IEEE Access
The journal for rapid open access publishing

Received October 21, 2015, accepted October 30, 2015, date of publication December 8, 2015,
date of current version January 12, 2016.

Digital Object Identifier 10.1109/ACCESS.2015.2506601

Regularized Weighted Circular Complex-Valued Extreme Learning Machine for Imbalanced Learning

SANYAM SHUKLA[1] AND RAM NARAYAN YADAV[2]
[1]Department of Computer Science and Engineering, Maulana Azad National Institute of Technology, Bhopal 462003, India
[2]Department of Electronics and Communication Engineering, Maulana Azad National Institute of Technology, Bhopal 462003, India

Corresponding author: S. Shukla (sanyamshukla@gmail.com)

ABSTRACT Extreme learning machine (ELM) is emerged as an effective, fast, and simple solution for real-valued classification problems. Various variants of ELM were recently proposed to enhance the performance of ELM. Circular complex-valued extreme learning machine (CC-ELM), a variant of ELM, exploits the capabilities of complex-valued neuron to achieve better performance. Another variant of ELM, weighted ELM (WELM) handles the class imbalance problem by minimizing a weighted least squares error along with regularization. In this paper, a regularized weighted CC-ELM (RWCC-ELM) is proposed, which incorporates the strength of both CC-ELM and WELM. Proposed RWCC-ELM is evaluated using imbalanced data sets taken from Keel repository. RWCC-ELM outperforms CC-ELM and WELM for most of the evaluated data sets.

INDEX TERMS Real valued classification, class imbalance problem, weighted least squares error, regularization, extreme learning machine, complex valued neural network.

I. INTRODUCTION

Real valued classification is a popular decision making problem, having wide practical application in various fields. Extreme Learning Machine (ELM) proposed by [1], is an effective machine learning technique for real valued classification. ELM is a single hidden layer feedforward network in which the weights between input and hidden layer are initialized randomly. ELM uses analytical approach to compute weights between hidden and output layer [1], which makes it faster compared to other gradient based classifiers. Various variants of ELM were recently proposed, which includes Incremental Extreme Learning Machine [2], Kernelized Extreme Learning Machine [3], Weighted Extreme Learning Machine (WELM) [4], Regularized Extreme Learning Machine [5], Complex Extreme Learning Machine [6], Circular Complex valued Extreme Learning Machine(CC-ELM) [7] etc. This work presents an extension of CC-ELM [7] and WELM [4].

CC-ELM [7] is a complex variant of ELM, which exploits the capabilities of complex valued neuron and uses fully complex activation function. Random feature mapping is the key idea in ELM for achieving universal approximation. CC-ELM uses random feature mapping while mapping the data from real domain to complex domain using circular transformation function. This complex valued data is further mapped to feature space. CC-ELM has two levels of random feature mapping. Random feature mapping [8] eliminates the problem of overfitting. It has been shown in [9]–[11] that complex valued neural network have better computational power and generalization ability than real valued neural network. Moreover, they have inherent orthogonal decision boundaries. For example, EX-OR problem can be solved easily by using a single complex valued neuron [12]. As a result of increase in the applications involving complex valued signals like telecommunication [13], [14], adaptive array signal processing [15], [16], medical imaging signals [17], [18] etc., many complex valued classifiers were developed. Recently Complex valued classifiers were also proposed and evaluated for real valued classification. It has been shown that complex valued classifiers outperforms real valued classifiers for real valued classification problems. Fully Complex valued Radial Basis Function classifier (FC-RBF) [19], [20], Fast Learning Complex-valued Neural Classifier (FLCNC) [21], Multi Layered Multi Valued Neural network (MLMVN) [22], Bilinear Branch-cut Complex-valued Extreme Learning Machine (BB-CELM),

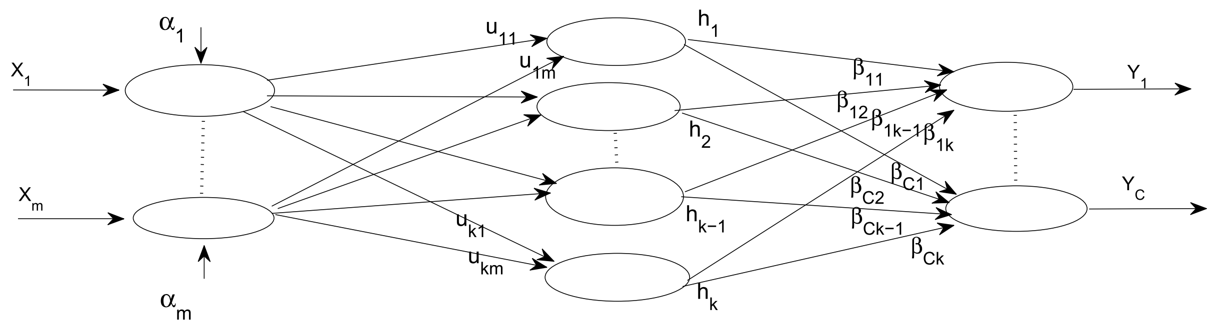

FIGURE 1. Architecture of CC-ELM.

Phase Encoded Complex-valued Extreme Learning Machine (PE-CELM) [23], CC-ELM [7] etc. are some of the complex-valued classifiers designed for real valued classification problems. CC-ELM outperforms other complex valued classifiers for real valued classification problems. It also performs well, when the dataset is imbalanced.

It has been observed that many practical classification problems have imbalanced data sets [24], [25]. If we classify such data most of the classifiers favors the majority class, due to which most of the instances belonging to minority class are misclassified. To deal with such dataset, various sampling approaches [26] as well as algorithmic approaches [4], [27], [28] have been developed. Sampling approaches include oversampling and undersampling techniques. Oversampling replicates a fraction of minority samples while undersampling approach reduces a fraction of majority samples to make dataset balanced. But there is problem with sampling approaches. Oversampling [29] increases redundancy of data and undersampling results in loss of information. In algorithmic approach, classifier design encompasses the measures to handle class imbalance. Most of the neural network based classifiers like FCRBF, CC-ELM [7], [19], [20] minimizes least square error to find optimal weights. Recently proposed WELM [4] minimizes weighted least square error function along with regualrization to find optimal weights between hidden and output layer. In this classifier, instances belonging to minority class are assigned more weights compared to instances of majority class. This increases the impact of minority samples. Finding optimal weighting scheme is a challenging task. WELM is evaluated using two generalized weighting schemes for assigning weights to the instances. Several variants of ELM employing regularization like [30] and [31] have been developed. A variant of regularized Extreme Learning Machine is proposed in [30] which is incremental. Regularized variants of ELM have been used for action recognition [32], large scale media content analysis [33], regression with missing values [34], face recognition [35] etc.

In this paper, Regularized Weighted Circular Complex Valued Extreme Learning Machine (RWCC-ELM) is proposed, which is an extension of CC-ELM and WELM. It differs from WELM as it is a complex valued classifier whereas, WELM is a real valued classifier. As RWCC-ELM is a complex valued classifier, this paper presents an extended derivation of the expression to find the weights between hidden and output layer in complex domain. RWCC-ELM differs from CC-ELM as it uses weighted least square error function along with regularization to find weights between hidden and output layer. RWCC-ELM assigns more weight to instances of minority class compared to that of majority class. This strengthens the relative impact of minority class, thereby increasing the overall performance. To remove the problem of overfitting, RWCC-ELM uses regularization. Some popular regularization methods are lasso (L1), ridge regression (L2), elastic net (combination of L1 and L2) etc. In our proposed classifier, ridge regression is used for regularization.

