International Journal of Engineering Science Invention
ISSN (Online): 2319 – 6734, ISSN (Print): 2319 – 6726
www.ijesi.org || PP—48-59

Extreme Learning Machine Classifier: a topical state-of-the-art survey

Chapala Maharana, Ch. Sanjeev Kumar Dash , Bijan Bihari Mishra
[1]*(Computer Science Department, Eastern Academy of Tech College/ BPUT, Odisha, India)*
[2]*(Computer Science Department, Department,* Silicon Institute of Technology
/ BPUT, Odisha, India)

Abstract : *Extreme Learning machine (ELM) is one of the successful approaches in machine learning that is used these days, particularly for performing pattern classification by many researchers and in many real world applications. It has the key strength of significantly low computational time. This is mostly efficient for training new classifiers. In its working the weights of the hidden and output nodes are randomly chosen and analytically determined, respectively. Too many or too few hidden nodes employed would lead to overfitting or underfitting issues in pattern classification. There is a topical survey of extreme learning machine classifiers for binary classification and multiclass classification.*

Keywords: *Extreme Learning Machine (ELM), Hidden neurons, Linear Discriminant Analysis (LDA), Principal Component Analysis (PCA), Single Layer Feed Forward Neural Networks (SLFN), Support Vector Machine (SVM).*

I. Introduction

According to the analysis of Huang et al. [1] in ELM the word "Extreme" is used to show the moving from general artificial intelligence learning techniques towards brain like learning which is self adaptive and very faster. ELM matches the speed and accuracy of biological learning mechanism. ELM works started according to the biological learning principles and neural network generalization performance theories [2]. Then after the development of ELM works implemented in Frank Rosenblatt's multilayer perceptrons [3], SVM [4], LS-SVM [5], fourier series, linear systems, numeral methods, matrix theories etc. but with necessary extensions [1]. Feedforward network with input output layers but without hidden layers is like the "brain" having eyes and nose that takes sense from outside world acting as input layer and output layers e.g. motor sensors but without "central" neurons. Such a brain has no learning and cognition capability. Neural network research is revived after hidden layers are emphasized in learning since 1980s. Hidden neurons of all networks are needed to be tuned. Many researchers are working hard on learning algorithms mainly by tuning hidden neurons since 1980s.

Both in machine learning and biological learning hidden neurons are important but do not need to be tuned. This results new technique called extreme learning machine (ELM). Extreme learning machine (ELM) invented by G.B. Huang in 2010 that represents a different machine learning technique (including single hidden layer feedforward network and multi hidden layer feedforward network) with the consideration of neural network generalization theory, control theory, matrix theory and linear system theory. This is possible by randomly generating hidden nodes. ELM is the machine learning technique where hidden neurons are not iteratively tuned. ELM is usually considered in neural network theory, control theory, matrix theory and linear system theory [6]. It is a new fast learning neural algorithm referred to as extreme learning machine (ELM) with additive hidden [7] nodes and radial basis function (RBF) kernels. This has been developed for single hidden layer feed forward networks (SLFNs). ELM has been applied to many real world applications and has been shown to generate good generalization performance at extremely high learning speed [7]. Nevertheless, the number of hidden nodes to be used when designing the classifier using ELM for handling different problems; remains a trial and error process. Single hidden layer of ELM covers number of neural networks but it is not limited to sigmoid networks and RBF networks which is referred to as "Generalised Single-Hidden-Layer Feedforward Networks (SLFNs)".

Support Vector Machine (SVM) [4] and the variants are extensively used in classification applications. SVM has mainly two learning features: 1) In SVM, the training data are mapped into a higher dimensional feature space through a nonlinear mapping function Φ(x), and 2) the standard optimization which is used to find the solution of maximizing the separating margin of two different classes in the feature space minimizing the training errors [8]. After the introduction of epsilon insensitive loss function, the SVM has been used to solve regression problems [9]. Training of SVM involves a quadratic programming problem for which it has polynomial time complexity. It is difficult to deal with large problems using single SVMs; rather than in large applications SVM mixtures are used.

Section I. covers Introduction that discusses about traditional classifiers issues and explains how ELM solves those problems. Section II. presents a brief review on ELM classifier and then its working principles. Section III. describes different types of ELM classifiers for binary classification and multiclass classification. Section IV. represents Conclusion.

II. An Overview Of ELM Classifier

Different operations of ELM are: Compression, Feature Learning, Clustering, Regression, and Classification.

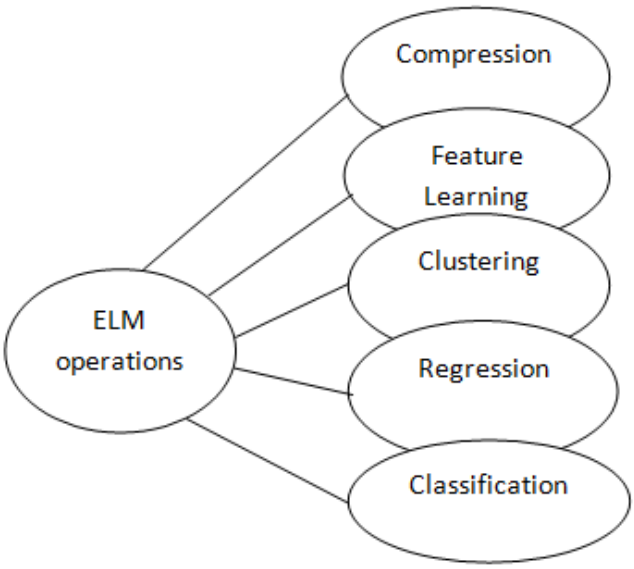

Figure1. ELM Operations

ELM algorithm is used for data classification problems; depends on three parameters [11]. They are (i) number of hidden neurons, ii) the input weights and (iii) the bias values. These are needed to be optimally chosen. Neural networks are mostly used in complex nonlinear mappings directly from the input sample. It has the disadvantage of more learning time.

