Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2014, Article ID 913897, 14 pages
http://dx.doi.org/10.1155/2014/913897

Research Article

Adaptive Linear and Normalized Combination of Radial Basis Function Networks for Function Approximation and Regression

Yunfeng Wu,[1] Xin Luo,[1] Fang Zheng,[1] Shanshan Yang,[1] Suxian Cai,[1] and Sin Chun Ng[2]

[1] *School of Information Science and Technology, Xiamen University, 422 Si Ming South Road, Xiamen, Fujian 361005, China*
[2] *School of Science and Technology, The Open University of Hong Kong, 30 Good Shepherd Street, Ho Man Tin, Kowloon, Hong Kong*

Correspondence should be addressed to Yunfeng Wu; y.wu@ieee.org

Received 2 November 2013; Revised 28 February 2014; Accepted 28 February 2014; Published 30 March 2014

Academic Editor: Wei Bian

This paper presents a novel adaptive linear and normalized combination (ALNC) method that can be used to combine the component radial basis function networks (RBFNs) to implement better function approximation and regression tasks. The optimization of the fusion weights is obtained by solving a constrained quadratic programming problem. According to the instantaneous errors generated by the component RBFNs, the ALNC is able to perform the selective ensemble of multiple leaners by adaptively adjusting the fusion weights from one instance to another. The results of the experiments on eight synthetic function approximation and six benchmark regression data sets show that the ALNC method can effectively help the ensemble system achieve a higher accuracy (measured in terms of mean-squared error) and the better fidelity (characterized by normalized correlation coefficient) of approximation, in relation to the popular simple average, weighted average, and the Bagging methods.

1. Introduction

Function approximation has been used in a variety of disciplines such as data mining, system identification, and forecasting [1]. Given a finite data set, the essential task of a function approximation problem is to interpret the appropriate relationship between multidimensional explanatory variables and the corresponding responses. Function approximation problems can be categorized into two major types [2]. First, for known target functions, the approximation theory investigates how to estimate the parameters of certain functions or how to closely match a target function via a particular class of rational functions with some desirable properties (inexpensive computation, continuity, integral or differential, limit values, etc.) [3]. Second, if the specific expression of the target function is unknown, instead, only a series of observations in the form of input-response pairs are available. To perform such an approximation, several numerical analysis techniques, for example, interpolation, extrapolation, regression analysis, and curve fitting, could be considered.

For two decades, artificial neural networks with the inner neurons activated by radial basis functions have been extensively applied in numerous practical applications [4–6]. The radial basis function network (RBFN) works by performing a nonlinear transformation from the inputs to a high-dimensional hidden space and produces the response through a linear output layer [7]. It has been justified that any continuous function on a compact interval can be interpolated toward an arbitrary accuracy by a well-devised RBFN with a sufficiently large number of hidden neurons [8, 9]. However, there still lacks a rigorous theoretical framework that specifies the routine to determine an optimal RBFN structure with the stated approximation properties. Furthermore, when dealing with high-dimensional training data, the RBFN approximation is subject to the risks of overfitting and "curse of dimensionality" [10].

Recently, ensemble methods have been recommended by the machine learning community [11–16]. Based on the principle of "divide and conquer" [10], an ensemble system combines a finite number of component neural networks (CNNs) to provide a consensus decision, with the aim to achieve some favorable performance (lower generalization error or higher accuracy superior to any single learning machine acting solely) [17]. The schemes of ensemble systems commonly follow the generative and nongenerative styles

[18]. The generative ensembles employ the resampling or filtering techniques to boost the training data with different underlying distributions. On the other hand, the nongenerative ensembles combine the CNNs trained from the same data set by using appropriate decision fusion strategies [15, 19] or combination rules [12, 14, 20].

The pioneering generative ensemble algorithms are Boosting [21] and Bagging (the acronym for "bootstrap aggregating") [22]. Boosting family algorithms repeatedly train a particular weak-learning machine with different distributed training data sets and then combine the local decisions. Freund and Schapire [21] proposed the AdaBoost algorithm in order to find a typical mapping function or hypothesis with a low error rate in relation to a given probability distribution of the training data. In spite of the effectiveness, the Boosting algorithms still have some drawbacks to implement regression tasks. First, the regression data have to be divided into many classification sets such that the number of classification instances becomes intensively larger in the boosted iterations. Second, the cost function has to be modified from one iteration to another, in order to adapt the boosted data sets. Therefore, the Boosting algorithms are very sensitive to noisy data and outliers [23]. The Bagging algorithm, on the other hand, introduces the bootstrap resampling procedure [24] into the neural network aggregation, with the purpose to increase the diversity [22]. By averaging the CNNs, the variance of Bagging ensemble would become much smaller than any of the CNNs. However, according to remarks of Breiman's work [22], Bagging stable component learners in an ensemble system can only slightly improve accuracy but would lead to greater computation expense.

From the last decade, nongenerative ensemble methods have also received broad attentions [20, 25–32]. Linear combinations are most frequently used in real world applications, in virtue of the simplicity and modest computation expense. Opitz and Maclin [33] suggested using the simple average (SA) combination rule for regression and the majority vote (MV) rule for classification. Ueda [20] used the optimal linear weights to combine neural network classifiers to improve classification performance. Fumera and Roli [25] provided the theoretical analysis of linear combinations and proposed a weighted average (WA) rule for multiple classifier systems. However, the effectiveness of aforementioned linear combination methods may be more or less affected by their inherent flaws in design [32]. For example, the SA can only work well when the component learners are with similar error rates, because it treats all the component learners equally. The linear weights derived by the WA method are based on the assumption that the component learners produce independent and identically distributed errors [34]. The theoretical superiority of the WA method is not yet guaranteed in practice, because the weights estimation may become rapidly skewed with small-size or noisy data sets [25]. Tresp and Taniguchi [35] suggested using the inverse of the variance depending on the input variables to generate the nonconstant weights in the linear combination. The variance-based weighting method is able to dynamically adjust the fusion weights with respect to different distributions of input variables. However, Ueda [20] pointed out that such a method has some serious pitfalls in the minimization of classification errors. On the other hand, the effectiveness of linear combinations may also be affected by some CNNs with poor performance. Zhou et al. [36] suggested that it may be better to select some CNNs with reliable performance over the training data to constitute an ensemble system. It is true that the CNNs with poor performance may mislead the ensemble system toward a higher accuracy, but it does not imply that these CNNs are useless at all. For example, a CNN may provide an excellent generalization for some parts of the data set but failed for the other parts. Such a phenomenon usually occurs due to overfitting; that is, the neural network is overtrained with poor generalization. Now there arises a question: can we retain a CNN in the ensemble system only when it produces good approximation for some parts of a function domain and discard it if it fails to attain the desired generalization accuracy? In the present study, we propose an adaptive linear and normalized combination (ALNC) method that can effectively combine the CNNs with the dynamic and normalized fusion weights, which can be obtained by solving a constrained quadratic programming problem.

2. Adaptive Linear and Normalized Combination (ALNC)

Suppose that an ALNC ensemble system contains K component RBFN approximators in total (see Figure 1). The kth component RBFN produces the approximation output, $f_k(\mathbf{x}^n)$, $k = 1, \ldots, K$, in response to the nth input instance vector $\mathbf{x}^n$, $n = 1, \ldots, N$. The ALNC system provides the ensemble approximation, $f_{\mathrm{ALNC}}(\mathbf{x}^n)$, by linearly combining the component RBFNs with the adaptive (instance-varying) normalized weights $w_k(\mathbf{x}^n)$, which can be formulated as

$$f_{\mathrm{ALNC}}(\mathbf{x}^n) = \sum_{k=1}^{K} w_k(\mathbf{x}^n) f_k(\mathbf{x}^n). \tag{1}$$

The fusion weights are subject to the nonnegativity and normalization constraints [32, 37–40], which can be written as

$$\sum_{k=1}^{K} w_k(\mathbf{x}^n) = 1, \quad w_k(\mathbf{x}^n) \geq 0. \tag{2}$$

Similar to most of the linear combination methods reported in the literature, the aim of the ANLC ensemble system is to provide the optimal solution of the nonnegative and normalized weights. The difference between the approximation of the kth component RBFN, $f_k(\mathbf{x}^n)$, and the target function response, $g(\mathbf{x}^n)$, is measured with the squared error as

$$e_k^2(\mathbf{x}^n) = [g(\mathbf{x}^n) - f_k(\mathbf{x}^n)]^2. \tag{3}$$

Then, the squared error of the ALNC ensemble approximation, $e_{\mathrm{ALNC}}(\mathbf{x}^n)$, is estimated in a similar way.

