A Radial Basis Function Approach to Pattern Recognition and Its Applications

Miyoung Shin[a] and Cheehang Park

Pattern recognition is one of the most common problems encountered in engineering and scientific disciplines, which involves developing prediction or classification models from historic data or training samples. This paper introduces a new approach, called the *Representational Capability* (RC) algorithm, to handle pattern recognition problems using radial basis function (RBF) models. The RC algorithm has been developed based on the mathematical properties of the interpolation and design matrices of RBF models. The model development process based on this algorithm not only yields the best model in the sense of balancing its parsimony and generalization ability, but also provides insights into the design process by employing a design parameter (δ). We discuss the RC algorithm and its use at length via an illustrative example. In addition, RBF classification models are developed for heart disease diagnosis.

I. INTRODUCTION

The term pattern recognition encompasses a wide range of information processing tasks of great practical significance. The spectrum of such tasks spans from speech recognition to handwritten character classification and from medical diagnosis to fault detection in nuclear fuel rods. Humans are particularly good at solving many such problems effortlessly and even subconsciously, e.g., recognizing telephone callers from their voice. On the other hand, other problems, such as reading bar codes, require machines to perform more accurately than humans can [1], [2]. The scientific discipline of building such machines for these tasks is the domain of pattern recognition.

The traditional formulation of pattern recognition tasks has been based on statistical methods. On the other hand, artificial neural networks are now extensively used for these tasks because of their good ability to handle large-scale practical problems. Radial Basis Function (RBF) models, the focus of this paper, is a class of neural networks and provides such ability. In addition, it has the very attractive feature of having both nonlinearity and linearity in the model which can be treated separately. Further, RBF models have been shown to possess very significant mathematical properties of universal and best approximation [3]. All these features make RBF models attractive for many applications. In fact, the range of fields in which this model has been employed is very impressive including geophysics, signal processing, meteorology, orthopedics, computational fluid dynamics and pattern recognition [4].

The problem of developing an RBF model is simply to determine the model parameters which will achieve the expected performance. Therefore, the most significant issues in constructing the RBF model are the criteria and algorithm for determining its parameters. Two important criteria are those of

Manuscript received August 28, 1999; revised May 13, 2000.

[a] Electronic mail: shinmy@etri.re.kr

parsimony and generalization ability of the model, i.e., the model should have as few parameters as possible and also provide good predictions for future inputs. An important aim of this paper is to present an efficient, new methodology for the design and evaluation of RBF models and illustrate its application in heart disease diagnosis.

This paper is organized as follows. A detailed mathematical description of the pattern recognition problem is given in Section II. The RBF model and the modeling issues are discussed in Section III. A new design methodology is briefly described in Section IV, and illustrated via a detailed example in Section V. An actual classification study for heart disease diagnosis is presented in Section VI. Finally, Section VII provides concluding remarks.

II. PATTERN RECOGNITION PROBLEM

1. Problem Definition

The task of pattern recognition is to construct a model that captures an unknown input-output mapping pattern on the basis of limited evidence about its nature. The evidence available is a set of labeled historic data and is called the training sample. We wish to construct the "best" model that is as close as possible to the true but unknown mapping function. This process of model development is commonly termed *training or modeling*.

Under the theme of the best model construction, the process of training is to determine the parameters such that the fitted model provides good fit to training data as well as good predictions on future data. Note that our objective is not to learn an exact representation of the data because that would lead to overfitting. For developing a good predictive model, it is essential to consider generalization capability of the fitted model for future, yet unseen, inputs for which output would not be known.

Now we state the pattern recognition problem and its objective analytically. Suppose we are given a training data set D,

$$D = \{(\mathbf{x}_i, y_i) : \mathbf{x}_i \in R^d, y_i \in R, i = 1,...,n\},$$

in which both inputs and their corresponding outputs are made available and the outputs are continuous or discrete values. The problem we address is to find a mapping function from the d-dimensional input space to the 1-dimensional output space based on the data set D. From a modeling perspective, we seek a model that provides the best fit to training data and the best prediction on future data while minimizing model complexity. The resulting model would be employed to predict output values y' for future observed inputs $\mathbf{x}'$ where only the inputs $\mathbf{x}'$ would be available, not the outputs y'.

In the regression setting where the output y has continuous values, it is assumed that y is related to $\mathbf{x}$ as follows:

$$y = h(\mathbf{x}) + \varepsilon, \tag{1}$$

where $h(\mathbf{x})$, called the target function, is a single valued deterministic function of $\mathbf{x}$ and ε is a random variable distributed according to $\varepsilon \sim p(\varepsilon \mid \mathbf{x})$. We assume $E[\varepsilon \mid \mathbf{x}] = 0$ for all $\mathbf{x}$, so that $h(\mathbf{x})$ can be written as

$$h(\mathbf{x}) = E[y \mid \mathbf{x}]. \tag{2}$$

On the other hand, in the classification setting, the output y assumes values on an ordered discrete set $y \in \{y_1, y_2, \cdots, y_L\}$. In the special case of binary classification $(\mathrm{L} = 2)$ without loss of generality we take $y \in \{0, 1\}$. Then

$$h(\mathbf{x}) = E[y \mid \mathbf{x}] = P(y = 1 \mid \mathbf{x}) = 1 - P(y = 0 \mid \mathbf{x}), \tag{3}$$

If a model f built from the training data D is employed as a predictor, prediction inaccuracy can be quantified as $E[y - f(\mathbf{x})]^2$. This represents the sum of squares of validation errors at $\mathbf{x}$, averaged over repeatedly realized training samples, each of size n, from the system being modeled. It can be easily shown that this error term has the following decomposition

$$E[y - f(\mathbf{x})]^2 = E[h(\mathbf{x}) - f(\mathbf{x})]^2 + E_\varepsilon[\varepsilon \mid \mathbf{x}]^2. \tag{4}$$

The second term in the above is independent of the target function $h(\mathbf{x})$ and the training data D. It simply reflects the irreducible validation error caused by the random nature of output variable y. The first term is the squared estimation error in the target function $h(\mathbf{x})$, averaged over the training samples. This depends on the target function $h(\mathbf{x})$ and the fitted model $f(\mathbf{x})$. It is this term that provides a measure of the effectiveness of $f(\mathbf{x})$ as a predictor of y. Therefore, our main objective is to determine the model parameters such that the squared estimation error $E[h(\mathbf{x}) - f(\mathbf{x})]^2$ is minimized.