The learning phase of ELM is completed in less than seconds for many datasets. The conventional learning algorithms for example feedforward back propagation network algorithm takes very long time to train the network. The ELM has better generalization performance than the gradient based learning e.g. Back propagation algorithm. Classical learning algorithms have problems of local minima, improper learning rate and overfitting etc. [12] which is avoided by some issues like weight decay and early stopping methods.

One key principle of the ELM is that one may randomly choose and fix the hidden node parameters. ELM is originally developed for the single-hidden layer feed forward neural networks (SLFNs) and then extended to the "generalized" SLFNs. After the hidden node parameters are chosen randomly, SLFN becomes a linear system where the output weights of the network can be analytically determined using simple generalized inverse operation of the hidden layer output matrices. One of the salient feature of ELM is that the weights from the input layer to the hidden layer are randomly generated [13] , [14]. The paper [11] further proves the random feature mapping theory rigorously.

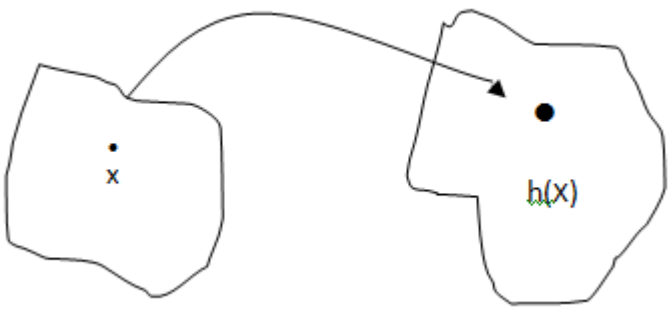

Figure2. ELM random feature mapping: h(x)

The hidden layer output function (hidden layer mapping, ELM feature space):

$h(x) = [G(a_1, b_1, x), \ldots, G(a_L, b_L, x)]$

The output functions of hidden nodes are limited to

1. Sigmoid: $G(a_i, b_i, x)=g(a_i.x+b_i)$
2. RBF: $G(a_i, b_i, x)=g(b_i\|x-a_i\|)$
3. Fourier Series: $G(a_i, b_i, x)= \cos(a_i.x+b_i)$

Conventional Random projection is just a specific case of ELM random feature mapping (ELM feature space) when linear additive hidden node is used. After the random nonlinear feature mapping in the hidden layer, the rest of ELM can be considered as a linear system [14]. Therefore, ELM has a closed form of solution due to the simple network structure and random hidden layer weights. The essence of the linear system used by ELM is to minimize the training error and the norm of connection weights from the hidden layer to the output layer at the same time [8]. Hence ELM has a good generalization performance according to the feed forward neural network theory [2, 15]. As a consequence, ELM has some desirable features, such as that hidden layer parameters need not be tuned. It has fast learning speed and good generalization performance. Additionally ELM has a unified framework for classification [8].

Random Projection: $G(a_i, b_i, x)=a_i.x$

ELM functions for Complex Networks are

- **Circular functions:**

$$:\tan(z) = \frac{e^{iz} - e^{-iz}}{i(e^{iz} + e^{-iz})} \text{.....(1)}$$

$$:\sin(z) = \frac{e^{iz} - e^{-iz}}{2i} \text{.....(2)}$$

- **Inverse circular functions:**

$$:\arctan(z) = \int_0^z \frac{dt}{1+t^2} \text{.....(3)}$$

$$:\arcsin(z) = \int_0^z \frac{dt}{(1-t)^{1/2}} \text{.....(4)}$$

$$:\arccos(z) = \int_0^z \frac{dt}{(1-t^2)^{1/2}} \text{.....(5)}$$

- **Hyperbolic functions:**

$$:\tanh(z) = \frac{e^{z} - e^{-z}}{i(e^{z} + e^{-z})} \text{.....(6)}$$

$$:\sinh(z) = \frac{e^{z} - e^{-z}}{2} \text{.....(7)}$$

- **Inverse hyperbolic functions:**

$$:\arctan h(z) = \int_0^z \frac{dt}{1-t^2} \text{.....(8)}$$

$$:\arcsin h(z) = \int_0^z \frac{dt}{(1+t)^{1/2}} \text{...(9)}$$

ELM usually has to randomly generate a great number of hidden nodes to achieve desirable performance. This leads to time consuming in test process, which is not helpful in real applications. Large numbers of hidden nodes also easily make the trained model stack into overfitting. The random selection of hidden neurons may cause the problem of either underfitting or overfitting. Overfitting arises because the network matches the data so closely as to lose its generalization ability over the test data [16]. The excessive hidden neurons will cause overfitting; that is, the neural networks have over estimate the complexity of the target problem [17]. Overfitting is caused due to many irrelevant neurons in the hidden layers present unnecessarily in the network. Underfitting is caused due to less number of neurons present in the network as

compared to the complexity of the problem data [16]. The ELM network has an extreme high learning speed due to the simple network structure and its closed form solution. Additionally, the randomness makes ELM not necessarily true about these hidden layer parameters iteratively. The output layer is a linear system, where the connection weights from the hidden layer to the output layer are learned by computing the Moore-Penrose generalised inverse [11].

Generally a classifier network that is too small lacks the capability of learning the training data of sufficiently well. On the other hand, a network that is too large could also overfit the training data; thus producing the poor generalization performance on unseen cases. Besides, an extremely large network also brings about larger prediction responses and unnecessary requirement for large memory as well as high cost for hardware implementation.

The essence of hidden layer functions is to map the data into a feature space, where the output layer can use a linear classifier to separate the data perfectly. Therefore, the hidden layer should extract discriminative features or some other data representations for classification tasks.

