# Transactions Letters

## A Reduced Multivariate Polynomial Model for Multimodal Biometrics and Classifiers Fusion

Kar-Ann Toh, *Senior Member, IEEE*, Wei-Yun Yau, *Member, IEEE*, and Xudong Jiang, *Member, IEEE*

***Abstract*—The multivariate polynomial model provides an effective way to describe complex nonlinear input–output relationships since it is tractable for optimization, sensitivity analysis, and prediction of confidence intervals. However, for high-dimensional and high-order problems, multivariate polynomial regression becomes impractical due to its huge number of product terms. This is especially true for the case of a full interaction model. In this paper, we propose a reduced multivariate polynomial model to circumvent the dimensionality problem with some compromise in its approximation capability. In multimodal biometrics and many classifiers fusion applications, as individual classifiers to be combined would have attained a certain level of classification accuracy, this reduced multivariate polynomial model can be used to combine these classifiers in the next level of classification taking their outputs as the inputs to the reduced multivariate polynomial model. The model is first applied to a well-known pattern classification problem to illustrate its classification capability. The reduced multivariate polynomial model is then applied to combine two biometric verification systems with improved receiver operating characteristics performance as compared to an optimal weighing method and a few commonly used classifiers.**

***Index Terms*—Biometrics, classification, data fusion and multivariate polynomials, pattern recognition.**

### I. Introduction

DUE TO A possible increase in degrees of freedom, the fusion of multiple biometrics and other classifiers may allow alleviation of problems intrinsic to individual classifiers. By exploiting the specialist capabilities of each classifier, a combined classifier may yield results which would not be possible in a single classifier. Fusion of multimodal biometrics and other classifiers is thus an important research topic in pattern recognition.

The biometrics verification problem can be considered as a classification problem wherein a decision is made upon whether or not a claimed identity is genuine with inference to some matching results. We thus treat the problem of combining multimodal biometrics as a classifier combination problem in this paper. According to the origination of various methods to combine classifiers, different terminologies have been adopted to reflect the essence of each approach. According to [1], these terminologies include combination of multiple classifiers, classifier fusion, mixture of experts, committees of neural networks, consensus aggregation, voting pool of classifiers, dynamic classifier selection, composite classifier system, classifier ensembles, divide-and-conquer classifiers, pandemonium system of reflective agents, and change-glasses approach to classifier selection.

Generally, the approaches for classifiers combination differ in terms of assumptions about classifier dependencies, type of classifier outputs, combining strategies and combining procedures [1]. Two main types of combination can be identified: *classifier selection* and *classifier fusion*. The difference between these two types lies on whether the classifiers are assumed to be complementary or competitive. Classifier selection assumes that each classifier is a "local expert" while classifier fusion assumes that all classifiers are trained over the entire feature space (see, e.g., [1]). In this paper, our focus will be on classifier fusion, and our main effort will be towards arriving at a fusion methodology that optimizes the accuracy of the combined decision.

According to the information adopted, three levels of combination can be identified [2], [3]: 1) abstract level; 2) rank level; and 3) measurement level. At the abstract level, the output information taken from each classifier is only a possible label for each pattern class, whereas at the rank level the output information taken from each classifier is a set of ordered possible labels which is ranked by decreasing confidence measure. At the measurement level, the output information taken from each classifier is a set of possible labels with associated confidence measure. In this way, with the measurement outputs taken from each individual system, the decision is brought forward to the final output of the combined system. We shall work at the measurement level to combine two biometric verification systems.

While the statistical approach (see, e.g., [4], [5]) has received considerable attention, many classifiers, predictors, or estimators [6] by themselves can be used for data fusion, and we shall briefly review some of the commonly used ones. Spline interpolation (see, e.g., [7]) possesses a good approximation capability, but the selection of its control points requires a great deal of knowledge regarding the distribution of data used. The feedforward neural network (FNN) has been shown to be a universal approximator (see, e.g., [8] and [9]), however, the training process remains much to be a trial-and-error effort since no learning algorithm can guarantee convergence to a global optimal solution within finite iterations for general application problems. The global FNN (GFNN) [10], [11] may provide possible speed-ups in the training process, but it is limited to applications that do not result in numerical ill-conditioning.

Manuscript received October 29, 2002; revised May 9, 2003.

The authors are with the Institute for Infocomm Research, Singapore 119613 (e-mail: katoh@i2r.a-star.edu.sg).

Digital Object Identifier 10.1109/TCSVT.2003.821974

The radial basis function network (RBFN) (see, e.g., [12]) has been widely used for approximation due to its structural simplicity. Typically, training of the RBFN involves selecting the hidden-layer neuron centers, choosing scaling parameters, and estimating the weights that connect the hidden and output layers. Although the weights can be estimated using the linear least-squares algorithm once the centers and the scaling parameters are fixed, selection of these centers and scaling parameters remains a nontrivial task. Other networks like ridge polynomial networks (RPN) (see, e.g., [13] and [14]), though they may exhibit a good approximation capability, have similar problems in training since the formulation is usually nonlinear. The general regression neural network (GRNN) (see, e.g., [15] and [16]) have good convergence and one-pass learning properties. However, it requires substantial computation to evaluate new points because a separate neuron is required for each sample. Moreover, estimation of the width parameter is nontrivial. The probabilistic neural network (PNN) [17], [18] is closely related to GRNN where it uses a spherical kernel as the radial basis function. Similar to the RBF, the approximation and estimation of the center and width parameters are nontrivial matters. The functional link network (FLN) using Chebyshev-polynomials (CPB) possesses the capability of a universal approximator [19] and reported a faster learning rate than the conventional feedforward/recurrent neural networks model. Since the network uses the recursive least-squares method with forgetting factor as learning algorithm, the choice of the forgetting factor affects the convergence. For high dimensional problems, the number of parameters to be estimated in this network tends to be be very large. In the more general FLN and high-order perceptrons (HOPs) using polynomial and power series expansions, the problem of having a huge number of parameters persists unless a computational intensive evolutional search is performed to reduce the model to an optimal subset of units [20].

Recently, the support vector machine (SVM) [21], [22] receives considerable attention due to its sound theoretical basis on optimizing the class boundary separation rather than the usual error objective. Solving the constrained optimization for nonlinear systems (usually the case for physical systems) in SVM results in a quadratic programming formulation. This quadratic formulation is normally solved in an iterative manner where the global solution is, again, not guaranteed.

The optimal weighting method (OWM) [23] provides an efficient way to combine different classifiers. However, it is limited to systems which can be separated by linear separation hyperplanes. As an extension to OWM, the multivariate polynomial model (MP) [24] provides an effective way to describe complex nonlinear input–output relationships since it is tractable for optimization, sensitivity analysis, and prediction of confidence intervals. 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 using a full interaction model. In view of these, our problem here is to derive a model that does not possess an exponentially increasing number of parameters with respect to the model order and number of inputs while at the same time preserving much of its approximation capability. We address this problem by proposing a reduced multivariate polynomial model (RM) where the number of parameters to be estimated increases *linearly* with the model order and the number of inputs. We shall demonstrate the classification capability of the RM using physical data. To circumvent possible multicollinearity among the classifiers and improve generalization, an optimal regularization search and a validation procedure are proposed. The main contributions of this paper thus include: 1) proposal of a reduced multivariate polynomial model for multimodal classifiers fusion and 2) optimal regularization for possible multicollinearity.

