ISSN:2320-0790

COMPUSOFT
An International Journal of Advanced Computer Technology

Classification Using Extreme Learning Machine

Soumya Sahoo[1], Sunil Kumar Mohapatra[2], Bijayalaxmi Panda[3]

[1]C.V Raman College of Engineering, Bhubaneswar, India
[2]C.V Raman College of Engineering, Bhubaneswar, India
[3]C.V Raman College of Engineering, Bhubaneswar, India

Abstract: Extreme Learning Machine (ELM) has become popular for solving classification problem due to its fast speed. The performance of ELM often relies on random input hidden node parameters. Neural network also uses artificial intelligence by adjusting weights and minimizing the error. The learning speed of feedforward neural network is very slow. Due to two slow gradient-based learning algorithms and iterative tuning of various parameters. This paper presents a comparative study of back propagation algorithm and an extremely fast ELM technique for single layer feedforward neural network which takes random hidden nodes and determines the output weights without iterative tuning. In theory, this algorithm tends to provides better performance at extremely fast learning speed.

Keywords: classification, neural network, extreme learning machine, moore penrose.

I. INTRODUCTION

Data mining is the computational process of finding patterns in various data sets by applying methods from the field of statistics, artificial intelligence etc. The main objective of the data mining process is to extract information from a data set and transform it into an understandable structure for further use. Data mining uses information from past data to analyze the result of specific problem or situation that may arise. Those data may be business data, production data or marketing data. Since data sets have grown in size and complexity like multivariate or categorical data, direct analysis is comparatively time consuming and hectic, Therefore automated data processing is preferred which is added with other discoveries in computer science, such as neural networks, cluster analysis, genetic algorithms , decision trees and support vector machines and extreme learning machines Data mining is the process of applying these methods for discovering the hidden patterns in large data sets. Feedforward neural networks have been widely used in many fields to analyze the nonlinear mappings from the input samples and to provide better models for classification and prediction which is difficult to using classical parametric techniques. But on another side it lacks faster learning algorithms. The traditional learning algorithms are usually slower than required. To avoid this we have proposed a simple learning algorithm for single layer feedforward neural network called **e**xtreme **l**earning machine (ELM) [1] whose learning speed can be thousands of times faster than traditional feedforward network learning algorithms like back-propagation algorithm while obtaining better generalization performance.

II. BACKGROUND

Guang-Bin Huang, Qin-Yu Zhu,and Chee-KheongSiew in 2006 used this learning algorithm called **e**xtreme **l**earning machine (ELM) for **s**ingle-hidden layer feedforward neural networks (SLFNs) which randomly chooses hidden nodes and analytically determines the output weights of SLFNs. In theory, this algorithm tends to provide good generalization performance at extremely fast learning speed. Their experimental results based on a few artificial and real benchmark function approximation and classification problems including very large complex applications show that the new algorithm can produce good generalization performance in most cases and can learn thousands of times faster than conventional popular learning algorithms for feedforward neural networks [1].

Guang-Bin Huang, Hongming Zhou, Xiaojian Ding, and Rui Zhang [2] shows that ELM provides a unified learning platform with a widespread type of feature mappings and can be applied in regression and multiclass classification applications directly, it from the optimization method point of view, ELM has milder optimization constraints compared to LS-SVM and PSVM and achieve suboptimal solutions, ELM can approximate any target continuous function and classify any disjoint regions.

III. Neural network

An Artificial Neural Network (ANN) is an information processing paradigm that is inspired from the biological nervous systems. Information is stored in the weight matrix W of a neural network. The knowledge possessed by neural network is contained in the values of the connections weights matrix W. Training is the act of presenting the network with some sample data and modifying the weights to better approximate the desired function. The supervised Training supplies the neural network with inputs and the desired outputs. Response of the network to the inputs is measured and the weights are modified to reduce the difference between the actual and desired outputs. Butthe traditional learning algorithms are usually far slower than required. It is not surprising to see that it may take several hours, several days, and even more time to train neural networks by using traditional methods.

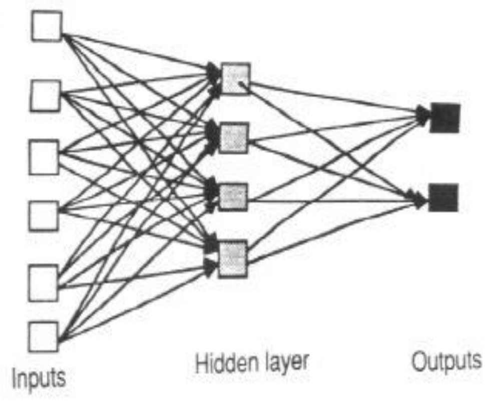

Figure 1

A. *Back Propagation Algorithm*

T The back propagation algorithm is used to find a local minimum of the error function. The network is initialized with randomly chosen weights. Our task is to compute the gradient recursively. The performance function of the neural network is normally chosen to be the mean squared error for each pattern on the training set which is given as:-

$$1/Ni=\sum_{i=1}^{N}(t_{pi}-o_{pi})^2$$

Where,
t_{pi} is the target value
o_{pi} is the output of the network
N is the no of neurons