2. Bias-Variance Components of Estimation Error

The squared estimation error can be decomposed into two components as follows [5], [6]:

$$E[h(\mathbf{x}) - f(\mathbf{x})]^2 = [E(f(\mathbf{x})) - h(\mathbf{x})]^2 + E[f(\mathbf{x}) - E[f(\mathbf{x})]]^2. \tag{5}$$

That is,

$$(\text{estimation error})^2 = (\text{bias})^2 + \text{variance}.$$

The first term on the right hand side of (5) is called *bias* squared and represents the squared deviation of the average of the fitted models over all data sets from the target function $h(\mathbf{x})$. That is, it measures the extent to which a fitted model, on the average, deviates from the true function. Thus, if a fitted

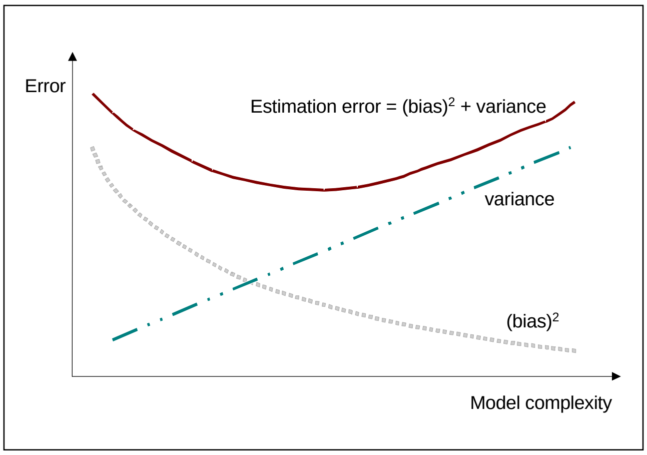

Fig. 1. Trend of estimation error and bias-variance components with respect to model complexity.

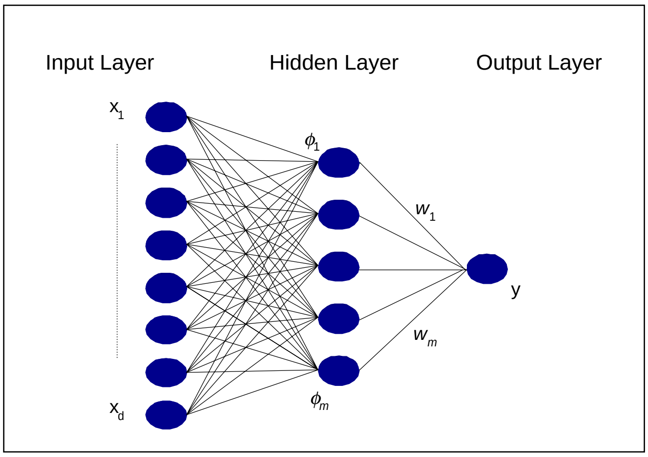

Fig. 2. RBF model structure.

model $f(\mathbf{x})$ is different from the true function $h(\mathbf{x})$, then $f(\mathbf{x})$ is said to be biased as an estimator of $h(\mathbf{x})$. The second term, called *variance,* reflects the sensitivity of $f(\mathbf{x})$ to training data D. It should be noted that the bias term is a measure of how well a fitted model explains the data, i.e., how well the model fits the training data. The variance term, on the other hand, provides a measure of performance of a fitted model on future data sets.

The error term $E[h(\mathbf{x}) - f(\mathbf{x})]^2$ could be large for two reasons. First, it could be large when the average of the fitted model $f(\mathbf{x})$ over all data sets, each of size n, is far off $h(\mathbf{x})$. Second, the fitted model $f(\mathbf{x})$ can vary widely from $h(\mathbf{x})$, over the training data D. That is, for some data set, a fitted model can be an excellent approximation of $h(\mathbf{x})$ while for other data set, it can be very far off $h(\mathbf{x})$. Either of the above circumstances would contribute large values to the squared estimation error term, which leads to an unreliable predictor $f(\mathbf{x})$. Thus, to minimize the estimation error, both the bias and the variance terms should be minimized.

In fact, both the terms cannot be minimzed at the same time because of their conflicting features. In general, if a model is complicated, it tends to produce large variance and small bias. On the other hand, if a model is simple, it tends to produce high bias and small variance. Figure 1 shows the general trend of the estimation error and the bias squared and variance terms with respect to model complexity. The objective is to find a model which is neither too simple nor too complex and provides an acceptable tradeoff between these two error terms.

III. RADIAL BASIS FUNCTION MODEL

1. Model Structure

For given data $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$, the RBF is a nonlinear model where the output estimate $\hat{y}$ for input vector $\mathbf{x}$ is represented by the following functional form

$$\hat{y} = f(\mathbf{x}) = \sum_{j=1}^{m} w_j \phi_j(\mathbf{x}) = \sum_{j=1}^{m} w_j \phi(\|\mathbf{x} - \mu_j\| / \sigma_j), \qquad (6)$$

where $\phi(\cdot)$ is a basis function and μ_i, σ_j are the center and the width of j-th basis functions, respectively; also, w_j is the weight associated with the j-th basis function output; and m is the number of basis functions. Thus, the RBF model is fully determined by $P = (m, \boldsymbol{\mu}, \sigma, \mathbf{w})$.