The paper is organized as follows. In Section II, the problem of combining multimodal classifiers is stated before some preliminaries on the optimal weighting method are provided. With these backgrounds in place, the multivariate polynomial model is introduced in Section III. Taking the measurement outputs of each biometric classifiers as input variables of the multivariate polynomials, several existing polynomial models are discussed in this section before a reduced model is derived in Section IV. In Section V, we present the weight decay regularization for possible multicollinearity among the classifiers to be combined. A search is also presented in this section to optimize the regularization parameter. In Section VI, a well-known pattern classification example is used to illustrate the performance of the reduced model in terms of classification capability. In Section VII, the proposed model is tested using physical data from the fingerprint and voice verification systems. Finally, in Section VIII, some concluding remarks are drawn.

## II. Problem Definition and Preliminaries

### A. Problem Definition

We define the problem of combining multimodal classifiers and a corresponding problem on multicollinearity as follows:

*1) The Problem of Combining Multimodal Classifiers:* Given $(l, m, n, p, q)$ as positive integers and consider two sets of data: a training set $\mathcal{S}_{\text{train}} = \{\boldsymbol{r}_i \in \mathcal{R}^p, \boldsymbol{s}_i \in \mathcal{R}^q\}$, $i = 1, \ldots, m$ and a test set $\mathcal{S}_{\text{test}} = \{\boldsymbol{r}_i \in \mathcal{R}^p, \boldsymbol{s}_i \in \mathcal{R}^q\}$, $i = 1, \ldots, n$ where $\boldsymbol{r}$ and $\boldsymbol{s}$ denote the feature vector and the class inference, respectively. Given a set of decision functions $\mathcal{F} = \{\hat{f}_j(\boldsymbol{r}, \boldsymbol{s})\}$, $j = 1 \ldots, l, l \ll m$ where each of its elements, $\hat{f}_j(\boldsymbol{r}, \boldsymbol{s})$ approximates a true function $f(\boldsymbol{r}, \boldsymbol{s})$ (assuming it exists) which classifies the data given by $\{\mathcal{S}_{\text{train}}, \mathcal{S}_{\text{test}}\}$. In biometric problems, the classification is labeled either as imposters or as genuine users. Given some $\mathcal{F}$ based on $\mathcal{S}_{\text{train}}$, our problem here is to find the best possible approximation of $f(\boldsymbol{r}, \boldsymbol{s})$ using this set of $\mathcal{F}$. Those $\mathcal{F}$ generated from $\mathcal{S}_{\text{test}}$ will be used to test the classification performance. We shall consider only the overdetermined system.

*2) Multicollinearity:* Assume that each false positive poses the same amount of risk and every false negative presents identical liability, and the system is under random attack. It remains an issue when combining a set of learned classifiers with correlation. The higher the degree of correlation, the larger amount of agreement or linear dependence among the classifiers will be. This correlation also reflects the amount of redundancy within the set of classifiers. Here, the problem of correlation which can produce unreliable estimates is referred to as *multicollinearity problem* [25].

### B. Linear Classifiers for Data Fusion

*1) Simple and Weighted Averaging:* When the individual classifiers produce continuous outputs, a simple way is to average out all the outputs when there is no bias against any of these individual classifiers and when these classifiers are mutually independent. This is termed the basic ensemble method (BEM) (see, e.g., [25]) which can be written as

$$\hat{f}_{\mathrm{BEM}} = \frac{1}{l}\sum_{j=1}^{l} \hat{f}_j(\boldsymbol{r},\boldsymbol{s}). \tag{1}$$

Equation (1) can also be written in terms of the *misfit function* for each $\hat{f}_j(\boldsymbol{r},\boldsymbol{s})$[25] as follows:

$$\hat{f}_{\mathrm{BEM}} = f(\boldsymbol{r},\boldsymbol{s}) - \frac{1}{l}\sum_{j=1}^{l} e_j(\boldsymbol{r},\boldsymbol{s}) \tag{2}$$

where

$$e_j(\boldsymbol{r},\boldsymbol{s}) = f(\boldsymbol{r},\boldsymbol{s}) - \hat{f}_j(\boldsymbol{r},\boldsymbol{s}), \quad j = 1,2,\ldots,l. \tag{3}$$

In practice, the mutual independency and unbiasness may not hold. A weighted averaging [called the generalized ensemble method (GEM)] can then be applied as follows if a certain classifier is known to be more accurate than another classifier:

$$\hat{f}_{\mathrm{GEM}}(\boldsymbol{\alpha}) = \sum_{j=1}^{l} \alpha_j \hat{f}_j(\boldsymbol{r},\boldsymbol{s}) = f(\boldsymbol{r},\boldsymbol{s}) - \sum_{j=1}^{l} \alpha_j e_j(\boldsymbol{r},\boldsymbol{s}) \tag{4}$$

where $\alpha_j, j = 1,\ldots,l$ (denoted as $\boldsymbol{\alpha}$) can be determined from

$$\alpha_j = \frac{\sum_{k=1}^{l} C_{jk}^{-1}}{\sum_{i=1}^{l}\sum_{k=1}^{l} C_{ik}^{-1}}, \quad C_{jk} = E[e_j(\boldsymbol{r},\boldsymbol{s})e_k(\boldsymbol{r},\boldsymbol{s})]. \tag{5}$$

Notice that in the above formulation

$$\sum_{j=1}^{l} \alpha_j = 1. \tag{6}$$

*2) OWM:* The GEM described above is a special case of OWM [23] minimizing the sum of squared error subject to the constraint given by (6). Consider

$$\hat{f}_{\mathrm{OWM}}(\boldsymbol{\alpha}) = \sum_{j=1}^{l} \alpha_j \hat{f}_j(\boldsymbol{r},\boldsymbol{s}) \tag{7}$$

and, assuming no constraints, the weights ($\boldsymbol{\alpha} = [\alpha_1,\ldots,\alpha_l]^T$) for OWM can be found from

$$\boldsymbol{\alpha} = (\hat{\mathbf{F}}^T\hat{\mathbf{F}})^{-1}\hat{\mathbf{F}}^T\mathbf{f} \tag{8}$$

where $\hat{\mathbf{F}} \in \mathcal{R}^{m\times l}$ denotes the Jacobian matrix of $\hat{f}_{\mathrm{OWM}}$ as follows:

$$\hat{\mathbf{F}} = \begin{bmatrix} \hat{f}_1(\boldsymbol{r}_1,\boldsymbol{s}_1) & \ldots & \hat{f}_l(\boldsymbol{r}_1,\boldsymbol{s}_1) \\ \vdots & \ldots & \vdots \\ \hat{f}_1(\boldsymbol{r}_m,\boldsymbol{s}_m) & \ldots & \hat{f}_l(\boldsymbol{r}_m,\boldsymbol{s}_m) \end{bmatrix} \tag{9}$$

and $\mathbf{f} = [f(\boldsymbol{r}_1,\boldsymbol{s}_1),\ldots,f(\boldsymbol{r}_m,\boldsymbol{s}_m)]$ from training data.

We shall use OWM as a basis of comparison since it represents a more general form of all the above estimators. Here it is noted that OWM involves computation of the inverse of a matrix, the problem of multicollinearity may arise if linear dependency among the elements of $\mathcal{F}$ is present.

## III. Multivariate Polynomial Regression (MP)

The OWM described above provides an effective way to linearly combine multiple classifiers. However, important interacting relationships among the data may be ignored, thereby giving rise to inaccurate results. To cater for possible nonlinear effects and interactions, the multivariate polynomial regression is considered.

Multivariate polynomial regression provides an effective way to describe complex nonlinear input–output relationships. Also, it is tractable for optimization, sensitivity analysis, and prediction of confidence intervals. A typical polynomial regression model contains the squared and higher order terms of the estimator variable. However, for high-dimensional and high-order problems, multivariate polynomial regression becomes impractical due to its prohibitive number of product terms. This is especially true for the case of using an interaction model. 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.