The rest of paper is organized as follows. Section II briefly describes related work. Section III presents proposed RWCC-ELM learning algorithm. Section IV gives details of data sets used, performance evaluation metrics, the results of the proposed learning algorithms and their analysis. Section V concludes the paper and outlines the future work.

II. RELATED WORK

A. CIRCULAR COMPLEX VALUED EXTREME LEARNING MACHINE (CC-ELM)

This section gives a brief description of CC-ELM [7], which is the foundation of our proposed algorithm. CC-ELM is a single hidden layer complex valued neural network with m input neurons, L hidden neurons and C output neurons. Number of input and output neurons is equal to the number of input features and number of classes respectively. It has nonlinear input, nonlinear hidden layer and linear output layer. Fig. 1 shows the architecture of CC-ELM. All weights in CC-ELM are complex valued. Given N observations $[(\mathbf{x_1}, t_1)(\mathbf{x_2}, t_2)\ldots(\mathbf{x_p}, t_p)\ldots(\mathbf{x_N}, t_N)]$, where $\mathbf{x_p} \in R^m$ refers to m-dimensional input feature and t_p refers to class label of p^{th} instance. CC-ELM maps the input data from real domain to complex domain uniformly with the help of following circular transformation function.

$$z_t = \sin(ax_t + ibx_t + \alpha_t), \quad t = 1, 2, \ldots, m \tag{1}$$

where, $(0 < a, b \leq 1)$ are scaling terms and $(0 < \alpha < 2\pi)$ are rotational bias term. Their values are chosen randomly and input features are normalized between [−1, 1]. Features are mapped to different quadrants by choosing different values of circular transformation parameter, α_t. The transformed input sample $\mathbf{x_i}$ is represented by $\mathbf{z_i}$. Target of $\mathbf{x_p}$ i.e., t_p is coded as vector $\mathbf{t_p}$, $[t_{p1} \cdots t_{pg} \cdots t_{pC}]^T$. Target output matrix, T ($N \times C$) for all instances in the training dataset is designed using the following class coding scheme.

$$T = \begin{pmatrix} \mathbf{t_1^T} \\ \vdots \\ \mathbf{t_N^T} \end{pmatrix} = \begin{pmatrix} t_{11} & \cdots t_{1k} \cdots & t_{1C} \\ \vdots & \vdots & \vdots \\ t_{p1} & \cdots t_{pk} \cdots & t_{pC} \\ \vdots & \vdots & \vdots \\ t_{N1} & \cdots t_{Nk} \cdots & t_{NC} \end{pmatrix} \tag{2}$$

$$Here, \begin{pmatrix} if\ t_p = k & t_{pk} = 1 + 1i, \\ else & t_{pk} = -1 - 1i, \\ & p = 1 \ldots N, k = 1 \ldots C \end{pmatrix}$$

Vector $\mathbf{T^k}$, $[t_{1k}, t_{2k}, \ldots, t_{Nk}]^T$ is used to represent the target output of k^{th} neuron. The neurons in the hidden layer employ fully complex sech activation function [20]. The response of j^{th} hidden neuron for the transformed input $\mathbf{z}$, $\mathbf{h_j(z)}$ is given by:

$$\mathbf{h_j(z)} = sech(\mathbf{u_j}^\mathbf{T}\mathbf{z} - \mathbf{v_j}) \tag{3}$$

Here, j = 1, 2 ... L, $\mathbf{u_j} \in C^m$ is the complex-valued weight vector and $\mathbf{v_j} \in C^m$ is the complex-valued center of the j^{th} hidden neuron. The superscript T represents the matrix transpose operator. Scaling factors $\mathbf{u_j}$ and $\mathbf{v_j}$ are selected randomly. The hidden layer output can be represented by a row vector $\mathbf{h(z)}$, $[h_1(\mathbf{z}), h_2(\mathbf{z}) \ldots h_L(\mathbf{z})]$ where $\mathbf{z}$ is the transformed input sample. The hidden layer response for all N training samples can be presented by the following ($N \times L$) matrix, H.

$$H = \begin{pmatrix} \mathbf{h(z_1)} \\ \vdots \\ \mathbf{h(z_N)} \end{pmatrix} \tag{4}$$

β_{kj} is the output weight connecting j^{th} hidden neuron and k^{th} output neuron. $\boldsymbol{\beta}^\mathbf{k}$, $[\beta_{k1}, \beta_{k2}, \ldots, \beta_{kj}, \ldots, \beta_{kL}]^T$ is the weight vector of k^{th} output neuron. $\boldsymbol{\beta}(L \times C)$ is the matrix of all weights connecting neurons of the hidden layer and output layer.

$$\boldsymbol{\beta} = [\boldsymbol{\beta}^\mathbf{1}, \boldsymbol{\beta}^\mathbf{2}, \ldots, \boldsymbol{\beta}^\mathbf{C}] \tag{5}$$

The neurons in the output layer employ a linear activation function. Therefore, target, $\mathbf{T} = \mathbf{H}\boldsymbol{\beta}$. The weights between the hidden and output layer, $\boldsymbol{\beta}$ are evaluated as follows:

$$\begin{aligned} \mathbf{H^H T} &= \mathbf{H^H H}\boldsymbol{\beta} \\ \boldsymbol{\beta} &= \left(\frac{\mathbf{H^H}}{\mathbf{H^H H}}\right)\mathbf{T} \\ \boldsymbol{\beta} &= \mathbf{H^+ T} \end{aligned} \tag{6}$$

Here, $\mathbf{H^+}$ is generalized Moore-Penrose inverse which provides unique least squares solution with minimum norm [36]. The predicted output, y^k of k^{th} output neuron for an instance $\mathbf{x}$ is given by:

$$y^k = \mathbf{h(z)}\boldsymbol{\beta}^\mathbf{k}, \quad k = 1, 2 \ldots C \tag{7}$$

The predicted output of all the output neurons for an instance $\mathbf{x}$, can be represented as $[y^1, y^2, \ldots, y^C]$. The predicted class label, c of a given test sample x, is the index number of the output node, whose real part of the output is maximum.

$$c = arg(\max(Real(y_k))), \quad k = 1, 2 \ldots C \tag{8}$$

The predicted output of all output neurons for all training instances, $\mathbf{Y}$ is given by following equation:

$$\mathbf{Y} = \mathbf{H}\boldsymbol{\beta}. \tag{9}$$

B. WEIGHTED EXTREME LEARNING MACHINE (WELM)

WELM [4] is a variant of ELM, which minimizes weighted least square error along with regularization to find the optimal weights between hidden and output layer. In WELM all the weights and bias are real valued. In [4] two generalized weighting schemes were proposed and evaluated. These generalized weighting scheme assign weights to the instances as per their class distribution. The two weighting schemes proposed by WELM are:

