# Sequential Projection-Based Metacognitive Learning in a Radial Basis Function Network for Classification Problems

Giduthuri Sateesh Babu, *Student Member, IEEE*, and Sundaram Suresh, *Senior Member, IEEE*

***Abstract*— In this paper, we present a sequential projection-based metacognitive learning algorithm in a radial basis function network (PBL-McRBFN) for classification problems. The algorithm is inspired by human metacognitive learning principles and has two components: a cognitive component and a metacognitive component. The cognitive component is a single-hidden-layer radial basis function network with evolving architecture. The metacognitive component controls the learning process in the cognitive component by choosing the best learning strategy for the current sample and adapts the learning strategies by implementing self-regulation. In addition, sample overlapping conditions and past knowledge of the samples in the form of pseudosamples are used for proper initialization of new hidden neurons to minimize the misclassification. The parameter update strategy uses projection-based direct minimization of hinge loss error. The interaction of the cognitive component and the metacognitive component addresses the what-to-learn, when-to-learn, and how-to-learn human learning principles efficiently. The performance of the PBL-McRBFN is evaluated using a set of benchmark classification problems from the University of California Irvine machine learning repository. The statistical performance evaluation on these problems proves the superior performance of the PBL-McRBFN classifier over results reported in the literature. Also, we evaluate the performance of the proposed algorithm on a practical Alzheimer's disease detection problem. The performance results on open access series of imaging studies and Alzheimer's disease neuroimaging initiative datasets, which are obtained from different demographic regions, clearly show that PBL-McRBFN can handle a problem with change in distribution.**

***Index Terms*— Alzheimer's disease, metacognitive learning, projection-based learning, radial basis function network classifier, self-regulatory thresholds.**

## I. Introduction

IN MACHINE learning, batch learning algorithms require complete training data to build the model. In most practical applications, especially in medical diagnosis, the complete training data describing the input-output relationship is not available *a priori*. For these problems, classical batch-learning algorithms are rather infeasible and instead sequential learning is employed. In a sequential learning framework, the training samples arrive one-by-one and the samples are discarded after the learning process. Hence, it requires less memory and computational time for the learning process. In addition, sequential learning algorithms automatically determine the minimal architecture that can accurately approximate the true decision function described by stream of the training samples [1].

Radial basis function (RBF) networks have been extensively used in a sequential learning framework because of their universal approximation ability and simplicity of architecture [2]. Recently, there has been renewed interest in single-hidden-layered RBF networks with least-square error training criterion, partly due to their modeling ability and partly due to the existence of efficient learning algorithms such as extreme learning machine (ELM) [3], and second-order training methods [4]. In recent years, researchers have been focusing on sequential learning algorithms for RBF networks through streams of data. A brief discussion of existing research work in a sequential learning framework is given below.

The first sequential learning algorithm introduced in the literature was resource allocation network (RAN) [5]. RAN uses novelty-based neuron growth to evolve the network architecture automatically and linear mean square algorithm to update the network parameters. Minimal RAN (MRAN) [6] adapted the similar approach, but it uses extended Kalman filter (EKF) for parameter update and pruning strategy to determine the compact network architecture. Growing and pruning RBF network [7] uses the neuron significance to select the neuron growth/prune criterion. On-line sequential ELM (OS-ELM) [8] is the sequential version of ELM using recursive least squares. OS-ELM randomly assigns input weights with fixed number of hidden neurons and uses minimum norm recursive least squares to determine the output weights analytically. The random selection of input weights with fixed number of hidden neurons affects the performance of OS-ELM significantly in case of sparse and imbalance datasets [9]. Sequential multicategory RBF network (SMC-RBF) combines the misclassification error, similarity measure within the class, and prediction error in neuron growth/parameter update criterion [2]. In SMC-RBF, it has been shown that growing/pruning criterion based on class-specific conditions improves the classification performance than conditions based on approximation error alone.

The aforementioned sequential learning algorithms in neural networks gain the knowledge about the information within

Manuscript received July 16, 2012; revised October 22, 2012; accepted October 22, 2012. Date of publication December 21, 2012; date of current version January 11, 2012. This work was supported in part by Nanyang Technological University and the Singapore Ministry of Defense under Grant MINDEF-NTU- 948JPP/11/02/05.

The authors are with the School of Computer Engineering, Nanyang Technological University, 639798 Singapore (e-mail: SBGiduthuri@ntu.edu.sg; ssundaram@ntu.edu.sg).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNNLS.2012.2226748

stream of data by using all the samples in the training dataset. Simply put, these algorithms have human information-processing abilities, such as perception, learning, remembering, judging, and problem solving, which are cognitive in nature. On the other hand, human learning studies suggested that the learning process is effective when the learners adopt self-regulation in the learning using metacognition [10], [11]. Metacognition means cognition about cognition. Precisely, in a metacognitive framework, learners think about their cognitive process, and improve and control it by developing new strategies using the information contained in their memory. If an RBF network is considered as a cognitive component in metacognitive framework to analyze its cognitive process and to choose suitable learning strategies adaptively, then it is referred to as metacognitive RBF network (McRBFN). Hence, there is a need to develop such McRBFN that is capable of deciding what-to-learn, when-to-learn, and how-to-learn from a stream of training data to devise the decision function by emulating the human self-regulated learning principles.

Self-adaptive RAN (SRAN) [12] is a sequential learning algorithm and also addresses the what-to-learn component of metacognition by selecting the significant samples using misclassification error and hinge loss error. The complex version of the above algorithm is complex-valued self-regulating RAN (CSRAN) [13]. It has been shown in [12] and [13], that selecting the significant samples and removing repetitive samples in learning helps to improve the generalization performance. Therefore, it is apparent that emulating the three components of metacognition with suitable learning strategies would improve the generalization ability of a neural network. The drawbacks in the above algorithms are as follows.

1) The selection of significant samples from stream of training data are based on simple error criterion which is not sufficient.
2) The allocation of new hidden neuron center without considering the amount of overlap with already existing neuron centers leads to misclassification.
3) The knowledge gained from past trained samples is not used in further learning.
4) These algorithms use computationally intensive EKF for parameter update.

Metacognitive neural network (McNN) [14] and metacognitive neuro-fuzzy inference system (McFIS) [15] address the first two issues efficiently by using three components of metacognition. However, McNN and McFIS use computationally intensive EKF for parameter updation and do not utilize the knowledge acquired from past trained samples. Similar works of metacognition in complex domain are reported in [16] and [17]. Recently proposed projection-based learning (PBL) in McRBFN [18] addresses the first three issues in batch framework except proper utilization of the past knowledge stored in the network and applied in detection of neurodegenerative diseases [19], [20]. Therefore, in this paper, we propose a fast and efficient sequential PBL algorithm for McRBFN.

There are several models of metacognition available in educational psychology and survey of these models of metacognition is reported in [21]. Nelson and Narens proposed a simple model of metacognition in [22] that clearly highlights the various self-regulated principles in human metacognition. Nelson and Narens model contains a cognitive component and a metacognitive component. The flow of information from the cognitive component to the metacognitive component is referred as a monitory signal, while the flow of information from the metacognitive component to the cognitive component is referred as a control signal. The control signal changes the state of the cognitive component or changes the cognitive component itself. On the other hand, monitory signal updates the metacognitive component about the state of cognitive component. Similar to the Nelson and Narens model of metacognition [22], McRBFN also has two components. A single-hidden-layer RBF network with evolving architecture is the cognitive component, and the metacognitive component contains a dynamic model of the cognitive component, knowledge measures, and self-regulated thresholds. The cognitive component learns from stream of training data by growing hidden neurons or updating the output weights of hidden neurons so as to approximate the decision function. When a new hidden neuron is added to the cognitive component, the Gaussian parameters (center and width) are determined based on the current training sample and the output weights are estimated using the PBL algorithm. Finding optimal output weights is first formulated as a linear programming problem from the principles of minimization and Real calculus. The PBL algorithm then converts the linear programming problem into a system of linear equations and provides a solution for the optimal output weights, corresponding to the minima of the error function. For classification problems, it has been proven theoretically in [23] and [24] that a classifier developed using a hinge-loss function estimates the posterior probability more accurately than a mean square error function. Hence, the PBL algorithm in the cognitive component of McRBFN employs hinge-loss error function. In addition, the PBL algorithm uses existing neurons in the cognitive component as pseudosamples. Thereby, the proposed algorithm exploits the knowledge stored in the network for proper initialization. When a sample is presented to the cognitive component of McRBFN, the metacognitive component of McRBFN measures the information contained in the current training sample with respect to the knowledge acquired in the cognitive component through its knowledge measures. The knowledge measures in the metacognitive component are predicted class label, maximum hinge error, and classwise significance. In kernel methods, spherical potential is widely used to determine whether all the trained samples are enclosed tightly by the Gaussian kernels [25]. Hence, the spherical potential is taken as classwise significance of a sample and it is measured as the squared distance between the sample and the hyperdimensional projection. In this paper, spherical potential is redefined to address the classification problems. Using the above-mentioned knowledge measures along with self-regulatory thresholds, the metacognitive component constructs two sample-based learning strategies and three neuron-based learning strategies. As each training sample is presented, the metacognitive component selects one of the five strategies such that the cognitive component learns the decision function

efficiently and achieves better classification performance. The metacognitive component also measures the amount of overlap between the current training sample and nearest neurons in the interclass and intraclass to determine new hidden parameters. The PBL algorithm for McRBFN to obtain the network parameters is referred to as PBL-McRBFN.

The performance of the proposed PBL-McRBFN classifier is evaluated through a set of benchmark binary and multicategory classification problems from the University of California, Irvine (UCI) machine learning repository [26]. We consider nine multicategory and five binary classification problems with varying values of imbalance factor (I.F). In all these problems, the performance of PBL-McRBFN is compared against the best-performing classifiers available in the literature. Further, we have also conducted a nonparametric Friedman test [27] to indicate the statistical significance in performance of PBL-McRBFN classifier. The results indicate PBL-McRBFN classifier outperforms the existing classifiers.

