral appointments at research centers in NTU from 1998 to 2002, he has worked for two years in the aerospace industry. Currently, he is affiliated with the Institute for Infocomm Research (Singapore). His research interests include biometrics and decision fusion, pattern classification, optimization, and neural networks. He has made several PCT filings related to biometric applications, and has actively published his works in above areas of interest. Over recent years, he has served as a reviewer for several international journals including *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *IEEE Transactions on Circuits and Systems-Part I*, and *IEEE Transactions on Systems, Man, and Cybernetics-Part B*. He is a senior member of the IEEE.

Quoc-Long Tran received the BSc degree from the Faculty of Technology at the Vietnam National University of Hanoi in 2002. He is currently awarded a scholarship from Institute for Infocomm Research to pursue the MEng degree in the Department of Electrical and Computer Engineering, National University of Singapore. His research interests include machine learning and pattern classification, especially for applications in multiple biometrics decision fusion.

Dipti Srinivasan (M'89 - SM'02) received the PhD degree in engineering from National University of Singapore (NUS Singapore). She worked as a postdoctoral researcher at the University of California, Berkeley from 1994 to 1995 before joining the National University of Singapore as an assistant professor with the Department of Electrical and Computer Engineering. Her research interest is in application of soft computing techniques to engineering optimization and control problems. She is a senior member of the IEEE.