The RBF model can also be considered as a three-layer network. The first layer (input layer) distributes the input vectors to each hidden unit in the second layer (hidden layer) without any multiplicative factors. The hidden layer has m hidden units, each of which represents a basis function and plays a role in performing nonlinear transformation of the input vector, producing a value as an output. In the third layer (output layer), the outputs from the m hidden units in the hidden layer are linearly combined with the weights to produce a model output.

A typical RBF model structure is shown in Fig. 2. It shows the connectivity between input, hidden and output layers. An input vector in d-dimensional space is transformed by the basis functions into an m-dimensional vector. That is, after passing through the m basis functions, the input vector x $= (x_1, x_2, \cdots, x_d)$ becomes the basis function output vector $\phi = (\phi_1(\mathbf{x}), \phi_2(\mathbf{x}), \cdots, \phi_m(\mathbf{x}))$ where $\phi_j(\cdot)$ is the output from the j-th basis function. The network output y corresponding to an input vector $\mathbf{x}$ is obtained by the summation of the weighted basis function outputs, i.e., $y = \sum_{j=1}^{m} w_j \phi_j(\mathbf{x})$.

A basis function $\phi(\cdot)$ in the RBF model can be seen as a transfer function similar to that in traditional neural networks. Yet, unlike the conventional transfer function, each basis function has the property that its responses to input vectors are

Table 1. Some radial basis function choices.

Basis Function	$\phi(r) = \phi(\lVert \mathbf{x} - \mu \rVert / \sigma)$
Gaussian	$\exp(-r^2/2)$
Thin plate spline	$r^2 \log r$
Inverse multiquadratic	$(r^2 + c^2)^{-1/2}$
Multiquadratic	$(r^2 + c^2)^{1/2}$
Cubic	r^3

radially symmetric with respect to the center μ_j and monotonic with increasing distance from the center. Some popular choices for basis functions and their expressions are given in Table. 1. Amongst these, the Gaussian is the most popular basis function because it has attractive mathematical properties and its hill-like shape is easy to control with parameter σ.

A Gaussian RBF model is defined by the number of basis functions (m), their centers $(\boldsymbol{\mu})$, widths (σ) and weights ($\mathbf{w}$) to the output layer. Amongst these, the parameters m, $\boldsymbol{\mu}$ and σ define the hidden layer, i.e., the nonlinearity of the RBF model, and $\mathbf{w}$ define the linear part as shown in Fig 2. The methods for determining these parameters are called training algorithms. The issue of finding the model parameters has been addressed by several authors over the past ten years. They include the clustering methods [7], orthogonal least squares [8], supervised gradient descent and regularization [1], [7]. Details of these can also be found in [9].

2. Modeling Issues

Recall from Section II that the objective of training is to determine model parameters so as to minimize the squared estimation error that can be decomposed into bias squared and variance. Since both components contribute to the estimation error and cannot be simultaneously minimized, we seek parameter values that give the best compromise between small bias and small variance.

In practice, during the modeling process, the bias squared and the variance cannot be computed because the computation requires knowledge of the true but unknown function. Instead, their trend can be analyzed from the shapes of the training and validation error curves corresponding to model performances on the training and validation data, respectively. The idealized relationship of these errors is graphically depicted in Fig 3. Here we clearly see the conceptual relationship between the expected training and validation errors, which describes the so-called bias-variance dilemma.

We note that the training error decreases with increasing model complexity because the more complicated model tends to provide a better fit to training data. On the other hand, validation error decreases with model complexity up to a certain point and then begins to increase. An increase in validation error implies that the increased model complexity causes overfitting. From the bias and variance perspective, underfitting means too much bias and small variance while overfitting means too much variance and small bias. For RBF, the region to the left of “best” model in Fig. 3 represents models with too few basis functions while the models to the right have too many basis functions.

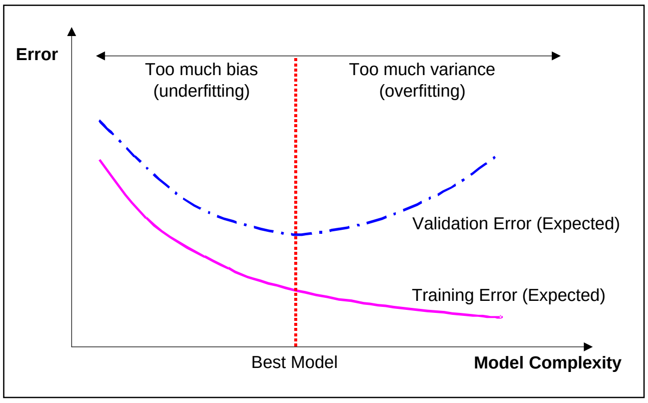

Fig. 3. Idealized depiction of training, validation errors and bias-variance trade-off.

In summary, we seek a model that is neither too simple nor too complex. A model that is too simple will suffer from underfitting because it does not learn enough from the data and hence provides a poor fit. On the other hand, a model that is too complicated would learn details including noise and thus suffer from overfitting. It cannot provide good generalization on unseen data. Hence, for an RBF modeling problem, a very significant issue is to determine an appropriate number of basis functions to achieve a good bias-variance compromise.

VI. RC BASED RBF MODEL DEVELOPMENT

In this section we describe our representational capability (RC) algorithm based on a new mathematical framework developed in [9]. The use of this algorithm in RBF model development guarantees the following:

The nonlinear parameters m, $\boldsymbol{\mu}$ and σ are determined first without reference to output values. Once these parameters are fixed, the linear parameter $(\mathbf{w})$ are determined by referencing target outputs.