Finally, the performance of the PBL-McRBFN classifier has also been evaluated on a practical Alzheimer's disease (AD) detection problem using magnetic resonance imaging (MRI). For performance evaluation, we consider well-known open access series of imaging studies (OASIS) [28] and Alzheimer's disease neuroimaging initiative (ADNI) [29] MRI databases. These two MRI databases are generated in similar imaging environment with different demographic patient conditions. Hence, we use these datasets to demonstrate the capability of learning from changing distribution. The proposed classifier approximates the relationship between morphometry features and class label. First, we demonstrate the performance of the proposed classifier on these datasets and compare with the existing results reported in the literature. Later, we show the ability to capture the change in distribution by adapting the classifier developed using OASIS with ADNI images. The results show that the classifier has the ability to learn efficiently from a stream of data.

The structure of this paper is as follows. Section II describes the metacognitive RBF network for classification problems. Section III presents the performance evaluation of PBL-McRBFN classifier on a set of benchmark and practical AD detection datasets in comparison with the best-performing classifiers available in the literature. Section IV summarizes the conclusions from this paper.

## II. McRBFN for Classification Problems

In this section, we describe the McRBFN for solving classification problems. The classification problem in sequential framework can be defined as follows: Given a stream of training data, $\{(\mathbf{x}^1, c^1), \ldots, (\mathbf{x}^t, c^t), \ldots\}$, where $\mathbf{x}^t = [x_1^t, \ldots, x_m^t]^T \in \Re^m$ is the $m$-dimensional input of the $t$th sample, $c^t \in \{1, \ldots, n\}$ is its class label, and $n$ is the total number of classes. The coded class labels $(\mathbf{y}^t = [y_1^t, \ldots, y_j^t, \ldots, y_n^t]^T) \in \Re^n$ are given by

$$y_j^t = \begin{cases} 1, & \text{if} \quad c^t = j \\ -1, & \text{otherwise} \end{cases} \quad j = 1, \ldots, n. \tag{1}$$

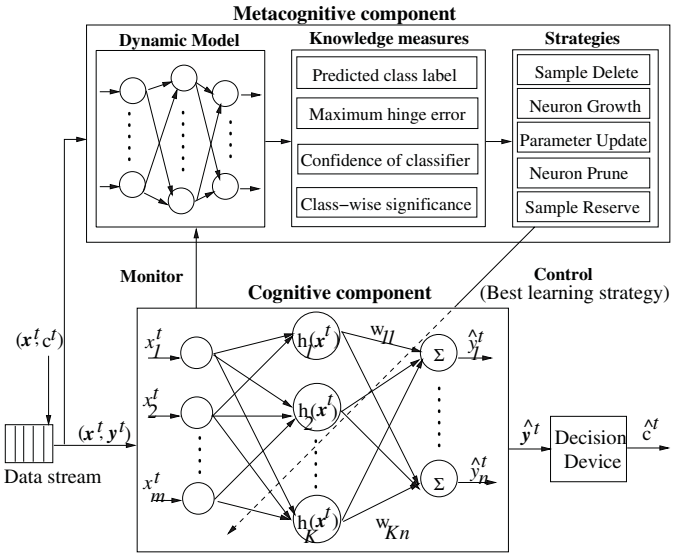


Fig. 1. Schematic diagram of metacognitive learning.

The objective of McRBFN classifier is to approximate the underlying decision function that maps $\mathbf{x}^t \in \Re^m \rightarrow \mathbf{y}^t \in \Re^n$. McRBFN begins with zero hidden neuron and selects suitable strategy for each sample.

### A. Metacognitive Learning in RBF Network

The schematic diagram of metacognitive learning in McRBFN is shown in Fig. 1. The cognitive component of McRBFN is an RBF network with evolving architecture. The metacognitive component of McRBFN contains a dynamic model of the cognitive component. When a new training sample is presented to the McRBFN, the metacognitive component of McRBFN estimates the knowledge present in the new training sample with respect to the cognitive component. Based on this information, the metacognitive component controls the learning process of the cognitive component by selecting suitable strategy for the current sample.

*1) Cognitive Component of McRBFN:* The cognitive component of McRBFN is a single-hidden-layered feed forward RBF network with a linear input and output layers. The neurons in the hidden layer employ the Gaussian activation function.

Without loss of generality, we assume that the McRBFN builds $K$ neurons from $t-1$ training samples. For a given input $\mathbf{x}^t$, the predicted output $(\widehat{y}_j^t)$ of McRBFN is

$$\widehat{y}_j^t = \sum_{k=1}^{K} w_{kj} h_k^t, \qquad j = 1, \ldots, n \tag{2}$$

where $w_{kj}$ is the weight connecting the $k$th hidden neuron to the $j$th output neuron and $h_k^t$ is the response of the $k$th hidden neuron to the input $\mathbf{x}^t$ is given by

$$h_k^t = \exp\left(-\frac{\|\mathbf{x}^t - \boldsymbol{\mu}_k^l\|^2}{(\sigma_k^l)^2}\right) \tag{3}$$

where $\boldsymbol{\mu}_k^l \in \Re^m$ is the center and $\sigma_k^l \in \Re^+$ is the width of the $k$th hidden neuron. Here, the superscript $l$ represents the corresponding class of the hidden neuron.

*a) PBL algorithm:* The projection-based learning algorithm works on the principle of minimization of error function and finds the optimal network output weights for which the error is minimum.

The considered error function is the sum of squared errors at output neurons of McRBFN. The error function for $i$th sample is defined as

$$J_i = \sum_{j=1}^{n}\left(y_j^i - \sum_{k=1}^{K} w_{kj} h_k^i\right)^2. \tag{4}$$

For $t$ training samples, the overall error function is defined as

$$J(\mathbf{W}) = \frac{1}{2}\sum_{i=1}^{t} J_i = \frac{1}{2}\sum_{i=1}^{t}\sum_{j=1}^{n}\left(y_j^i - \sum_{k=1}^{K} w_{kj} h_k^i\right)^2 \tag{5}$$

where $h_k^i$ is the response of the $k$th hidden neuron for $i$th training sample.

The optimal network output weights ($\mathbf{W}^* \in \Re^{K\times n}$) are estimated such that the total error reaches its minimum

$$\mathbf{W}^* = \arg \min_{\mathbf{W}\in\Re^{K\times n}} J(\mathbf{W}). \tag{6}$$

The optimal $\mathbf{W}^*$ corresponding to the minima of the error function ($J(\mathbf{W}^*)$) is obtained by equating the first-order partial derivative of $J(\mathbf{W})$ with respect to the output weight to zero

$$\frac{\partial J(\mathbf{W})}{\partial w_{pj}} = 0, \qquad p = 1,\ldots,K;\ \ j = 1,\ldots,n. \tag{7}$$

Equating the first partial derivative to zero and re-arranging

$$\sum_{k=1}^{K}\sum_{i=1}^{t} h_k^i h_p^i w_{kj} = \sum_{i=1}^{t} h_p^i y_j^i \tag{8}$$

which can be represented in matrix form as

$$\mathbf{AW} = \mathbf{B} \tag{9}$$

where the projection matrix $\mathbf{A} \in \Re^{K\times K}$ is given by

$$a_{kp} = \sum_{i=1}^{t} h_k^i h_p^i, \qquad k = 1,\ldots,K;\ \ p = 1,\ldots,K \tag{10}$$

and the output matrix $\mathbf{B} \in \Re^{K\times n}$ is

$$b_{pj} = \sum_{i=1}^{t} h_p^i y_j^i, \qquad p = 1,\ldots,K;\ \ j = 1,\ldots,n. \tag{11}$$

Equation (9) gives the set of $K\times n$ linear equations with $K\times n$ unknown output weights $\mathbf{W}$.

The solution for $\mathbf{W}$ obtained as a solution to the set of equations as given in (9) is minimum, if $(\partial^2 J/\partial {w_{lp}}^2) > 0$. The second derivative of the error function ($J$) with respect to the output weights is given by

$$\frac{\partial^2 J(\mathbf{W})}{\partial {w_{lp}}^2} = \sum_{i=1}^{t} h_p^i h_p^i \ = \ \sum_{i=1}^{t} |h_p^i|^2 > 0. \tag{12}$$

As the second derivative of the error function $J(\mathbf{W})$ is positive, the following observations can be made from (12).

i) $J$ is a convex function.
ii) The obtained output weight $W^*$ is the weight corresponding to the minima of the error function ($J$).

The solution for the system of equations in (9) can be determined as follows:

$$\mathbf{W}^* = \mathbf{A}^{-1}\mathbf{B}. \tag{13}$$

*2) Metacognitive Component of McRBFN:* The metacognitive component contains a dynamic model of the cognitive component, knowledge measures, and self-regulated thresholds. During the learning process, the metacognitive component monitors the cognitive component and updates its dynamic model of the cognitive component. When a new training sample $(\mathbf{x}^t, \mathbf{y}^t)$ is presented to McRBFN, the metacognitive component of McRBFN estimates the knowledge present in the new training sample with respect to the cognitive component using its knowledge measures. The metacognitive component uses predicted class label ($\widehat{c}^t$), maximum hinge error ($E^t$), confidence of classifier ($\hat{p}(c^t|\mathbf{x}^t)$), and classwise significance ($\psi_c$) as the measures of knowledge in the new training sample. The self-regulated thresholds are adapted to capture the knowledge present in the new training sample. Based on the knowledge measures and self-regulated thresholds, the metacognitive component chooses one of the two sample-based learning strategies or three neuron-based learning strategies to learn the current sample accurately.

The knowledge measures are defined as shown below.