In the following, to simplify the expression as well as to avoid possible confusion, the notation of individual classifiers or estimators $\hat{f}_j(\boldsymbol{r},\boldsymbol{s}), j = 1,\ldots,l$ to be combined will be replaced by $x_j, j = 1,\ldots,l$ as polynomial inputs (i.e., $x_j = \hat{f}_j(\boldsymbol{r},\boldsymbol{s})$).

### A. Multinomial (MN): A Special Case of Multivariate Polynomials

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

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

where the summation is taken over all nonnegative integers $n_1,n_2,\ldots,n_l$ for which $n_1+n_2+\cdots+n_l = r$ where $r$ is the order of approximation. Suppose there are a total of $K+1$ terms in this multinomial model. A possible application using this multinomial model for classifier combination is to estimate the weighting parameter vector $\boldsymbol{\alpha} = [\alpha_0,\alpha_1,\ldots,\alpha_K]^T$ from

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

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 as follows:

$$(\mathrm{MN-II}): \hat{f}_{\mathrm{MN}}(\boldsymbol{\alpha}) = \alpha_0 + \sum_{j=1}^{r} \alpha_j (x_1+x_2+\cdots+x_l)^j. \tag{12}$$

### B. Approximation Capability: An Empirical Case Study

The approximation capability of polynomials is well known from the Weierstrass approximation theorem [26] which states that every continuous function defined on an interval can be

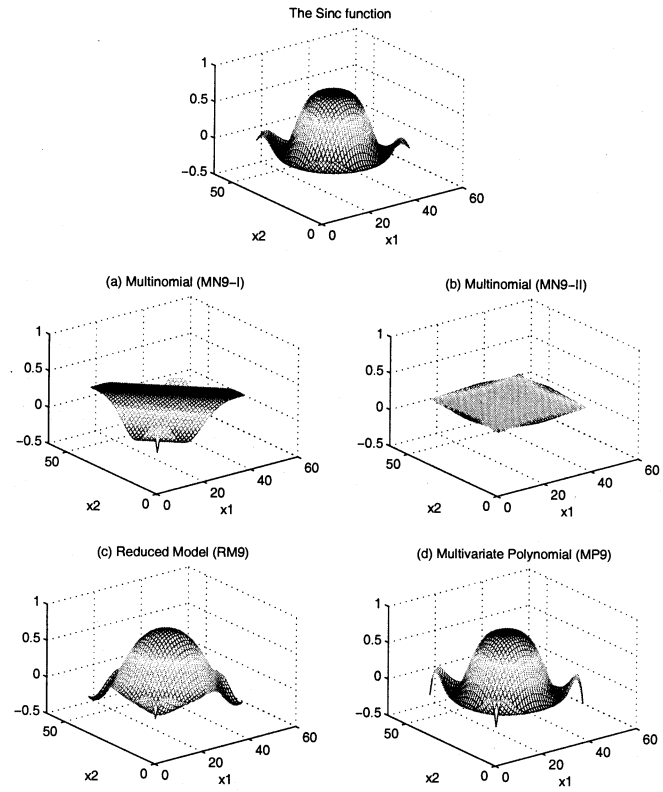


Fig. 1. Sinc function approximation using ninth-order models: (a) MN9-I, (b) MN9-II, (c) RM9, and (d) MP9.

approximated as closely as desired by a polynomial function. However, as mentioned, the number of terms increases tremendously with the number of inputs and the order of the polynomials to be approximated.

The multinomial model MN-I offers a greatly reduced number of terms but suffers from approximation deficiency because only high-order terms are used. The lumped multinomial model MN-II, though having a significantly reduced number of terms which is only dependent on the order of approximation, also suffers from approximation deficiency as the inputs only have a "lumped" effect on the approximation. This can be seen from the approximation of the *sinc* function [top of Fig. 1] using a ninth-order model in each case (MN9-I and MN9-II) in Fig. 1(a) and (b).

## IV. A Reduced Multivariate Polynomials Model

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

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

It is noted that this gives rise to a nonlinear estimation model where the weight parameters $(\alpha_{jk}, j = 1, \ldots, r, k = 1, \ldots, l)$ may not be estimated in a straightforward 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.

Assume two points $\boldsymbol{\alpha}$ and $\boldsymbol{\alpha}_1$ on the multinomial function which is differentiable. By the mean value theorem, the multinomial function $f(\boldsymbol{\alpha}) = (\alpha_{j1}x_1 + \alpha_{j2}x_2 + \cdots + \alpha_{jl}x_l)^j$, $j = 2, \ldots, r$ 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 $0 \leq \beta \leq 1$. Let $\boldsymbol{x} = [x_1, \ldots, x_l]^T$. With an appropriate choice of terms based on (14), omitting the 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:

$$\begin{aligned}\hat{f}_{\mathrm{RM}'} = {} & \alpha_0 + \sum_{j=1}^{l} \alpha_j x_j \\ & + \sum_{j=1}^{r} \alpha_{l+j}(x_1 + x_2 + \cdots + x_l)^j \\ & + \sum_{j=2}^{r} (\boldsymbol{\alpha}_j^T \cdot \boldsymbol{x})(x_1 + x_2 + \cdots + x_l)^{j-1}, \\ & l, r \geq 2 \end{aligned} \quad (15)$$

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:

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

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. 2. A linear relationship between the number of terms and the model order $(r)$ or model dimension $(l)$ is observed for the proposed reduced model. For comparison purposes, the same figure includes the number of terms plotted over the model order for a full multivariate polynomial model with input dimension two $(l = 2)$.

The above reduced model can be used in a least-squares error objective function and a solution form similar to (8) can be obtained. For simplicity and without loss of generality, consider the following second-order reduced bivariate polynomial model taking $x_1$ from output of biometric-A and $x_2$ from output of biometric-B:

$$\begin{aligned}\hat{f}_{\mathrm{RM2}} = {} & \alpha_0 + \alpha_1 x_1 + \alpha_2 x_2 + \alpha_3 x_1^2 + \alpha_4 x_2^2 \\ & + \alpha_5(x_1 + x_2) + \alpha_6(x_1 + x_2)^2 \\ & + \alpha_7 x_1(x_1 + x_2) + \alpha_8 x_2(x_1 + x_2). \end{aligned} \quad (17)$$

The matrix $\hat{\mathbf{F}}$ can then be obtained as (18), shown at the bottom of the following page, and $\mathbf{f} = [f_1, \ldots, f_m]^T$ is the known desired output vector from training data. In (18),

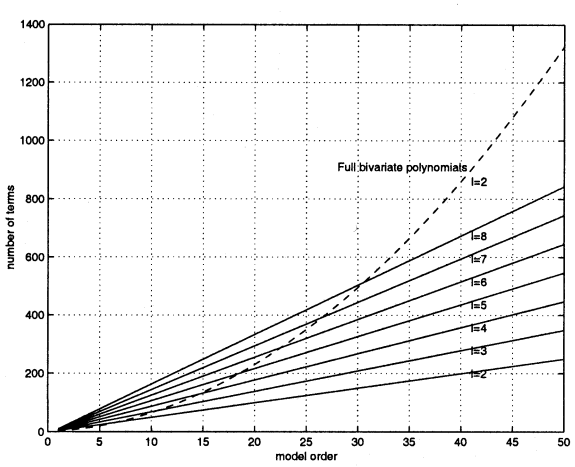


Fig. 2. Number of terms plotted over model order for different input dimensions.

the first and second subscripts of the matrix elements $x_{j,k}$, $j = 1, 2, k = 1, \ldots, m$ indicate the number of biometric systems and the the number of instances respectively. The model coefficients $\alpha_0, \ldots, \alpha_8$ can be obtained using (8) and they can be used to predict future decision outcomes given instances of outputs from biometric-A and biometric-B ($x_1$ and $x_2$).