First Weighting Approach W1:

$$w_i = 1/q_k \quad Here, \quad k = t_i, \quad i \in 1, 2 \ldots N \\ k \in 1, 2 \ldots C \tag{10}$$

Second Weighting Approach W2:

$$\begin{aligned} q_{avg} &= \sum_{k=1}^{C} q_k / C \\ w_i &= 1/q_k, \quad if\ (q_k <= q_{avg}) \\ w_i &= 0.618/q_k, \quad if\ (q_k > q_{avg}) \end{aligned}$$

Weight, w_i is assigned to the i^{th} instances. Here, q_k is the total number of instances belonging to k^{th} class. Instances belonging to minority class will be assigned weights equal to $1/q_i$ in both the weighting schemes. Second weighting scheme assigns less weight to majority class instances compared to first weighting scheme. In WELM the problem of finding output layer weights is formulated as follows.

$$\begin{aligned} &Minimize: \frac{1}{2}\|\boldsymbol{\beta}\|^2 + \lambda \mathbf{W} \frac{1}{2} \sum_{i=1}^{N} \|\boldsymbol{\xi}_i\|^2 \\ &Subject\ to: \mathbf{h(x_i)}\boldsymbol{\beta} = t_i^T - \xi_i^T, \quad i = 1, \ldots, N \end{aligned} \tag{11}$$

The first term of the objective function is regularization term, also known as structural risk $||\boldsymbol{\beta}||^2$ and the second term is weighted least square error, also known as empirical risk $||\boldsymbol{\xi}||^2$. Structural risk depends on margin separating classes [5]. The regularization parameter, λ is used to control the trade off between the two risks. $\mathbf{W}$ is an N × N diagonal matrix whose diagonal elements are w_i. On solving the above quadratic optimization problem [4] has derived the following equations to find the weight between hidden and output layer. **For the case, when the number of training samples is less than the number of hidden neurons**

$$\boldsymbol{\beta} = \mathbf{H^T}(\mathbf{I}/\lambda + \mathbf{WHH^T})^{-1}\mathbf{WT} \tag{12}$$

For the case, when the number of training samples is greater than the number of hidden neurons

$$\boldsymbol{\beta} = (\mathbf{I}/\lambda + \mathbf{H^T WH})^{-1}\mathbf{H^T WT}. \tag{13}$$

III. PROPOSED RWCC-ELM

RWCC-ELM is a regularized weighted version of CC-ELM which incorporates the strength of both CC-ELM and WELM. The main features of this classifier are: (1) Deals with imbalanced data set. (2) Minimize overfitting problem using regularization. (3) Generalized weight matrix, so dataset specific tuning of weights, to be assigned to instances is not required. (4) Capable of both binary and multiclass classification. (5) Complex valued classifier with orthogonal decision boundary. The classification problem is defined in the same way as that of CC-ELM. The notations used in following section are same as that of section 2. To handle class imbalance problem, RWCC-ELM minimizes weighted least square error function. In RWCC-ELM the instances belonging to different classes are assigned different weights using equation (10). Instances belonging to minority class are assigned more weight compared to instances belonging to majority class. This enables minority class to have significant contribution in weighted least square error. This reduces the misclassification of minority class samples and results in overall increase in performance of classifier. Weighted least square error is given by:

$$\mathbf{W}.\boldsymbol{\xi}^{\mathbf{2}} = \sum_{k=1}^{C}\sum_{i=1}^{N} w_i.\xi_{ik}{}^2 = \begin{pmatrix} w_1 & 0 & \cdots & 0 \\ 0 & w_2 & \cdots & 0 \\ \vdots & \vdots & \cdots & \vdots \\ 0 & 0 & \cdots & w_N \end{pmatrix} \cdot \begin{pmatrix} (\xi_{11})^2 & (\xi_{1k})^2 & \cdots & (\xi_{1C})^2 \\ \vdots & \vdots & \cdots & \vdots \\ (\xi_{p1})^2 & (\xi_{pk})^2 & \cdots & (\xi_{pC})^2 \\ \vdots & \vdots & \cdots & \vdots \\ (\xi_{N1})^2 & (\xi_{Nk})^2 & \cdots & (\xi_{NC})^2 \end{pmatrix}$$

Here, weight matrix, $\mathbf{W}$ is an $N \times N$ diagonal matrix, where, N is number of instances. Diagonal element, w_i is the weight assigned to i^{th} instance and ξ_{ik} is the error of i^{th} instance for k^{th} class. $\boldsymbol{\xi}^{\mathbf{k}} = [\xi_{ik} \ldots \xi_{Nk}]^T$ is the vector used to represent error of all the instances corresponding to k^{th} class. Smaller values of weights lead to more generalized solution. For this, RWCC-ELM uses ridge regression. RWCC-ELM is different from CC-ELM as it minimizes weighted least square error function along with regularization to obtain optimal weights.

WELM is a real valued classifier which minimizes weighted least squares error along with regularization. WELM formulates this as optimization problem which is given in equation (11). The proposed classifier also minimizes weighted least square error along with regularization. But the proposed classifier is a complex valued classifier which has complex values weights, bias and input. The formulation given by equation (11) is not valid for complex valued weight and input. The proposed RWCCELM formulates the equivalent optimization function in complex domain which is given in equation (14). The architecture of RWCC-ELM is same as CC-ELM. RWCC-ELM uses circularly complex transformation function (1) to map data to complex domain. It uses fully complex, sech activation function (3) same as that of CC-ELM. RWCC-ELM differs from CC-ELM only in the way to compute weights between the hidden and output neurons. The objective function, to find the weights between the hidden and the output neurons of RWCC-ELM, is formulated as follows:

$$\begin{aligned} &Minimize: \ (\boldsymbol{\beta}^{\mathbf{k}})^{\mathbf{H}}\boldsymbol{\beta}^{\mathbf{k}} + \lambda(\xi^{\mathbf{k}})^{\mathbf{H}}\mathbf{W}\xi^{\mathbf{k}} \\ &Subject\ to: \ \mathbf{h}(\mathbf{z_i})\boldsymbol{\beta}^{\mathbf{k}} = t_{ik} - \xi_{ik}, \quad i = 1, \ldots N, k \in 1, \ldots C \end{aligned} \tag{14}$$

Here, superscript H indicates complex conjugate transpose. The first term of the objective function is regularization term, also known as structural risk $(\boldsymbol{\beta}^k)^H \boldsymbol{\beta}^k$ and the second term is weighted least square error, also known as empirical risk $W(\xi^k)^H \xi^k$. Structural risk depends on margin separating class [5]. These risks are to be minimized. The regularization parameter, λ is used to control the trade off between the two risks. The above optimization problem has a real valued objective function with N complex valued equality constraints. The Lagrangian function for the above optimization problem can be formulated as per guidelines given in [37] and [38].