*a) Predicted class label ($\widehat{c}^t$):* Using the predicted output ($\widehat{\mathbf{y}}^t$) it can be obtained as

$$\widehat{c}^t = arg \max_{j\in 1,\ldots,n} \widehat{y}_j^t. \tag{14}$$

*b) Maximum hinge error ($E^t$):* McRBFN uses the hinge loss error ($\mathbf{e^t} = [e_1^t,\ldots,e_j^t,\ldots,e_n^t]^T) \in \Re^n$ and is defined as

$$e_j^t = \begin{cases} y_j^t - \widehat{y}_j^t & \text{if } \ y_j^t \widehat{y}_j^t < 1 \\ 0 & \text{otherwise} \end{cases} \qquad j = 1,\ldots,n. \tag{15}$$

Using the hinge loss error $\mathbf{e^t}$, the maximum hinge error ($E^t$) can be obtained as

$$E^t = \max_{j\in 1,2,\ldots,n} \left|e_j^t\right|. \tag{16}$$

*c) Confidence of classifier ($\hat{p}(c^t|\mathbf{x}^t)$):* The confidence level of classification or predicted posterior probability is given as

$$\hat{p}(j|\mathbf{x}^t) = \frac{\min(1, \max(-1, \hat{y}_j^t)) + 1}{2}, \qquad j = c^t. \tag{17}$$

*d) Classwise significance ($\psi_c$):* The classwise distribution plays a vital role and it will influence the performance the classifier significantly [2]. Hence, we use the measure of the spherical potential of the new training sample $\mathbf{x}^t$ belonging to class $c$ with respect to the neurons associated with same class (i.e., $l = c$). Let $K^c$ be the number of neurons associated

with the class $c$, then classwise spherical potential or classwise significance ($\psi_c$) is defined as

$$\psi_c = \frac{1}{K^c} \sum_{k=1}^{K^c} h\left(\mathbf{x}^t, \boldsymbol{\mu}_k^c\right). \tag{18}$$

The spherical potential explicitly indicates the knowledge contained in the sample, a higher value of spherical potential (close to one) indicates that the sample is similar to the existing knowledge in the cognitive component, and a smaller value of spherical potential (close to zero) indicates that the sample is novel. For more details on the classwise significance, one may refer to [14].

*3) Learning Strategies:* The metacognitive component devises various learning strategies using the knowledge measures and the self-regulated thresholds that address the basic principles of self-regulated learning (i.e., what-to-learn, when-to-learn, and how-to-learn). The metacognitive part controls the learning process in the cognitive component by selecting one of the following five learning strategies for the new training sample.

a) *Sample Delete Strategy:* If the new training sample contains information similar to the knowledge present in the cognitive component, then delete the new training sample from the training dataset without using it in the learning process.
b) *Neuron Growth Strategy:* Use the new training sample to add a new hidden neuron in the cognitive component. During neuron addition, sample overlapping conditions are identified to allocate a new hidden neuron appropriately.
c) *Parameter Update Strategy:* The new training sample is used to update the parameters of the cognitive component. PBL is used to update the parameters.
d) *Neuron Pruning Strategy:* Very less contributed neurons in a class over a period of same class samples are deleted from the cognitive component.
e) *Sample Reserve Strategy:* If the new training sample contains some information but not significant, they can be used at later stage of the learning process for fine tuning the parameters of the cognitive component.

The principle behind these five learning strategies are described in detail below.

*a) Sample delete strategy:* When the predicted class label of the new training sample is same as the actual class label and the estimated posterior probability is close to 1, then the new training sample does not provide additional information to the classifier and can be deleted from training sequence without being used in learning process. The sample deletion criterion is given by

$$\widehat{c}^t == c^t \quad \text{AND} \quad \hat{p}(c^t|\mathbf{x}^t) \geq \beta_d. \tag{19}$$

The metacognitive deletion threshold ($\beta_d$) controls the number of samples participating in the learning process. The sample deletion strategy prevents learning of the samples with similar information, and thereby, avoids overtraining and reduces the computational effort. In our simulation studies, it is selected in the interval of [0.9–0.95].

*b) Neuron growth strategy:* When a new training sample contains significant information and the predicted class label is different from the actual class label, then one needs to add a new hidden neuron to represent the knowledge contained in the sample. The neuron growth criterion is given by

$$\left(\widehat{c}^t \neq c^t \quad \text{OR} \quad E^t \geq \beta_a\right) \quad \text{AND} \quad \psi_c(\mathbf{x}^t) \leq \beta_c \tag{20}$$

where $\beta_c$ is the metacognitive knowledge measurement threshold and $\beta_a$ is the self-adaptive metacognitive addition threshold. The terms $\beta_c$ and $\beta_a$ first allow the samples with significant knowledge for learning then use the other samples for fine tuning. The $\beta_a$ is adapted as follows:

$$\beta_a := \delta\beta_a + (1-\delta)E^t \tag{21}$$

where $\delta$ is the slope that controls rate of self-adaptation and is set close to one. The $\beta_a$ adaptation allows McRBFN to add neurons only when the presented samples to the cognitive network contains significant information. In our simulation studies, $\beta_c$ is selected in the interval of [0.3–0.7] and $\beta_a$ is selected in the interval of [1.3–1.7]. Details on influence of these parameters are given in [14].

When a new neuron $(K+1)$ is added, the existing sequential learning algorithms initialize center based on the new training sample and width based on the distance with nearest neuron. The new training sample may overlap with existing neurons in other classes or will be a distinct cluster far away from the nearest neuron in the same class. Therefore, one has to consider the amount of overlap with respect to existing neurons while initializing the new neuron parameters. Also, the existing sequential learning algorithms estimate new neuron output weights based on the error of the new training sample. In sequential learning framework, the output weight initialization using only the new training sample influences the performance significantly, due to absence of knowledge contained in the past samples. In the proposed McRBFN, the above-mentioned issues are dealt with as follows.

i) The new neuron output weights will be estimated using existing knowledge of past trained samples stored in the network.
ii) The new neuron center and width parameters will be initialized on the basis of new training sample distances to existing interclass and interclass nearest neurons.

Let $nrS$ be the nearest hidden neuron in the intraclass and $nrI$ be the nearest hidden neuron in the interclass as

$$nrS = \arg \min_{l==c;\forall k} \|\mathbf{x}^t - \boldsymbol{\mu}_k^l\|; \quad nrI = \arg \min_{l\neq c;\forall k} \|\mathbf{x}^t - \boldsymbol{\mu}_k^l\|. \tag{22}$$

Let the Euclidian distances between the new training sample to $nrS$ and $nrI$ are given as follows:

$$d_S = ||\mathbf{x}^t - \boldsymbol{\mu}_{nrS}^c||; \quad d_I = ||\mathbf{x}^t - \boldsymbol{\mu}_{nrI}^l||. \tag{23}$$

We can determine the new hidden neuron center ($\boldsymbol{\mu}_{K+1}^c$) and width ($\sigma_{K+1}^c$) parameters on the basis of overlapping and nonoverlapping conditions using the nearest neuron distances as follows.

i) *Distinct Sample:* If ($d_S >> \sigma^c_{nrS}$ AND $d_I >> \sigma^l_{nrI}$) then the new training sample does not overlap with any class cluster, and is from a distinct cluster

$$\boldsymbol{\mu}^l_{K+1} = \mathbf{x}^t; \quad \sigma^l_{K+1} = \kappa\sqrt{(\mathbf{x}^t)^T\mathbf{x}^t}, \qquad l = c \tag{24}$$

where $\kappa$ is a overlap factor of the hidden units, which lies in the range $0.5 \le \kappa \le 1$.

ii) *Nonoverlapping:* If the intraclass and interclass distance ratio is $< 1$, then the sample does not overlap with the other classes

$$\boldsymbol{\mu}^l_{K+1} = \mathbf{x}^t; \quad \sigma^l_{K+1} = \kappa\|\mathbf{x}^t - \boldsymbol{\mu}^c_{nrS}\|, \qquad l = c. \tag{25}$$

iii) *Minimum Overlapping With the Interclass:* If the intraclass and interclass distance ratio is in range 1–1.5, then the sample has minimum overlapping with the other class

$$\begin{aligned}\boldsymbol{\mu}^l_{K+1} &= \mathbf{x}^t + \zeta(\boldsymbol{\mu}^c_{nrS} - \boldsymbol{\mu}^l_{nrI})\\ \sigma^l_{K+1} &= \kappa\|\boldsymbol{\mu}^c_{K+1} - \boldsymbol{\mu}^c_{nrS}\|, \qquad l = c\end{aligned} \tag{26}$$

where $\zeta$ is center shift factor. In our simulation studies, it is selected in the interval of [0.01–0.1].

iv) *Significant Overlapping With the Interclass:* If the intraclass and interclass distance ratio is more than 1.5, then the sample has significant overlapping with the other class

$$\begin{aligned}\boldsymbol{\mu}^l_{K+1} &= \mathbf{x}^t - \zeta(\boldsymbol{\mu}^l_{nrI} - \mathbf{x}^t)\\ \sigma^l_{K+1} &= \kappa\|\boldsymbol{\mu}^c_{K+1} - \boldsymbol{\mu}^l_{nrI}\|, \qquad l = c.\end{aligned} \tag{27}$$

The above-mentioned center and width determination conditions help in minimizing the misclassification in McRBFN.

When a neuron is added to McRBFN, the output weights are estimated using PBL with past knowledge stored in the network as pseudosamples, i.e., centers and associated class labels as pseudosamples.