LDA [18] is probably the most commonly used method to extract discriminative features. "Small Sample Size" (SSS) problem and Gaussian distribution assumption of equal covariance and different means. Su et al. [19] proposed a projection pursuit based LDA method. The method showed that difference vectors of between class samples have a strong discriminative property for classification tasks.

Salient Features of Extreme Learning Machines (ELM) are:

- "Simple Math is Enough" i.e., ELM is a simple tuning free three step algorithm.
- Learning speed of ELM is extremely fast.
- The hidden node parameters are independent of training data and independent of each other
- Hidden nodes of ELM need not be tuned.
- ELM network generate hidden nodes before seeing the training data.

Basic Structure of ELM

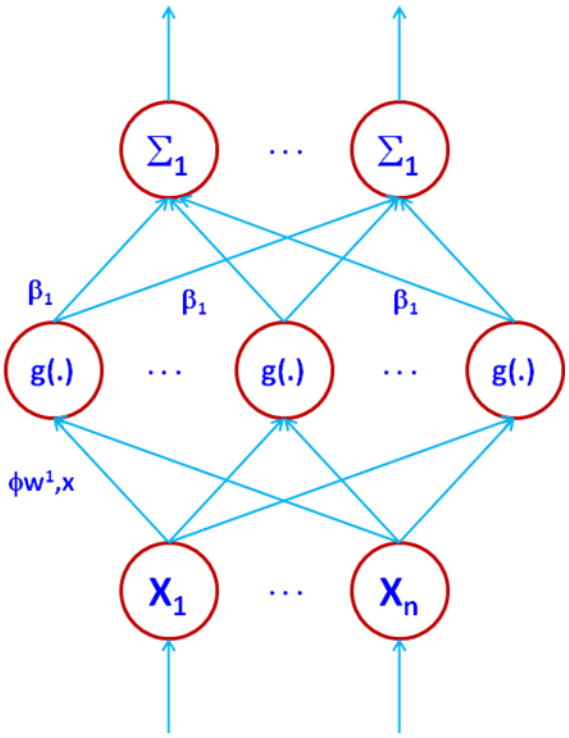

Figure3. Illustration of the structure of ELM neural network.

ELM training Algorithm.

Input: training set $\{(x_k, t_k) | x_k \varepsilon \mathbf{R}^n, t_k \varepsilon \mathbf{R}^m, k=1,---N\}$

Output: SLFN parameters.

1: Randomly generate hidden node parameters (a_k,b_k), k=1,…,L where a_k and b_k are the input weight and bias values and L is the hidden node number.

2: Calculate the hidden layer output matrix **H.**

3: Calculate the output weight matrix β:

$\beta=\mathbf{H}^{+}T$

Where

$$\mathbf{H}=\begin{pmatrix} h(x_1) \\ \vdots \\ h(x_3) \end{pmatrix}=\begin{pmatrix} G(a_1,b_1,x_1) & \dots & G(a_L,b_L,x_1) \\ \vdots & \cdots & \vdots \\ G(a_1,b_1,x_N) & \cdots & G(a_L,b_L,x_N) \end{pmatrix}_{N\times L} \quad \beta=\begin{pmatrix} \beta_1^T \\ \vdots \\ \beta_L^T \end{pmatrix}_{L\times m} \quad \textbf{and} \quad \mathbf{T}=\begin{pmatrix} t_1^T \\ \vdots \\ t_N^T \end{pmatrix}_{N\times m}$$

T is the training sample vector; **H** is called the hidden layer output matrix of the neural network. Once the hidden layers are randomly generated, the parameters (a_i, b_i) remain fixed. Training an SLFN is equivalent to finding a least squares solution β of the linear system.

: $H\beta=T$ …… (10)

: $\beta = H^{\dagger}T$……. (11)

Where $\mathbf{H}^{\dagger}$ is the Moore- Penrose generalized inverse of matrix [20].

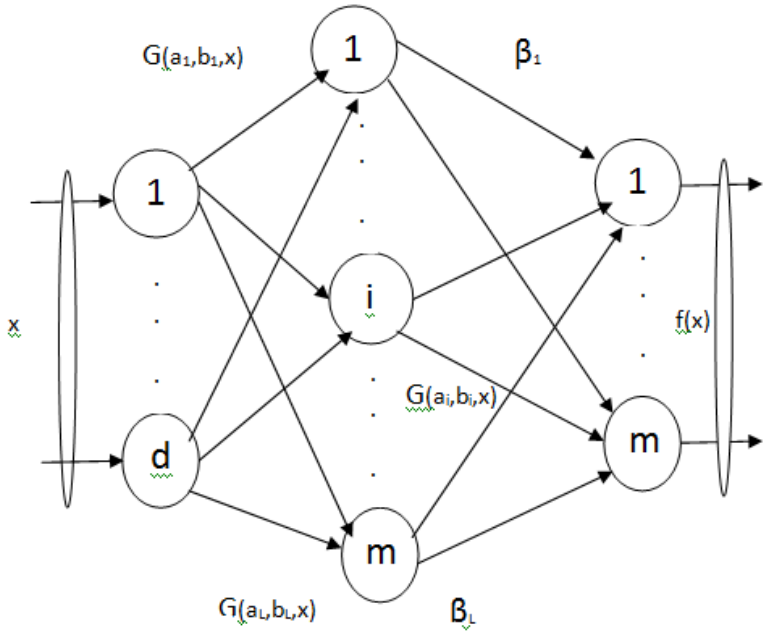

Figure3. SLFN as used by ELM [21].

Essential considerations of ELM

- High Accuracy
- Least User Intervention
- Real Time Learning (in seconds, milliseconds, even in microseconds)

ELM for Threshold Networks

- Binary/ Threshold node:

$$g(x)=\begin{cases} 1, x \geq 0 \\ 0, x < 0 \end{cases}$$

- Threshold networks are usually trained by Backpropagation and its variants indirectly in the past three decades. There was no direct learning solution to threshold networks in the past 60 years.
- Threshold unit can be approximated by sigmoid unit g(x) = 1/ (1+exp(-λx)) with sufficiently large gain parameter λ.
- With ELM, threshold networks can be trained directly.