The Lagrangian function for the above optimization problem, $L_{RWCCELM}$ is as follows:

$$\begin{aligned} L_{RWCCELM} &= (\boldsymbol{\beta}^{\mathbf{k}})^{\mathbf{H}}\boldsymbol{\beta}^{\mathbf{k}} + \lambda(\boldsymbol{\xi}^{\mathbf{k}})^{\mathbf{H}}\mathbf{W}\boldsymbol{\xi}^{\mathbf{k}} \\ &\quad -2Real\sum_{i=1}^{N}\alpha_i(\mathbf{h}(\mathbf{z_i})\boldsymbol{\beta}^{\mathbf{k}} - t_{ik} + \xi_{ik}) \\ &= (\boldsymbol{\beta}^{\mathbf{k}})^{\mathbf{H}}\boldsymbol{\beta}^{\mathbf{k}} + \lambda(\boldsymbol{\xi}^{\mathbf{k}})^{\mathbf{H}}\mathbf{W}\boldsymbol{\xi}^{\mathbf{k}} \\ &\quad -\sum_{i=1}^{N}\alpha_i(\mathbf{h}(\mathbf{z_i})\boldsymbol{\beta}^{\mathbf{k}} - t_{ik} + \xi_{ik}) \\ &\quad -\sum_{I=1}^{N}\alpha_i^*(\mathbf{h}^*(\mathbf{z_i})\boldsymbol{\beta}^{\mathbf{k}*} - t_{ik}^* + \xi_{ik}^*) \end{aligned}$$

The Karush-Kuhn-Tucker (KKT) optimality conditions are obtained by taking partial derivative with respect to variables

$\boldsymbol{\beta}^{\mathbf{k}}$, ξ_{ik}, α_i and equating them to zero.

$$\begin{aligned}
\delta/\delta\boldsymbol{\beta}^{\mathbf{k}} \Rightarrow (\boldsymbol{\beta}^{\mathbf{k}})^{\mathbf{H}} &= \sum_{i=1}^{N}\alpha_i\mathbf{h}(\mathbf{z_i})\\
(\boldsymbol{\beta}^{\mathbf{k}})^{\mathbf{H}} &= \boldsymbol{\alpha}^{\mathbf{T}}\mathbf{H}\\
\boldsymbol{\beta}^{\mathbf{k}} &= (\boldsymbol{\alpha}^{\mathbf{T}}\mathbf{H})^{\mathbf{H}} = \mathbf{H}^{\mathbf{H}}\boldsymbol{\alpha}^* \qquad (15a)\\
\delta/\delta\xi_{ik} \Rightarrow \alpha_i &= \lambda w_i(\xi_{ik})^*\\
\boldsymbol{\alpha} &= \lambda\mathbf{W}(\boldsymbol{\xi}^{\mathbf{k}})^*\\
\boldsymbol{\alpha}^* &= \lambda\mathbf{W}\boldsymbol{\xi}^{\mathbf{k}} \qquad (15b)\\
\boldsymbol{\xi}^{\mathbf{k}} &= \boldsymbol{\alpha}^*/\lambda\mathbf{W} \qquad (15c)\\
\delta/\delta\alpha_i \Rightarrow \xi_{ik} &= t_{ik} - \mathbf{h}(\mathbf{z_i})\boldsymbol{\beta}^{\mathbf{k}}\\
\boldsymbol{\xi}^{\mathbf{k}} &= \mathbf{T}^{\mathbf{k}} - \mathbf{H}\boldsymbol{\beta}^{\mathbf{k}} \qquad (15d)
\end{aligned}$$

Different solutions to the aforementioned KKT conditions can be obtained. Using (15), we have

- **For the case, where number of training samples is less than number of hidden neurons:**
Substituting (15a) and (15c) in (15d), we have

$$\begin{aligned}
\boldsymbol{\alpha}^*/\lambda\mathbf{W} &= \mathbf{T}^{\mathbf{k}} - \mathbf{H}\mathbf{H}^{\mathbf{H}}\boldsymbol{\alpha}^*\\
\mathbf{T}^{\mathbf{k}} &= \boldsymbol{\alpha}^*[\mathbf{I}/\lambda\mathbf{W} + \mathbf{H}\mathbf{H}^{\mathbf{H}}]\\
\boldsymbol{\alpha}^* &= \frac{\mathbf{T}^{\mathbf{k}}}{\mathbf{I}/\lambda\mathbf{W} + \mathbf{H}\mathbf{H}^{\mathbf{H}}} \qquad (16)
\end{aligned}$$

Substituting value of $\boldsymbol{\alpha}^*$ from (16) into (15a), we have

$$\begin{aligned}
\boldsymbol{\beta}^{\mathbf{k}} &= \mathbf{H}^{\mathbf{H}}\frac{\mathbf{T}^{\mathbf{k}}}{\mathbf{I}/\lambda\mathbf{W} + \mathbf{H}\mathbf{H}^{\mathbf{H}}}\\
\boldsymbol{\beta}^{\mathbf{k}} &= \mathbf{H}^{\mathbf{H}}\frac{\mathbf{W}\mathbf{T}^{\mathbf{k}}}{\mathbf{I}/\lambda + \mathbf{W}\mathbf{H}\mathbf{H}^{\mathbf{H}}}\\
\boldsymbol{\beta}^{\mathbf{k}} &= \mathbf{H}^{\mathbf{H}}(\mathbf{I}/\lambda + \mathbf{W}\mathbf{H}\mathbf{H}^{\mathbf{H}})^{-1}\mathbf{W}\mathbf{T}^{\mathbf{k}} \qquad (17)
\end{aligned}$$

The above equation can be rewritten in the following form:

$$\boldsymbol{\beta} = \mathbf{H}^{\mathbf{H}}(\mathbf{I}/\lambda + \mathbf{W}\mathbf{H}\mathbf{H}^{\mathbf{H}})^{-1}\mathbf{W}\mathbf{T} \qquad (18)$$

- **For the case, where number of training samples is greater than number of hidden neurons:**
If the number of training data is very large, for example, it is much larger than the dimensionality of the feature space, N $\gg$ L, we have an alternative solution. From (15a) and (15b), we have

$$\boldsymbol{\beta}^{\mathbf{k}} = \mathbf{H}^{\mathbf{H}}\lambda\mathbf{W}\boldsymbol{\xi}^{\mathbf{k}}$$

Substituting the value of $\boldsymbol{\xi}^k$ from (15d), we have

$$\begin{aligned}
\boldsymbol{\beta}^{\mathbf{k}} &= \mathbf{H}^{\mathbf{H}}\lambda\mathbf{W}[\mathbf{T}^{\mathbf{k}} - \mathbf{H}^{\mathbf{k}}\boldsymbol{\beta}^{\mathbf{k}}]\\
\boldsymbol{\beta}^{\mathbf{k}} &= \mathbf{H}^{\mathbf{H}}\lambda\mathbf{W}\mathbf{T}^{\mathbf{k}} - \mathbf{H}^{\mathbf{H}}\lambda\mathbf{W}\mathbf{H}^{\mathbf{k}}\boldsymbol{\beta}^{\mathbf{k}}\\
\boldsymbol{\beta}^{\mathbf{k}} &= \mathbf{H}^{\mathbf{H}}\lambda\mathbf{W}\mathbf{T}^{\mathbf{k}}/[\mathbf{I} + \mathbf{H}^{\mathbf{H}}\lambda\mathbf{W}\mathbf{H}^{\mathbf{k}}]\\
\boldsymbol{\beta}^{\mathbf{k}} &= \mathbf{H}^{\mathbf{H}}\mathbf{W}\mathbf{T}^{\mathbf{k}}/[\mathbf{I}/\lambda + \mathbf{H}^{\mathbf{H}}\mathbf{W}\mathbf{H}^{\mathbf{k}}]\\
\boldsymbol{\beta}^{\mathbf{k}} &= (\mathbf{I}/\lambda + \mathbf{H}^{\mathbf{H}}\mathbf{W}\mathbf{H})^{-1}\mathbf{H}^{\mathbf{H}}\mathbf{W}\mathbf{T}^{\mathbf{k}} \qquad (19)
\end{aligned}$$