When a new neuron is added to the cognitive component, the size of matrix $\mathbf{A}$ as defined in (10) is increased from $K \times K$ to $(K+1) \times (K+1)$

$$\mathbf{A}^t_{(K+1)\times(K+1)} = \left[\begin{array}{c|c} \mathbf{A}^{t-1}_{K\times K} + \left(\mathbf{h}^t\right)^T\mathbf{h}^t & \mathbf{a}^T_{K+1} \\ \hline \mathbf{a}_{K+1} & a_{K+1,K+1}\end{array}\right] \tag{28}$$

where $\mathbf{h}^t = [h^t_1, h^t_2, \ldots, h^t_K]$ is a vector of the existing $K$ hidden neurons response for new ($t$th) training sample. In sequential learning, the samples are discarded after learning, but the information present in the past samples are stored in the network. The centers of neuron provides the distribution of past samples in feature space. These centers $\{\boldsymbol{\mu}^l_1, \ldots, \boldsymbol{\mu}^l_K\}$ can be used as pseudosamples to capture the effect of past samples. Hence, existing hidden neurons are used as pseudosamples to calculate $\mathbf{a}_{K+1}$ and $a_{K+1,K+1}$ terms. $\mathbf{a}_{K+1} \in \Re^{1\times K}$ is assigned as

$$a_{K+1,p} = \sum_{i=1}^{K+1} h^i_{K+1}h^i_p, \qquad p = 1, \ldots, K \tag{29}$$

and $a_{K+1,K+1} \in \Re^+$ value is assigned as

$$a_{K+1,K+1} = \sum_{i=1}^{K+1} h^i_{K+1}h^i_{K+1}. \tag{30}$$

The size of matrix $\mathbf{B}$ is increased from $K \times n$ to $(K+1) \times n$

$$\mathbf{B}^t_{(K+1)\times n} = \left[\begin{array}{c} \mathbf{B}^{t-1}_{K\times n} + \left(\mathbf{h}^t\right)^T\left(\mathbf{y}^t\right)^T \\ \mathbf{b}_{K+1}\end{array}\right] \tag{31}$$

and $\mathbf{b}_{K+1} \in \Re^{1\times n}$ is a row vector assigned as

$$b_{K+1,j} = \sum_{i=1}^{K+1} h^i_{K+1}\tilde{y}^i_j, \qquad j = 1, \ldots, n \tag{32}$$

where $\tilde{y}^i$ is the pseudooutput for the $i$th pseudosample or hidden neuron ($\boldsymbol{\mu}^l_i$) given as

$$\tilde{y}^i_j = \begin{cases} 1, & \text{if} \quad l = j \\ -1, & \text{otherwise}\end{cases} \quad j = 1, \ldots, n. \tag{33}$$

Finally, the output weights are estimated as

$$\left[\begin{array}{c}\mathbf{W}^t_K \\ \mathbf{w}^t_{K+1}\end{array}\right] = \left(\mathbf{A}^t_{(K+1)\times(K+1)}\right)^{-1}\mathbf{B}^t_{(K+1)\times n} \tag{34}$$

where $\mathbf{W}^t_K$ is the output weight matrix for $K$ hidden neurons, and $\mathbf{w}^t_{K+1}$ is the vector of output weights for new hidden neuron after learning from $t$th sample. The inverse of a matrix $\mathbf{A}^t_{(K+1)\times(K+1)}$ is calculated recursively using matrix identities as

$$\begin{aligned}\left(\mathbf{A}^t_{(K+1)\times(K+1)}\right)^{-1} &= \left[\begin{array}{c|c}\left(\mathbf{A}^t_{K\times K}\right)^{-1} & 0 \\ \hline 0 & 0\end{array}\right]\\ &\quad + \frac{1}{\Delta}\left[\begin{array}{c}\left(\mathbf{A}^t_{K\times K}\right)^{-1}\mathbf{a}^T \\ 1\end{array}\right]\left[\begin{array}{c}\left(\mathbf{A}^t_{K\times K}\right)^{-1}\mathbf{a}^T \\ 1\end{array}\right]^T\end{aligned} \tag{35}$$

where

$$\begin{aligned}\mathbf{A}^t_{K\times K} &= \mathbf{A}^{t-1} + (\mathbf{h}^t)^T\mathbf{h}^t\\ \Delta &= a_{K+1,K+1} - \mathbf{a}_{K+1}\left(\mathbf{A}^{t-1}_{K\times K}\right)^{-1}\mathbf{a}^T_{K+1}\end{aligned}$$

and $\left(\mathbf{A}^t_{K\times K}\right)^{-1}$ is calculated as

$$\left(\mathbf{A}^t_{K\times K}\right)^{-1} = \left(\mathbf{A}^{t-1}\right)^{-1} - \frac{\left(\mathbf{A}^{t-1}\right)^{-1}\left(\mathbf{h}^t\right)^T\mathbf{h}^t\left(\mathbf{A}^{t-1}\right)^{-1}}{1 + \mathbf{h}^t\left(\mathbf{A}^{t-1}\right)^{-1}\left(\mathbf{h}^t\right)^T}. \tag{36}$$

After calculating inverse of matrix in (34) using (35) and (36), the resultant equations are

$$\begin{aligned}\mathbf{W}^t_K &= \left[\mathbf{I}_{K\times K} + \frac{\left(\mathbf{A}^{t-1}_{K\times K}\right)^{-1}\mathbf{a}^T_{K+1}\mathbf{a}_{K+1}}{\Delta}\right]\\ &\quad\times\left[\mathbf{W}^{t-1}_K + \left(\mathbf{A}^{t-1}_{K\times K}\right)^{-1}\left(\mathbf{h}^t\right)^T\left(\mathbf{y}^t\right)^T\right]\\ &\quad - \frac{\left(\mathbf{A}^{t-1}_{K\times K}\right)^{-1}\mathbf{a}^T_{K+1}\mathbf{b}_{K+1}}{\Delta}\end{aligned} \tag{37}$$

$$\mathbf{w}^t_{K+1} = -\frac{\mathbf{a}_{K+1}\left(\mathbf{W}^{t-1}_K + \left(\mathbf{A}^{t-1}_{K\times K}\right)^{-1}\left(\mathbf{h}^t\right)^T\left(\mathbf{y}^t\right)^T\right)}{\Delta} + \frac{\mathbf{b}_{K+1}}{\Delta}. \tag{38}$$

*c) Parameters update strategy:* The current ($t$th) training sample is used to update the output weights of the cognitive component ($\mathbf{W}_K = [\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_K]^T$), if the following criterion is satisfied:

$$c^t == \widehat{c}^t \quad \text{and} \quad E^t \geq \beta_u \tag{39}$$

where $\beta_u$ is the self-adaptive metacognitive parameter update threshold. For the selection of $\beta_u$ value, one may refer to [14].

The $\beta_u$ is adapted based on the maximum hinge error as

$$\beta_u := \delta\beta_u + (1-\delta)E^t \tag{40}$$

where $\delta$ is the slope that controls the rate of self-adaption of parameter update and is set close to one.

When a sample is used to update the output weight parameters, the PBL algorithm updates the output weight parameters as follows:

$$\frac{\partial J(\mathbf{W}_K^t)}{\partial w_{pj}} = \frac{\partial J(\mathbf{W}_K^t)}{\partial w_{pj}} + \frac{\partial J_t(\mathbf{W}_K^t)}{\partial w_{pj}} = 0$$

where

$$p = 1, \ldots, K; \qquad j = 1, \ldots, n. \tag{41}$$

Equating the first partial derivative to zero and re-arranging (41), we get

$$\left(\mathbf{A}^{t-1} + \left(\mathbf{h}^t\right)^T \mathbf{h}^t\right) \mathbf{W}_K^t - \left(\mathbf{B}^{t-1} + \left(\mathbf{h}^t\right)^T \left(\mathbf{y}^t\right)^T\right) = 0. \tag{42}$$

By substituting $\mathbf{B}^{t-1} = \mathbf{A}^{t-1}\mathbf{W}_K^{t-1}$ and $\mathbf{A}^t = \mathbf{A}^{t-1} + (\mathbf{h}^t)^T\mathbf{h}^t$ and adding or subtracting the term $\left(\mathbf{h}^t\right)^T \mathbf{h}^t \mathbf{W}_K^{t-1}$ on both sides, (42) is reduced to

$$\mathbf{W}_K^t = \left(\mathbf{A}^t\right)^{-1} \left(\mathbf{A}^t \mathbf{W}_K^{t-1} + \left(\mathbf{h}^t\right)^T \left(\left(\mathbf{y}^t\right)^T - \mathbf{h}^t \mathbf{W}_K^{t-1}\right)\right). \tag{43}$$

Finally, the output weights are updated as

$$\mathbf{W}_K^t = \mathbf{W}_K^{t-1} + \left(\mathbf{A}^t\right)^{-1} \left(\mathbf{h}^t\right)^T \left(\mathbf{e}^t\right)^T \tag{44}$$

where $\mathbf{e}^t$ is the hinge loss error for $t$th sample obtained from (15). From (15) and (44), we can see that only the nonzero hinge loss error producing output neurons connections weights are updated. Since, the parameter update employs hinge error, the resultant network approximates the posterior probability very well.

*d) Neuron pruning strategy:* If the contribution of a neuron in the same class is lesser than a pruning threshold $\beta_p$ for $N_p$ consequent samples in the same class $l$, then that neuron is insignificant and can be removed from the network. The contribution of the $k$th neuron is defined as

$$r_k^l = \frac{h\left(\mathbf{x}^i, \boldsymbol{\mu}_k^l\right)}{\max_j \; h\left(\mathbf{x}^i, \boldsymbol{\mu}_j^l\right)} \tag{45}$$

where $\beta_p$ is the metacognitive neuron pruning threshold. If $\beta_p$ is chosen closer to zero, then pruning occurs seldom and all the added neurons will remain in the network irrespective of their contribution to the network output. If $\beta_p$ is chosen closer to one, then pruning occurs frequently, resulting in oscillations and insufficient neurons to capture the distribution from stream of training samples. When a neuron is pruned from the network, the dimensionality of the $\mathbf{A}^t$ and $\mathbf{B}^t$ is reduced by removing the respective rows/columns corresponding to the pruned neuron.