Fig. 1(c) shows the approximated $sinc$ function using a ninth-order model from (16) (RM9). The approximation capability in this case is visibly seen to be comparable to that of the full ninth-order multivariate polynomials (MP9) as shown in Fig. 1(d).

## V. Optimizing the Weight-Decay Regularization

### A. Handling the Multicollinearity Problem

As mentioned, minimization of the sum of squared errors using (8) may result in a multicollinearity problem when a heavy linear dependency of data is present. A simple approach to provide numerical stability is to perform a weight decay regularization as follows:

$$\begin{aligned} s_{\mathrm{MPR}}(\boldsymbol{\alpha}) &= \sum_{i=1}^{m} e_i^2(\boldsymbol{\alpha}) + b\|\boldsymbol{\alpha}\|_2^2 \\ &= \sum_{i=1}^{m} \left[ f(\boldsymbol{r}_i, \boldsymbol{s}_i) - \hat{f}_{\mathrm{RM}}(\boldsymbol{r}_i, \boldsymbol{s}_i, \boldsymbol{\alpha}) \right]^2 \\ &\quad + b\|\boldsymbol{\alpha}\|_2^2 \end{aligned} \tag{19}$$

where $\|\cdot\|_2$ denotes the $L_2$-norm and $b$ is a regularization constant.

Minimizing the new objective function (19) results in

$$\boldsymbol{\alpha} = (\hat{\mathbf{F}}^T\hat{\mathbf{F}} + b\mathbf{I})^{-1}\hat{\mathbf{F}}^T\mathbf{f} \tag{20}$$

where $\hat{\mathbf{F}} \in \mathcal{R}^{m\times l}$ as defined in (9), $\mathbf{f} \in \mathcal{R}^{m\times 1}$ is the training output, and $\mathbf{I}$ is a $(l \times l)$ identity matrix. It is noted that this addition of a bias term into the least-squares regression model is also termed as *ridge regression* [24].

The addition of the bias term affects the total mean squared error of the original estimator. For a large value of $b$ selected, a large value of the bias will be incurred to the total mean squared error for training [24]. Depending on individual applications, the effect of $b$ on the validation error varies from case to case. Hence the selection of $b$ becomes an important issue for good generalization.

### B. Optimizing the Regularization Parameter

A conventional approach to determine the biasing constant $b$ is to obtain a simultaneous plot of the estimated parameters over different values of $b$, usually between 0 and 1, and look for the smallest value of $b$ where these estimated parameter traces are deemed to first becoming stable [24]. In this paper, we seek to select $b$ by optimizing the validation error. Instead of using the more complex gradient-based search [27], we adopt a simple single directional search using a validation formulation which can globally locate the optimal $b$ within the interval $[0, 1]$.

Given the training data set $\mathcal{S}_{\mathrm{train}}$, we further partition it into two portions namely: the subtraining set $\mathcal{S}_{\mathrm{tr}}$ and the validation set $\mathcal{S}_v$. The subtraining set will be used to compute the weight parameter $\boldsymbol{\alpha}$ and the validation set will be used to determine the regularization parameter $b$.

Consider the estimate given by $\hat{\mathbf{f}} = \hat{\mathbf{F}}\boldsymbol{\alpha} = \hat{\mathbf{F}}(\hat{\mathbf{F}}^T\hat{\mathbf{F}} + b\mathbf{I})^{-1}\hat{\mathbf{F}}^T\mathbf{f}$ using the subtraining data set. With some algebraic manipulation, the sum of squared errors for the subtraining data can be written as

$$SSE_{\mathrm{tr}}(b) = \mathbf{f}^T[\mathbf{I} - \hat{\mathbf{F}}(\hat{\mathbf{F}}^T\hat{\mathbf{F}} + b\mathbf{I})^{-1}\hat{\mathbf{F}}^T]\mathbf{f}. \tag{21}$$

For the case of validation data $\hat{\mathbf{F}}_v$, the new estimate becomes $\hat{\mathbf{f}}_v = \hat{\mathbf{F}}_v\boldsymbol{\alpha} = \hat{\mathbf{F}}_v(\hat{\mathbf{F}}^T\hat{\mathbf{F}} + b\mathbf{I})^{-1}\hat{\mathbf{F}}^T\mathbf{f}$. The sum of squared errors for the validation output $\mathbf{f}_v$ is thus

$$\begin{aligned} \mathrm{SSE_v}(b) &= (\mathbf{f}_v - \hat{\mathbf{f}})^T(\mathbf{f}_v - \hat{\mathbf{f}}) \\ &= [\mathbf{f}_v - \hat{\mathbf{F}}_v(\hat{\mathbf{F}}^T\hat{\mathbf{F}} + b\mathbf{I})^{-1}\hat{\mathbf{F}}^T\mathbf{f}]^T \\ &\quad \times [\mathbf{f}_v - \hat{\mathbf{F}}_v(\hat{\mathbf{F}}^T\hat{\mathbf{F}} + b\mathbf{I})^{-1}\hat{\mathbf{F}}^T\mathbf{f}]. \end{aligned} \tag{22}$$

By performing a single-directional search on $\mathrm{SSE_v}(b)$ for $b$ over the interval $[0, 1]$, an optimal value of $b$ can be found. Here, we note that optimizing the formulation in (22) with respect to $b$ implicitly optimizes $\boldsymbol{\alpha}$ as well because $\boldsymbol{\alpha} = (\hat{\mathbf{F}}^T\hat{\mathbf{F}} + b\mathbf{I})^{-1}\hat{\mathbf{F}}^T\mathbf{f}$.

### C. Optimizing Cross Validations

It is noted that the above optimally regularized model may be sensitive to different data partitions in cross validations. To achieve a more stable result than that using just a single cross-

$$\hat{\mathbf{F}} = \begin{bmatrix} 1 & x_{1,1} & x_{2,1} & x_{1,1}^2 & x_{2,1}^2 & (x_{1,1}+x_{2,1}) & (x_{1,1}+x_{2,1})^2 & x_{1,1}(x_{1,1}+x_{2,1}) & x_{2,1}(x_{1,1}+x_{2,1}) \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x_{1,m} & x_{2,m} & x_{1,m}^2 & x_{2,m}^2 & (x_{1,m}+x_{2,m}) & (x_{1,m}+x_{2,m})^2 & x_{1,m}(x_{1,m}+x_{2,m}) & x_{2,m}(x_{1,m}+x_{2,m}) \end{bmatrix} \tag{18}$$

validation point, we can empirically select the best operating point based on the distribution of the various optimal settings.

As mentioned, we partition the data into three parts: sub-training data $\mathcal{S}_{\rm tr}$, validation data $\mathcal{S}_v$ and test data $\mathcal{S}_{\rm test}$. Here we note that $\mathcal{S}_{\rm train} = \{\mathcal{S}_{\rm tr}, \mathcal{S}_v\}$. The sub-training and the validation data $\{\mathcal{S}_{\rm tr}, \mathcal{S}_v\}$ are used to obtain the training and regularization parameters. The remaining data $\mathcal{S}_{\rm test}$, which are not used in both training and validation, will then be used to test the performance of the classifier.