Since the nonnegative weights are normalized in the ALNC ensemble, the target function response $g(\mathbf{x}^n)$ can be

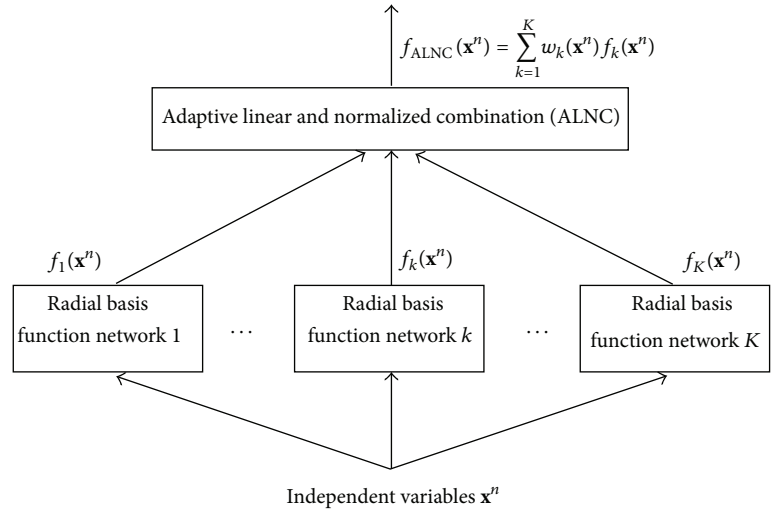

FIGURE 1: Illustration of adaptive linear and normalized combination (ALNC) of radial basis function networks for function approximation.

split and combined with the weighting multipliers, $w_k(\mathbf{x}^n)$; that is, $g(\mathbf{x}^n) = \sum_{k=1}^{K} w_k(\mathbf{x}^n) g(\mathbf{x}^n)$. Thus, the approximation error term of the ALNC ensemble system, $e_{\mathrm{ALNC}}(\mathbf{x}^n)$, is derived as follows:

$$\begin{aligned} e_{\mathrm{ALNC}}^2(\mathbf{x}^n) &= \left[g(\mathbf{x}^n) - \sum_{k=1}^{K} w_k(\mathbf{x}^n) f_k(\mathbf{x}^n) \right]^2 \\ &= \left[\sum_{k=1}^{K} w_k(\mathbf{x}^n) g(\mathbf{x}^n) - \sum_{k=1}^{K} w_k(\mathbf{x}^n) f_k(\mathbf{x}^n) \right]^2 \\ &= \left\{ \sum_{k=1}^{K} w_k(\mathbf{x}^n) \left[g(\mathbf{x}^n) - f_k(\mathbf{x}^n) \right] \right\} \\ &\quad \times \left\{ \sum_{j=1}^{K} w_j(\mathbf{x}^n) \left[g(\mathbf{x}^n) - f_j(\mathbf{x}^n) \right] \right\} \\ &= \sum_{k=1}^{K} \sum_{j=1}^{K} w_k(\mathbf{x}^n) w_j(\mathbf{x}^n) \\ &\quad \times \left[g(\mathbf{x}^n) - f_k(\mathbf{x}^n) \right] \left[g(\mathbf{x}^n) - f_j(\mathbf{x}^n) \right] \\ &= \sum_{k=1}^{K} \sum_{j=1}^{K} w_k(\mathbf{x}^n) w_j(\mathbf{x}^n) e_k(\mathbf{x}^n) e_j(\mathbf{x}^n). \end{aligned} \tag{4}$$

By virtue of (2) and (4), the minimization of the ALNC ensemble error on the nth input data is equivalent to the constrained quadratic programming (CQP) problem specified as follows:

$$\begin{aligned} &\text{minimize} \quad e_{\mathrm{ALNC}}^2(\mathbf{x}^n) = \sum_{k=1}^{K} \sum_{j=1}^{K} w_k(\mathbf{x}^n) w_j(\mathbf{x}^n) e_k(\mathbf{x}^n) e_j(\mathbf{x}^n), \\ &\text{subject to} \quad \sum_{k=1}^{K} w_k(\mathbf{x}^n) = 1, \quad w_k(\mathbf{x}^n) \geq 0. \end{aligned} \tag{5}$$

We may use the Lagrange multiplier method [41] to solve this CQP problem by defining the cost function as

$$\begin{aligned} &C\left(w_1(\mathbf{x}^n), \ldots, w_K(\mathbf{x}^n), \lambda(\mathbf{x}^n)\right) \\ &\quad = \sum_{k=1}^{K} \sum_{j=1}^{K} w_k(\mathbf{x}^n) w_j(\mathbf{x}^n) e_k(\mathbf{x}^n) e_j(\mathbf{x}^n) \\ &\qquad - \lambda(\mathbf{x}^n) \left[\sum_{k=1}^{K} w_k(\mathbf{x}^n) - 1 \right], \end{aligned} \tag{6}$$

where the nonnegative coefficient $\lambda(\mathbf{x}^n)$ denotes the Lagrange multiplier, the value of which varies from one input data vector to another.

According to the weak Lagrangian principle [41], the optimum solution, $\{\mathbf{w}^*(\mathbf{x}^n), \lambda^*(\mathbf{x}^n)\}$, is the stationary point of the cost function given in (6) and satisfies the following unique equations [40]:

$$\begin{aligned} &\frac{\partial C\left(w_1(\mathbf{x}^n), \ldots, w_K(\mathbf{x}^n), \lambda(\mathbf{x}^n)\right)}{\partial w_k(\mathbf{x}^n)} \\ &\quad = 2 \sum_{j=1}^{K} w_j(\mathbf{x}^n) e_k(\mathbf{x}^n) e_j(\mathbf{x}^n) - \lambda(\mathbf{x}^n) = 0, \\ &\frac{\partial C\left(w_1(\mathbf{x}^n), \ldots, w_K(\mathbf{x}^n), \lambda(\mathbf{x}^n)\right)}{\partial \lambda(\mathbf{x}^n)} \\ &\quad = \sum_{k=1}^{K} w_k(\mathbf{x}^n) - 1 = 0. \end{aligned} \tag{7}$$

Then, the optimal solution of the ALNC weights, $w_k^*(\mathbf{x}^n)$, can be derived by solving the CQP problem as

$$w_k^*(\mathbf{x}^n) = \frac{\sum_{j=1}^{K} e_k^{-1}(\mathbf{x}^n) e_j^{-1}(\mathbf{x}^n)}{\sum_{i=1}^{K} \sum_{j=1}^{K} e_i^{-1}(\mathbf{x}^n) e_j^{-1}(\mathbf{x}^n)}, \quad k = 1, \ldots, K. \tag{8}$$

Note that the optimal fusion weights are determined by the errors of component RBFNs, when the RBFN parameters are specified and the target function response is given.

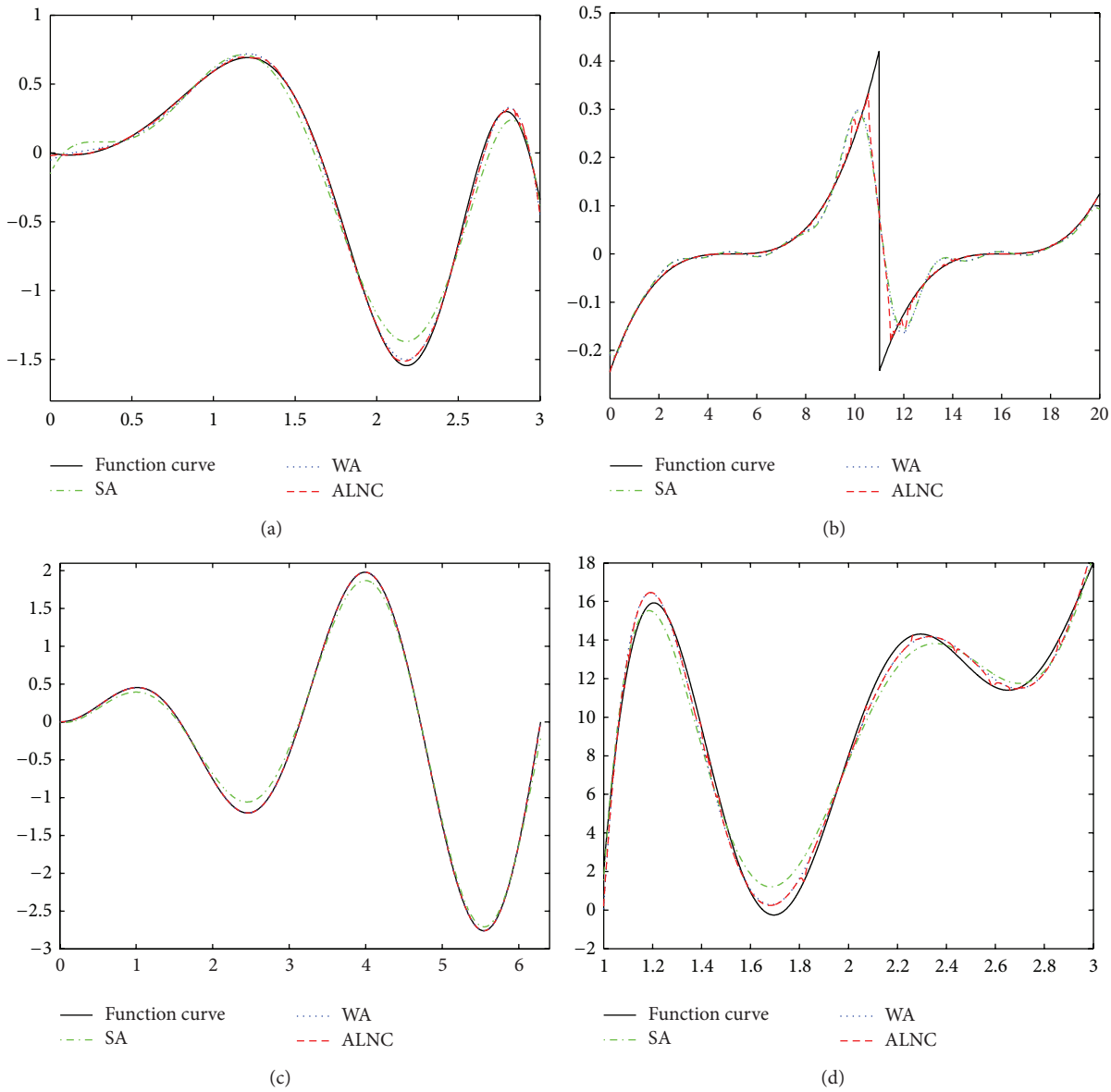

FIGURE 2: Plots of the approximations for the two-dimensional functions: (a) Zigzag, (b) Rhythm, (c) SinCos, and (d) ExpSin. Target function curve (solid line), simple average (SA) approximation (dash-dot line), weighted average (WA) approximation (dotted line), and adaptive linear and normalized combination (ALNC) approximation (dashed line).