The above equation can be rewritten as

$$\boldsymbol{\beta} = (\mathbf{I}/\lambda + \mathbf{H}^{\mathbf{H}}\mathbf{W}\mathbf{H})^{-1}\mathbf{H}^{\mathbf{H}}\mathbf{W}\mathbf{T} \qquad (20)$$

Both (18) and (20) can be used to find $\boldsymbol{\beta}$. Solution of a system of linear equations having singular coefficient matrix can lead to inaccurate results. So here, it is preferable to use generalized Moore-Penrose inverse which will give fast and accurate result. The algorithm of RWCC-ELM is described in Algorithm 1.

Algorithm 1 Regularized Weighted Circular Complex Valued Extreme Learning Machine

Input:
Training Set S, having N observations $[(\mathbf{x_1}, t_1)$ $(\mathbf{x_2}, t_2)..(\mathbf{x_p}, t_p)$ $...(\mathbf{x_N}, t_N)]$, where $\mathbf{x_p} \in R^m$ refers to m-dimensional input feature and t_p refers to class label of p^{th} instance, $t_p \in 1, 2.........C$.

Training Phase

Step 1: Map the real valued data to complex domain using equation (1).

Step 2: Compute the target matrix, T using the class coding scheme (2).

Step 3: Choose number of neurons, L as per complexity of problem. Also choose a reasonable value of regularization parameter, λ. We have performed a grid search on $L = [10, 20, ...1000]$ and $\lambda = [2^{-18}, 2^{-16}...2^{50}]$ to achieve optimal results.

Step 4: Initialize the weights between input layer and hidden layer, u_j and centers of j^{th} neuron, v_j with complex numbers randomly.

Step 5: Compute hidden layer output matrix, H using (3) and (4).

Step 6: Compute weight matrix, W using equation (10).

Step 7: Calculate weights between hidden and output layer, $\boldsymbol{\beta}$ using either equation (18) or (20).

Output: RWCC-ELM model which consist of a, b, $\boldsymbol{\alpha}$, L, $\mathbf{U}$, $\mathbf{V}$, $\mathbf{W}$ and $\boldsymbol{\beta}$

Testing Phase

Input: Testing set, S1 having R observations $[(\mathbf{x}_1', t_1')(\mathbf{x}_2', t_2')$ $..(\mathbf{x}_p', t_p')...(\mathbf{x}_R', t_R')]$, where $\mathbf{x}_p' \in R^m$ refers to m-dimensional input feature and t_p' refers to class label of p^{th} instance, $t_p' \in 1, 2.........C$.

Step 1: Map the testing data to complex domain using circularly complex mapping function given in (1). Use values of a, b and $\boldsymbol{\alpha}$ which were obtained as output of training phase.

Step 2: Compute hidden layer output matrix, $\mathbf{H}'$ using values of L,$\mathbf{U}$, $\mathbf{V}$, $\mathbf{W}$ obtained as output of training phase.

Step 3: Calculate the predicted output,$\mathbf{Y}$ as follows:

$$\mathbf{Y}' = \mathbf{H}'\boldsymbol{\beta} \qquad (21)$$

Step 4: Determine the predicted class label using (8).

Step 5: Evaluate testing performance using predicted class label and the known, actual class label.

IV. PERFORMANCE EVALUATION

In the following section, the proposed RWCC-ELM is compared with CC-ELM and WELM for various real valued classification problems. All experiments are carried out using

IEEE Access

TABLE 1. Specification of datasets.

Datasets	Number of Attributes	Number of Classes	Number of Train Instances	Number of Test Instances	Imbalance Ratio
abalone19	8	2	3339	835	128.87
abalone9_18	8	2	584	147	16.68
ecoli1	7	2	268	68	3.36
ecoli2	7	2	268	68	5.46
ecoli3	7	2	268	68	8.19
ecoli4	7	2	268	68	13.84
glass0	9	2	173	43	2.06
glass1	9	2	171	43	1.82
glass2	9	2	171	43	10.39
glass4	9	2	171	43	15.47
glass5	9	2	171	43	22.81
glass6	9	2	171	43	6.38
haberman	3	2	244	62	2.68
iris0	4	2	120	30	2
new-thyroid1	5	2	172	43	5.14
new-thyroid2	5	2	172	43	4.92
balance	4	3	500	125	5.88
contraceptive	9	3	1178	295	1.89
ecoli	7	8	269	67	71.5
hayes-roth	4	3	106	26	1.7
newthyroid	5	3	172	43	4.84
thyroid	21	3	576	144	36.94
yeast	8	10	1187	297	23.15
bupa	6	2	276	69	1.38
glass	9	7	171	43	8.44
ionosphere	33	2	280	71	1.8
pima	8	2	614	154	1.87
segment	19	7	1848	462	1
vehicle	18	4	676	170	1.10
wisconsin	9	2	546	137	1.86

Matlab 7.1 running on PC with Intel core i5 processor, 3.20 GHz CPU and 2 GB RAM. The averaged results evaluated by running proposed algorithm for 10 independent trials are presented in this section.

A. DATA SPECIFICATION

To demonstrate the performance of RWCC-ELM, experiments were conducted on 20 binary and 10 multiclass imbalanced datasets of varying Imbalance Ratio (IR). These datasets with five fold cross validation, are downloaded from Keel dataset repository [39]. The specifications of datasets used are shown in Table 1. IR is evaluated as follows.

$$IR = \frac{\max(q_k)}{\min(q_k)} \quad Here, k = 1, 2 \ldots, C \tag{22}$$

The attributes of all datasets are normalized in the range $[-1, 1]$.

B. PERFORMANCE EVALUATION METRICS

The result of binary classification can be categorized into four categories: True Positive (TP), True Negative (TN), False Positive (FP) and False Negative (FN). Overall accuracy, η_{ova} is defined as:

$$\eta_{ova} = \frac{TP + TN}{TP + FP + TN + FN} = \frac{\textit{Number of correctly classified samples}}{\textit{Total number of samples}} \tag{23}$$

For a binary classification problem having 98 instances belonging to negative class and 2 instances belonging to positive class, a classifier which classifies all the instances to negative class would achieve 98 percent accuracy. Overall accuracy is not an effective measure to deal with class imbalance problem. *G-mean*, which is a function of sensitivity and specificity, is an effective measure to deal with the class imbalance problem. Sensitivity and specificity are recall of the positive and negative class respectively. They are defined as follows:

$$\textit{Sensitivity} = TP/(TP + FN) \tag{24}$$

$$\textit{Specificity} = TN/(TN + FP) \tag{25}$$

$$\textit{G-mean} = \sqrt{\textit{Sensitivity} \times \textit{Specificity}} \tag{26}$$

G-mean for multiclass problem is defined as follows:

$$\textit{G-mean} = \left(\prod_{k=1}^{C} R_k \right)^{\frac{1}{C}} \tag{27}$$

Here R_k represents the recall of k^{th} class.