TABLE I
SPECIFICATION OF BENCHMARK BINARY AND MULTICLASS DATASETS

| Datasets | No. of Features | No. of Classes | No. of Samples | | I.F | |
|---|---|---|---|---|---|---|
| | | | Training | Testing | Training | Testing |
| LD | 6 | 2 | 200 | 145 | 0.17 | 0.14 |
| PIMA | 8 | 2 | 400 | 368 | 0.22 | 0.39 |
| BC | 9 | 2 | 300 | 383 | 0.26 | 0.33 |
| HEART | 13 | 2 | 70 | 200 | 0.14 | 0.1 |
| ION | 34 | 2 | 100 | 251 | 0.28 | 0.28 |
| IS | 19 | 7 | 210 | 2100 | 0 | 0 |
| IRIS | 4 | 3 | 45 | 105 | 0 | 0 |
| WINE | 13 | 3 | 60 | 118 | 0 | 0.29 |
| VC | 18 | 4 | 424[a] | 422 | 0.1 | 0.12 |
| GI | 9 | 6 | 109[a] | 105 | 0.68 | 0.77 |
| AE | 5 | 4 | 62 | 137 | 0.1 | 0.33 |
| GCM | 98 | 14 | 144 | 46 | 0.22 | 0.39 |
| LETTER | 16 | 26 | 13333 | 6667 | 0.06 | 0.1 |
| SI | 6 | 9 | 9108 | 262144 | 0 | 0.87 |

[a] As suggested in [2], the samples are repeated three times.

*e) Sample reserve strategy:* If the new training sample does not satisfy either the deletion or the neuron growth or the cognitive component parameters update criterion or neuron pruning, then the current sample is pushed to the rear of the training sequence. Since McRBFN modifies the strategies based on the current sample knowledge, these samples may be used in later stage.

Ideally, training process stops when no further sample is available in data stream. However, in real-time, training stops when samples in the reserve remains same.

In PBL-McRBFN, sample delete strategy addresses the what-to-learn by deleting insignificant samples from training dataset, neuron growth strategy, parameters update strategy, and neuron pruning strategy address the how-to-learn efficiently by which the cognitive component learns from the samples, and self-adaptive nature of metacognitive thresholds in addition to the sample reserve strategy address the when-to-learn by presenting the samples in the learning process according to the knowledge present in the sample.

## III. PERFORMANCE EVALUATION OF PBL-MCRBFN CLASSIFIER

In this section, we first present the performance comparison of the proposed PBL-McRBFN with the best-performing sequential learning algorithm reported in the literature McNN [14], batch ELM [9], and standard Support Vector Machine (SVM) [30] on real-world benchmark binary and multicategory classification datasets from the UCI machine learning repository [26]. Next, we use the PBL-McRBFN to detect AD from a stream of MRI scans.

### A. Performance Comparison on Benchmark Datasets

*1) Benchmark Datasets:* To extensively verify the performance of the proposed algorithm, we have chosen datasets with small and large number of samples, low- and high-dimensional features, and balanced and unbalanced datasets in both binary and multiclassification problems. The detailed specifications of five binary and nine multiclassification datasets are given in Table I. Note that the datasets are taken from UCI machine learning repository, except for satellite imaging [24], global cancer map using microarray gene expression [31], and acoustic emission [32] datasets. The sample imbalance in training and testing is measured using I.F and is defined as

$$\text{I.F} = 1 - \frac{n}{N} \times \min_{j=1,\ldots,n} N_j \tag{46}$$

where $N_j$ is the total number of training samples in class $j$ and $N = \sum_{j=1}^{n} N_j$.

For efficient comparison, we present them under the following categories as described below.

*a) Binary class datasets:* All the considered binary class datasets have high sample imbalance and are grouped into two categories.

i) *Low Dimensional:* Liver disorders (LD), PIMA Indian diabetes (PIMA), and breast cancer (BC) are having low-dimensional features with relatively smaller number of training samples.

ii) *High Dimensional:* Heart disease (HEART) and ionosphere (ION) datasets are having smaller number of training samples with high-dimensional features.

*b) Multiclass datasets:* Considered nine multiclass datasets are grouped into three categories.

i) *Well Balanced:* Image segmentation (IS), iris classification (IRIS), and wine determination (WINE) datasets have equal number of training samples per class. These datasets are having varying number of features and training and testing samples.

ii) *Imbalanced:* Vehicle classification (VC), glass identification (GI), and acoustic emission classification (AE) datasets have lower dimensional features and highly imbalanced training samples. The global cancer mapping (GCM) using microarray gene expression is having high-dimensional features with high sample imbalance.

iii) *Large Number of Samples:* Letter recognition (LETTER) and satellite image (SI) classification datasets have relatively large number of samples and classes.

*2) Simulation Environment:* For this performance comparison study, experiments are conducted for the PBL-McRBFN, McNN, ELM, and SVM classifiers on all the datasets in MATLAB 2011 on a desktop PC with Intel Core 2 Duo, 2.66-GHz CPU and 3-GB RAM. The tunable parameters of PBL-McRBFN and McNN are chosen using cross-validation on the training datasets. For ELM classifier [3], the number of hidden neurons are obtained using the constructive-destructive procedure presented in [33]. The simulations for batch SVM with Gaussian kernels are carried out using the LIBSVM package in C [34]. For SVM classifier, the parameters $(c, \gamma)$ are optimized using grid search technique.

TABLE II
PERFORMANCE COMPARISON OF PBL-McRBFN WITH McNN, ELM, AND SVM ON BINARY CLASS DATASETS

| Data set | Classifier | No. of Neurons | No. of Samples Used | Testing | |
|---|---|---|---|---|---|
| | | | | $\eta_o$ | $\eta_a$ |
| LD | SVM | 141[a] | 200 | 71.03 | 70.21(4) |
| | ELM | 100 | 200 | 72.41 | 71.41(3) |
| | McNN | 68 | 110 | 73.79 | 71.60(2) |
| | PBL-McRBFN | 87 | 116 | 73.10 | 72.63(1) |
| PIMA | SVM | 221[a] | 400 | 77.45 | 76.43(3) |
| | ELM | 100 | 400 | 76.63 | 75.25(4) |
| | McNN | 76 | 193 | 80.16 | 77.31(1) |
| | PBL-McRBFN | 100 | 162 | 79.62 | 76.67(2) |
| BC | SVM | 24[a] | 300 | 96.61 | 97.06(3) |
| | ELM | 66 | 300 | 96.35 | 96.48(4) |
| | McNN | 9 | 27 | 97.39 | 97.85(1.5) |
| | PBL-McRBFN | 13 | 45 | 97.39 | 97.85(1.5) |
| HEART | SVM | 42[a] | 70 | 75.5 | 75.10(4) |
| | ELM | 36 | 70 | 76.50 | 75.91(3) |
| | McNN | 26 | 46 | 80.50 | 79.65(2) |
| | PBL-McRBFN | 20 | 69 | 81.50 | 81.47(1) |
| ION | SVM | 43[a] | 100 | 91.24 | 88.51(3) |
| | ELM | 32 | 100 | 89.64 | 87.52(4) |
| | McNN | 20 | 39 | 95.62 | 95.60(2) |
| | PBL-McRBFN | 18 | 58 | 96.41 | 96.47(1) |

[a] Number of support vectors.

*3) Performance Measures:* The overall and average classification efficiencies are used as quantitative evaluation measures in this paper. The confusion matrix $Q$ is used to obtain the class-level performance and global performance of the various classifiers. Class-level performance is measured by the percentage classification $(\eta_j)$ which is defined as

$$\eta_j = \frac{q_{jj}}{N_j} \times 100\% \tag{47}$$

where $q_{jj}$ is the total number of correctly classified samples in the class $j$. The global measures used in the evaluation are the average per-class classification accuracy $(\eta_a)$ and the overall classification accuracy $(\eta_o)$ defined as

$$\eta_a = \frac{1}{n}\sum_{j=1}^{n} \eta_j, \qquad \eta_o = \frac{\sum_{j=1}^{n} q_{jj}}{N} \times 100\%. \tag{48}$$

*4) Performance Comparison:*

*a) Binary class datasets:* The performance measures such as overall $(\eta_o)$, average $(\eta_a)$ testing efficiencies, number of neurons, and samples used for PBL-McRBFN, McNN, ELM, and SVM classifiers on all the five binary class datasets are reported in Table II. From the performance comparison results in Table II, one can see that in case of low-dimensional LD and PIMA datasets, the proposed PBL-McRBFN uses fewer samples for training and achieves significantly better generalization performance approximately 1% improvement over McNN, 1%–2% improvement over ELM and SVM