Concerning the error term of the ALNC ensemble system, substituting (8) into (4) yields

$$e^2_{\mathrm{ALNC}}(\mathbf{x}^n) = \sum_{k=1}^{K}\sum_{j=1}^{K} w_k(\mathbf{x}^n)\, w_j(\mathbf{x}^n)\, e_k(\mathbf{x}^n)\, e_j(\mathbf{x}^n) = \frac{1}{\sum_{i=1}^{K}\sum_{j=1}^{K} e_i^{-1}(\mathbf{x}^n)\, e_j^{-1}(\mathbf{x}^n)}. \tag{9}$$

It is clear that $e_k^2(\mathbf{x}^n)$ and $e^2_{\mathrm{ALNC}}(\mathbf{f}^n)$ are both nonnegative; that is, $e_k^2(\mathbf{x}^n) \geq 0$ and $\sum_{i=1}^{K}\sum_{j=1}^{K} e_i^{-1}(\mathbf{x}^n) e_j^{-1}(\mathbf{x}^n) \geq 0$. To compare the ALNC ensemble error with a component RBFN error, we may compute the division operator as

$$\frac{e_k^2(\mathbf{x}^n)}{e^2_{\mathrm{ALNC}}(\mathbf{x}^n)} = 1 + \sum_{\substack{i=1\\ i\neq k}}^{K}\sum_{j\neq k}^{K} \frac{e_k^2(\mathbf{x}^n)}{e_i(\mathbf{x}^n)\, e_j(\mathbf{x}^n)} \geq 1. \tag{10}$$

According to (10), it can be inferred that the ALNC ensemble system, with the optimal fusion weights, is more likely to outperform any of its component RBFNs.

3. Experiments

3.1. Data Description. The data sets tested in our experiments are twofold. The first eight data sets are synthetic

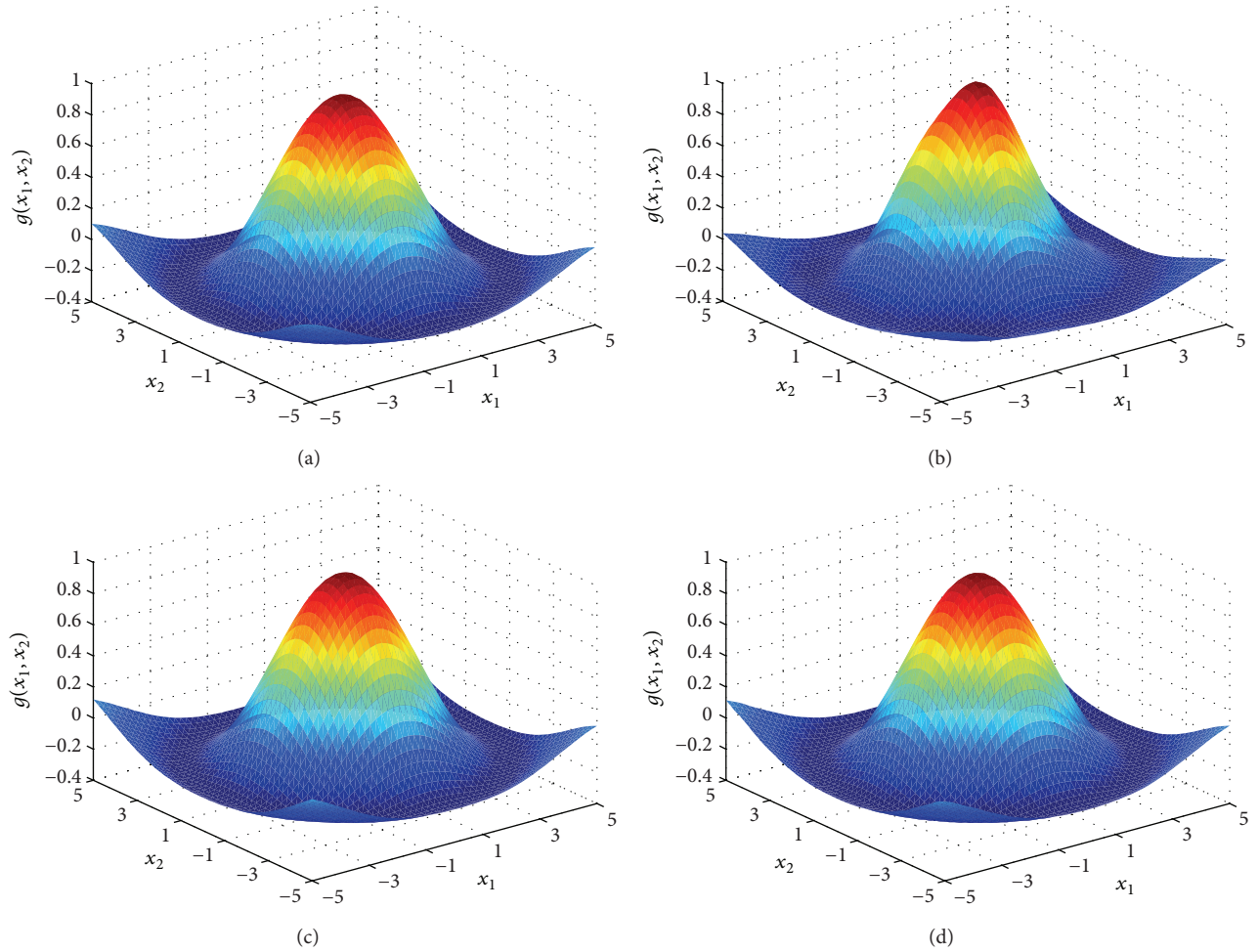

Figure 3: Plots of the approximations for 3-D Mexican Hat: (a) target function, (b) simple average (SA) approximation, (c) weighted average (WA) approximation, and (d) adaptive linear and normalized combination (ALNC) approximation.

functions without noise, among which the Zigzag, Rhythm, SinCos, and ExpSin are two-dimensional functions and the 3-D Mexican Hat, Gabor, SwingCos, and Exponential are multivariate functions with two independent variables. The details of particular function expressions, domains, and the size of samples, with regard to these eight synthetic sets, are specified in Table 1, in which $U[a, b]$ indicates a uniform distribution over the interval from a to b.

The other six benchmark multivariate regression sets listed in Table 2 were obtained from the University of California at Irvine (UCI) machine learning repository [42] and Carnegie Mellon University (CMU) StatLib library (available online at http://lib.stat.cmu.edu/datasets/), respectively.

Abalone. The task is to predict the age of abalone from seven physical measurements (the nominal attribute "sex" in the original UCI data set was not included in the set of input attributes in our experiments).

Housing. The Housing data set, which contains 2 integers and 11 continuous attributes, describes the housing values in suburbs of Boston.

Auto-MPG. This data set concerns city-cycle fuel consumption in miles per gallon. From the original data set, the string attribute "car name" (unique for each instance) and six instances with missing values were removed in our experiments.

Stock. The data set describes daily stock prices of 10 aerospace companies from January 1988 through October 1991. The task is to predict the stock price of the first company from the other nine.