The set $\mathcal{S}_{\rm train}$ can be partitioned into $\mathcal{S}_{\rm tr}$ and $\mathcal{S}_v$ in various ways, typically $N(\mathcal{S}_{\rm tr})/N(\mathcal{S}_{\rm train}) \geq 0.5$ with $N(\mathcal{S}_v) = N(\mathcal{S}_{\rm train}) - N(\mathcal{S}_{\rm tr})$ where $N(\bullet)$ denotes the number of elements within the set $\bullet$.

In this application we use $N(\mathcal{S}_{\rm tr})/N(\mathcal{S}_{\rm train}) = 0.9$ and perform a corresponding ten-fold cross-validation using the data sets $\{\mathcal{S}_{tr}^i, \mathcal{S}_v^i\}_i$, $i = 1, \ldots, 10$. For each set of cross validation labeled as $i$, the optimal regularization parameter $b_i$ is determined with the corresponding weight vector $\boldsymbol{\alpha}_i$. Using these 10 sets of optimized parameters, we obtain 10 sets of training outputs ($\hat{\mathbf{f}}_i = \hat{\mathbf{F}}\boldsymbol{\alpha}_i$, $i = 1, \ldots, 10$) and respective optimized regularization parameter $b_i$, $i = 1, \ldots, 10$. To obtain a good representation of data distribution, the experiments can be performed $M$ times using different random partitions of $\mathcal{S}_{\rm train}$ and $\mathcal{S}_{\rm test}$. A distribution of $b$ can be plotted and the most populated value of optimized $b$ can be selected for the final test.

*Remark 1:* The above procedure provides a structural way to obtain a very likely good choice of the regularization parameter $b$. However, for large data sets, the computational requirement could be high. For simplicity reason in such applications, we note that an empirical choice of $b \in [10^{-4}, 10^{-3}]$ could produce reasonably good results from our experience.

□

## VI. A Classification Example: Iris Data

In this example, the reduced multivariate polynomial model (16) is tested using the well-used benchmark IRIS data from the UCI Machine Learning Repository [28] to illustrate its classification capability before applying it to the data fusion problem. Here we treat the extracted feature patterns in this example as the outputs of some estimators for fusion. Features extraction is an important part in a classification system since it involves a certain decision process to select relevant features to be used. This experiment is an important complement of our data fusion experiments in Section VII since the IRIS data used here are a prevalent common benchmark.

The data set contains 150 samples belonging to three subspecies of dimension 4. Each of these three subspecies or classes occupies 50 samples. One class is well separated from the other two classes. The available 150 samples were randomly partitioned into a training set and a test set, each containing 75 samples. A total of $M = 40$ trials were performed for the randomly partitioned training and test sets. The four input features were entered as $(x_1, \ldots, x_4)$ in the reduced model and matrix $\hat{\mathbf{F}}$ was obtained using the reduced model which has expansion terms arranged similar to (18). The regularized solution form as in (20) was used to compute the polynomial coefficients. The RM with order 6 (RM6) was chosen for the validation experiment. Here

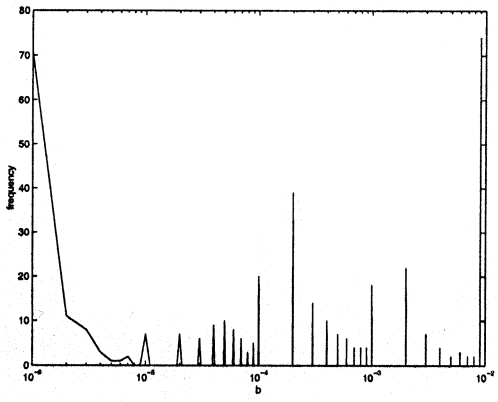


Fig. 3. Iris data: distribution of regularization parameter $b$.

we seek to optimize the regularization parameter by the validation method presented in Section V-B. The search for $b$ was constrained within the interval $[10^{-6}, 10^{-2}]$ since beyond this range the bias could be either too large or too small to have desired effect. It is noted that a large $b$ value will result in a large bias from the optimal solution to the original formulation and a small $b$ value will have insignificant regularization effect. Fig. 3 shows the distribution of optimal values of 10-fold validated $b$ for the 40 trials for RM6. Discarding those undesirable peaks around the search boundaries, here we see that the optimal values of $b$ are concentrated around $2 \times 10^{-4}$, and for consistency we shall use this value for the following study.

For the next set of experiments, we shall observe the classification accuracy with respect to model orders. The average classification errors were recorded for the reduced model for different model orders (RM2–RM6). Table I shows the average training and test errors over 40 trials for respective model orders RM2 through RM6 with respective statistics ($\sigma$: standard deviation). These results outperform the best reported test result ($\%Error_{\rm train} = 2.4$, $\sigma_{\rm train} = 2.2\%$, $Error_{\rm test} = 4.8$, $\sigma_{\rm test} = 1.2$) from the reformulated radial basis neural networks as seen in [29]. Moreover, in [29], only five trials were performed for each averaged error value and training of neural networks is a nontrivial process. For a better picture on the performance, a 10-fold cross-validation test is also included in Table I using similar regularization setting.

Table II compares RM6 with several commonly used classifiers: Naive–Bayes (NBayes), SVM, and Neural Network (Neural). The Naive–Bayes method [30] assumed Gaussian distribution and independency of the input data. A 10th-order polynomial kernel was adopted for SVM learning [31] since it was found to outperform other kernels (linear and RBF) in this example. The neural network used a $(4 \times 4 \times 1)$ structure and was trained using the method proposed by [10] to achieve possible global optimality. As seen from the 40-trials results, RM6 outperforms all other classifiers for both training and test data. The 10-fold validation results show that the Neural Network method with appropriate regularization and tuning [10] can produce better results than all the compared methods. The SVM for some instances appear to be overly trained in the 10-fold process. In terms of computing effort, the Neural Network, SVM, and Naive–Bayes, respectively, took about

TABLE I
IRIS DATA: AVERAGE ERRORS FOR DIFFERENT MODEL ORDERS

| Model | 40-trials | | 10-fold | |
|---|---|---|---|---|
| | $\%Error_{train}$ $(\sigma_{train})$ | $\%Error_{test}$ $(\sigma_{test})$ | $\%Error_{train}$ | $\%Error_{test}$ |
| RM2 | 2.90(0.0159) | 4.17 (0.0148) | 2.59 | 4.67 |
| RM3 | 2.20(0.0137) | 3.63 (0.0154) | 2.00 | 4.00 |
| RM4 | 1.67(0.0141) | 3.10 (0.0136) | 2.00 | 3.33 |
| RM5 | 1.47(0.0131) | 2.77 (0.0126) | 1.85 | 3.33 |
| RM6 | 1.37(0.0126) | 2.80 (0.0124) | 1.63 | 3.33 |

TABLE II
IRIS DATA: AVERAGE ERRORS FOR DIFFERENT METHODS

| Model | 40-trials | | 10-fold | |
|---|---|---|---|---|
| | $\%Error_{train}$ $(\sigma_{train})$ | $\%Error_{test}$ $(\sigma_{test})$ | $\%Error_{train}$ | $\%Error_{test}$ |
| NBayes | 3.60 (0.0139) | 4.37 (0.0157) | 3.93 | 4.00 |
| SVM | 3.00 (0.0153) | 3.83 (0.0163) | 3.48 | 5.33 |
| Neural | 2.47 (0.0111) | 3.20 (0.0138) | 2.00 | 0.67 |
| RM6 | 1.37 (0.0126) | 2.80 (0.0124) | 1.63 | 3.33 |

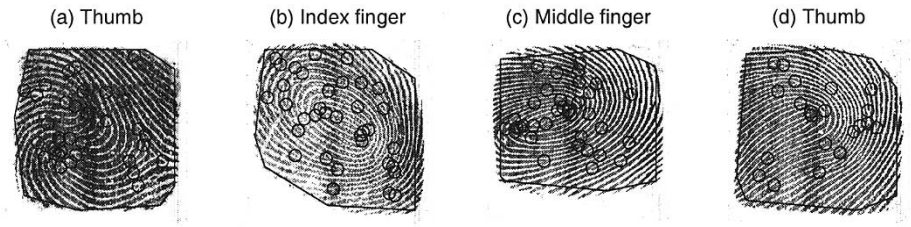


Fig. 4. Fingerprint image samples.