TABLE III
PERFORMANCE COMPARISON OF PBL-MCRBFN WITH MCNN, ELM, AND SVM ON MULTICATEGORY DATASETS

| Data set | Classifier | No. of Neurons | No. of Samples Used | Testing $\eta_o$ | Testing $\eta_a$ |
|---|---|---|---|---|---|
| IS | SVM | 127[a] | 210 | 91.38 | 91.38(3) |
| | ELM | 49 | 210 | 90.23 | 90.23(4) |
| | McNN | 49 | 93 | 93.38 | 93.38(2) |
| | PBL-McRBFN | 50 | 89 | 94.19 | 94.19(1) |
| IRIS | SVM | 13[a] | 45 | 96.19 | 96.19(3.5) |
| | ELM | 10 | 45 | 96.19 | 96.19(3.5) |
| | McNN | 5 | 22 | 97.14 | 97.14(2) |
| | PBL-McRBFN | 6 | 20 | 98.10 | 98.10(1) |
| WINE | SVM | 36[a] | 60 | 97.46 | 98.04(3.5) |
| | ELM | 10 | 60 | 97.46 | 98.04(3.5) |
| | McNN | 9 | 27 | 98.3 | 98.49(2) |
| | PBL-McRBFN | 11 | 29 | 98.31 | 98.69(1) |
| VC | SVM | 340[a] | 620 | 70.62 | 68.51(4) |
| | ELM | 150 | 620 | 77.01 | 77.59(3) |
| | McNN | 146 | 359 | 77.72 | 78.72(2) |
| | PBL-McRBFN | 175 | 318 | 78.91 | 79.09(1) |
| GI | SVM | 183[a] | 336 | 70.47 | 75.61(4) |
| | ELM | 80 | 336 | 81.31 | 87.43(2) |
| | McNN | 73 | 117 | 85.71 | 87.03(3) |
| | PBL-McRBFN | 71 | 115 | 84.76 | 92.72(1) |
| AE | SVM | 22[a] | 62 | 98.54 | 97.95(4) |
| | ELM | 10 | 62 | 99.27 | 98.91(2) |
| | McNN | 5 | 20 | 99.27 | 98.91(2) |
| | PBL-McRBFN | 5 | 9 | 99.27 | 98.91(2) |
| GCM | SVM | 107[a] | 98 | 76.08 | 82.62 |
| | ELM | 55 | 98 | 76.08 | 73.57 |
| | PBL-McRBFN | 72 | 109 | 93.47 | 91.67 |
| LETTER | SVM | 4429[a] | 13333 | 92.94 | - |
| | ELM | - | 13333 | 93.51 | - |
| | PBL-McRBFN | 1654 | 3346 | 95.42 | 95.44 |
| SI | SVM | 1298[a] | 9108 | 92.21 | 90.14 |
| | ELM | 1500 | 9108 | 88.39 | - |
| | PBL-McRBFN | 1274 | 2445 | 90.14 | 91.19 |

[a] Number of support vectors.

with less number of neurons. In case of simple BC dataset, PBL-McRBFN uses fewer samples for training and achieves slightly better generalization performance approximately 1% improvement over ELM and SVM with less number of neurons and same performance as McNN. In case of high-dimensional HEART and ION datasets, PBL-McRBFN uses fewer samples for training and achieves better generalization performance 1%–2% improvement over McNN, 6%–9% improvement over SVM and ELM. The overlapping conditions and class-specific criterion in learning strategies of PBL-McRBFN helps in capturing the knowledge accurately in case of high sample imbalance problems.

*b) Multicategory datasets:* The overall ($\eta_o$) and average ($\eta_a$) testing efficiencies, number of neurons, and samples used for PBL-McRBFN, ELM, and SVM classifiers on all the nine multicategory datasets are reported in Table III. From Table III, we can see that PBL-McRBFN performs significantly better than the ELM and SVM on all the nine multicategory datasets. In case of well-balanced IS, IRIS, and WINE datasets, PBL-McRBFN uses only 42%–50% training samples to achieve better generalization performance approximately 2%–3% improvement over McNN, SVM, and ELM with the less number of neurons. Metacognitive sample deletion criteria helps in removing redundant samples from the training set and thereby improves the generalization performance. For highly unbalanced datasets, one can see that the proposed PBL-McRBFN is able to achieve significantly better performance than the other classifiers. In case of VC and GI datasets, PBL-McRBFN uses fewer samples and achieves better generalization performance approximately 1%–5% improvement over McNN and ELM, and 7%–12% improvement over SVM. In case of low-dimensional AE dataset, PBL-McRBFN achieves slightly better generalization performance 1% improvement over SVM, similar to McNN and ELM. In case of high-dimensional GCM dataset, PBL-McRBFN achieves significantly better generalization performance approximately 13% improvement over SVM and ELM. In case of large samples LETTER and SI datasets, the generalization performance of PBL-McRBFN is better than ELM and SVM by approximately 2%–3% and 3% using less number of training samples and neurons. Due to computational intensive EKF for parameter update, McNN experiences memory problem for large problems like GCM, LETTER, and SI. Hence, the results for McNN in these problem are not presented here.

From Tables II and III, we can say that the proposed PBL-McRBFN improves generalization performance under wide range of sample imbalance datasets.

*5) Statistical Comparison:* To compare the performance of the proposed PBL-McRBFN classifier with McNN, ELM, and SVM classifiers on various benchmark datasets, we employ a nonparametric Friedman test followed by the Benforroni-Dunn test as described in [27]. The Friedman test compares whether the mean of individual experimental condition differs significantly from the aggregate mean across all conditions. If the measure $F$-score is greater than the $F$-statistic at 95% confidence level, then one rejects the equality of mean hypothesis, i.e., the classifiers used in our paper perform similarly on different dataset. If nonparametric Friedman test rejects the equality hypothesis, then pairwise post hoc should be conducted to test in which the mean is different from others. In our paper, we have used four different classifiers ranks based on average testing efficiency on 11 different datasets from Table I.

The $F$-score obtained using nonparametric Friedman test is 42.75, which is greater than the $F$-statistic at 95% confidence level ($F_{3,30,0.025}$ is 3.542), i.e., $42.75 > 3.542$. Hence, one can reject mean equality hypothesis at a confidence level of 95%.

Next, we conduct a pairwise comparison using a Benforroni-Dunn test to highlight the performance significance of PBL-McRBFN classifier with respect to other classifiers. Here, the proposed PBL-McRBFN classifier is used as a control. The critical difference (CD) is calculated is 1.317 at 95% confidence level. From Tables II and III, we can obtain the average ranks for all four classifiers and are PBL-McRBFN:

1.25, McNN: 1.86, ELM: 3.18, and SVM: 3.72. The difference in average rank between the proposed PBL-McRBFN classifier and the other three classifiers are PBL-McRBFN-McNN: 0.61, PBL-McRBFN-ELM: 1.93, and PBL-McRBFN-SVM: 2.47. Note that difference in average rank for PBL-McRBFN-ELM and PBL-McRBFN-SVM pairs is greater than CD at 95% confidence level, i.e., $1.93 > 1.317$ and $2.47 > 1.317$. The difference in average rank for PBL-McRBFN-McNN pair is less than CD at 95% confidence level, i.e., $0.61 < 1.317$. Hence, we can say that PBL-McRBFN performs slightly better than the McNN classifier and significantly better than ELM and SVM classifiers with a confidence level of 95%.

### *B. AD Detection Using PBL-McRBFN*

AD is one of the most common causes of dementia. AD is a progressive, neurodegenerative disorder that leads to memory loss, problems in learning, confusion, and poor judgment. Early detection of AD using noninvasive methods (brain imaging) plays a major role in providing treatment that may slow down its progress. MRI is the most important brain imaging procedure that provides accurate information about the shape and volume of the brain. The problem of early detection of AD using MRI can be formulated as binary classification and can be solved using machine learning techniques [35].

*1) Related Works:* The studies of analyzing MRI scans can be categorized into two classes: region-of-interest (ROI) methods [36] and whole-brain morphometric methods [37]. In ROI methods, the volumetric measurements of specific brain regions are used to detect AD. Studies have shown that the tissue loss in the hippocampus and the entorhinal cortex could be indicators for early AD. Major shortcomings in the use of the ROI methods are dependent on tracer expertise and are erroneous. To overcome these shortcomings, whole-brain morphometric methods have been employed for accurate AD detection. Voxel-based morphometry (VBM) is a completely automatic image analysis approach for identifying the amount of gray matter or white matter differences between the normal persons and AD patients [38]. The steps involved in the VBM to identify significant differences between the groups of images are spatial normalization, segmentation, smoothing, and statistical analysis [39]. The VBM analysis is used in this paper to identify the regional differences in gray matter among groups of persons and to extract morphometric features from MRI scans.

*2) AD Datasets:* OASIS database [28] has cross-sectional collection of 416 persons covering the adult life-span including persons with AD in an early-stage. The data includes 218 persons aged from 18 to 59 years and 198 from 60 to 96 years. Among 198 elderly persons considered in this paper, 98 had no AD, i.e., clinical dementia rating (CDR) is zero. Seventy persons were diagnosed as very mild AD (CDR = 0.5), 28 persons were diagnosed as mild AD (CDR = 1), and two persons as moderate AD (CDR = 2). ADNI database [29] has a cross-sectional collection of 422 elderly persons aged from 55 to 91 years. Among 422 elderly persons considered in this database, 226 were diagnosed as normal (CDR = 0), 95 persons were diagnosed as very mild AD (CDR = 0.5), and 101 persons were diagnosed as mild AD (CDR = 1).

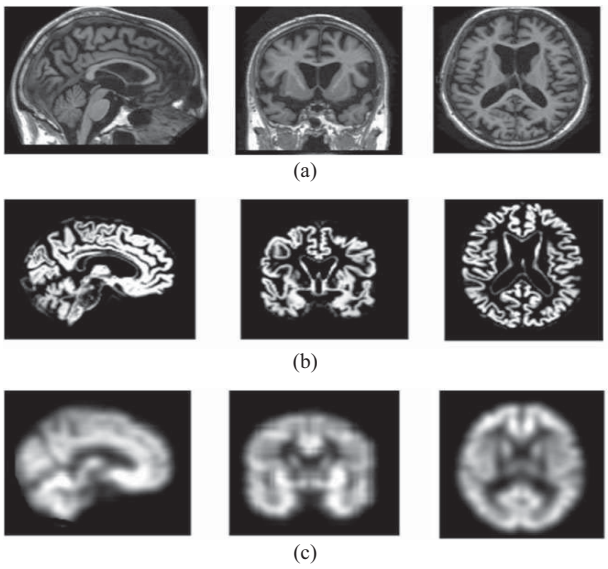


Fig. 2. Results of the unified segmentation and smoothing steps performed on MRI of an AD patient. From right: sagittal, coronal, and axial views. (a) MRI of an AD patient. (b) Segmented gray matter tissue class. (c) Smoothed gray matter image.