Bolt. This is a relatively small data set retrieved from a trial on the effects of machine adjustments on the time to count bolts (a type of automotive accessory). The task is to predict

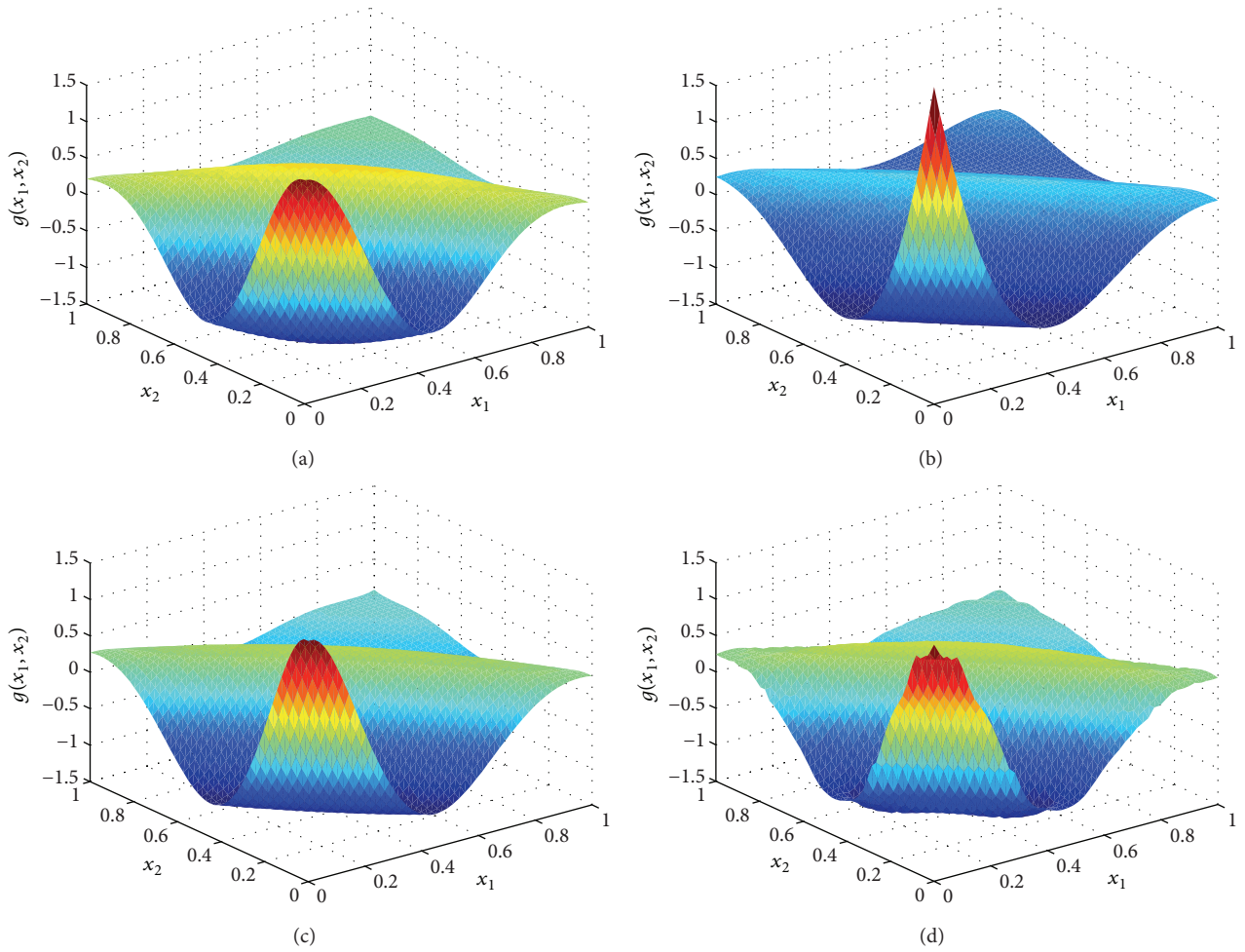

FIGURE 4: Plots of the approximations for Gabor: (a) target function, (b) simple average (SA) approximation, (c) weighted average (WA) approximation, and (d) adaptive linear and normalized combination (ALNC) approximation.

TABLE 1: Description of the synthetic data sets for function approximation.

Function name	Target function expression	Distribution of independent variables	Size of samples
Zigzag	$g(x) = \sin x^2 - 0.25x$	$x \sim U[0, 3]$	3000
Rhythm	$g(x) = \left[\frac{\text{mod}(x, 11) - 5}{8}\right]^3$	$x \sim U[0, 20]$	1000
SinCos	$g(x) = x \sin x \cos x$	$x \sim U[0, 2\pi]$	2000
ExpSin	$g(x) = 2x^2 + \exp\left[\frac{\pi}{x}\right] \sin(2\pi x)$	$x \sim U[1, 3]$	4000
3-D Mexican Hat	$g(x_1, x_2) = \frac{\sin\sqrt{x_1^2 + x_2^2}}{\sqrt{x_1^2 + x_2^2}}$	$x_1 \sim U[-5, 5]$ $x_2 \sim U[-5, 5]$	2500
Gabor	$g(x_1, x_2) = \frac{\pi}{2} \exp\left[-2(x_1^2 + x_2^2)\right] \cos 2\pi(x_1 + x_2)$	$x_1 \sim U[0, 1]$ $x_2 \sim U[0, 1]$	2500
SwingCos	$g(x_1, x_2) = x_1 + x_2 + \cos 2\pi x_1 + \cos 2\pi x_2$	$x_1 \sim U[-1, 1]$ $x_2 \sim U[-1, 1]$	2500
Exponential	$g(x_1, x_2) = x_1 \exp\left[-(x_1^2 + x_2^2)\right]$	$x_1 \sim U[-2, 2]$ $x_2 \sim U[-2, 2]$	2500

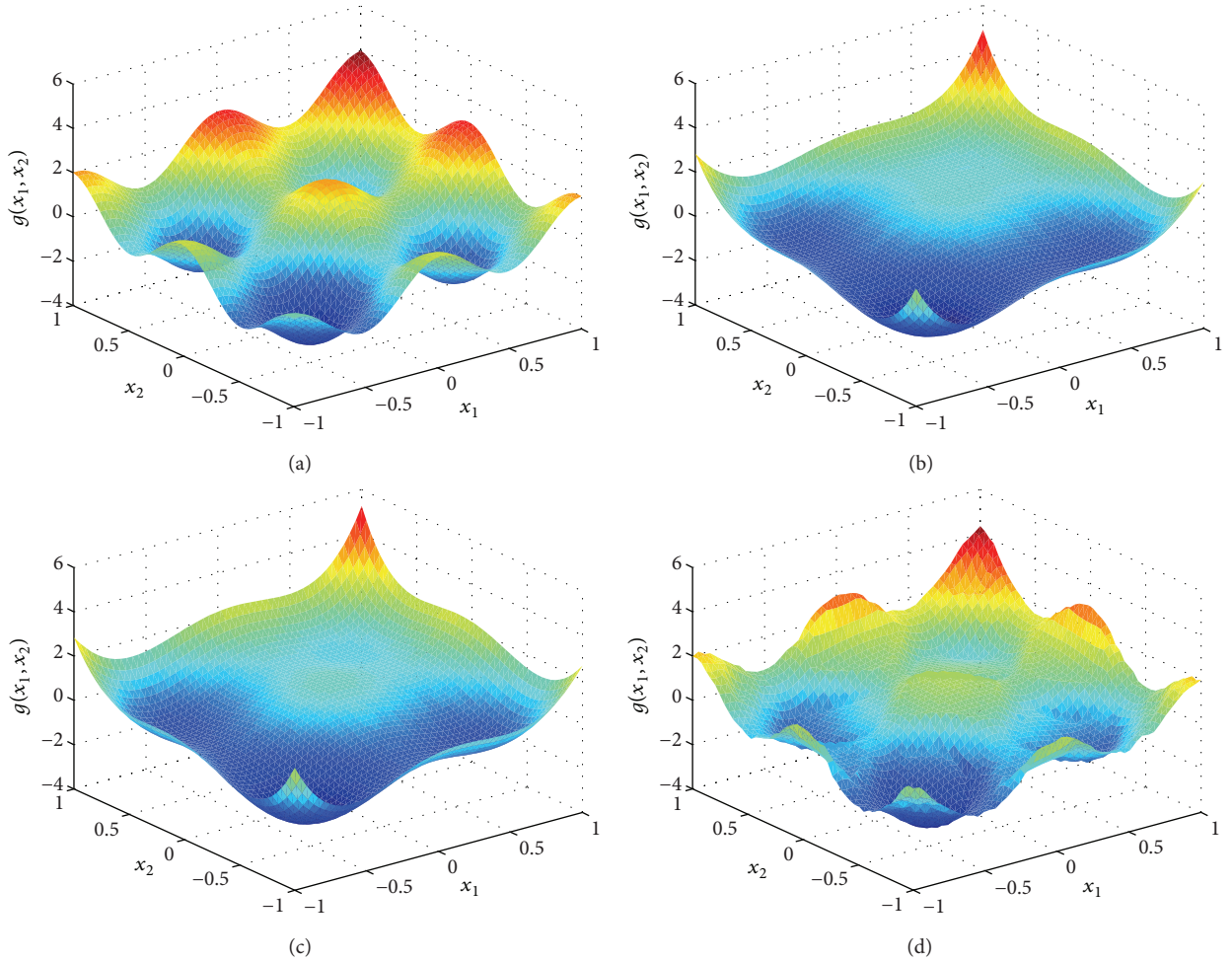

Figure 5: Plots of the approximations for SwingCos: (a) target function, (b) simple average (SA) approximation, (c) weighted average (WA) approximation, and (d) adaptive linear and normalized combination (ALNC) approximation.

Table 2: Description of the regression data sets.

Data set name	Number of attributes	Size of instances	Data source
Abalone	7	4177	UCI
Housing	13	506	UCI
Auto-MPG	7	392	UCI
Stock	9	950	StatLib
Bolt	7	40	StatLib
CPS-85-Wages	10	534	StatLib

the correct time to count 20 bolts from 7 attributes associated with the machine adjustment settings.

CPS-85-Wages. The CPS-85-Wages data were obtained from the Current Population Survey (CPS) of 534 people. Such a survey provides the information on wages and other aspects of the workers, including years of education, region of residence, gender, years of work experience, union membership, age, race, occupational status, sector, and marital status.