The basis function centers $(\boldsymbol{\mu})$ selected by this algorithm are a subset of input vectors $(\mathbf{x})$.

1. Design Objectives and Conceptual Considerations

To achieve our goal of developing the best model, we formulate

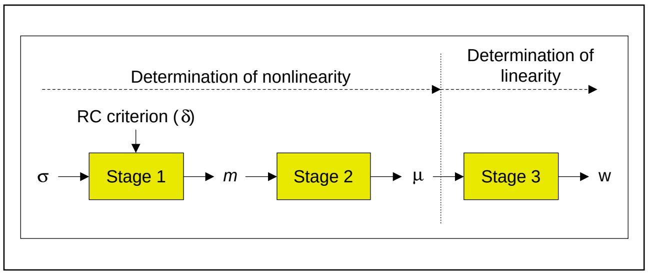

Fig. 4. A three-stage RBF modeling process of RC based algorithm.

the modeling problem as a three stage process shown in Fig. 4. In Stage 1 the RC algorithm selects m for a given σ and a specified value of δ. For the selected (m, σ) pair, the centers are determined in Stage 2. This completes the design of the hidden layer, i.e., the nonlinear part of the RBF model. Finally, in Stage 3, for the selected nonlinear parameters, the linear parameters are determined by the least squares method.

As mentioned above, the mathematical basis for the RC algorithm is discussed in [9]. Here we provide a brief conceptual description of the underlying theory. First, in Stage 1 of Fig. 4, we ask the following question. For given granularity σ, how many basis functions can cover the input space adequately when we know that 100% coverage can be provided only by all the n inputs? To answer this, we introduce the term representational capability (δ) which is defined relative to the entire input space spanned by the data $\mathbf{x}$. We find that, with m much smaller than n, we can provide a very high level of representation. For example, in Section VI, we will see that $m=13$ provides 99% $(\delta=1\%)$ representation of the input space based on $n=152$.

The next question, posed in Stage 2 of Fig. 4 is, "Which m of the n input vectors should be used to provide the best design?" The answer is to choose those m vectors that are the farthest from other, leading to a model having the maximal representational capability with m basis functions. In addition, the centers selected in this way provide structural stabilization, another important property of a good model.

2. RC Algorithm

The specific steps leading to the three stage model development process of Fig. 4 are summarized in the following. The mathematical details can be found in [9].

RC Algorithm

Step 1: Select a range of values for width σ and a value for the RC measure δ. Heuristically we take $0 \le \sigma \le \sqrt{d/2}$ where d is the number of input variables. Also, δ is usually taken to be 0.1% to 1.0%.

Step 2: Determine a value of m that satisfies the δ criterion. This step involves singular value decomposition of the interpolation matrix computed from the $(n \times d)$ input data matrix.

Step 3: Determine the centers of the m basis functions such that these centers could maximize structural stability provided by the selected model complexity, m. This step involves the use of QR factorization.

Step 4: Compute weights using the pseudo inverse and estimate output values.

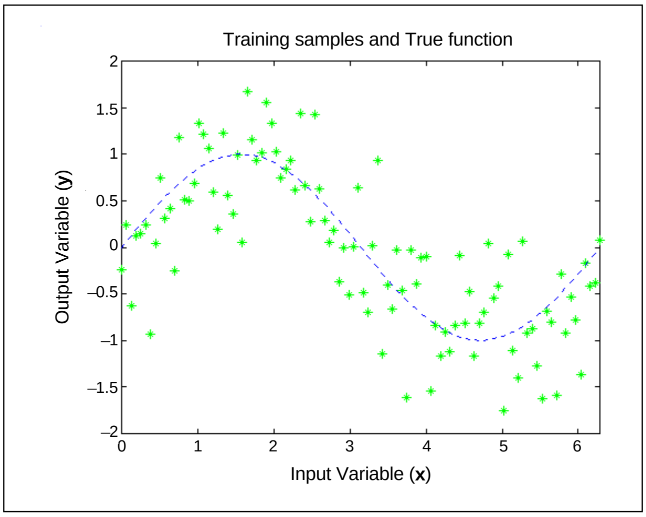

Fig. 5. Training data and the true function.

V. AN ILLUSTRATIVE EXAMPLE

In this section we illustrate the RC algorithm by constructing an RBF model from simulated data for a sine function with an error term (ε) added to it [10]:

$$y = \sin(x) + \varepsilon$$

We generate 100 data points from the above function by taking x in $[0, 2\pi]$ and ε to be a Gaussian distribution with mean zero and standard deviation 0.5. The x values are selected as the start points of 100 equal intervals in $[0, 2\pi]$ so that $x_i = 2\pi(\frac{i-1}{100}), i = 1, 2, \ldots, 100$. The 100 simulated data points $D = \{(x_i, y_i)\}_{i=1}^{100}$ along with the superimposed true curve are shown in Fig 5. Due to the high variance chosen here for the error term, the simulated y_i's exhibit considerable deviation from the true function $\sin(x)$.

For preprocessing, x_i and y_i are transformed to lie in the interval [0, 1]. We do so by subtracting the smallest x_i value from x_i's and the smallest y_i value from y_i's and then dividing the resulting values by the ranges of x and y, respectively.

1. Parameter Determination by RC Algorithm

Step 1: selection of σ and δ

Since $d = 2$, we take $0 \le \sigma \le \sqrt{1/2}$ and $0.1 \le \delta \le 1.0$.