ELM solutions

1. Ridge regression theory based ELM

   $f(x)=h(x)\beta=h(x)H^T(HH^T)^{-1}T=>h(x)\ H^T(\frac{I}{C}+HH^T)^{-1}\ T$ and …….. (12)

   $f(x)=h(x)\beta= h(x)(\ H^TH)^{-1}H^TT=>\ h(x)(\ \frac{I}{C}+\ H^TH)^{-1}\ H^TT$…… (13)

2. Equivalent ELM optimization formula

Minimize: $L_{P_{ELM}} = \frac{1}{2}\|\beta\|^2 + C\frac{1}{2}\sum_{i=1}^{N}\|\xi_i\|^2$ .....(14)

Subject to: $h(x_i)\beta = t_i^T + \xi_i^T, \forall i$ .....(15)

Here I is the unit matrix, C is a user specified parameter that provides a tradeoff between the minimization of the training errors and the maximization of the marginal distance. H is the hidden layer output matrix, T is the training data target matrix.

In the standard ELM algorithm proposed in [23], the number of hidden nodes, Ñ, to use is a decision to be made by the designer of the classifier network. Often a trial and error process is conducted in advance, in order to arrive at networks that produce good generalization performances. To avoid this P-ELM algorithm is proposed as a systematic and automated approach for defining the hidden node size Ñ in the design of classifier networks.

III. Types Of ELM Classifiers

3.1 Pruned_ELM (PELM)

The main idea lies in identifying the degree of relevance between the hidden nodes and the class labels with the use of some statistical measure and then removing the irrelevant or low relevant hidden nodes to arrive at compact networks while keeping the generalization ability of the network uncompromised.

Initially P-ELM begins with a large number of hidden neurons where the weight parameters are randomly assigned to obtain its response of input vectors. Hidden nodes are pruned that is kept intact basing on the relevance of each hidden node in finding prediction accuracy of the classifier. P-ELM is sensitive to the relevance threshold level used i.e., the classifier prediction accuracy level varies according to the threshold used in it. Threshold is identified based on the Akaike information criterion (AIC).

P-ELM identifies the degree of relevance between the hidden nodes and the class labels by using statistical measure. Then it removes the low relevant hidden nodes to find the compact network without compromising the generalization ability of it

At the beginning in P-ELM an initial network of (Ñ) hidden node size is assigned. All hidden node parameters i.e., ((c_i ,ai), i=1,…, Ñ) are randomly assigned in the standard methodology of ELM. Matrix H is obtained by finding the responses of hidden nodes. Statistical relevance of each hidden node that contribute true class label is then identified. Here two statistical measures namely Chi-squared (χ^2), [29, 30] and information gain (IG) [31, 32] are used for revealing the statistical relevance of hidden nodes in arriving at the true class labels. Responses of the hidden nodes are discretized according to the entropy based discretization method [33] before the statistical methods are put in place.

P-ELM Algorithm. [7] Given a training set $\aleph = \{(x_k, t_k) | x_k \in \mathbf{R}^n, t_k \in \mathbf{R}^m, k=1, \ldots, N\}$, activation function g, an initial larger hidden node size Ñ, and a relevance threshold base γ ($\gamma_1, \gamma_2, \ldots, \gamma_q$). AIC is given by,

$$AIC(i) = 2N_{val}\ln(\delta_i^2 / N_{val}) + S_i, i = 1, 2, ..., q$$

δ_i

where N_{val} is the number of validation data; δ_i is the classification error for the i[th] relevance threshold; S_i is the hidden node size for the i[th] relevance threshold. AIC is used to achieve a trade-off between prediction accuracy and the structural complexity of the final network.

P-ELM Algorithm

1. Separate the training set into two non-overlapping subsets for learning and validation.
2. Randomly assign hidden node parameters (c_i, a_i), i=1,…, Ñ .
3. Calculate the hidden layer output matrix H using the learning subset.
4. Calculate the statistical relevance for the hidden nodes using the statistical measures (χ^2 or IG) then sort them in descending order.
5. For each relevance threshold γ_i (i =1,2,…. q).
6. Select S* according to min (AIC).
7. Retrain network S^* with the whole training set.
- Calculate the hidden layer output matrix H* using the training set.
- Calculate the output weight β*:β*=(H*)ŦT.
8. Evaluate the performance of S* on unseen testing data set.

The statistical methods χ^2 and IG are used for pruning the insignificant nodes which are denoted as P-ELM1 and P-ELM2, respectively.

3.2 Twin Extreme Learning Machine (TELM)

Expanding the SVM, twin support vector machine [23] is used that targets at deriving two nonparallel separating hyperplanes. Each hyperplane reaches the smallest distance to one of the two classes and makes its distance the other class as large as possible. It has faster learning by solving two reduced-sized QPPs. Twin extreme learning machine (TELM) [24] learns two nonparallel separating hyperplanes in the ELM feature space for data classification. TELM is used to minimize its distance to one of the two classes for each hyperplane and keeps it far away from the other class. TELM tries to minimize both the training error and the sum of squares of the distance from one hyperplane to one of the two classes.

There are three main differences between traditional ELM and TELM for binary classification problem:

1. TELM aims at generating two nonparallel separating hyperplanes, however, ELM finds only one separable hyperplane.
2. ELM tends to reach zero training errors, however in TELM training errors are not equal to zero. This leads to better generalization in testing data.
3. TELM has two objective functions and needs to solve two quadratic programming problem (QPPs).