3.2. Settings of Component Radial Basis Function Networks. The details of the component RBFNs involved in the ALNC ensemble system are presented as follows. The number of the sensory neurons in the input layer is equal to the dimensions of independent variables. The radial basis function kernel function is defined as

$$\varphi(\mathbf{x}, \mathbf{c}_l) = \exp\left(-\log_e \frac{2\|\mathbf{x} - \mathbf{c}_l\|^2}{\sigma^2}\right), \tag{11}$$

where $\mathbf{c}_l$ denotes the center vector for the lth hidden neuron and σ is the spread parameter that determines the width of the area in the input space to which each hidden neuron responds. The output layer is linear, and the responses of the component RBFNs are sent to the succeeding linear combination. We employed a total of 30 component RBFNs to approximate the synthetic functions. The first 10 component RBFNs had the same spread parameter of 1.0, and the number of hidden neurons increased from 1 to 10, for each neural network. Regarding the second 10 component RBFNs (with the spread of 2.0) and the remaining 10 component RBNFs (with the spread of 3.0), their hidden neuron numbers were

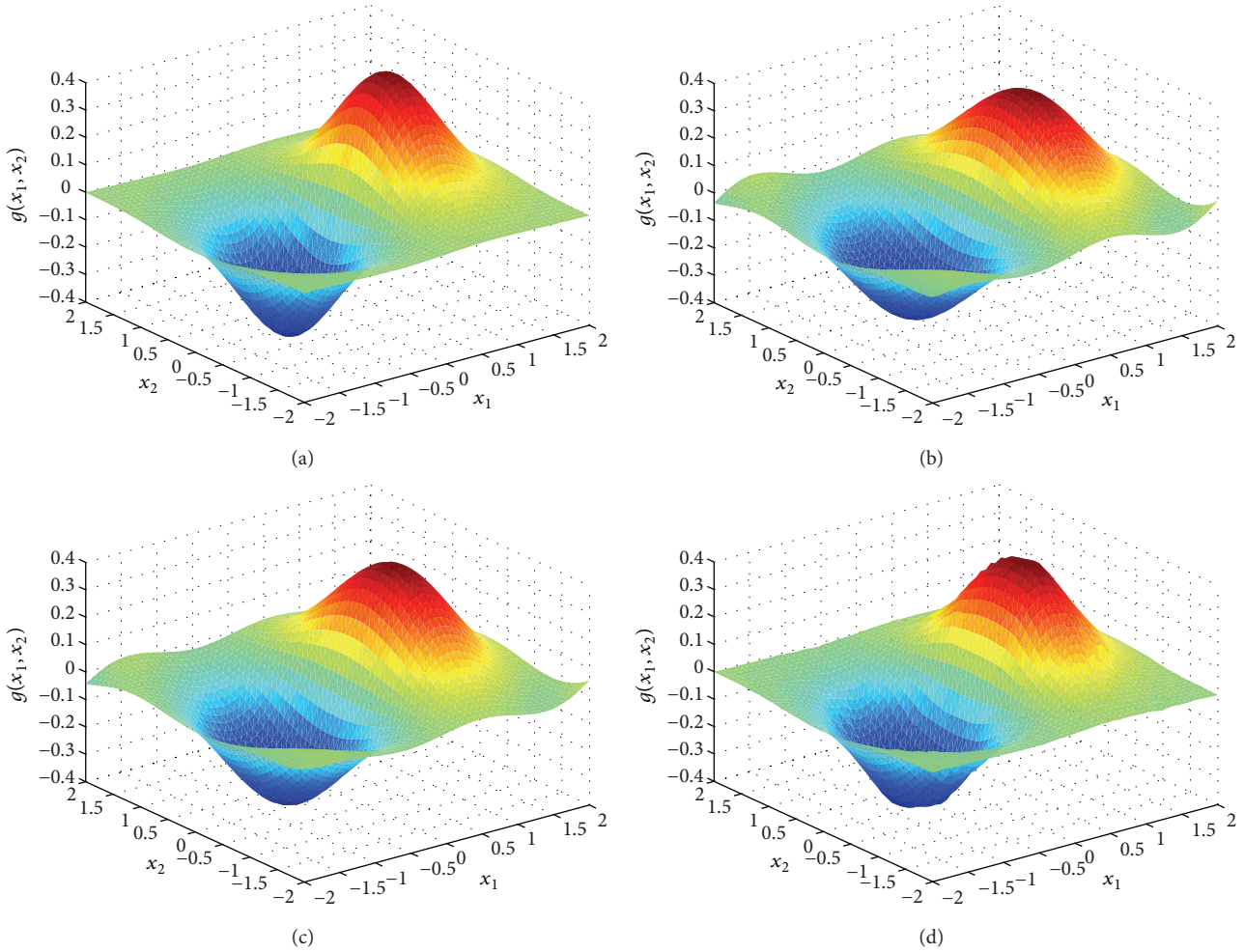

FIGURE 6: Plots of the approximations for Exponential: (a) target function, (b) simple average (SA) approximation, (c) weighted average (WA) approximation, and (d) adaptive linear and normalized combination (ALNC) approximation.

incremental from 11 to 20 and from 21 to 30, respectively. In the regression experiments, the ensemble system combined three component RBNFs. The number of hidden neurons was equal to the dimensions of the input independent variables, and the spread parameter varied from 1.0 to 3.0 for each network. Each component RBFN was trained with the orthogonal least-squares algorithm [43], which offers a systematic method for center selection and it significantly reduces the complexity of the RBFN.

3.3. Other Experiment Settings. For the purpose of approximation performance comparison, we also implemented the popular SA and WA combination rules on the synthetic data sets and the Bagging algorithm on the benchmark regression data sets. In the regression experiments, the ALNC method used the same bootstrap resampling procedure as the Bagging for fair comparison purpose. The ALNC ensemble with such a data preprocessing procedure is presented as Bootstrap-ALNC hereafter.

The computer programs were performed on a laptop with a CPU processor of 1.86 GHz speed and 1.5 GB RAM memory. Each experiment was repeatedly carried out for 50 times to provide the results in statistical sense. In order to compare the computational efficiency, we also recorded the computation time consumption (CTC) in milliseconds (ms) of each fusion method.

3.4. Quantitative Performance Evaluation Criteria. The approximation accuracy in our experiments was measured in terms of mean-squared error (MSE); that is,

$$\text{MSE} = \frac{1}{N}\sum_{n=1}^{N}\left[g(\mathbf{x}^n) - f(\mathbf{x}^n)\right]^2 = \frac{1}{N}\sum_{n=1}^{N} e^2\left(\mathbf{x}^n\right), \tag{12}$$

where $f(\mathbf{x}^n)$ denotes the approximation of the component RBFNs, the Bagging, or Bootstrap-ALNC.

The similarity between the approximator output and the target function, referred to as the approximation fidelity, was computed with the normalized correlation coefficient (NCC) as

$$\text{NCC} = \frac{\sum_{n=1}^{N} g\left(\mathbf{x}^n\right) f\left(\mathbf{x}^n\right)}{\sqrt{\sum_{n=1}^{N} g^2\left(\mathbf{x}^n\right) \sum_{n=1}^{N} f^2\left(\mathbf{x}^n\right)}} \times 100\%. \tag{13}$$

4. Results

Figure 2 illustrates the approximation curves provided by the SA, WA, and ALNC methods, with regard to the synthetic two-dimensional functions. Compared with the output curves of the SA, the curves produced by the WA and ALNC methods are closer to the target functions in the input range from 1.8 to 2.5 for the Zigzag function, from 7 to 13 for the Rhythm function, from 1.8 to 3.2 for the SinCos function, and from 1.5 to 2 for the ExpSin function, respectively. Three ensemble methods provide different 3-D surface approximation results, with respect to the functions of 3-D Mexican Hat (see Figure 3), Gabor (see Figure 4), SwingCos (see Figure 5), and Exponential (see Figure 6), respectively.

Concerning the 3-D Mexican Hat, the peak area approximated by the SA method, as depicted in Figure 3(b), is slightly skewed versus the target function in Figure 3(a), whereas such skewness does not appear in the results of the WA and ALNC methods. For the Gabor, the output surface shape of the SA fusion is severely distorted, as shown in Figure 4(b). The surfaces produced by the WA fusion, plotted in Figure 4(c), and the ALNC method, plotted in Figure 4(d), are much better than those of the SA method. In addition, the WA fusion surface is even smoother than the ALNC output surface. From Figures 5(b) and 5(c), we may observe that the output surface of the SA or WA fusion fails to match the SwingCos function such that several local crests and troughs in the approximated surface central region are missing. On the contrary, the ALNC method is able to predict the locations of these local crests and troughs, although its numerical estimate is not very precise. According to Figure 6, the surface regions around the crest and trough predicted by the SA and WA fusion methods are fluctuating, whereas the ALNC provides an approximation surface relatively closer to the Exponential target function in Figure 6(d).

The quantitative results on the synthetic data sets listed in Table 3 indicate that the MSE and NCC values obtained with all three ensemble methods are remarkably better than those of the component RBFNs on average, especially for the Zigzag, SinCos, ExpSin, and Gabor data sets. Even when approximating the SinCos function, the WA and ALNC both perfectly achieve almost zero error and 100% output fidelity. In addition, the ALNC consistently outperforms the SA fusion and is also superior to the WA method in most experiments (see the best performance results highlighted in Table 3).