C. PARAMETER SETTING

For both RWCC-ELM and CC-ELM, the parameters of circular complex transformation function (1) are chosen randomly. In order to achieve optimal results, a grid search on number of hidden neurons, L on $[10, 20, \ldots 990, 1000]$ and regularization parameter, λ on $[2^{-18}, 2^{-16} \ldots 2^{50}]$ is conducted for RWCC-ELM. The effect of these parameters on the performance of classifier is shown in Fig. 2. For CC-ELM, optimal number of hidden neurons is searched by varying L on $[10, 20 \ldots 1000]$.

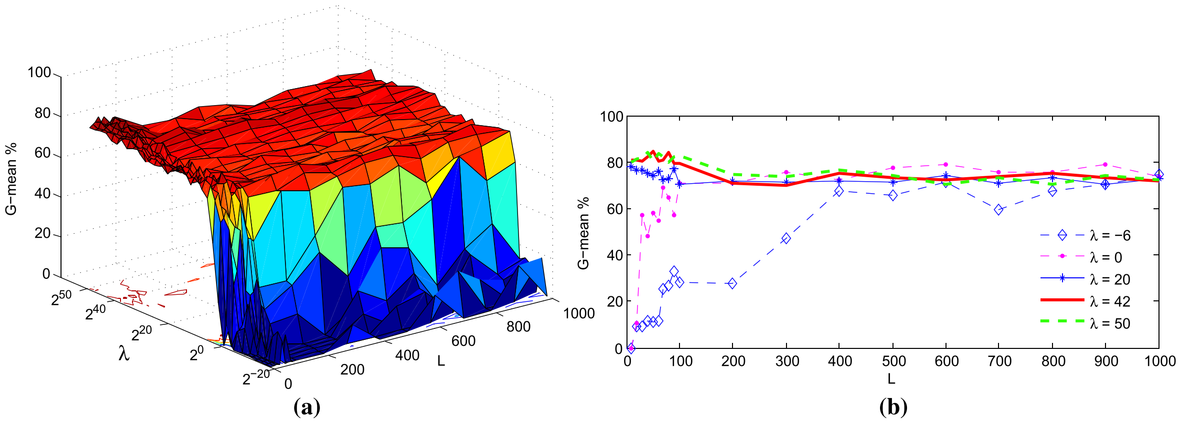

FIGURE 2. Display of test G-mean of glass0 dataset for RWCC-ELM (a) when λ and L varies. (b) when L varies.

TABLE 2. Testing accuracy in terms of G-mean.

Activation Function->		Sigmoid	Gaussian Kernel	Sech				
DATA	IR	RW1ELM	RW1ELM	CC-ELM		RWCC-ELM		
		G-mean%	G-mean%	G-mean%	L	G-mean %	L	λ
abalone19	128.87	77.19	74.47	0	10	**85.49**	10	44
abalone9-18	16.68	87.99	89.76	81.58	890	**95.39**	80	50
ecoli1	3.36	90.69	91.04	89.24	40	**91.72**	130	14
ecoli2	5.46	93.91	94.09	93.05	760	**94.7**	50	42
ecoli3	8.19	90.17	89.6	81.04	380	**92.1**	100	6
ecoli4	13.84	97.83	98.24	93.26	240	**98.88**	250	16
glass0	2.06	81.17	**85.65**	80.4	100	84.63	50	42
glass1	1.82	78.31	**80.35**	77.28	130	78.23	110	40
glass2	10.39	80.33	82.59	74.05	210	**87.85**	160	48
glass4	15.47	91.34	91.17	94.41	900	**98.74**	30	40
glass5	22.81	95.99	96.51	99.26	70	**99.51**	110	40
glass6	6.38	95.72	94.04	**97.99**	80	**97.99**	90	42
haberman	2.68	65.11	66.26	54.88	160	**69.85**	20	8
iris0	2	100	**100**	**100**	10	**100**	10	6
new-thyroid1	5.14	99.44	99.72	98.31	50	**100**	20	8
newthyroid2	4.92	**99.72**	**99.72**	97.03	40	**99.72**	50	10
balance	5.88	66.05	**83.82**	27.2	890	80.04	300	30
contraceptive	1.89	50.85	55.53	53.61	90	**57.11**	100	48
ecoli	71.5	35.98	17.61	17.25	60	**48.7**	250	50
hayes-roth	1.7	81.4	85.08	88	20	**87.43**	840	38
new-thyroid	4.84	96.49	**99.55**	89.34	40	96.56	520	34
thyroid	36.94	72.46	72.02	72.63	340	**79.81**	250	40
yeast	23.15	0	**48.77**	0	10	46.99	10	46
bupa	1.38	70.54	71.63	72.05	40	**73.10**	60	46
glass	8.44	73.04	70.34	51.46	60	**75.74**	30	34
ionosphere	1.8	70.98	90.05	89.43	90	**91.58**	30	44
pima	1.87	74.92	75.96	73.06	30	**75.43**	70	32
segment	1	93.61	97.75	96.61	690	**97.80**	710	46
vehicle	1.10	79.92	84.64	84.24	280	**85.06**	310	40
wisconsin	1.86	97.25	97.61	97.80	20	**97.96**	70	30

D. EXPERIMENT RESULTS

1) CC-ELM vs RWCC-ELM

RWCC-ELM differs from CC-ELM only in the way to find weights between hidden and output neuron. CC-ELM can be considered as a special case of RWCC-ELM, when the value of regularization parameter, λ is equal to ∞ and weight matrix, W is replaced by an identity matix. It can be seen from Table 2 that RWCC-ELM outperforms CC-ELM for all the evaluated datasets.

2) WELM vs RWCC-ELM

RWCC-ELM differs from WELM as it is a complex valued classifier. RWCC-ELM uses equations 18 and 20 to find $\boldsymbol{\beta}$. These equations are reduced to equations 12 and 13 respectively, when complex conjugate transpose operator, H is replaced by transpose operator, T. For real valued data, hermitian operator can be replaced by transpose operator. WELM with weighting scheme given by (10) is represented as RW1ELM. The testing efficiencies of WELM for binary

TABLE 3. Testing overall accuracy.