TELM algorithm [24]

Given a training set $\mathbf{X}= \{(x_i, t_i)|x_i \in \square^{d}$, $t_i=\{+1, -1\}$, $i=1, \ldots N\}$, activation function G(x), and the number of hidden node is L. U and V are used to denote the hidden layer output of the data points belonging to class +1 and class -1, respectively.

$$U = \begin{pmatrix} u_1 \\ \vdots \\ u_{m_1} \end{pmatrix} = \begin{pmatrix} h_1(x_1) & \cdots & h_L(x_1) \\ \vdots & \ddots & \vdots \\ h_1(x_{m_2}) & \cdots & h_L(x_{m_2}) \end{pmatrix}$$

$$V = \begin{pmatrix} v_1 \\ \vdots \\ v_{m_2} \end{pmatrix} = \begin{pmatrix} h_1(x_1) & \cdots & h_L(x_1) \\ \vdots & \ddots & \vdots \\ h_1(x_{m_2}) & \cdots & h_L(x_{m_2}) \end{pmatrix}$$

$$\max_{\alpha} \; e_2^{T}\alpha - \frac{1}{2}\alpha^{T}V(U^{T}U+\varepsilon I)^{-1}V^{T}\alpha \ldots(1)$$

$$s.t.\ 0 \le \alpha_i \le c_1, \quad i = 1,\ldots,m_2.$$

$$\max_{\gamma} \; e_1^{T}\gamma - \frac{1}{2}\gamma^{T}U(V^{T}V+\varepsilon I)^{-1}U^{T}\gamma) \ldots(2)$$

$$s.t. \quad 0 \le \gamma_i \le c_2, \quad i = 1,\ldots,m_1$$

$$\max_{\alpha} \; e_2^{T}\alpha - \frac{1}{2}\alpha^{T}S(R^{T}R+\varepsilon I)^{-1}S^{T}\alpha) \ldots(3)$$

$$s.t. \quad i = 1,\ldots,m_2.$$

$$0 \le \alpha_i \le c_1$$

$$\max_{\gamma} \; e_1^{T}\gamma - \frac{1}{2}\gamma^{T}R(S^{T}S+\varepsilon I)^{-1}R^{T}\gamma) \ldots(4)$$

$$s.t. \quad 0 \le \gamma_i \le c_2, \; i = 1,\ldots,m_1$$

$$(\beta_1 = -(U^{T}U+\varepsilon I)^{-1}V^{T}\alpha) \ldots(5)$$

$$(\beta_2 = -(V^{T}V+\varepsilon I)^{-1}U^{T}\gamma) \ldots(6)$$

$$(\mu_1 = -(R^{T}R+\varepsilon I)^{-1}S\alpha) \ldots(7)$$

$$(\mu_2 = -(S^{T}S+\varepsilon I)^{-1}R^{T}\gamma) \ldots(8)$$

$$(f(x) = \arg\min_{r=1,2} d_r(x) = \arg\min_{r=1,2} |\beta_r^T h(x)|)...(9)$$

1. Initiate an ELM network with L hidden nodes using the random input weight w_i and bias b_i.
2. Construct input matrices A and B. Calculate their hidden layer output matrices U and V respectively for linear TELM and R and S respectively for nonlinear TELM.
3. Construct convex QPPs (1) and (2) for linear TELM then (3) and (4) for nonlinear TELM.
4. Obtain Lagrange multipliers α and γ by solving two QPPs.
5. For linear TELM, calculate the output weights β_1 and β_2 using (5) and (6) for nonlinear TELM calculate

   $\mu_1 and \mu_2 u \sin g(7) and (8)$
6. Calculate the perpendicular distance of data point x from the separating hyperplane using (9) then assign the x to class i (i=+1, -1).

3.3 Class Specific Cost Regulation ELM (CCR-ELM)

The classification methods for imbalanced data are only focused on binary classification. Multiclass classification suffers from imbalanced data distribution problem. There the majority classes and minority classes are difficult to be identified. To avoid such problems class-specific cost regulation ELM (CCR-ELM) [27] can be used for classification problems with imbalanced data distributions. This introduces class specific regulation cost for misclassification of each class in terms of performance index. This reduces the effects of number of class samples and even the effects of dispersion degree of data. This method can be directly used for binary classification and multiclass classification. This is done to keep the separation boundary in the ideal position so that it can rebalance the proportion of the two classes.

Imbalanced ratio (IR) is defined to present the imbalanced degree of the dataset

For binary classification: $IR = \frac{N_+}{N_-}$.....(1)

For multiclass classification: $IR = \frac{N_{min}}{N_{max}}$...(2)

Where N_+ and N_- are positive sample numbers and the negative sample numbers, N_{min} and N_{max} are the sample number of the class with maximum sample and the class with minimum sample, respectively.

3.3.1 CCR-ELM for binary classification

Without loss of generality, binary classification problem is considered. There are lots of negative samples and few positive samples in imbalanced binary classification problem. Both positive samples and negative samples are assigned with same value of C [a user specified parameter that provides tradeoff between minimization of training errors and maximization of marginal distance]. ELM solves the optimization problem to get better generalization performance which is written mathematically as

$Minimize:$

$$: L_{P_{ELM}} = \frac{1}{2}\|\beta\|^2 + \frac{1}{2}C\sum_{i=1}^{N}\xi_i^2 .....(3)$$

$$s.t.,\ h(x_i)\beta = t_i - \xi_i,\ i = 1,...,N$$

Where $\xi_i = [\xi_{i,m},...,\xi_{i,m}]$ is the training error vector of the m output nodes with respect to training sample x_i. for correct classification of imbalanced data the separation boundary is let to be near the ideal position as far as possible. CCR-ELM sets two parameters C^+ for minority positive samples and C^- for majority negative samples. Number of positive samples is l_1, and the number of majority negative samples is l_2, then CCR- ELM is described as

$$: \min(\frac{1}{2}\|\beta\|^2 + \frac{1}{2}C^+\sum_{i=1|t_1=+1}^{l_1}\xi_i^2 + \frac{1}{2}C^-\sum_{i=1|t_1=-1}^{l_2}\xi_i^2)..(4)$$

C^+ and C^- are used as the regulation cost for misclassification in the performance index specifically for majority negative samples and minority positive samples.

$$\|\varepsilon_+\|^2 = \sum_{i=1|t_1=+1}^{l_1}\xi_i^2 ...(5) \qquad \|\varepsilon_-\|^2 = \sum_{i=1|t_1=-1}^{l_2}\xi_i^2 ....(6)$$

where ε_+ stands for sum error of minority positive samples and ε_- stands for sum error of majority negative samples. Henceforth CCR-ELM uses the optimization equation

$$Min(\frac{1}{2}\|\beta\|^2+\frac{1}{2}C^+\|\varepsilon_+\|^2+\frac{1}{2}C^-\|\varepsilon_-\|^2)...(7)$$

for its description.