The weight updates in standard BPN, is given a by:-
$\Delta W = W\text{ (new)} - W\text{(old)} - \eta\,\delta E/\delta W$
Where, η is the learning rate

IV. Extreme Learning Machine

The ELM algorithm was originally proposed by Huang *et al.* in [3] and it makes use of the single layer feedforward neural network. The main concept behind the ELM [4][5] lies in the random initialization of the neural network weights and biases, where the input weights and biases do not need to be adjusted. According to Huang and Babri a **s**ingle-hidden layer feedforward neural network with at most *N* hidden nodes and with almost any nonlinear activation function can exactly learn *N* distinct observations. Here the input weights and hidden layer biases need to be adjusted in all these previous theoretical research works as well as in almost all practical learning algorithms of feedforward neural networks. In case of single layer feedforward neural networks with random hidden nodes, it is described as: For N distinct arbitrary distinct samples $(x_j, t_j) \varepsilon R_m x R_n$, standard SLFNs with R hidden nodes and output functioning (x) are mathematically modeled as:

$$\sum_{i=1}^{R}\beta jG(a_j,b_j,x_k)=t_k$$

Where, (aj , bj) are hidden node parameters. β_j is the weight vector connecting the ith hidden node and output node.

A. *Moore–Penrose generalized inverse*

A matrix G of order n x m is the Moore–Penrose generalized [1] inverse of matrix A of order m x n, if AGA = A; GAG = G; $(AG)^T$=AG; $(GA)^T$=GA

B. *Mathematical Model*

$\sum_{j=1}^{R}\beta\,jG(a_j,b_j,x_k)=t_k$For j=1….N is equivalent to Hβ=T
Where H is called the hidden layer output matrix of the SLFN. The i^{th} column of H is the output of the i^{th} hidden node with respect to inputs x_1, x_2 …. x_n. [1][6].

$$H=\begin{pmatrix} h(x_1) \\ \vdots \\ h(x_N) \end{pmatrix} = \begin{pmatrix} G(a_1,b_1,x_1) & \dots & G(a_R,b_R,x_1) \\ \vdots & & \vdots \\ G(a_1,b_1,x_N) & \dots & G(a_R,b_R,x_N) \end{pmatrix}$$

$$\beta=\begin{pmatrix} \beta_1^T \\ : \\ \beta_R^T \end{pmatrix} \quad \text{and} \quad T=\begin{pmatrix} t_1^T \\ : \\ t_N^T \end{pmatrix}$$

C. Algorithm for computing Hidden Layer Output Matrix Using ELM

1) Generate random number of Hidden Nodes.
2) Generate random weights on the basis of number of Hidden neurons and input data matrix.
3) Input random Bias matrix.
4) Compute H=Input weight* Input data matrix.
5) Compute H=H+Bias matrix.
6) Calculate the output of the H matrix using activation function.
7) Calculate output weight as H^+ * target matrix. Where $\mathbf{H}^+$is called the Moore – Penrose generalized inverse of the hidden layer output matrix **H**.
8) Process the test matrix in the similar manner like the training set input matrix.
9) Forecast _Result= Processed test matrix*output weight

V. EXPERIMENTAL RESULTS

We have considered five datasets from the UCI repository for the simulation purpose. Basically we have selected 70% data for training purpose and 30% for testing purpose. The number of hidden neurons has been varied in both the cases for getting better results. Basically the numbers of hidden layer neurons are taken on the basis of size and attributes of datasets. The output neurons are taken on the basis of number of classes. For iris and wine datasets we have taken 2 output neurons as they are having 3 classes and for glass dataset we have taken 3 output neurons as it is having 6 classes. For Haberman and Bupa liver disorder We have taken 1 output neuron as they are having 2 classes only. The final class is decided on the basis of binary equivalent values of all output neurons. The performance of the ELM learning algorithm is compared with the back propagations algorithms of feedforward neural neural network for classification purpose. The activation function used in our proposed algorithms is a bipolar sigmoidal or tansig function which is also same in case of backpropagation algorithm. All the inputs and outputs have been normalized into the range [0-1]. The learning time of ELM is mainly spent on calculating the Moore–Penrose generalized inverse of the hidden layer output matrix H. We have performed various comparisons like training time computations, accuracy calculations, mean square error computations to show the variations between back propagation and ELM learning methods. The MAE is calculated as follows.

$$\text{MAE}=\sum_{i=1}^{N}(Target_i\text{-}Forecast_i)/\text{N}.$$

The details of datasets are given below.