Considering the benchmark regression data sets, Figure 7 and Table 4 show that the MSEs produced by the Bootstrap-ALNC in all six experiments are consistently lower than those of the prevailing Bagging or a single RBFN on average, especially the Bootstrap-ALNC reductions versus the Bagging the MSE values of 8.59% (2.8209/32.8325), 6.7% (28.5683/426.5834), and 17.54% (3.8174/21.7593) on the Stock, Bolt, and CPS-85-Wages data sets, respectively. The NCC improvements of the Bootstrap-ALNC over the Bagging are also noticeable in Figure 8 and Table 4. Such results indicate that the proposed ALNC method, along with the bootstrap resampling procedure, is competent to solve practical regression problems.

The CTC parameter indicates how much a fusion method would occupy the CPU computing resources, which involves the total elapsed time of the training of component RBFNs and the optimization of fusion weights. Thus, we only list the CTC values of the fusion methods in Tables 3 and 4. By comparing the CTC results on different data sets, we may find that the ALNC method consumed the least CPU execution time for almost all the function approximation and regression data sets, whereas the WA method occupied the most CPU resources. The WA fusion method would require some additional CPU execution time because it has to estimate the error distributions of the component RBFNs. The ALNC method can directly compute the fusion weights with the instantaneous errors of the component RBFNs to improve the efficiency. From Table 3, the CTC values of the SA method on the ExpSin and SwingCos data sets are smaller than those of the ALNC method. But it is worth noting that the ALNC fusion produces much better ExpSin approximation curve (see Figure 2) and SwingCos surface (see Figure 5) than the SA method.

5. Discussion

The performance evaluation results measured by the MSE and NCC parameters have demonstrated the effectiveness of the proposed ALNC method for function approximation and regression. It is worth noting that the MSE values could be influenced by the dependent variable scales; for example, the MSE results on the Bolt data set (Bagging: 426.5834; Bootstrap-ALNC: 398.0151) are much larger than those on the Abalone data set (Bagging: 4.6753; Bootstrap-ALNC: 4.4646); however, the performance improvement trends (the ALNC versus the SA or WA, the Bootstrap-ALNC versus the Bagging) are evident. Such amelioration effects of function approximation and regression are reflected qualitatively in the geometric patterns of the two-dimensional function curves in Figure 2 and three-dimensional surface plots in Figures 3–6. On the other hand, although some criticisms on the MSE criterion exist in the literature [44], this metric still has several excellent properties such as simplicity, valid Euclidean distance measure, and energy of the data errors, which makes the MSE widely used as a favorable metric in optimization, statistics, and data analysis.

From Table 3, it can be observed that the Bagging ensemble can only slightly improve the performance of the component RBFNs. The robustness of the RBFN is considered as the primary cause of such a phenomenon. According to the remarks of Breiman's work [22] "Bagging stable learners is not a good idea," because the robustness of the stable learners leads to more computational complexity but with little performance amelioration rewards.

The simple average (SA) and weighted average (WA) fusion strategies combine the component learners with the fixed or predefined fusion weights. In the meantime, they neglect that in some situations individual learners may be able to produce good approximations for some particular

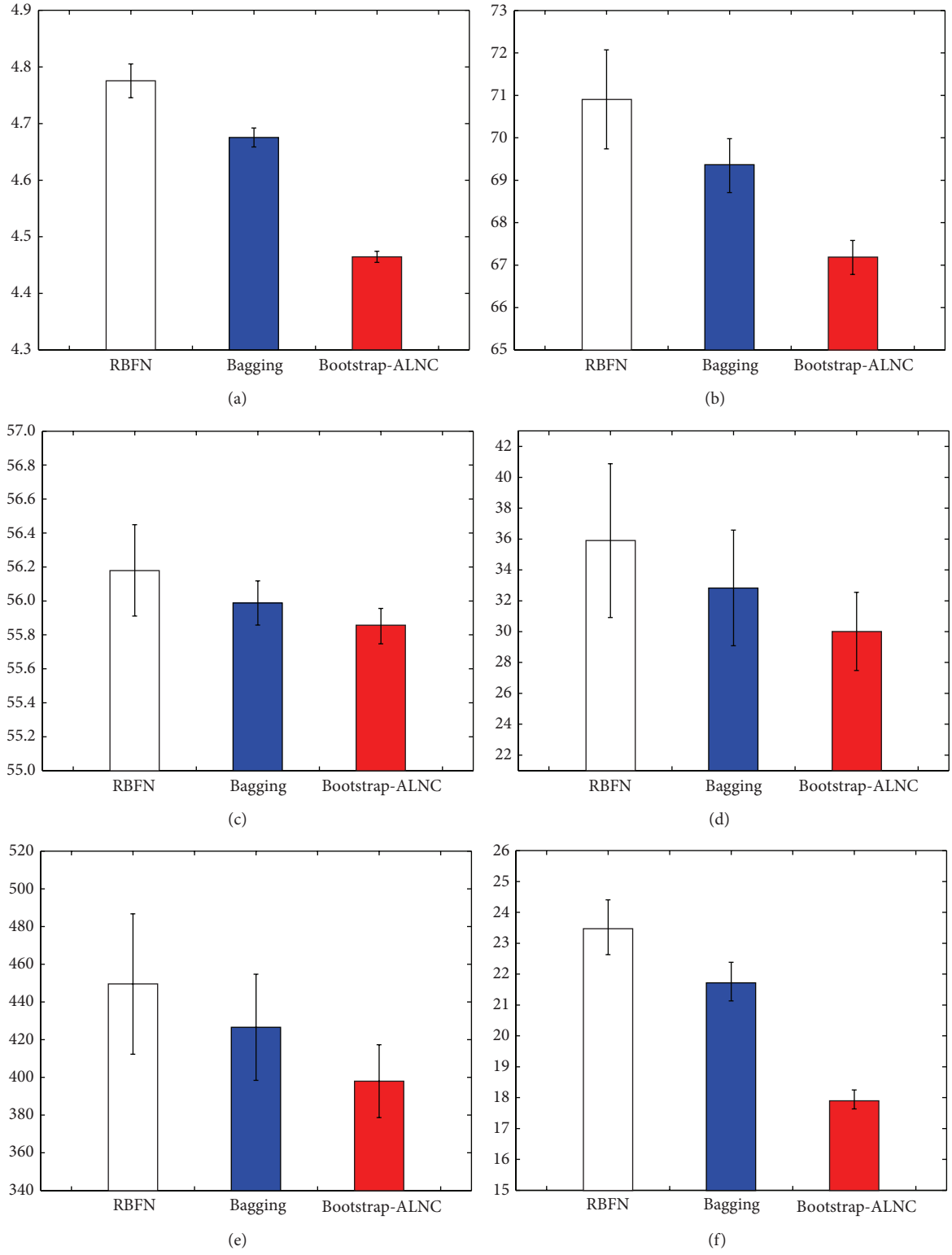

Figure 7: Bar graphics of the regression mean-squared errors produced by the component radial basis function networks (RBFNs), the Bagging, and the Bootstrap-ALNC on the data set: (a) Abalone, (b) Housing, (c) Auto-MPG, (d) Stock, (e) Bolt, and (f) CPS-85-Wages.