CCR-ELM uses the following for binary classification from $f(x)=signh(x)\beta$:

$$\begin{cases} signh(x)H^T(\frac{I}{C^+}+\frac{I}{C^-}+H^TH)^\dagger T, N<L...(8) \\ signh(x)(\frac{I}{C^+}+\frac{I}{C^-}+H^TH)^\dagger H^TT, \ N>L...(9) \end{cases}$$

3.3.2 CCR-ELM for multiclass classification

For multiclass classification with D classes, a parameter C^d is set for dth class (d=1, 2,…, D). The multiclass classification optimization equation is

$$\min\ (\frac{1}{2}\|\beta\|^2+\frac{1}{2}\sum_{d=1}^{D}C^d\sum_{i=1|t_i=d}^{l_d}\xi_i^2)...(10)$$

$$s.t., h(x_i)\ \beta=t_i-\xi_i,\ i=1,...,N$$

The sum error of the d[th] class is ε_d

$$\|\varepsilon_d\|^2=\sum_{i=1|t_i=d}^{l_d}\xi_i^2...(11)$$

The output weight vector β of CCR-ELM is found for multiclass classification as

$$\beta=\begin{cases} H^T(\sum_{d=1}^{D}\frac{I}{C^d}+H^TH)^\dagger T, N<L \\ \\ (\sum_{d=1}^{D}\frac{I}{C^d}+H^TH)^\dagger H^TT, N>L \end{cases} ...(12)$$

3.4 Multilayer Extreme Learning Machine (ML-ELM)

This is one of the classification systems used for classification of motor imagery electroencephalogram (EEG). This is one of the most important technologies used for BCI [26]. In this system, the combination of PCA and LDA is chosen as the feature extraction method and the ML-ELM [26] algorithm is used for classification. ML-ELM has better performance than ELM.

In the ML-ELM system, feature vectors of original EEG signals are extracted using the combination method of PCA and LDA. The feature vectors are sent to the classification part based on ML-ELM. The result of classification is regarded as the criteria to evaluate the system.

The combination of PCA and LDA [18] procedure of feature extraction method to select features as following:

1. Perform PCA on channel A_1 and A_2 of data set to obtain compact PCA features. Then determine the dimension which is reduced according to Accuracy Contribution Rate (ACR) of new space's first few dimensions.
2. Perform LDA on 16-dimensional features which are converted to 1 dimension as they are binary class.
3. Combine LDA features of Channel A1 and A2 to form a new feature of 18 dimensions as the input to ML-ELM.

Multilayer ELM is a multiplayer neural network. It is based on ELM auto-encoder as their hidden layer hidden layer weights are initialized with ELM-AE. It performs layer-wise unsupervised training.

ML-ELM hidden layer activation functions can be either linear and nonlinear in piecewise. If number of nodes L_k in k^{th} layer and L_{k-1} in $(k-1)^{th}$ hidden layer are equal then g (sigmoidal function) is chosen as linear otherwise nonlinear.

$$H^K = g((\beta^k)H^{k-1})$$

H^K is the k^{th} hidden layer output matrix. The input layer x can be considered as the 0^{th} hidden layer, for k=0. The output of connections between last hidden layer node and output node t is calculated analytically using regularized least squares. The parameters hidden nodes and activation function can be set directly as mentioned in [20].

The structure of Multilayer ELM

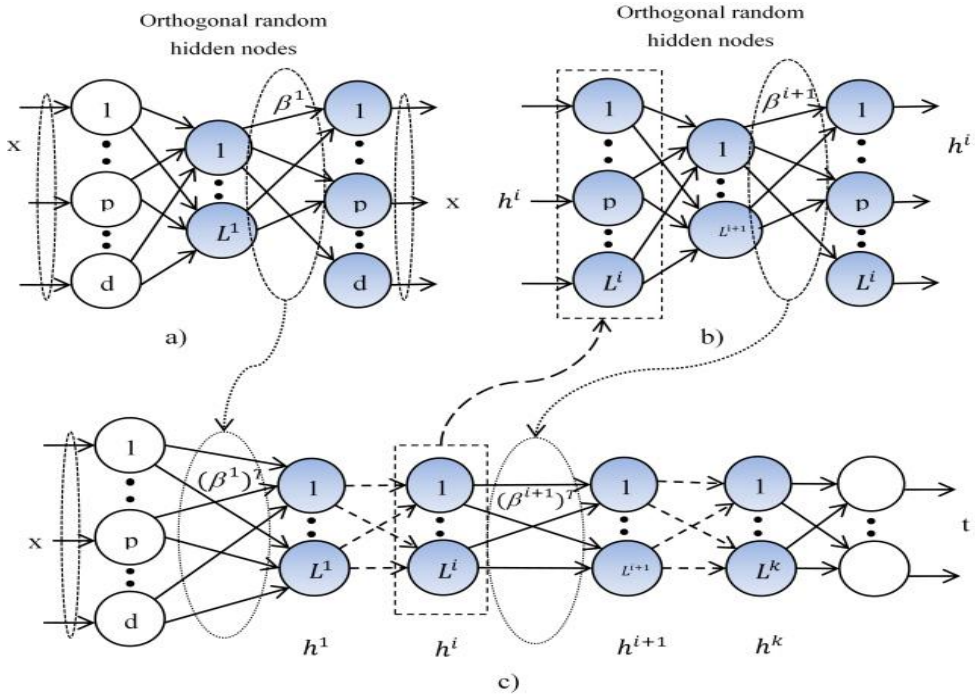

Figure4. Adding layers in ML-ELM [45]