Step 2: determination of m

To determine m, we first compute an interpolation matrix for each σ and perform its singular value decomposition (SVD). This yields a diagonal matrix of decreasing singular values $s_1 \ge s_2 \ge \cdots \ge s_n \ge 0$. From these, we determine m from the following:

$$m = \max_{1 \le i < n} \{i; s_{i+1} \le s_1 \times \frac{\delta}{100}\},$$

where (100-δ)% is the chosen RC criterion. Thus, we get one m value for each chosen σ, i.e., we get (σ, m) pairs.

For the example data, since the number of input data points is 100 and there is only one input variable x_i for each input, the 100×100 interpolation matrix G is constructed by substituting the input points x_i 's $(i = 1, \cdots, 100)$ in the following:

$$G = \begin{bmatrix} \exp(-\frac{\|x_1 - x_1\|^2}{2\sigma^2}) & \exp(-\frac{\|x_1 - x_2\|^2}{2\sigma^2}) & \cdots & \exp(-\frac{\|x_1 - x_{100}\|^2}{2\sigma^2}) \\ \exp(-\frac{\|x_2 - x_1\|^2}{2\sigma^2}) & \exp(-\frac{\|x_2 - x_2\|^2}{2\sigma^2}) & \cdots & \exp(-\frac{\|x_2 - x_{100}\|^2}{2\sigma^2}) \\ \vdots & \vdots & \vdots & \vdots \\ \exp(-\frac{\|x_{100} - x_1\|^2}{2\sigma^2}) & \exp(-\frac{\|x_{100} - x_2\|^2}{2\sigma^2}) & \cdots & \exp(-\frac{\|x_{100} - x_{100}\|^2}{2\sigma^2}) \end{bmatrix}. \qquad (7)$$

If we use $\sigma = 0.4$, the interpolation matrix becomes

$$G = \begin{bmatrix} 1.0000 & 0.9997 & \cdots & 0.0468 & 0.0439 \\ 0.9997 & 1.0000 & \cdots & 0.0498 & 0.0468 \\ 0.9987 & 0.9997 & \cdots & 0.0529 & 0.0498 \\ 0.9971 & 0.9987 & \cdots & 0.0563 & 0.0529 \\ 0.9949 & 0.9971 & \cdots & 0.0598 & 0.0563 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0.0498 & 0.0529 & \cdots & 0.9997 & 0.9987 \\ 0.0468 & 0.0498 & \cdots & 1.0000 & 0.9997 \\ 0.0439 & 0.0468 & \cdots & 0.9997 & 1.0000 \end{bmatrix}. \qquad (8)$$

Table 2. Values of (σ, m) pairs for δ=0.01%.

RBF	σ	m
I	0.2	9
II	0.3	7
III	0.4	5
IV	0.5	5
V	0.6	4
VI	0.7	4

For our design, suppose we use $\delta = 0.01\%$. Then, for each σ, we compute G as above and the SVD of G yields the matrix of singular values from which the value of m is obtained. For illustration, we choose six values of width, $\sigma = (0.2 : 0.1 : 0.7)$. The resulting m values for each σ are given in Table 2.

Step 3: determination of μ

As pointed out earlier, the QR decomposition with column pivoting is used to determine the m centers for each of the (σ, m) combinations obtained above [11]. The set of basis function centers determined by this method for the six (σ, m) pairs is shown in Table 3. From a visual inspection of these values, we note that in each case the selected basis function centers seem to provide a good coverage of the normalized input domain of x. In other words, the centers selected cover the input range [0, 1] illustrating one of the strong mathematical properties of our methodology.

The design matrix is next obtained by extracting from the interpolation matrix G the columns with centers in Table 3. Thus, for the case of $\sigma = 0.4$ and $m = 5$, the five columns that correspond to the five centers 0, 0.2020, 0.5051, 0.8081, and 1.0 in Table 3 would constitute the 100×5 design matrix. These columns are in the interpolation matrix of (8) but are not explicitly shown here.

Table 3. Listing of basis function centers.

RBF	(σ, m)		centers								
	σ	M	μ_1	μ_2	μ_3	μ_4	μ_5	μ_6	μ_7	μ_8	μ_9
I	0.2	9	0.0	1.0	0.9192	0.0808	0.2121	0.7879	0.6465	0.3535	0.5051
II	0.3	7	0.0	1.0	0.8990	0.1111	0.2929	0.7071	0.5051		
III	0.4	5	0.0	1.0	0.8081	0.2020	0.5051				
IV	0.5	5	0.0	1.0	0.8182	0.1919	0.5051				
V	0.6	4	0.0	1.0	0.7172	0.3030					
VI	0.7	4	0.0	1.0	0.7273	0.2929					

Table 4. Listing of weights for the six models.

RBF	$(\sigma,\ m)$		Weights								
	σ	m	w_1	w_2	w_3	w_4	w_5	w_6	w_7	w_8	w_9
I	0.2	9	2.75	11.68	–20.69	–5.48	6.92	18.34	–14.55	–7.44	10.61
II	0.3	7	22.73	28.74	–53.41	–47.96	52.30	52.99	–52.59		
III	0.4	5	–3.81	0.32	1.13	6.44	–3.61				
IV	0.5	5	–14.19	–1.97	8.55	24.52	–16.95				
V	0.6	4	–11.21	12.70	–23.49	22.67					
VI	0.7	4	–26.59	29.39	–55.29	53.08					

Step 4: computation of $\mathbf{w}$ and estimation of $\mathbf{y}$

Once the design matrix has been determined, we can compute the m weights of the connections to the output unit by the pseudo inverse method. Specifically, for the design matrix Φ, the weights are given as

$$\mathbf{w} = \Phi^{+}\mathbf{y}. \tag{9}$$

Here Φ^{+} denotes the pseudo inverse of Φ and $\mathbf{y}$ is the observed output vector. For our example, $\mathbf{w}$ is $m \times 1$, Φ is $100 \times m$ and $\mathbf{y}$ is 100×1. The m weights for each of the six models are given in Table 4.