Activation Function->	Sigmoid	Gaussian Kernel	Sech				
DATA	RW1ELM	RW1ELM	CC-ELM		RWCC-ELM		
	$\eta_{ova}\%$	$\eta_{ova}\%$	$\eta_{ova}\%$	L	$\eta_{ova}\%$	L	λ
abalone19	93.56	98.8	**99.23**	10	**99.23**	10	-18
abalone9_18	93.68	95.72	**97.12**	130	94.53	20	4
ecoli1	89.89	90.19	**92.57**	10	90.17	10	34
ecoli2	94.94	95.54	95.17	760	**95.83**	80	40
ecoli3	89.28	91.08	**93.75**	330	92.55	40	6
ecoli4	97.32	97.92	**98.81**	20	98.51	580	0
glass0	79.45	**85.5**	81.32	10	82.72	50	42
glass1	78.99	**81.77**	78.98	70	77.12	110	40
glass2	87.81	92.06	92.06	10	**92.53**	210	0
glass4	95.81	96.28	**97.67**	40	**97.67**	30	40
glass5	98.14	98.14	98.6	30	**99.07**	40	42
glass6	97.67	97.67	99.06	80	**99.07**	90	36
haberman	75.16	72.87	75.82	10	**77.77**	730	-2
new-thyroid1	99.04	99.54	98.6	30	**100**	20	8
new-thyroid2	99.54	**99.54**	99.07	40	**99.54**	10	14
pima	76.56	76.04	**78.77**	40	78.25	20	40
wisconsin	97.66	97.51	97.51	20	**98.39**	340	22
balance	92.16	87.36	**93.6**	50	87.2	10	30
contraceptive	56.55	54.85	**57.36**	50	57.03	220	44
ecoli	86.33	81.54	**82.77**	60	82.43	200	44
hayes-roth	81.82	85.58	**87.86**	20	87.09	840	38
new-thyroid	98.61	**99.07**	95.35	40	98.14	10	30
thyroid	92.64	**94.03**	93.75	80	92.92	170	0
bupa	70.96	72.17	**74.78**	40	73.25	60	44
glass	73.44	**73.86**	70.11	30	70.09	30	38
ionosphere	90.87	92.61	92.33	40	**93.18**	50	32
pima	75.58	75.78	**79.29**	10	76.14	70	32
segment	93.92	97.79	96.67	690	**98.81**	710	40
vehicle	81.57	85.70	84.87	280	**86.05**	310	46
wisconsin	97.18	97.66	97.65	20	**97.70**	50	44

TABLE 4. Wilcoxon test results.

	RW1ELM_Sigmoid	RW1ELM_Gaussian	CC-ELM
RW1ELM_Sigmoid			
RW1ELM_Gaussian	**0.0108**		
CC-ELM	**0.1215**	0.0018	
RWCC-ELM	4.7123 $\times$ E-06	0.0044	5.2561 $\times$ E-06

datasets in Table 2 and Table 3 are reproduced from [4]. Testing efficiencies of WELM for multiclass datasets have been obtained by experimentation using the same parameter setting as in [4]. It can be seen from Table 2 that RWCC-ELM surmounts both, RW1ELM using Sigmoid kernel function and RW1ELM using Gaussian kernel function for most of the datasets.

3) STATISTICAL TEST

For further comparision of the proposed classifier with WELM and CC-ELM, Wilcoxon signed rank test is conducted. For this test, the threshold value is set to 0.05. The results of test are shown in Table 4. WELM classifier using weighting scheme (10) and sigmoid node is referred as RW1ELM_Sigmoid. WELM classifier using weighting scheme (10) and Gaussain kernel is referred as RW1ELM_Gaussian. If the p-Value is less than 0.05, then there is significant difference between the two algorithms. The smaller the p-Value, the difference is more statistically significant. Looking at results in Table 4, it is clear that RWCC-ELM surmounts CC-ELM and WELM using sigmoid activation function and gaussian kernel.

4) PERFORMANCE IN TERMS OF OVERALL ACCURACY

Performance in terms of overall accuracy is shown in Table 3. It can be seen from the table that CC-ELM outperforms RWCC-ELM for 14 datasets out of 30 datasets in terms of overall accuracy. For these 14 datasets, RWCC-ELM outperforms CC-ELM in terms of G-mean at the cost of small drop in overall accuracy. For the remaining 16 datasets, RWCC-ELM outperforms both in terms of G-mean and overall accuracy.

V. CONCLUSION

This paper proposes and evaluates RWCC-ELM, which is single layer complex valued neural network designed for imbalanced real valued classification problems. It incorporates the strength of both CC-ELM and WELM. It uses the same circular transformation function as CC-ELM to map the real valued data to complex domain. Like CC-ELM, it also has complex valued weight and bias. It also uses fully complex sech activation function in the hidden layer. It differs from CC-ELM in the way to compute the weights between hidden and output layer. Like WELM it minimizes weighted least square error along with regularization term to find weights between hidden and output layer. As RWCC-ELM is a complex valued classifier, this paper presents an extended derivation of the expression to find the weights between hidden and output layer in complex domain.

The performance of proposed RWCC-ELM is evaluated on several Keel repository datasets and compared with CC-ELM and WELM. RWCC-ELM superceeds all other classifiers for most of the evaluated datasets. The superiority of RWCC-ELM is also revealed by Wilcoxon signed-rank test. As RWCC-ELM is a complex valued classifier, it can also be used when the input is complex valued, by ommitting the circular transformation phase. The future work may include applying RWCC-ELM on real world applications having complex valued input with large variation in class distribution.