(Figure4a.) ELM-AE output weights β^1 with respect to input data x are the first layer weights of ML-ELM.
(Figure4b.) The output weights β^{i+1} of ELM-AE with respect to i^{th} hidden layer output h^i of ML-ELM are the $(i+1)^{th}$ layer weights of ML-ELM.
(Figurre4c.) The ML-ELM output layer weights are calculated using regularized least squares.

IV. Conclusion

We conclude that ELM classifiers are faster Machine learning classifiers. They are widely used in real world applications for their generalized property. Impact of hidden layer neurons is negligible in the generalization of the ELM model. P-ELM algorithm is insensitive to the initial number of hidden nodes. But a large number of initial hidden nodes generally leads to increased computational burden in P-ELM. TELM has same or better performance on the binary classification problem compared to other famous classification algorithms for example ELM, SVM and TSVM [34]. TELM performs much better on multiclass classification problem especially when number of sample features is large. CCR-ELM can significantly improve the classification performance compared with the original ELM with imbalanced data distribution. ML-ELM performs well for binary class signals for motor imagery.

References

[1]. G.B. Huang, *An insight into extreme learning machines: random neurons, random features and kernels*, Cogn Comput. Springer Paper, *6(3)*, 2014, 376-90.
[2]. P.L. Bartlett, *The sample complexity of pattern classification with neural networks: the size of the weights is more important than the size of the network*, IEEE Trans Inform Theory, 44(2), 1998, 525-36.
[3]. F. Rosenblatt, *The perceptron: a probabilistic model for information storage and organization in the brain*, Psychol Rev, 65(6), 1958, 386-408
[4]. C. Cortes, V. Vapnik, *Support vector Networks,* Mach Learn. *20(3)*, 1995; 273-97.
[5]. J.A.K Suykens, J. Vandewalle, *Least squares support vector machine classifiers*, Neural Process Lett., *9(3*), 1999, 293-300.

[6]. G.B. Huang, *What are Extreme Learning Machines? Filling the Gap Between Frank Rosenblatt's Dream and John von Neumann's Puzzle*, Springer Paper Cogn Comput. *7*, 2015, 263–278.

[7]. Hai-Jun Rong, Yew-Soon ong*, Ah-Hwee Tan, Zexuan Zhu, *A fast pruned- extreme learning machine for classification problem*, Elsevier Science Direct Neurocomputing, *72*, 2008, pp.359-366.

[8]. G.-B. Huang, H. Zhou, X.Ding, and R. Zhang, *Extreme Learning Machine for Regression and Multiclass Classification*, in IEEE Transactions on Systems, Man, and Cybernetics- Part B, *42(2)*, April 2012.

[9]. H. Drucker, C.J. Burges, L. Kaufman, A. Smola, V.Vapnik, M.Mozer, J. Jordan, and T. Petscbe, Eds. Cambridge, MAMIT Press, *Support vector regression machines*, Neural Information Processing Systems, *9*, 1997 155-161.

[10]. N.M. Dash, R. Priyadarshini, S. Rout, *Performance Comparision of Back propagation Neural Network and Extreme Learning machine for Multinomial Classification Task* , COMPUSOFT, (IJACT), *2(11)*, Nov. 2013.

[11]. G.-B. Huang, Q.-Y. Zhu and C.-K. Siew, *Extreme learning machine: Theory and applications*, Neurocomputing, 70(1-3),Dec. 2006, 489-501

[12]. M.C. Sezgin, Z. Dokur, T. Olmez, M. Korurek, Classification *of Respiratory Sound by using An Neural Network*, Proceedings-23rd Annual Conference IEEE/ EMBS, 2001.

[13]. W. Zhu, J. Miao, L. Qing, *Constrained Extreme Learning Machines: A Study on Classification Cases* , Natural Science Foundation of China (No. 61175115 and 61272320) and President Fund of Graduate University of Chinese Academy of Sciences (No. Y35101CY00).

[14]. G.B. Huang, Q.Y. Zhu, C.K. Siew, *Extreme learning machine: A new learning scheme of feedforward neural network,* Proceedings of International Joint Conference on Neural Networks (IJCNN2004), 2 (Budapest, Hungary), Jul 2004. 985-990.

[15]. B. Widrow, A. Greenblat, Y. Kim and D. Park, *The No- Prop algorithm: A new learning algorithm for multilayer neural networks*, Neural Networks, *37*, 2013, 182-188.

[16]. F.S. Panchal, M. Panchal, *review of methods on selecting number of hidden nodes in artificial neural network*, in International Journal of Computer Science and Mobile Computing, A monthly Journal of Computer Science and Information Technology. Foram S.Panchal et al, International Journal of Computer Science and Mobile Computing, 3(11), Nov 2014, 455-464.

[17]. K. Jinchuan and L. Xinzhe, *Empirical analysis of hidden neurons in neural network modeling for stock prediction*, in proceedings of the Pacific-Asia Workshop on Computational Intelligence and Industrial Application, *2*, Dec 2008, 828-832.

[18]. K.P. Murphy, *Machine Learning a probabilistic perspective*, The MIT Press 2012.

[19]. Y. Su, S. Shan, X. Chen, W. Gao, *Classifiability based discriminatory projection pursuit*, IEEE Trans. Neural Networks, *22(12)*, 2011, 2050-2061.