At this point, each of the RBF model is completely defined in the sense that all the parameters $P = (m, \boldsymbol{\mu}, \sigma, \mathrm{w})$ have been determined. Next, we use these models to compute the corresponding output $\hat{y}$'s by using (6). The required values of σ, m and $\boldsymbol{\mu}$ are given in Table 3. Thus, the fitted RBF model for $m = 5$ and $\sigma = 0.4$ (model III) can be written as below. Here x_N and y_N are normalized values of x and y, respectively.

$$\begin{aligned}\hat{y}_N &= -3.81\exp\left(-\frac{\|x_N - 0.0\|^2}{2(0.4)^2}\right) + 0.32\exp\left(-\frac{\|x_N - 1.0\|^2}{2(0.4)^2}\right)\\ &+ 1.13\exp\left(-\frac{\|x_N - 0.8081\|^2}{2(0.4)^2}\right) + 6.44\exp\left(-\frac{\|x_N - 0.2020\|^2}{2(0.4)^2}\right)\\ &- 3.61\exp\left(-\frac{\|x_N - 0.5051\|^2}{2(0.4)^2}\right)\end{aligned} \tag{10}$$

The output $\hat{y}$ is obtained by first computing $\hat{y}_N$ from above and then unnormalizing each value. For model III, a plot of the outputs $\hat{y}$, obtained by unnormalizing $\hat{y}_N$ computed above, is shown as the solid line in Fig. 6. The simulated data and the true sine function are also shown in this figure.

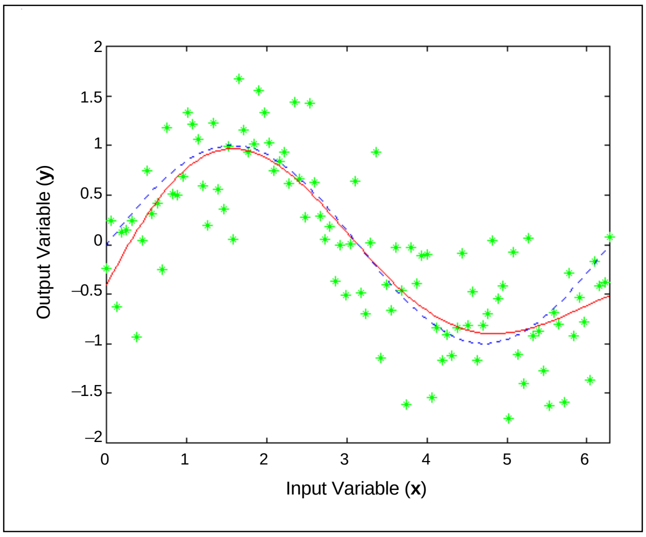

Fig. 6. Fitted RBF model III (m = 5, σ = 0.4).

2. Model Selection

A. Evaluate approximation ability

We use mean squared error based on the training data as a measure to evaluate the approximation ability of the fitted model. A large value of this error would indicate poor approximation ability and a small value would indicate that the fitted model provides a good approximation of the observed or training data.

In Table 5, we show training error values for the six RBF models. By looking at m and training error alone, we note that the error values decrease with increasing m. This is to be expected since a large number of basis functions would provide a better approximation than a smaller value. However, the purpose of training is not just to minimize training error but also to ensure good generalization ability, which we also consider next.

Table 5. Training and validation errors for candidate models.

RBF	(σ, m) combinations		MSE	
	σ	m	Training	Validation
I	0.2	9	.2189	.2973
II	0.3	7	.2228	.2947
III	0.4	5	.2347	.2776
IV	0.5	5	.2357	.2773
V	0.6	4	.2395	.2763
VI	0.7	4	.2399	.2772

B. Evaluate generalization ability

Recall that our goal is to build an RBF model of the process that generates the data and also provides good predictions for new inputs. Therefore, it is essential to evaluate the generalization ability of the trained models. For this purpose, we generated a validation data set of $D_v = \{(x_k^v, y_k^v)\}_{k=1}^{1000}$ based on the same sine function from which the training data set was generated. The generalization ability of the six RBF models developed above is assessed by the mean validation squared error:

$$E_{valid} = \frac{1}{1000}\sum_{k=1}^{1000}(y_k^v - \hat{y}_k^v)^2.$$

Here y_k^v 's are the simulated data in the validation set and $\hat{y}_k^v$'s are the RBF model estimates for $x_k^v, k = 1, 2, \ldots, 1000$, respectively. The computed E_{valid} values for the six models are shown in Table 5. This error decreases slightly from model VI to model V and then increases monotonically through models IV, III, II and I. Based on this pattern alone, it seems that the minimum validation error occurs for model V.

C. Select a final model

A common criterion for choosing a model is to consider training and validation errors associated with the models being considered. We show the plots of training and validation errors in the (σ, m) plane in Fig. 7. Note that models III, IV, V and VI are almost identical in their performance and any of them would be a good choice. If we consider model complexity as well as MSE values, however, model V would be selected over the others. Thus, based on our analyses and the chosen criteria, we select model V with $m = 4$ and $\sigma = 0.6$, as the trained RBF model for this data set.

Note that the model selected here represents a balance between the bias and variance properties, discussed in Section II-2. More specifically, the above model selection process tries to find a design that is close to the "best model" in the idealized depiction of Fig. 3.