3250, 3.3, and 2.5 times the RM6's computing time to obtain the required parameters under similar computing conditions.

## VII. COMBINING FINGERPRINT AND SPEAKER CLASSIFIERS

In this section, we perform experiments on the multivariate polynomial fusion using physical data from two biometrics: fingerprint and voice data. We shall compare the performances of RM with OWM and a few commonly used classifiers, in the form of receiver operating characteristic (ROC) curves. As only two biometrics are to be combined, we shall limit the order of the RM model up to three in this experiment to avoid over-modeling.

### A. Fingerprint Verification

In general, an automatic fingerprint identification or verification (see, e.g., [32]–[35]) system consists of three main processing stages namely, *image acquisition*, *feature extraction*, and *matching*. In image acquisition, query and template database images are acquired through various input devices. Development over the years has seen through means that mechanically scan the ink based fingerprints into the computer system, to means which directly capture the fingerprints using sophisticated solid state sensors. With fingerprint images which could be distorted or contaminated with noise, the automated system seeks to *extract* characteristic *features* which are discriminating for different fingers and yet invariant with respect to image orientation for same fingers. The final stage of fingerprint identification is to search and verify matching image pairs.

Our representation for the fingerprint consists of a global structure and a local structure. The global structure consists of positional and directional information of ridge endings and ridge bifurcations. The local structure consists of relative information of each detected minutia with other neighboring minutiae. Fingerprint verification is then performed by comparing the minutia information between two templates [36]. Fig. 4 shows some samples of the fingerprint images with detected minutiae and area of interest segmentation. The interested readers are referred to [36] and [37] for details of minutiae detection and matching.

### B. Speaker Verification

Speaker verification seeks to determine whether an unknown voice matches the known voice of a speaker with known identity. It is a subset of the more general problem of speaker recognition which includes the task of speaker identification (see e.g., [38]). Operation of the above systems can either be in fixed-text mode or in free-text mode. In fixed-text mode, a predetermined text is required to be recited for reliable comparison whereas in free-text mode, speech utterances of unrestricted text can be

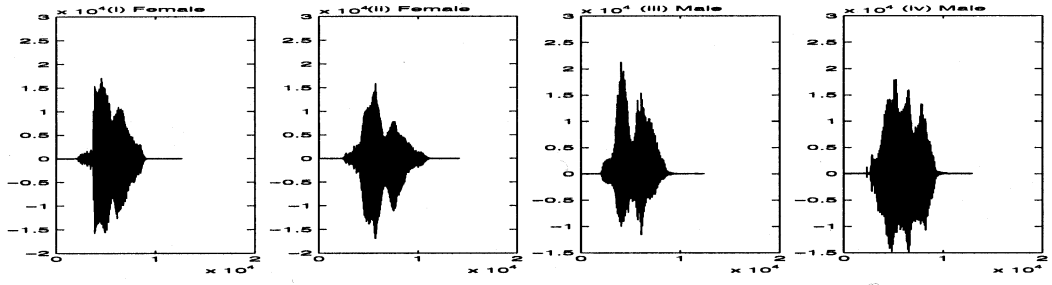


Fig. 5. Voice samples.

accepted. The fixed-text system provides a more precise and reliable comparison between two utterances of the same text than that of the free-text system since it works under a better controlled environment. The fixed-text system are thus primarily used in access control applications and the free-text systems are more for surveillance and other applications [38].

In this application, the fixed-text mode and the template matching method is adopted for speaker verification. Comparison of two utterances is performed by aligning the two templates at corresponding points in time. To cater for difference in duration of the two utterances, the dynamic time warping (DTW) method is adopted when minimizing a distance metric between two feature sets extracted from the speech data. Fig. 5 shows some samples of voice data uttering the word "zero." The interested reader is referred to [39] for more details about the system (see also [38] and [40] for similar matching designs).

### C. Combining Fingerprint and Speaker Verification Systems

In this experiment, both the databases for fingerprint verification and speaker verification consist of 16 different identities; each comes with six different fingers or six different words with each fingerprint or word containing ten samples. A total of 960 samples were thus used for each fingerprint and voice verification system. The fingerprint images were collected using Veridicom's $i$ Touch sensor and the voice data were taken from TIDIGIT database. An arbitrary one-to-one correspondences were taken between the two biometric databases. It is noted here that six different fingers taken from the same person are uncorrelated in their minutia distribution and hence we can reasonably assume that the fingerprint database contains 96 different users ($16 \times 6$). As for the speech database, the same person uttering six different words may have some correlation among the words as compared to those from six different persons. Apparently, this may result in lowering the matching performance of the speaker verification system. Since our focus is on matching performance improvement using two or more "not-so-high-performance" biometrics, we nevertheless assume a total of 96 different users ($16 \times 6$) for both databases in the following experiments.

Both the fingerprint and speech databases are partitioned into two equal sets for training and testing, i.e., set-1 with 480 ($16 \times 6 \times 5$) samples for training and, set-2 with 480 ($16 \times 6 \times 5$) samples for testing. These training and test sets each contains a total of 960 $(96 \times 5 \times 4/2)$ matching scores generated among the genuine-users and 45 000 $([(16 \times 15)/2] \times [(6 \times 5)/2] \times (5 \times 5))$ matching scores generated across the imposters for both biometrics. The reason for using only 45 000 out of the total possible 114 000 $(96 \times 95 \times 5 \times 5/2)$ imposter scores is to have a smaller difference between the sizes of the imposter and the genuine user data sets. The fingerprint match score $(\hat{f}_1)$ and the speaker match score $(\hat{f}_2)$ from the training set are entered as polynomial variables $x_1$ and $x_2$ respectively to form the matrix $\hat{\mathbf{F}}$ which contains the reduced polynomial expansion terms. The polynomial coefficients are solved using the regularized objective which has a solution form similar to (20). These polynomial coefficients are then used to predict the combined out-come for test data based on the reduced model function (16).

Depending on individual implementation, the matching output ranges for different modalities may differ significantly. For such cases, numerical sensitivity may be affected and hence a score normalization should be performed between the outputs of different modalities. For reasonably small differences between the scores like in our case (fingerprint-match scores in $\in [0, 115]$, speaker-match scores $\in [0, 2.05]$), the weighting parameters can be adapted automatically without serious floating point truncation and hence no normalization is needed. Figs. 6 and 7 show the original matching performances for the training and test sets, respectively, for individual fingerprint verification and speaker verification, using the above mentioned database, before multimodal fusion. In what follows, we shall observe the effects of regularization and different model orders on the fusion performance.

### D. Effects of Regularization

A search for an optimal regularization parameter was performed for the three models OWM, RM2,and RM3 using the procedure outlined in Section V-B with arbitrary partition of training data. The empirical search for all three models resulted in having $b \approx 10^{-4}$. A plot of the ROC curve does not reflect visible difference between the regularized models and the non-regularized models. This shows that multicollinearity may not be a serious problem in this application.