REFERENCES

[1] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Extreme learning machine: Theory and applications," *Neurocomputing*, vol. 70, nos. 1–3, pp. 489–501, 2006.
[2] G.-B. Huang, M.-B. Li, L. Chen, and C.-K. Siew, "Incremental extreme learning machine with fully complex hidden nodes," *Neurocomputing*, vol. 71, nos. 4–6, pp. 576–583, 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] W. Zong, G.-B. Huang, and L. Chen, "Weighted extreme learning machine for imbalance learning," *Neurocomputing*, vol. 101, pp. 229–242, Feb. 2013.
[5] W. Deng, Q. Zheng, and L. Chen, "Regularized extreme learning machine," in *Proc. IEEE Symp. Comput. Intell. Data Mining (CIDM)*, Mar./Apr. 2009, pp. 389–395.
[6] M.-B. Li, G.-B. Huang, P. Saratchandran, and N. Sundararajan, "Fully complex extreme learning machine," *Neurocomputing*, vol. 68, nos. 1–4, pp. 306–314, Oct. 2005.
[7] R. Savitha, S. Suresh, and N. Sundararajan, "Fast learning circular complex-valued extreme learning machine (CC-ELM) for real-valued classification problems," *Inf. Sci.*, vol. 187, pp. 277–290, Mar. 2012.
[8] G. Huang, G.-B. Huang, S. Song, and K. You, "Trends in extreme learning machines: A review," *Neural Netw.*, vol. 61, pp. 32–48, Jan. 2015.
[9] T. Nitta, "The computational power of complex-valued neuron," in *Artificial Neural Networks and Neural Information Processing* (Lecture Notes in Computer Science), vol. 2714, Japan: Springer, 2003, pp. 993–1000.
[10] T. Nitta, "Orthogonality of decision boundaries in complex-valued neural networks," *Neural Comput.*, vol. 16, no. 1, pp. 73–97, 2004.
[11] T. Nitta, "On the inherent property of the decision boundary in complex-valued neural networks," *Neurocomputing*, vol. 50, pp. 291–303, Jan. 2003.
[12] T. Nitta, "Solving the XOR problem and the detection of symmetry using a single complex-valued neuron," *Neural Netw.*, vol. 16, no. 8, pp. 1101–1105, Oct. 2003.
[13] X. Wang, H. Lin, J. Lu, and T. Yahagi, "Channel equalization using complex-valued recurrent neural network," in *Proc. Int. Conf. Info-Tech Info-Net (ICII)*, vol. 3. Beijing, China, 2001, pp. 498–503.
[14] K. Burse, R. N. Yadav, and S. C. Shrivastava, "Channel equalization using neural networks: A review," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 40, no. 3, pp. 352–357, May 2010.
[15] R. Savitha, S. Vigneswaran, S. Suresh, and N. Sundararajan, "Adaptive beamforming using complex-valued radial basis function neural networks," in *Proc. IEEE Region 10 Conf. (TENCON)*, Jan. 2009, pp. 1–6.
[16] A. B. Suksmono and A. Hirose, "Intelligent beamforming by using a complex-valued neural network," *J. Intell. Fuzzy Syst.*, vol. 15, nos. 3–4, pp. 139–147, Dec. 2004.
[17] L.-C. Shi and B.-L. Lu, "EEG-based vigilance estimation using extreme learning machines," *Neurocomputing*, vol. 102, pp. 135–143, Feb. 2013.
[18] Y. Özbay, S. Kara, F. Latifoğlu, R. Ceylan, and M. Ceylan, "Complex-valued wavelet artificial neural network for Doppler signals classifying," *Artif. Intell. Med.*, vol. 40, no. 2, pp. 143–156, 2007.
[19] R. Savitha, S. Suresh, N. Sundararajan, and H. J. Kim, "A fully complex-valued radial basis function classifier for real-valued classification problems," *Neurocomputing*, vol. 78, no. 1, pp. 104–110, 2012.
[20] R. Savitha, S. Suresh, and N. Sundararajan, "A fully complex-valued radial basis function network and its learning algorithm," *Int. J. Neural Syst.*, vol. 19, no. 4, pp. 253–267, 2009.
[21] R. Savitha, S. Suresh, and N. Sundararajan, "A fast learning complex-valued neural classifier for real-valued classification problems," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul./Aug. 2011, pp. 2243–2249.
[22] I. Aizenberg and C. Moraga, "Multilayer feedforward neural network based on multi-valued neurons (MLMVN) and a backpropagation learning algorithm," *Soft Comput.*, vol. 11, no. 2, pp. 169–183, Jan. 2007.
[23] R. Savitha, S. Suresh, and N. Sundararajan, "Fast learning complex-valued classifiers for real-valued classification problems," *Int. J. Mach. Learn. Cybern.*, vol. 4, no. 5, pp. 469–476, Oct. 2013.
[24] R. N. Lichtenwalter and N. V. Chawla, "Adaptive methods for classification in arbitrarily imbalanced and drifting data streams," in *New Frontiers in Applied Data Mining* (Lecture Notes in Computer Science), vol. 5669. Berlin, Germany: Springer, 2010, pp. 53–75.
[25] D. P. Williams, V. Myers, and M. S. Silvious, "Mine classification with imbalanced data," *IEEE Geosci. Remote Sens. Lett.*, vol. 6, no. 3, pp. 528–532, Jul. 2009.
[26] Z.-H. Zhou and X.-Y. Liu, "Training cost-sensitive neural networks with methods addressing the class imbalance problem," *IEEE Trans. Knowl. Data Eng.*, vol. 18, no. 1, pp. 63–77, Jan. 2006.
[27] S. Suresh, S. Saraswathi, and N. Sundararajan, "Performance enhancement of extreme learning machine for multi-category sparse data classification problems," *Eng. Appl. Artif. Intell.*, vol. 23, no. 7, pp. 1149–1157, Oct. 2010.
[28] 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.
[29] H. He and E. A. Garcia, "Learning from imbalanced data," *IEEE Trans. Knowl. Data Eng.*, vol. 21, no. 9, pp. 1263–1284, Sep. 2009.
[30] Z. Shao and M. J. Er, "An online sequential learning algorithm for regularized extreme learning machine," *Neurocomputing*, vol. 173, pp. 778–788, Jan. 2016.
[31] S. Zhou, X. Liu, Q. Liu, S. Wang, C. Zhu, and J. Yin, "Random Fourier extreme learning machine with $\ell_{2,1}$-norm regularization," *Neurocomputing*, vol. 174, pp. 143–153, Jan. 2016.
[32] A. Iosifidis, A. Tefas, and I. Pitas, "Regularized extreme learning machine for multi-view semi-supervised action recognition," *Neurocomputing*, vol. 145, pp. 250–262, Dec. 2014.
[33] A. Iosifidis, A. Tefas, and I. Pitas, "Regularized extreme learning machine for large-scale media content analysis," *Procedia Comput. Sci.*, vol. 53, pp. 420–427, Aug. 2015.
[34] Q. Yu, Y. Miche, E. Eirola, M. van Heeswijk, E. Séverin, and A. Lendasse, "Regularized extreme learning machine for regression with missing data," *Neurocomputing*, vol. 102, pp. 45–51, Feb. 2013.
[35] Y. Peng, S. Wang, X. Long, and B.-L. Lu, "Discriminative graph regularized extreme learning machine and its application to face recognition," *Neurocomputing*, vol. 149, pp. 340–353, Feb. 2015.
[36] J. M. Ortega, *Matrix Theory*. New York, NY, USA: Plenum, 1986.
[37] D. H. Brandwood, "A complex gradient operator and its application in adaptive array theory," *IEE Proc. F (Commun., Radar, Signal Process.)*, vol. 130, no. 1, pp. 11–16, Feb. 1983.
[38] K. Kreutz-Delgado, "Lecture supplement on complex vector calculus: Parameter estimation I," Dept. Elect. Comput. Eng., Univ. California, San Diego, CA, USA, Tech. Rep. ECE275A, 2006.
[39] J. Alcalá-Fdez *et al.*, "KEEL data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework," *J. Multiple-Valued Logic Soft Comput.*, vol. 17, nos. 2–3, pp. 255–287, 2011.

SANYAM SHUKLA received the B.E. degree in computer science and engineering from the Shri Govindram Seksaria Institute of Technology and Science, Indore, India, in 2002, and the M.Tech. and Ph.D. degrees from the Department of Computer Science and Engineering, Maulana Azad National Institute of Technology, Bhopal, in 2009 and 2015, respectively. Since 2005, he has been an Assistant Professor with the Department of Computer Science and Engineering, Maulana Azad National Institue of Technology. He has authored over ten papers in journals and conference proceedings. His research interests include data mining, neural networks, machine learning, optimization techniques, and evolutionary algorithms.

RAM NARAYAN YADAV received the B.E. degree in electronics from Motilal Nehru Regional Engineering College, Allahabad, India, in 1993, the M.Tech. degree in digital communication from the Maulana Azad College of Technology, Bhopal, India, in 1997, and the Ph.D. degree from the Department of Electrical Engineering, IIT Kanpur, India, in 2005. He has been with the Department of Electronics and Communication Engineering, Maulana Azad College of Technology, since 1997, where he is currently a Professor. He has authored several papers in journals and conferences of international repute, including the IEEE TRANSACTIONS. His research interests are communication systems, image processing, pattern recognition, and artificial neural networks and their engineering applications. He is an Associate Member of the Institution of Electronics and Communication Engineers, New Delhi, and the Institution of Communication Engineers and Information Technologists, New Delhi. He has supervised several B.Tech., M.Tech., and Ph.D. theses. He is a Reviewer of the reputed journals, including the journals of the IEEE, Elsevier, Springer, and Taylor and Francis.