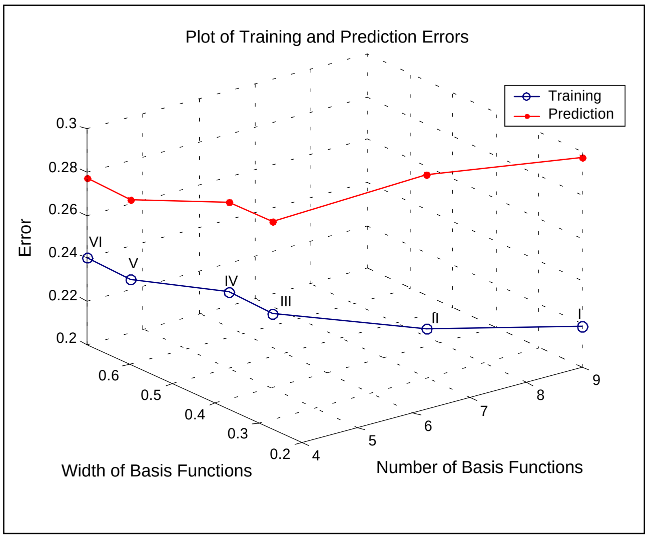

Fig.7. Training and validation errors for sine models.

VI. HEART DISEASE CLASSIFICATION

The problem of heart disease diagnosis is to predict the seriousness of the disease based on data such as age, blood pressure and chest pain. For this purpose, we develop RBF classifiers using public domain data about heart patients and their diagnosis results from the Cleveland Clinic Foundation [12], which we call data set CL. The objective here is to decide whether at least one of four major vessels is reduced in diameter by more than 50%. This binary classification (output is yes or no) is made from personal data which consists of 35 input values for each patient. These include age, sex, chest pain type, blood pressure, etc. Preprocessing is done to handle real, integer, ordinal and nominal attributes. Properly each variable is then normalized to lie in the range 0 to 1. The details of the attribute values and preprocessing conventions are given in [12]. The data set CL consists of 303 patients. We divide it into three subsets consisting of 152 patients for training, 76 for validation and 75 for test. Note that the test set is used to evaluate the performance of a classifier on future data while the validation error is used for model selection.

1. RBF Classifiers

Recall that our methodology uses the following sequence of modeling activities. For the chosen δ and σ, we use training data to determine the smallest m that satisfies δ criterion and the corresponding parameters $\boldsymbol{\mu}$ and $\mathbf{w}$. Based on these selected parameters, the training error is calculated for the RBF model. Then we compute the validation error for the validation data

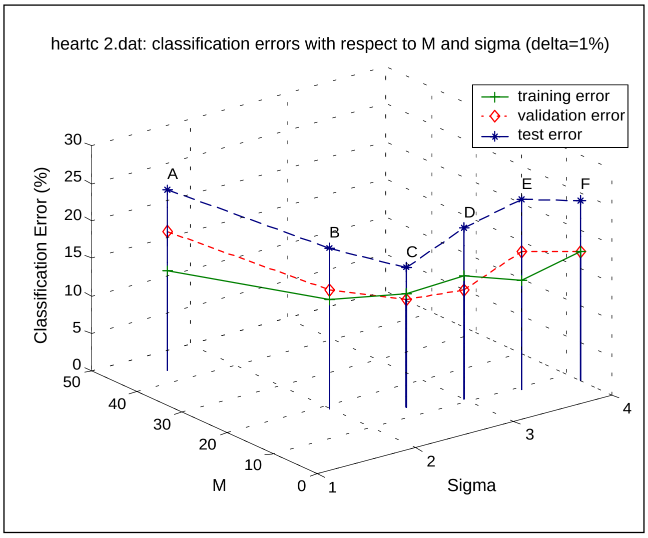

Fig. 8. Model selection process for heart disease classifier (σ = 1 %).

Table 6. Heart disease classification results for data set CL (δ = 1 %).

Model	σ	*m*	Training	Validation
A	1.5	44	13.2	18.4
B	2.0	19	14.5	15.8
C	2.5	13	15.1	14.5
D	3.0	11	16.5	14.5
E	3.5	9	14.5	18.4
F	4.0	7	17.1	17.1

and study the error behavior in order to select the best RBF classifier. The objective is to find a classifier with a good compromise between the bias and variance errors discussed in Section II. From a practical point of view, we expect this selection process depicted in Fig. 8 to yield a classifier as close as possible to the best model. This classifier is then employed to compute the test error for the test data. All errors are the percentage of misclassified patterns.

The analysis and modeling results for data set CL are summarized in Table 6 for $\delta = 1\%$ and six values of $\sigma = (1.5$: 0.5:4.0). The resulting six models (A to F) have a range of m from 7 to 44, training error varies from 13.1 to 17.1 and validation error has a range of 14.5 to 18.4. Note that the error rates here are not monotonic because both σ and m are changing simultaneously. The various training and validation errors as a joint function of σ and m are shown in Fig. 8 for models A to F. Using minimum validation error criterion for model selection, we find that model C $(m = 13, \sigma = 2.5)$ is the most desirable model in this case.

Table 7. Heart disease classification results for data set CL (δ = 0.5 %).

Model	σ	*m*	Training	Validation
A	1.5	65	9.2	18.4
B	2.0	32	12.5	17.1
C	2.5	18	13.8	11.8
D	3.0	13	15.8	15.8
E	3.5	13	14.5	15.8
F	4.0	12	14.5	15.8

Table 8. Heart disease classification results for data set CL (δ = 0.1 %).

Model	σ	*m*	Training	Validation
A	1.5	105	1.3	21.1
B	2.0	70	5.9	19.7
C	2.5	47	10.5	19.7
D	3.0	32	10.5	18.4
E	3.5	23	13.8	17.1
F	4.0	19	13.8	15.8

Table 9. Heart disease classification results for data set CL.