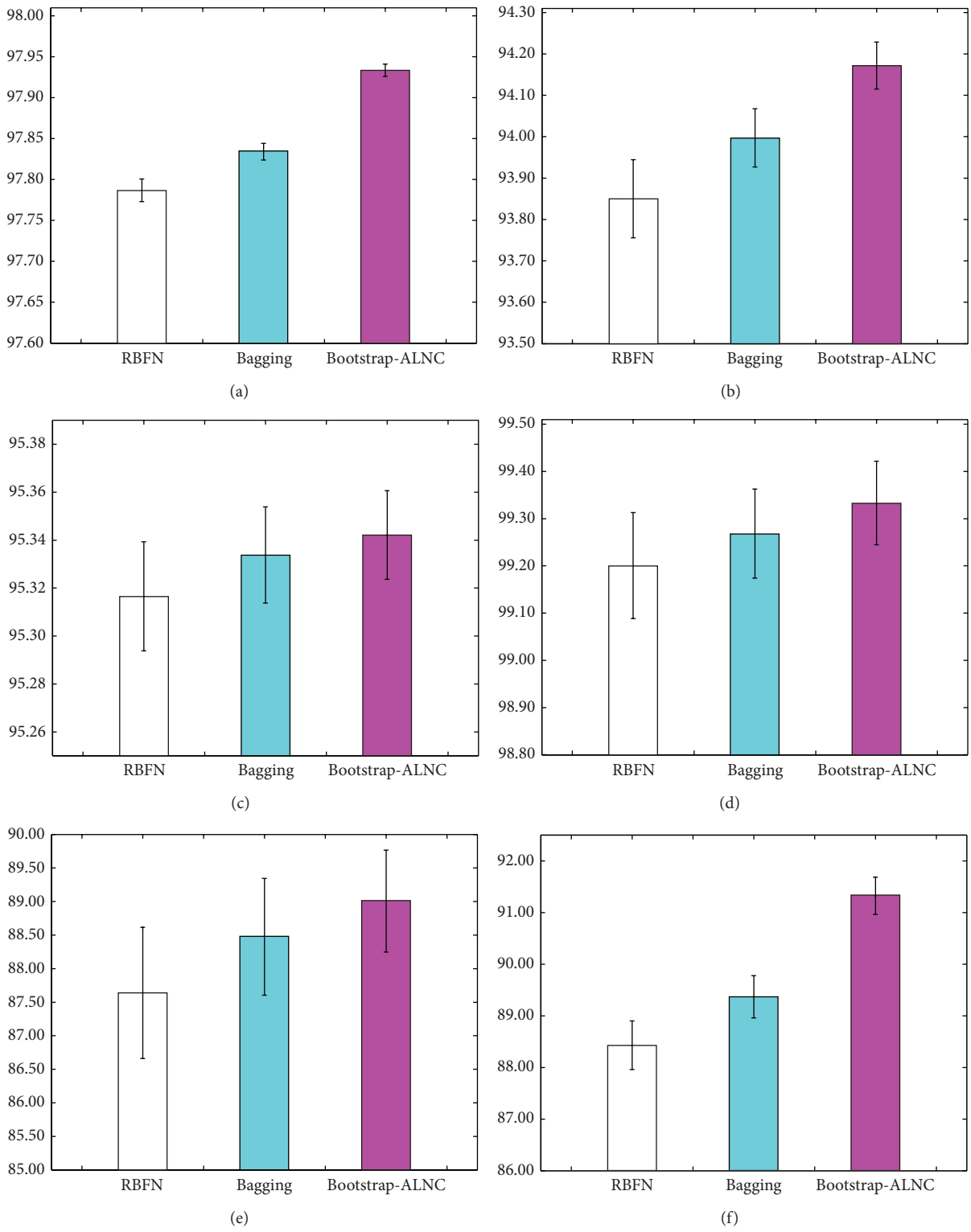

Figure 8: Bar graphics of the regression normalized correlation coefficients (in percentage) produced by the component radial basis function networks (RBFNs), the Bagging, and the Bootstrap-ALNC on the data set: (a) Abalone, (b) Housing, (c) Auto-MPG, (d) Stock, (e) Bolt, and (f) CPS-85-Wages.

Table 3: Results of function approximation experiments in terms of mean-squared error (MSE), normalized correlation coefficient (NCC), and computation time consumption (CTC in ms).

Function name	Evaluation	RBFN	SA	WA	ALNC
Zigzag	MSE	0.0295 ± 0.0735	0.0062 ± 0.0164	0.0007 ± 0.0049	**0.0004 ± 0.0011**
	NCC (%)	96.3055 ± 9.6683	99.4763 ± 9.3546	99.9250 ± 6.3702	**99.9571 ± 4.2214**
	CTC (ms)		7.42 ± 1.89	8.84 ± 0.92	**7.28 ± 0.95**
Rhythm	MSE	0.0029 ± 0.0010	0.0024 ± 0.0013	0.0023 ± 0.0012	**0.0020 ± 0.0009**
	NCC (%)	87.5134 ± 5.0192	89.7075 ± 3.2103	90.1384 ± 2.8523	**91.5593 ± 1.8064**
	CTC (ms)		8.71 ± 1.12	9.33 ± 1.74	**8.54 ± 1.03**
SinCos	MSE	0.0634 ± 0.2023	0.0055 ± 0.0047	**0.0000 ± 0.0002**	**0.0000 ± 0.0002**
	NCC (%)	97.7722 ± 7.2694	99.9010 ± 5.3231	**100.0000 ± 2.0012**	**100.0000 ± 2.0012**
	CTC (ms)		9.01 ± 1.24	9.24 ± 1.01	**8.52 ± 1.36**
ExpSin	MSE	2.6070 ± 5.9641	0.5745 ± 0.2563	0.2246 ± 0.1094	**0.1793 ± 0.0521**
	NCC (%)	98.9156 ± 2.5046	99.7826 ± 1.3005	99.9097 ± 0.7421	**99.9992 ± 0.5096**
	CTC (ms)		**8.42 ± 1.88**	9.12 ± 3.65	9.07 ± 2.09
3-D Mexican Hat	MSE	0.0056 ± 0.0092	0.0016 ± 0.0022	**0.0001 ± 0.0003**	**0.0001 ± 0.0003**
	NCC (%)	96.7638 ± 5.4163	99.2510 ± 2.3316	99.9973 ± 0.05433	**99.9992 ± 0.0097**
	CTC (ms)		12.67 ± 1.04	13.69 ± 1.06	**11.15 ± 0.98**
Gabor	MSE	0.0526 ± 0.0591	0.0284 ± 0.0087	0.0044 ± 0.0027	**0.0030 ± 0.0015**
	NCC (%)	85.9035 ± 19.5016	94.5788 ± 12.0816	99.1086 ± 7.3341	**99.4828 ± 4.2035**
	CTC (ms)		12.41 ± 1.35	12.97 ± 1.39	**11.79 ± 1.19**
SwingCos	MSE	0.6108 ± 0.2585	0.4614 ± 0.2448	0.3663 ± 0.1802	**0.2018 ± 0.0097**
	NCC (%)	79.6648 ± 9.7284	86.2856 ± 7.0205	89.2254 ± 6.6522	**94.3319 ± 4.3316**
	CTC (ms)		**11.38 ± 0.74**	13.03 ± 0.89	11.65 ± 0.81
Exponential	MSE	0.0016 ± 0.0022	0.0007 ± 0.0003	0.0004 ± 0.0001	**0.0001 ± 0.0001**
	NCC (%)	96.3979 ± 5.2557	98.5697 ± 3.4681	99.2090 ± 1.0506	**99.7315 ± 0.6233**
	CTC (ms)		12.46 ± 1.52	13.18 ± 2.13	**11.53 ± 1.08**

Table 4: Regression results in terms of mean-squared error (MSE), normalized correlation coefficient (NCC), and computation time consumption (CTC in ms).

Data set name	Evaluation	RBFN	Bagging	Bootstrap-ALNC
Abalone	MSE	4.7755 ± 0.0298	4.6753 ± 0.0167	**4.4646 ± 0.0098**
	NCC (%)	97.7867 ± 0.0139	97.8340 ± 0.0102	**97.9335 ± 0.0075**
	CTC (ms)		11.22 ± 3.51	**9.78 ± 0.85**
Housing	MSE	70.9045 ± 1.1676	69.3435 ± 0.6351	**67.1779 ± 0.4001**
	NCC (%)	93.8500 ± 0.0944	93.9970 ± 0.0704	**94.1718 ± 0.0568**
	CTC (ms)		8.62 ± 1.33	**7.11 ± 0.88**
Auto-MPG	MSE	56.1797 ± 0.2687	55.9879 ± 0.1305	**55.8517 ± 0.1038**
	NCC (%)	95.3166 ± 0.0228	95.3338 ± 0.0201	**95.3422 ± 0.0185**
	CTC (ms)		6.33 ± 0.71	**5.65 ± 0.57**
Stock	MSE	35.8967 ± 4.9799	32.8325 ± 3.7468	**30.0116 ± 2.5311**
	NCC (%)	99.2006 ± 0.1122	99.2685 ± 0.0942	**99.3332 ± 0.0886**
	CTC (ms)		8.31 ± 1.05	**7.28 ± 0.86**
Bolt	MSE	449.5107 ± 37.2104	426.5834 ± 28.1695	**398.0151 ± 19.3207**
	NCC (%)	87.6304 ± 0.9843	88.4725 ± 0.8702	**89.0087 ± 0.7596**
	CTC (ms)		4.32 ± 0.58	**3.48 ± 0.22**
CPS-85-Wages	MSE	23.5167 ± 0.8882	21.7593 ± 0.6251	**17.9419 ± 0.3058**
	NCC (%)	88.4300 ± 0.4704	89.3703 ± 0.4107	**91.3259 ± 0.3622**
	CTC (ms)		7.78 ± 1.24	**6.95 ± 1.06**

portions of the data set but incompetent for the rest of the data. The ALNC method is an instance-varying technique that concentrates its ensemble capability more on the local data points. This method can adaptively adjust the fusion weights, which explores the highest potential of the component learners toward precise approximations from one input instance to another. Thus, the function approximation or regression task over the entire data set can be split into several subtasks, in which the fusion strategy can seek for the most competent local learners. Nevertheless, despite its effectiveness, the ALNC method has some limitations. Because the ALNC method is still a supervised learning technique, the ALNC method is only limited to be suited for the function approximation and regression applications, rather than solving prediction or forecast problems, in which the desired references are not available to optimize the fusion parameters.

6. Conclusion

The adaptive linear and normalized combination (ALNC) is able to adaptively combine the component learners with the optimized fusion weights by solving the constrained quadratic programming problem. Depending on the performance of component learners on a specific input instance, the ALNC can exclusively select the best component learner or discard the worse one in the ensemble so as to provide the best approximation result. The property of instance-varying fusion weights allows the ALNC method to focus more on local approximation, although it sometimes leads to a slightly wrinkled approximated surface output. In general, the experimental results of low error rate and high fidelity percentage on the synthetic and benchmark data sets demonstrated the effectiveness and prominent advantages of the proposed ALNC method. Furthermore, the ALNC method is also promising for practical applications in the fields of pattern recognition, although it is known that the mean-squared error is not a very suitable performance measure for classification problems [45]. The future work could be directed toward an extended ensemble learning algorithm in accordance with other performance evaluation criteria, for the design of multiple classifier systems.