[20]. G.-B. Huang, Q.-Y. Zhu and C.-K. Siew, Extreme learning machine: Theory and applications, Neurocomputing, Dec. 2006, *70(1-3)*, .489-501

[21]. M.C. Sezgin, Z. Dokur, T. Olmez, M. Korurek, *Classification of Respiratory Sound by using An Neural Network*, Proceedings-23rd Annual Conference IEEE/ EMBS. (2001).

[22]. W. Zhu, J. Miao, L. Qing, *Constrained Extreme Learning Machines: A Study on Classification Cases* , Natural Science Foundation of China (No. 61175115 and 61272320) and President Fund of Graduate University of Chinese Academy of Sciences (No. Y35101CY00).

[23]. G.B. Huang, Q.Y. Zhu, C.K. Siew, *Extreme learning machine: A new learning scheme of feedforward neural networks*, Proceedings of International Joint Conference on Neural Networks (IJCNN2004), Jul 2004, *2 (Budapest, Hungary)*, 985-990.

[24]. B. Widrow, A. Greenblat, Y. Kim and D. Park, *The No- Prop algorithm: A new learning algorithm for multilayer neural networks*, Neural Networks, 2013, *37*,182-188.

[25]. F.S. Panchal, M. Panchal, *review of methods on selecting number of hidden nodes in artificial neural network*, in International Journal of Computer Science and Mobile Computing, A monthly Journal of Computer Science and Information Technology. Foram S.Panchal et al, International Journal of Computer Science and Mobile Computing Nov. 2014, 3(11)455-464.

[26]. K. Jinchuan and L. Xinzhe, *Empirical analysis of hidden neurons in neural network modeling for stock prediction,* in proceedings of the Pacific-Asia Workshop on Computational Intelligence and Industrial Application, Dec 2008; *2*, 828-832.

[27]. K.P. Murphy, *Machine Learning a probabilistic perspective*, The MIT Press 2012.

[28]. Y. Su, S. Shan, X. Chen, W. Gao, Classifiability based discriminatory projection pursuit, IEEE Trans. Neural Networks, 2011, *22(12*), 2050-2061.

[29]. P.Courrieu, *Fast computation of moore-penrose inverse matrices*, 2008 arXivpreprint arXiv, 0804- 4809.

[30]. J.L. Fandino, P.Q. Barriuso, D. B. Heras, and F. Arguello, *Efficient ELM Based Techniques for the classification of Hyperspectral Remote sensing Images on Commodity GPUs*, in IEEE Journal of selected topics in Applied Earth Observations and Remote Sensing. June 2015, *8(6)*.

[31]. J. L. Fandiño, P. Q. Barriuso, D. B. Heras, F. Argüello, *Efficient ELM-Based Techniques for the Classification of Hyperspectral Remote Sensing Images on Commodity GPUs*, IEEE Journal Of Selected Topics in Applied Earth Observations And Remote Sensing, Jun 2015, *8(6)*.

[32]. R. Khemchandani, S. Chandra, et al.. In *Twin support vector machines for pattern classification*, in IEEE Trans. Pattern anal. Mach Intell. 2007, *29(5)*, 905-910.

[33]. Y. Wan, S. Song, G. Huang, S. Li, *Twin Extreme learning machines for pattern classification*, in Neurocomputing, Elsevier Paper, Neurocomputing 2017, *260*, 235-244.

[34]. S. Tamura, M. Tateishi, *Capabilities of a four layered feedforward neural network: four layers versus three*, IEEE Trans. Neural Networks, 1997, *8(2)*, 251-255

[35]. Lijuan Duan, Menghu Bao, Jun Miao, Yanhui Xu and Juncheng Chen. *Classification based on Multilayer Extreme Learning Machine for Motor Imagery Task from EEG signals*, Elsevier Procedia Computer Science 7th Annual International Conference on Biologically Inspired Cognitive Architectures, BICA 2016, *88*, 176-184.

[36]. W.Xiao, J.Zhang, Y. Li, S. Zhang, W. Yang, *Class-specific cost regulation extreme learning machine for imbalanced classification*, in Neurocomputing, Elsevier paper, Neurocomputing, 2017, *261*, 70-82.

[37]. I. d. Campo, V. Martinez, F. Orosa, J. Echanobe, e. Asua, K. Basterretxea *Piecewise Multi-linear Fuzzy Extreme Learning Machine for the Implementation of Intelligent Agents*, IEEE paper 2017, 978-1-5090-6182-2.

[38]. H.-Q. Liu, J.-Y. Li, L.-S. Wong, *A comparative study on feature selection and classification methods using gene expression profiles and proteomic patterns*, Genome Informatics 2002, *13*, 51–60.

[39]. Z.-X. Zhu, Y.-S. Ong, M. Dash, *Wrapper-filter feature selection algorithm using a memetic framework*, IEEE Trans. Systems, Man Cybern., Part B, 2007, 37(1), 70–76.

[40]. R.B. Ash, Information Theory, Wiley, 1965.

[41]. I.H. Witten, E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques*, second ed., Morgan Kaufmann, San Francisco, 2005.
[42]. U.M. Fayyad, K.B. Irani, *Multi-interval discretization of continuous valued attributes for classification learning*, Proceedings of theThirteenth International Joint Conference on Artificial Intelligence,San Francisco, CA, Morgan Kaufmann, Los Altos, CA, 1993, 1022–1029.
[43]. R. Khemchandani , S. Chandra , et al. , *Twin support vector machines for pattern classification*, IEEE Trans. Pattern Anal. Mach. Intell. 2007, *29(5)*, 905–910.
[44]. I.T. Jolliffe, *Principal Component Analysis*, second ed., New York, Springer-Verlag, 2002.
[45]. L.L.C. Kasun, G.B. Huang et al, *Representational Learning with ELMs for Big Data [J] IEEE Intelligent Systems*, 2013.