δ (%)	σ	*m*	Training	Validation	Test
1	2.5	13	15.1	14.5	18.7
0.5	3.5	13	14.5	15.8	18.7
0.1	4.0	19	13.8	15.8	19.7

Similar analyses were done for $\delta = 0.5\%$ and 0.1%. The corresponding tabular results are shown in Tables 7 and 8, respectively. For the three δ values, the selected models and σ, m and errors are given in Table 9. Based on the results, the most appropriate RBF classifier would be $\sigma = 2.5, m = 13$, for $\delta = 1\%$, with validation classification error of 14.5 %.

The test errors for the models for each δ are also shown in Table 9 which happens to be the same for each model. To gain further insight into the behavior of the three error terms (training, validation, test), we have also shown the test errors in Tables 7, 8 and 9 for $\delta = 1\%$, 0.5 % and 0.1 %, respectively. These patterns of errors as a function of (σ, m) provide a deeper understanding of the RBF classifier development and selection, especially when compared with the theoretical and ideal depictions of Section II-2 and III-2 (Figs. 1 and 3).

VII. CONCLUSIONS

In this paper we presented a new approach for pattern recognition problems using the radial basis function model. Our objective was to derive a pattern classifier with desirable bias-variance properties systematically and efficiently. Towards this goal, we formulated the problem as a three-stage process. In the first stage the number of basis functions, for a given width, was determined using the representational capability criterion. The centers were selected from the input vectors in the second stage such that they provide structural stabilization, another property of a good classifier. The weights were obtained in the third stage by a pseudo inverse.

The use of the new methodology and the RC algorithm were illustrated via a detailed example from simulated data. Also, RBF classifiers were developed for heart disease prediction from well known data sets from the Cleveland Clinic Foundation.

ACKNOWLEDGEMENT

We want to express our thanks to the anonymous reviewers and the Editor of ETRI Journal for many valuable suggestions that have improved the paper.

REFERENCES

[1] C. M. Bishop, *Neural Networks for Pattern Recognition*. Oxford University Press, 1995.

[2] B. D. Ripley, *Pattern Recognition and Neural Networks*. Cambridge University Press, 1996.

[3] F. Girosi, and T. Poggio, "Networks and the Best Approximation Property," *Biological Cybernetics*, Vol. 63, 1990, pp. 169–176.

[4] M. J. D. Powell, "The Theory of Radial Basis Function Approximation in 1990," *Advances in Numerical Analysis, Wavelets, Subdivision Algorithms and Radial Basis Functions*, edited by W. A. Light, Oxford University Press, Vol. 2, 1992, pp. 105–210.

[5] J. H. Friedman, "On Bias, Variance, 0/1-loss and the Curse-of-dimensionality," *Data Mining and Knowledge Discovery*, Vol. 1, 1997, pp. 55–77.

[6] S. Geman, E. Bienenstock, and R. Doursat, "Neural Networks and the Bias/Variance Dilemma," *Neural Computation*, Vol. 4, 1992, pp. 1–58

[7] J. Moody, and C. J. Darken, "Fast Learning in Networks of Locally-tuned Processing Units," *Neural Computation*, Vol. 1, 1989, pp. 281–294.

[8] S. Chen, C. F. N. Cowan, and P. M. Grant, "Orthogonal Least Squares Learning Algorithm for Radial Basis Function Networks," *IEEE Transactions on Neural Networks,* Vol. 2, No. 2, March 1991, pp. 302–309.

[9] M. Shin, *Design and Evaluation of Radial Basis Function Model for Function Approximation*, Ph.D Dissertation, Syracuse University, 1998.

[10] D.Bertsimas, D. Gamarnik, and J. N. Tsitsiklis, "Estimation of Time-Varying Parameters in Statistical Models; an Optimization Approach," In *COLT 97* Nashville, Tennessee, USA, 1997, pp. 314–324.

[11] G. H. Golub, and C. F. Van Loan, *Matrix Computations*. The Johns Hopkins University Press, The Third Edition, 1996.

[12] L. Prechelt, "PROBEN1-A Set of Neural Network Benchmark Problems and Benchmarking Rules," Technical Reports 21/94, Universitat Karlsruhe, Germany, September 1994.

[13] M. Shin, and A. L. Goel, "An RBF Classifier Based Framework for Software Quality Evaluation," *Proceedings of the International Conference on Computational Intelligence for Modeling, Control and Automation*, Vienna, Austria, February 1999.

[14] M. Shin, and A. L. Goel, "Knowledge Discovery and Validation in Software Metrics Databases," *Proceedings of Data Mining and Knowledge Discovery: Theory, Tools and Technology*, Orlando, Florida, April 1999.

[15] V. N. Vapnik, *Statistical Learning Theory,* John Wiley, 1998.

Miyoung Shin received the B.S. and M.S. degrees in computer science from Yonsei University in Seoul, Korea, in 1991 and 1993, respectively, and the Ph.D degree in computer and information science from Syracuse University, New York, USA, in May 1998. For her outstanding Ph.D. work, she awarded the All-University Doctoral Prize from Syracuse University. She is currently working as a senior member of technical staff at the Electronics and Telecommunication Research Institute in Korea. Her research interests include data mining algorithms, pattern recognition, radial basis function modeling and evaluation for a variety of problems in medical diagnosis, and software engineering.

Cheehang Park received the B.S. degree in applied physics from Seoul National University, Korea in 1974, the M.S. degree in computer science from Korea Advance institute of Science and Technology, Korea in 1980, and the Ph.D. degree in computer science from University of Paris 6, France in 1987. During the last 10 years, he has been involved as project leader in several large projects such as Multimedia Computer System Development and High Speed Parallel Computer System Development. His research interests include multimedia systems, distributed system, middleware, groupware, network virtual computing, and mobile agent architecture. He is currently Executive Director of Information Support Division, Electronics and Telecommunications Research Institute.