*3) Feature Extraction Using VBM:* The VBM is a voxelwise comparison of local tissue volume of gray matter within or among groups of persons using MRI scans. Here, MRI scans undergo various preprocessing steps before the voxelwise parametric tests are carried out on them.

The steps involved in VBM analysis are unified segmentation, smoothing, and statistical testing, in the order as in unified segmentation model proposed by [38]. The unified segmentation step is a generative modeling approach, in which tissue segmentation, bias correction, and image registration are combined in a single model. The unified segmentation framework combines deformable tissue probability maps with a Gaussian mixture model. The MR brain images of both the AD patients and healthy persons are segmented into gray matter tissue class. The segmented and normalized gray matter images are then smoothed by convolving with an isotropic Gaussian kernel. VBM was performed using the statistical parametric map (SPM) software package [40]. A 10-mm full-width at half-maximum Gaussian kernel was employed. Fig. 2 shows three planar views (sagittal, coronal, and axial) of the original images and images after every stages in VBM analysis. The maximum intensity projections of the significant voxels obtained for OASIS dataset in sagittal, coronal, and axial views are shown in Fig. 3.

*4) Performance Comparison:*

*a) Individual OASIS and ADNI datasets:* Significant voxels extracted from VBM analysis are taken as features. The VBM extracted 19 789 features on OASIS dataset and 23 797 features on ADNI dataset. For classification study, 10 random trials of features sets are used as input to the PBL-McRBFN classifier. In each trial, 50% of the samples are randomly chosen for training set, the rest as testing set. Table IV

TABLE IV
CLASSIFICATION PERFORMANCE OF PBL-MCRBFN ON AD DATASET FROM OASIS AND ADNI

| Data set | No. of Features | No. of Samples Used | | No. of Neurons | | Testing | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | Accuracy (%) | | | $F$-Measure | |
| | | Mean | Deviation | Mean | Deviation | Mean | Deviation | Best | Mean | Deviation |
| OASIS | 19 879 | 81.8 | 9.49 | 42.8 | 3.11 | 75.75 | 1.01 | 76.77 | 0.74 | 0.02 |
| ADNI | 23 797 | 136.6 | 20.10 | 63.6 | 6.84 | 85.49 | 0.98 | 86.26 | 0.84 | 0.01 |

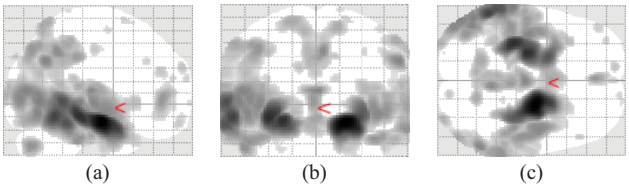


Fig. 3. Maximum intensity projections of the significant areas with increased gray matter density in the healthy persons relative to the AD patients. (a) Sagittal view. (b) Coronal view. (c) Axial view.

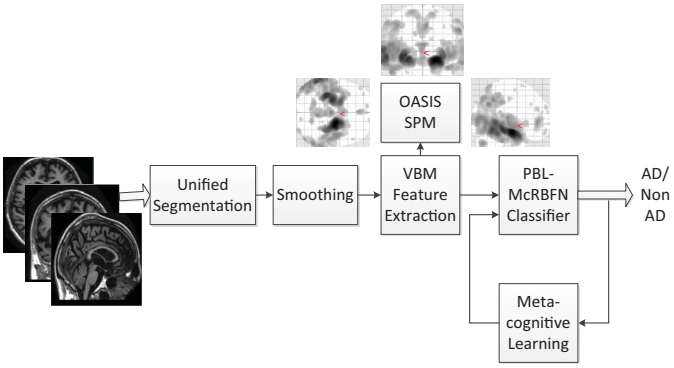


Fig. 4. Schematic diagram of PBL-McRBFN application on extracted ADNI dataset features from OASIS dataset SPM.

presents the mean results obtained from 10 random trials on both OASIS and ADNI datasets. The best PBL-McRBFN classification accuracy on OASIS dataset is 76.77%. Compared with the results reported in [41] and [42], the PBL-McRBFN classification accuracy on 19 879 features set is 10% more than the PCA-SVM approach (66.98%) [41], 7% more than the IPCA-SVM approach (69.7%) [41], and 14% more than the ICA-SVM approach (62.8%) [42].

The best PBL-McRBFN classification accuracy on ADNI dataset is 86.26%. Compared with the results reported in [43] and [44], the PBL-McRBFN classification accuracy on complete 23 797 features set is 8% more than the ICA-SVM approach on ADNI whole-brain images (78.4%) [43], 4% more than the ICA-SVM approach on ADNI gray matter images (82.4%) [43], 5% more than the SVM approach on ANDI images with hippocampal volume features (81%) [44], and 4% more than the SVM approach on ADNI images with cortical thickness features (82%) [44].

*b) Nonstationary ADNI dataset from OASIS SPM:* From the description of OASIS and ADNI datasets, we can see that these datasets are collected from the different demographic people with different geographic locations. Hence, the databases represent variation in the data distribution. In this, we study the performance of proposed classifier under varying distribution. For this purpose, SPM obtained using OASIS database is used to extract the VBM features from ADNI database. The extracted 19 879 features from ADNI samples are tested with best classifier developed using OASIS database. The schematic diagram of cross-sectional study is shown in Fig. 4. Such cross-sectional study will prevent computationally intensive VBM feature extraction and unify the diagnosis mechanism. If all the 422 samples from the ADNI dataset are used as testing samples to the best PBL-McRBFN classifier that is trained on OASIS database, then the testing accuracy is 62.3%. If 25% of samples from the ADNI database are used to adapt the PBL-McRBFN classifier using metacognitive principles and tested on the remaining 75% of samples, then the testing accuracy is 77.4%. From the above results, we can infer that best PBL-McRBFN classifier trained on one data distribution (OASIS database) with adaptation using fewer samples from ADNI achieves significant testing accuracy on unseen samples.

## IV. CONCLUSION

In this paper, we presented a sequential learning algorithm using human metacognitive principles. The metacognitive component selects appropriate learning strategy for the cognitive RBF classifier to achieve better generalization performance with minimal computational effort. The metacognitive component adapts the learning process appropriately by implementing self-regulation, and hence it decides what-to-learn, when-to-learn, and how-to-learn efficiently. In addition, PBL accurately estimates the output weight by direct minimization of hinge loss error and the overlapping conditions present in neuron growth strategy helps in proper initialization of new hidden neuron parameters, which minimizes the misclassification error. The performance of the proposed PBL-McRBFN classifier was evaluated using the benchmark binary and multiclassification datasets and also a practical problem of AD detection. The statistical nonparametric Friedman test based on 11 benchmark datasets clearly indicated that the proposed PBL-McRBFN classifier achieves slightly better performance than McNN, and significantly better performance than ELM and SVM classifiers. In practical AD detection problem, the proposed PBL-McRBFN classifier showed 7%–14% improvement over reported results on OASIS and 4%–8% improvement over ADNI databases. Based on simulation studies conducted on benchmark and practical problems, we can infer that human metacognitive principles in learning algorithm improves the performance significantly. Finally, the performance evaluation on ADNI database with PBL-McRBFN classifier trained on

OASIS database shows that the proposed PBL-McRBFN can also achieve significant results on the nonstationary problems.

## ACKNOWLEDGMENT

The authors would like to thank the anonymous reviewers for their valuable comments and suggestions, which helped to improve the quality of this paper. They would also like to thank Dr. B. S. Mahanand, Sri Jayachamarajendra College of Engineering, Mysore, India, for guidance and help in voxel-based morphometry feature extraction technique.

## REFERENCES

[1] E. Y. Cheu, C. Quek, and S. K. Ng, "ARPOP: An appetitive reward-based pseudo-outer-product neural fuzzy inference system inspired from the operant conditioning of feeding behavior in Aplysia," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 23, no. 2, pp. 317–329, Feb. 2012.

[2] S. Suresh, N. Sundararajan, and P. Saratchandran, "A sequential multi-category classifier using radial basis function networks," *Neurocomputing*, vol. 71, no. 1, pp. 1345–1358, 2008.

[3] G.-B. Huang, H. Zhou, X. Ding, and R. Zhang, "Extreme learning machine for regression and multiclass classification," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 42, no. 2, pp. 513–529, Apr. 2012.

[4] T. Xie, H. Yu, J. Hewlett, P. Rozycki, and B. Wilamowski, "Fast and efficient second-order method for training radial basis function networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 23, no. 4, pp. 609–619, Apr. 2012.

[5] J. C. Platt, "A resource allocation network for function interpolation," *Neural Comput.*, vol. 3, no. 2, pp. 213–225, 1991.

[6] L. Yingwei, N. Sundararajan, and P. Saratchandran, "A sequential learning scheme for function approximation using minimal radial basis function neural networks," *Neural Comput.*, vol. 9, no. 2, pp. 461–478, 1997.

[7] G.-B. Huang, P. Saratchandran, and N. Sundararajan, "An efficient sequential learning algorithm for growing and pruning RBF (GAP-RBF) networks," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 34, no. 6, pp. 2284–2292, Dec. 2004.

[8] N.-Y. Liang, G.-B. Huang, P. Saratchandran, and N. Sundararajan, "A fast and accurate online sequential learning algorithm for feedforward networks," *IEEE Trans. Neural Netw.*, vol. 17, no. 6, pp. 1411–1423, Nov. 2006.

[9] S. Suresh, R. V. Babu, and H. J. Kim, "No-reference image quality assessment using modified extreme learning machine classifier," *Appl. Soft Comput.*, vol. 9, no. 2, pp. 541–552, 2009.

[10] W. P. Rivers, "Autonomy at all costs: An ethnography of metacognitive self-assessment and self-management among experienced language learners," *Mod. Lang. J.*, vol. 85, no. 2, pp. 279–290, 2001.