Conflict of Interests

There is no conflict of interests.

Acknowledgments

This work was supported by the Fundamental Research Funds for the Central Universities of China (Grant no. 2010121061), the Natural Science Foundation of Fujian (Grant no. 2011J01371), and the National Natural Science Foundation of China (Grant no. 81101115). Yunfeng Wu was supported by the 2013 Program for New Century Excellent Talents in Fujian Province University.

References

[1] T. J. Rivlin, *An Introduction to the Approximation of Functions*, Dover, Mineola, NY, USA, 1981.

[2] N. I. Achieser, *Theory of Approximation*, Dover, Mineola, NY, USA, 2004.

[3] J. Ramsay and B. Silverman, *Functional Data Analysis*, Springer, New York, NY, USA, 1997.

[4] R. M. Rangayyan and Y. F. Wu, "Screening of knee-joint vibroarthrographic signals using statistical parameters and radial basis functions," *Medical and Biological Engineering and Computing*, vol. 46, no. 3, pp. 223–232, 2008.

[5] R. M. Rangayyan and Y. Wu, "Analysis of vibroarthrographic signals with features related to signal variability and radial-basis functions," *Annals of Biomedical Engineering*, vol. 37, no. 1, pp. 156–163, 2009.

[6] R. J. Schilling, J. J. Carroll Jr., and A. F. Al-Ajlouni, "Approximation of nonlinear systems with radial basis function neural networks," *IEEE Transactions on Neural Networks*, vol. 12, no. 1, pp. 1–15, 2001.

[7] M. D. Buhmann, *Radial Basis Functions*, Cambridge University, Cambridge, UK, 2003.

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

[9] J. Park and I. W. Sandberg, "Universal approximation using radialbasis- function networks," *Neural Computation*, vol. 3, no. 2, pp. 246–257, 1991.

[10] S. Haykin, *Neural Networks: A Comprehensive Foundation*, Prentice Hall, Englewood Cliffs, NJ, USA, 2nd edition, 1998.

[11] L. K. Hansen and P. Salamon, "Neural network ensembles," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, no. 10, pp. 993–1001, 1990.

[12] S. Hashem and B. Schmeiser, "Improving model accuracy using optimal linear combinations of trained neural networks," *IEEE Transactions on Neural Networks*, vol. 6, no. 3, pp. 792–794, 1995.

[13] K. Woods, W. Philip Kegelmeyer, and K. Bowyer, "Combination of multiple classifiers using local accuracy estimates," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 4, pp. 405–410, 1997.

[14] J. Kittler, M. Hatef, R. P. W. Duin, and J. Matas, "On combining classifiers," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 3, pp. 226–239, 1998.

[15] L. I. Kuncheva, "A theoretical study on six classifier fusion strategies," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 2, pp. 281–286, 2002.

[16] A. Rahman and B. Verma, "Novel layered clustering-based approach for generating ensemble of classifiers," *IEEE Transactions on Neural Networks*, vol. 22, no. 5, pp. 781–792, 2011.

[17] Y. Wu and J. I. Arribas, "Fusing output information in neural networks: ensemble performs better," in *Proceedings of the 25th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC '03)*, pp. 2265–2268, September 2003.

[18] G. Valentini and F. Masulli, "Ensembles of learning machines," in *Neural Nets*, vol. 2486 of *Lecture Notes in Computer Science*, pp. 3–20, 2002.

[19] A. Sinha, H. Chen, D. G. Danu, T. Kirubarajan, and M. Farooq, "Estimation and decision fusion: a survey," *Neurocomputing*, vol. 71, no. 13–15, pp. 2650–2656, 2008.

[20] N. Ueda, "Optimal linear combination of neural networks for improving classification performance," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 2, pp. 207–215, 2000.

[21] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of Computer and System Sciences*, vol. 55, no. 1, pp. 119–139, 1997.

[22] L. Breiman, "Bagging predictors," *Machine Learning*, vol. 24, no. 2, pp. 123–140, 1996.

[23] G. Ridgeway, "The state of boosting," *Computing Science and Statistics*, vol. 31, pp. 172–181, 1999.

[24] B. Efron and R. Tibshirani, *An Introduction to the Bootstrap*, Chapman and Hall, New York, NY, USA, 1993.

[25] G. Fumera and F. Roli, "A theoretical and experimental analysis of linear combiners for multiple classifier systems," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 6, pp. 942–956, 2005.

[26] Y. Wu, J. He, Y. Man, and J. I. Arribas, "Neural network fusion strategies for identifying breast masses," in *Proceedings of the IEEE International Joint Conference on Neural Networks (IJCNN '04)*, pp. 2437–2442, July 2004.

[27] Y. Wu, C. Wang, S. C. Ng, A. Madabhushi, and Y. Zhong, "Breast cancer diagnosis using neural-based linear fusion strategies," in *Neural Information Processing*, vol. 4234 of *Lecture Notes in Computer Science*, pp. 165–175, 2006.

[28] Y. Wu and S. C. Ng, "Breast tissue classification based on unbiased linear fusion of neural networks with normalized weighted average algorithm," in *Proceedings of the International Joint Conference on Neural Networks (IJCNN '07)*, pp. 2846–2850, Orlando, Fla, USA, August 2007.

[29] Y. Wu and C. Wang, "Linear least-squares fusion of multilayer perceptrons for protein localization sites prediction," in *Proceedings of the IEEE 32nd Annual Northeast Bioengineering Conference*, pp. 157–158, Easton, Pa, USA, April 2006.

[30] Y. Wu, Y. Ma, X. Liu, and C. Wang, "A bootstrap-based linear classifierfusion system for protein subcelluar location prediction," in *Proceedings of the 28th Annual International Conference of IEEE Engineering in Medicine and Biology Society (EMBC '06)*, pp. 4229–4232, New York, NY, USA, 2006.

[31] Y. Wu and S. C. Ng, "Unbiased linear neural-based fusion with normalized weighted average algorithm for regression," in *Advances in Neural Networks*, vol. 4493 of *Lecture Notes in Computer Science*, pp. 664–670, 2007.

[32] Y. Wu, Y. Zhou, S.-C. Ng, and Y. Zhong, "Combining neural-based regression predictors using an unbiased and normalized linear ensemble model," in *Proceedings of the International Joint Conference on Neural Networks (IJCNN '08)*, pp. 3955–3960, Hong Kong, June 2008.

[33] D. Opitz and R. Maclin, "Popular ensemble methods: An Empirical Study," *Journal of Artificial Intelligence Research*, vol. 11, pp. 169–198, 1999.

[34] K. Tumer and J. Ghosh, "Analysis of decision boundaries in linearly combined neural classifiers," *Pattern Recognition*, vol. 29, no. 2, pp. 341–348, 1996.

[35] V. Tresp and M. Taniguchi, "Combining estimators using non-constant weighting functions," in *Advances in Neural Information Processing Systems*, G. Tasuaro, D. Touretzky, and T. Leen, Eds., vol. 7, pp. 419–426, MIT Press, Cambridge, Mass, USA, 1995.

[36] Z.-H. Zhou, J. Wu, and W. Tang, "Ensembling neural networks: many could be better than all," *Artificial Intelligence*, vol. 137, no. 1-2, pp. 239–263, 2002.

[37] Y. Wu and S.-C. Ng, "Adaptively fusing neural network predictors toward higher accuracy: a case study," in *Proceedings of the IEEE International Conference on Computational Intelligence for Measurement Systems and Applications (CIMSA '09)*, pp. 273–276, Hong Kong, May 2009.

[38] Y. Wu and S. Krishnan, "Combining least-squares support vector machines for classification of biomedical signals: a case study with knee-joint vibroarthrographic signals," *Journal of Experimental and Theoretical Artificial Intelligence*, vol. 23, no. 1, pp. 63–77, 2011.

[39] Y. Wu, S. Cai, M. Lu, and S. Krishnan, "An artificial-neural-networkbased multiple classifier system for knee-joint vibration signal classification," in *Advances in Computer, Communication, Control and Automation*, Y. W. Wu, Ed., vol. 121 of *Lecture Notes in Electrical Engineering*, pp. 235–242, Springer, Heidelberg, Germany, 2011.

[40] S. Cai, S. Yang, F. Zheng, M. Lu, Y. Wu, and S. Krishnan, "Knee joint vibration signal analysis with matching pursuit decomposition and dynamic weighted classifier fusion," *Computational and Mathematical Methods in Medicine*, vol. 2013, Article ID 904267, 11 pages, 2013.

[41] S. G. Nash and A. Sofer, *Linear and Nonlinear Programming*, McGraw-Hill, Columbus, Ohio, USA, 1995.

[42] A. Asuncion and D. J. Newman, "UCI machine learning repository," 2007, http://archive.ics.uci.edu/ml/.

[43] 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, pp. 302–309, 1991.

[44] Z. Wang and A. C. Bovik, "Mean squared error: lot it or leave it? A new look at signal fidelity measures," *IEEE Signal Processing Magazine*, vol. 26, no. 1, pp. 98–117, 2009.

[45] C. M. Bishop, *Pattern Recognition and Machine Learning*, Springer, New York, NY, USA, 2006.