### E. Effects of Model Orders

Fig. 8 shows the training and test results for multimodal data fusion using OWM, RM2, and RM3. As in Section VI, these results are compared to several commonly used classifiers: Naive–Bayes [30], SVM [31], and Neural Network [10]. The SVM used the RBF kernel as it was found to have good convergence as compared to the polynomial kernel. For the Neural

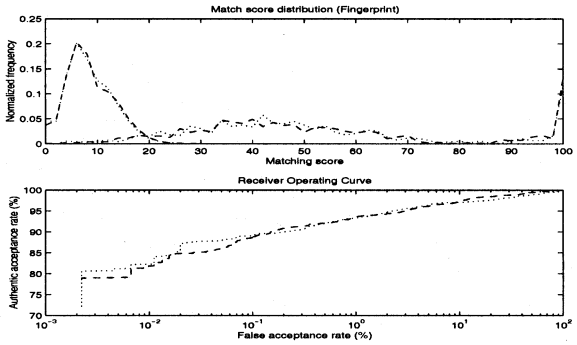


Fig. 6. Matching curves for fingerprint verification training (dashed lines) and test (dotted lines) sets.

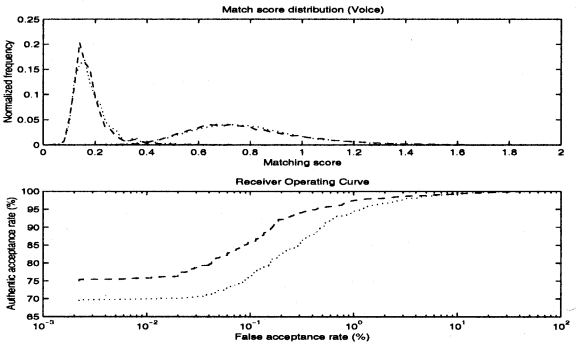


Fig. 7. Matching curves for speaker verification training (dashed lines) and test (dotted lines) sets.

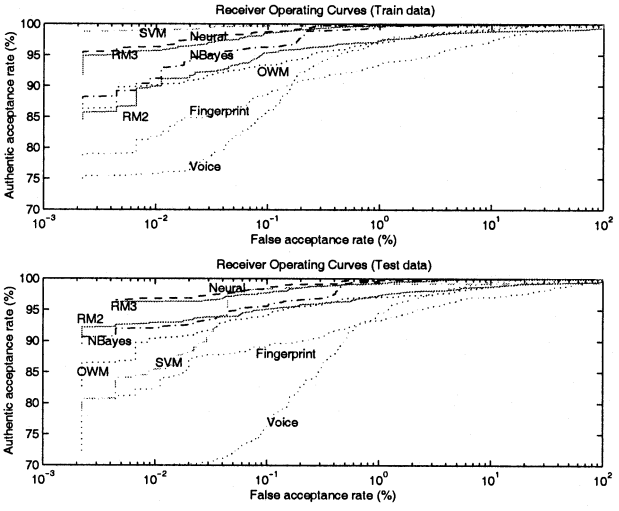


Fig. 8. ROC curves for fingerprint (dotted), voice (dotted), OWM (dotted), Naive–Bayes (dashed–dotted), neural network (dashed), SVM (dashed–dotted), RM2 (continuous), and RM3 (continuous).

Network, a $(2 \times 2 \times 1)$ structure was selected as it was found to be sufficient for the approximation. The Naive–Bayes assumed a Gaussian distribution of data and independency between the outputs of the two biometrics. The ROC curves for the original fingerprint and speaker verifications are also included in the same plots for comparison. It can be seen from these plots that OWM improves significantly over the entire operating range of

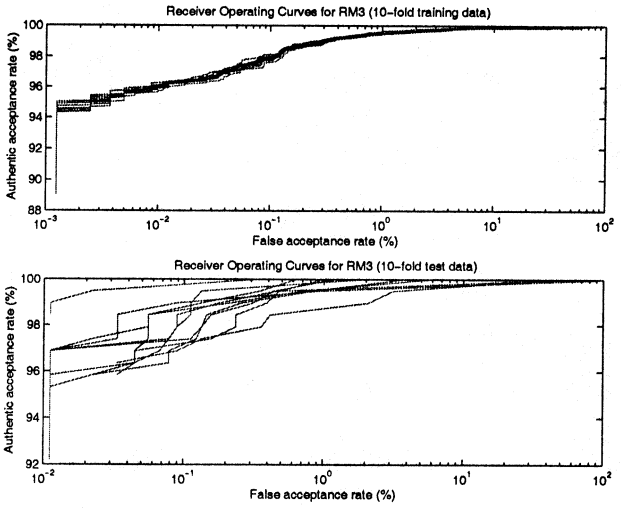


Fig. 9. Tenfold ROC curves for RM3.

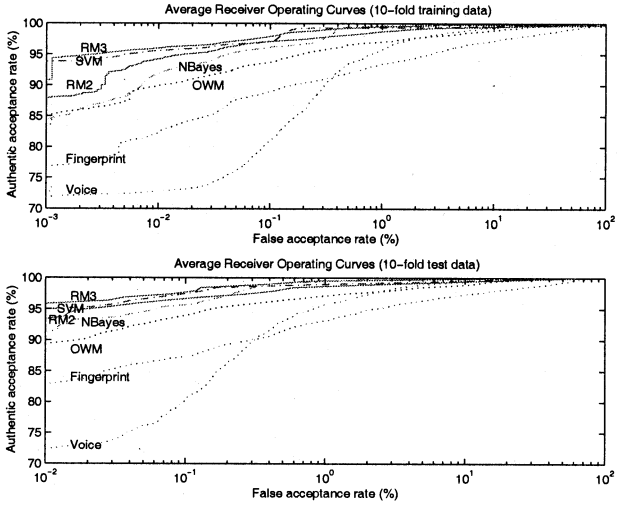


Fig. 10. Average tenfold ROC curves for fingerprint (dotted), voice (dotted), OWM (dotted), Naive–Bayes (dashed-dotted), SVM (dashed–dotted), RM2 (continuous) and RM3 (continuous).

individual fingerprint and speaker verifications. As the model order increases using RM2 and RM3, the improvement is even more significant. For RM3, the authentic acceptance rates are all found to be over 95%. The SVM shows excellent training results, but the test results show that it has been over-fitted. The Neural Network gives the best test result with RM3 followed by closely. In terms of computing effort, the Neural Network, SVM, and Naive–Bayes, respectively, took about 239, 3600, and 0.27 times the computing time of RM3 to obtain the required parameters under similar computing conditions.

In addition to the above results using equal partitioning of the entire data set into a training set and a test set, a tenfold cross-validation test process is performed to provide a better picture of the fusion accuracy. Fig. 9 shows the ROC curves for training and test sets using RM3. For clarity purpose, the average ROC curves of the tenfold results from the compared methods are shown in Fig. 10 except for that using the Neural Network method because numerical ill-conditioning was found during its training. It is clear from this figure that RM3 and SVM

outperform all other compared methods for both training and test data.

## VIII. Conclusion

In this paper, we proposed a reduced multivariate polynomial model where the number of parameters increases linearly with model order and number of inputs. Even though the approximation capability of this model is found to be compromised from an empirical example, we found that it can be well applied for multimodal classifiers fusion. The main advantage of this model over more complex neural-network-like models is its straightforward model parameter computation. The model is first applied to a well-known pattern classification problem to illustrate its classification capability. This is followed by a biometrics fusion problem combining fingerprint and voice data. The combined results of the proposed model show superiority of performance over the optimal weighting method and a few commonly used classifiers in terms of the ROCs. The proposed model is found to give a performance comparable to that of a neural network with a much faster computing speed.