[11] R. Isaacson and F. Fujita, "Metacognitive knowledge monitoring and self-regulated learning: Academic success and reflections on learning," *J. Scholarship Teach. Learn.*, vol. 6, no. 1, pp. 39–55, 2006.

[12] S. Suresh, K. Dong, and H. J. Kim, "A sequential learning algorithm for self-adaptive resource allocation network classifier," *Neurocomputing*, vol. 73, nos. 16–18, pp. 3012–3019, 2010.

[13] S. Suresh, R. Savitha, and N. Sundararajan, "A sequential learning algorithm for complex-valued self-regulating resource allocation network-CSRAN," *IEEE Trans. Neural Netw.*, vol. 22, no. 7, pp. 1061–1072, Jul. 2011.

[14] G. S. Babu and S. Suresh, "Meta-cognitive neural network for classification problems in a sequential learning framework," *Neurocomputing*, vol. 81, pp. 86–96, Apr. 2012.

[15] K. Subramanian and S. Suresh, "A meta-cognitive sequential learning algorithm for neuro-fuzzy inference system," *Appl. Soft Comput.*, vol. 12, no. 11, pp. 3603–3614, 2012.

[16] R. Savitha, S. Suresh, and N. Sundararajan, "Metacognitive learning in a fully complex-valued radial basis function neural network," *Neural Comput.*, vol. 24, no. 5, pp. 1297–1328, 2012.

[17] R. Savitha, S. Suresh, and N. Sundararajan, "A meta-cognitive learning algorithm for a fully complex-valued relaxation network," *Neural Netw.*, vol. 32, pp. 209–218, Aug. 2012.

[18] G. S. Babu, R. Savitha, and S. Suresh, "A projection based learning in meta-cognitive radial basis function network for classification problems," in *Proc. Int. Joint Conf. Neural Netw.*, 2012, pp. 2907–2914.

[19] G. S. Babu, S. Suresh, and B. S. Mahanand, "Alzheimer's disease detection using a projection based learning meta-cognitive RBF network," in *Proc. Int. Joint Conf. Neural Netw.*, 2012, pp. 408–415.

[20] G. S. Babu and S. Suresh, "Parkinson's disease prediction using gene expression—A projection based learning meta-cognitive neural classifier approach," *Expert Syst. Appl.*, 2012, to be published.

[21] M. T. Cox, "Metacognition in computation: A selected research review," *Artif. Intell.*, vol. 169, no. 2, pp. 104–141, 2005.

[22] T. O. Nelson and L. Narens, *Metamemory: A Theoritical Framework and New Findings*. Boston, MA: Allyn & Bacon, 1992.

[23] T. Zhang, "Statistical behavior and consistency of classification methods based on convex risk minimization," *Ann. Stat.*, vol. 32, no. 1, pp. 56–85, 2004.

[24] S. Suresh, N. Sundararajan, and P. Saratchandran, "Risk-sensitive loss functions for sparse multi-category classification problems," *Inf. Sci.*, vol. 178, no. 12, pp. 2621–2638, 2008.

[25] H. Hoffmann, "Kernel PCA for novelty detection," *Pattern Recognit.*, vol. 40, no. 3, pp. 863–874, 2007.

[26] C. Blake and C. Merz. (1998). *UCI Repository of Machine Learning Databases*. Dept. Information & Computer Sciences, Univ. California, Irvine [Online]. Available: http://archive.ics.uci.edu/ml/

[27] J. Demsar, "Statistical comparisons of classifiers over multiple datasets," *J. Mach. Learn. Res.*, vol. 7, pp. 1–30, Dec. 2006.

[28] D. S. Marcus, T. H. Wang, J. Parker, J. G. Csernansky, J. C. Morris, and R. L. Buckner, "Open access series of imaging studies (OASIS): Cross-sectional MRI data in young, middle aged, nondemented, and demented older adults," *J. Cognit. Neurosci.*, vol. 19, no. 9, pp. 1498–507, 2007.

[29] S. G. Mueller, M. W. Weiner, L. J. Thal, R. C. Petersen, C. Jack, W. Jagust, J. Q. Trojanowski, A. W. Toga, and L. Beckett, "The Alzheimer's disease neuroimaging initiative," *Neuroimag. Clinics North Amer.*, vol. 15, no. 4, pp. 869–877, 2005.

[30] C. Cortes and V. Vapnik, "Support-vector networks," *Mach. Learn.*, vol. 20, no. 3, pp. 273–297, 1995.

[31] S. Ramasamy, P. Tamayo, R. Rifkin, S. Mukherjee, C.-H. Yeang, M. Angelo, C. Ladd, M. Reich, E. Latulippe, J. P. Mesirov, T. Poggio, W. Gerald, M. Loda, E. S. Lander, and T. R. Golub, "Multiclass cancer diagnosis using tumor gene expression signatures," *Proc. Nat. Acad. Sci.*, vol. 98, no. 26, pp. 15149–15154, 2001.

[32] S. Suresh, S. N. Omkar, V. Mani, and C. Menaka, "Classification of acoustic emission signal sources using genetic programming," *J. Aerosp. Sci. Technol.*, vol. 56, pp. 26–40, Apr. 2004.

[33] S. Suresh, S. Omkar, V. Mani, and T. G. Prakash, "Lift coefficient prediction at high angle of attack using recurrent neural network," *Aerosp. Sci. Technol.*, vol. 7, no. 8, pp. 595–602, 2003.

[34] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Trans. Intell. Syst. Technol.*, vol. 2, no. 3, pp. 1–27, 2011.

[35] S. Kloppel, C. M. Stonnington, C. Chu, B. Draganski, R. I. Scahill, J. D. Rohrer, N. C. Fox, C. R. Jack, Jr., J. Ashburner, and R. S. J. Frackowiak, "Automatic classification of MR scans in Alzheimer's disease," *Brain*, vol. 131, no. 3, pp. 681–689, 2008.

[36] C. R. Jack, R. C. Petersen, P. C. O'Brien, and E. G. Tangalos, "MR-based hippocampal volumetry in the diagnosis of Alzheimer's disease," *Neurology*, vol. 42, no. 1, pp. 183–188, 1992.

[37] B. Magnin, L. Mesrob, S. Kinkingnéhun, M. Pélégrini-Issac, O. Colliot, M. Sarazin, B. Dubois, S. Lehéricy, and H. Benali, "Support vector machine-based classification of Alzheimer's disease from whole-brain anatomical MRI," *Neuroradiology*, vol. 51, no. 2, pp. 73–83, 2009.

[38] J. Ashburner and K. Friston, "Unified segmentation," *Neuroimage*, vol. 26, no. 3, pp. 839–851, 2005.

[39] B. S. Mahanand, S. Suresh, N. Sundararajan, and M. A. Kumar, "Identification of brain regions responsible for Alzheimer's disease using a self-adaptive resource allocation network," *Neural Netw.*, vol. 32, pp. 313–322, Aug. 2012.

[40] *SPM8*. (2011). Wellcome Trust Center for Neuroimaging, Inst. Neurology, London, U.K. [Online]. Available: http://www.fil.ion.ucl.ac.uk/spm/

[41] Y. Fan and D. Shen, "Integrated feature extraction and selection for neuroimage classification," *Proc. SPIE, Med. Imag.*, vol. 7259, p. 72591U, Mar. 2009.

[42] W. Yang, H. Xia, B. Xia, L. M. Lui, and X. Huang, "ICA-based feature extraction and automatic classification of AD-related MRI data," in *Proc. 6th Int. Conf. Natural Comput.*, vol. 3. Aug. 2010, pp. 1261–1265.

[43] W. Yang, X. Chen, H. Xie, and X. Huang, "ICA-based automatic classification of magnetic resonance images from ADNI data," in *Proc. Int. Conf. Life Syst. Model. Simul. Intell. Comput.*, 2010, pp. 340–347.

[44] R. Wolz, V. Julkunen, J. Koikkalainen, E. Niskanen, D. P. Zhang, D. Rueckert, H. Soininen, and J. Lötjönen, "Multi-method analysis of MRI images in early diagnostics of Alzheimer's disease," *PLOS ONE*, vol. 6, no. 10, p. e25446, Oct. 2011.

**Giduthuri Sateesh Babu** (S'12) received the B.Tech. degree in electrical and electronics engineering from Jawaharlal Nehru Technological University, Hyderabad, India, in 2007, and the M.Tech. degree in electrical engineering from the Indian Institute of Technology—Delhi, Delhi, India, in 2009. He is currently pursuing the Ph.D. degree with the School of Computer Engineering, Nanyang Technological University, Singapore.

He was a Senior Software Engineer with the Samsung Research and Development Center, Noida, India, from 2009 to 2010. His current research interests include machine learning, cognitive computing, neural networks, control systems, and optimization and medical informatics.

**Sundaram Suresh** (SM'08) received the B.E. degree in electrical and electronics engineering from Bharathiyar University, Coimbatore, India, in 1999, and the M.E. and Ph.D. degrees in aerospace engineering from the Indian Institute of Science, Bangalore, India, in 2001 and 2005, respectively.

He was Post-Doctoral Researcher with the School of Electrical Engineering, Nanyang Technological University, Singapore, from 2005 to 2007. From 2007 to 2008, he was with INRIA-Sophia Antipolis, Sophia Antipolis, France, as an ERCIM Research Fellow. He was with Korea University, Seoul, Korea, as a Visiting Faculty of industrial engineering. From January 2009 to December 2009, he was with the Department of Electrical Engineering, Indian Institute of Technology—Delhi, Delhi, India, as an Assistant Professor. He has been an Assistant Professor with the School of Computer Engineering, Nanyang Technological University since 2010. His current research interests include machine learning, cognitive modeling, flight control, applied game theory, optimization, and computer vision.

Dr. Suresh received the Best Young Faculty Award for the Year 2009 from the Indian Institute of Technology—Delhi.