## Acknowledgment

The authors would like to thank Y. Xu and C. Li for providing the data and classifier related to speaker verification.

## References

[1] L. I. Kuncheva, J. C. Bezdek, and R. Duin, "Decision templates for multiple classifier design: An experimental comparison," *Pattern Recognit.*, vol. 34, no. 2, pp. 299–314, 2001.

[2] R. Brunelli and D. Falavigna, "Personal identification using multiple cues," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 17, pp. 955–966, Oct. 1995.

[3] L. Hong and A. Jain, "Integrating faces and fingerprints for person identification," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 20, pp. 1295–1307, Dec. 1998.

[4] J. Kittler, M. Hatef, R. P. W. Duin, and J. Matas, "On combining classifiers," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 20, pp. 226–239, Mar. 1998.

[5] L. I. Kuncheva, "A theoretical study on six classifier fusion strategies," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 24, pp. 281–286, Feb. 2002.

[6] T. Poggio and F. Girosi, "Networks for approximation and learning," *Proc. IEEE*, vol. 78, pp. 1481–1497, 1990.

[7] L. L. Schumaker, *Spline Functions: Basic Theory*. New York: Wiley, 1981.

[8] G. Cybenko, "Approximations by superpositions of a sigmoidal function," *Math. Cont. Signal Syst.*, vol. 2, pp. 303–314, 1989.

[9] K. Hornik, M. Stinchcombe, and H. White, "Multi-layer feedforward networks are universal approximators," *Neural Networks*, vol. 2, no. 5, pp. 359–366, 1989.

[10] K -A. Toh, "Deterministic global optimization for FNN training," IEEE Trans. Syst., Man Cybern. B, vol. 33, pp. 977–983, June 2003, to be published.

[11] K.-A. Toh, "Global optimization by monotonic transformation," *Computat. Optim. Applicat.*, vol. 23, pp. 77–99, Oct. 2002.

[12] F. Schwenker, H. A. Kestler, and G. Palm, "Radial-basis-function netowrks: Learning and applications," in *Proc. 4th Int. Conf. Knowledge-Based Intelligent Engineering Systems & Allied Technologies*, Brighton, U.K., 2000, pp. 33–43.

[13] Y. Shin and J. Ghosh, "Ridge polynomial networks," *IEEE Trans. Neural Networks*, vol. 6, no. 3, pp. 610–622, 1995.

[14] Y.-H. Pao and Y. Takefuji, "Functional-link net computing: Theory, system architecture, and functionalities," *Computer*, vol. 25, no. 5, pp. 76–79, 1992.

[15] D. F. Soecht, "A general regression neural network," *IEEE Trans. Neural Networks*, vol. 2, no. 6, pp. 568–576, 1991.

[16] T. Masters and W. Land, "A new training algorithm for the general regression neural network," in *Proc. IEEE Int. Conf. Systems, Man, and Cybernetics*, vol. 3, 1997, pp. 1990–1994.

[17] D. F. Soecht, "Probabilistic neural networks," *Neural Networks*, vol. 3, pp. 109–118, 1990.

[18] A. Zaknich, "Introduction to the modified probabilistic neural network for general signal processing applications," *IEEE Trans. Signal Processing*, vol. 46, pp. 1980–1990, July 1998.

[19] T.-T. Lee and J.-T. Jeng, "The chebyshev-polynomial-based unified model neural networks for function approximation," *IEEE Tran. Syst., Man Cybern.*, vol. 28, pp. 925–935, June 1998.

[20] A. Sierra, J. A. Macías, and F. Corbacho, "Evolution of functional link networks," *IEEE Trans. Evol. Comput.*, vol. 5, no. 1, pp. 54–65, 2001.

[21] V. N. Vapnik, *Statistical Learning Theory*. New York: Wiley-Interscience, 1998.

[22] C. J. C. Burges, "A tutorial on support vector machines for pattern recognition," *Data Mining and Knowledge Discovery*, vol. 2, no. 2, pp. 121–167, 1998.

[23] N. Ueda, "Optimal linear combination of neural networks for improving classification performance," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 22, pp. 207–215, Feb. 2000.

[24] J. Neter, M. H. Kutner, C. J. Nachtsheim, and W. Wasserman, *Applied Linear Regression Models*, 3rd ed. Chicago, IL: Irwin, 1996.

[25] C. J. Merz, "A principal components approach to combining regression estimates," *Machine Learning*, vol. 36, pp. 9–32, 1999.

[26] W. R. Wade, *An Introduction to Analysis*, 2nd ed. Upper Saddle River, NJ: Prentice-Hall, 2000.

[27] R. Eigenmann and J. A. Nossek, "Gradient based adaptive regularization," in *Proc. IEEE Signal Processing Workshop: Neural Networks for Signal Processing*, 1999, pp. 87–94.

[28] C. L. Blake and C. J. Merz. (1998) UCI Repository of Machine Learning Databases. Univ. California Dept. of Information and Computer Sciences, Irvine. [Online]. Available: http://www.ics.uci.edu/~mlearn/ML-Repository.html

[29] N. B. Karayiannis, "Reformulated radial basis neural networks trained by gradient descent," *IEEE Trans. Neural Networks*, vol. 10, no. 3, pp. 657–671, 1999.

[30] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*, 2nd ed. New York: Wiley, 2001.

[31] J. Ma, Y. Zhao, and S. Ahalt. (2002) OSU SVM Classifier Matlab Toolbox (Ver 3.00). The Ohio State University. [Online]. Available: http://eewww.eng.ohio-state.edu/~maj/osu_svm/

[32] A. Jain, L. Hong, and R. Bolle, "On-line fingerprint verification," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 19, pp. 302–313, Apr. 1997.

[33] A. K. Jain, L. Hong, S. Pankanti, and R. Bolle, "An identity-authentication system using fingerprints," *Proc. IEEE*, vol. 85, pp. 1365–1388, 1997.

[34] N. K. Ratha, K. Karu, S. Chen, and A. K. Jain, "A real-time matching system for large fingerprint databases," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 18, pp. 799–812, Aug. 1996.

[35] U. Halici *et al.*, "Introduction to fingerprint recognition," in *Intelligent Biometric Techniques in Fingerprint and Face Recognition*. ser. CRC Press Int. Series on Computational Intelligence, L. C. Jain *et al.*, Eds. Boca Raton, FL: CRC Press, 1999, pp. 3–34.

[36] X. Jiang and W. Y. Yau, "Fingerprint minutiae matching based on the local and global structures," in *Proc. 15th Int. Conf. Pattern Recognition*, vol. 2, 2000, pp. 1042–1045.

[37] X. Jiang, W. Y. Yau, and W. Ser, "Detecting the fingerprint minutiae by adaptive tracing the gray-level ridge," *Pattern Recognit.*, vol. 34, no. 5, pp. 999–1013, 2001.

[38] J. M. Naik, "Speaker verification: A tutorial," *IEEE Commun. Mag.*, pp. 42–48, Jan. 1990.

[39] C. Li and R. Venkateswarlu, "High accuracy connected digits recognition system with less computation," in *Proc. 6th World Multiconf. Systemics, Cybernetics and Informatics (SCI 2002)*, Orlando, FL, July 2002.

[40] L. Rabiner and J. Juang, *Fundamentals of Speech Recognition*. Englewood Cliffs, NJ: Prentice-Hall, 1993.