Table 1: Description of datasets

Datasets	**Number of Records**	**Number of Attributes**	**Number of Classes**
Iris	150	4	3
Wine	178	13	3
Haberman	306	3	2
Glass	214	10	6
Bupa Liver Disorder	345	6	2

Table 2: Division of datasets

Name of the datasets	**Distribution**	**Number of Records**	**Records Distribution**					
			Class1	Class 2	Class 3	Class 4	Class5	Class 6
Iris	**Training Set**	105	35	35	35			
	Testing Set	45	15	15	15			
Wine	**Training Set**	125	42	42	41			
	Testing Set	53	16	17	16			
Haberman	**Training Set**	215	108	107				
	Testing Set	91	45	46				
Glass	**Training Set**	150	25	25	25	25	25	25
	Testing Set	64	11	11	11	11	10	10
Bupa Liver Disorder	**Training Set**	241	120	121				
	Testing Set	104	52	52				

Table 3: Neural Network Parameters

Parameter Name	**Value**
Value of Learning Parameter	0.3
Value of Momentum	0.9
Activation function	tansig

Table 4: ELM Parameters

Datasets	**No.of Hidden Layer Neurons(BPN)**	**No.of Hidden Layer Neurons(ELM)**	**Number of Output Neurons**
Iris	8	25	2
Wine	10	35	2
Haberman	12	35	1
Glass	15	40	3
Bupa Liver Disorder	10	23	1

Table5: Classification Accuracy Through BPN and ELM

Datasets	No. of Hidden Layer Neurons(BPN)	No. of Hidden Layer Neurons(ELM)	Number of misclassifications using BPN	Number of misclassifications using ELM	Accuracy% using BPN	Accuracy %using ELM%
Iris	8	25	4	3	91.33	93.33
Wine	10	35	3	2	94.339	96.22
Haberman	12	35	11	9	87.912	92.307
Glass	15	40	6	4	90.625	93.75
Bupa Liver Disorder	10	23	13	10	87.5	90.384

Table 6: Comparison of Training Time (in seconds) of BPN and ELM

Datasets	BPN	ELM
Iris	0.3825	0.0071
Wine	1.2413	0.1211
Haberman	0.7456	0.2062
Glass	1.6782	0.4762
BupaLiver Disorder	0.7521	0.1005

Datasets	BPN(output neuron1)	ELM(output neuron1)	BPN(output neuron2)	ELM(output neuron2)	BPN(output neuron3)	ELM(output neuron3)
Iris	0.3401	0.3213	0.1887	0.1764		
Wine	0.2692	0.2343	0.2254	0.2070		
Haberman	0.4575	0.2113				
Glass	0.3526	0.3154	0.3454	0.3123	0.5436	0.3212
Bupa Liver Disorder	0.0883	0.0821				

Bartlett's [7] theory on the generalization performance of feedforward neural networks states for feedforward neural networks reaching smaller training error, the smaller the norm of weights gives the better generalization performance. In conventional approaches all the parameters of the feedforward networks need to be adjusted and thus there exists the dependency between different layers of parameters. Previously gradient descent-based methods have mainly been used in various learning algorithms for feedforward neural networks. But we can see that gradient descent-based learning methods are generally very slow due to improper learning steps or may easily converge to local minima as well as many iterative learning steps may be required by such learning algorithms in order to obtain better learning performance. The gradient-based learning algorithms like back-propagation can be used for feedforward neural networks which have more than one hidden layers while the ELM algorithm is only valid for single-hidden layer feedforward networks. Therefore ELM needs much less training time compared to popular BP. Accuracy sometimes depends on number of attributes and size of datasets. is basically de The prediction accuracy of ELM is usually slightly better than BP in many applications. Compared with BP and other computational methods ELM can be implemented easily since there is no parameter to be tuned except an insensitive parameter L. It should be noted that many nonlinear activation functions can be used in ELM. ELM needs more hidden nodes than BP but the simulation result tells that ELM and BP have much shorter response time to unknown data than other approaches. Therefore, the proposed learning algorithm tends to have good generalization performance for feedforward neural networks. Both the algorithms are showing approximately same MAE but the ELM performance is better in all the cases .We have observed the performance of different datasets over 25 simulations.

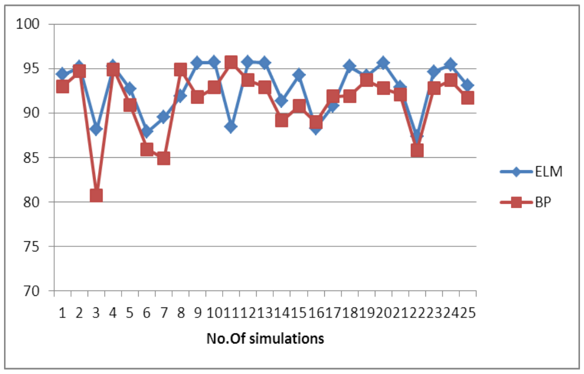

Figure 2: Performance comparison of iris

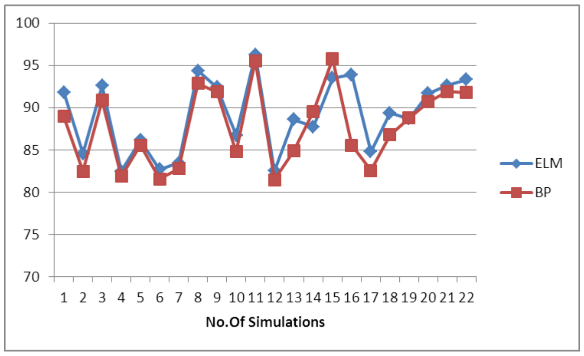

Figure 3: Performance comparison of Wine

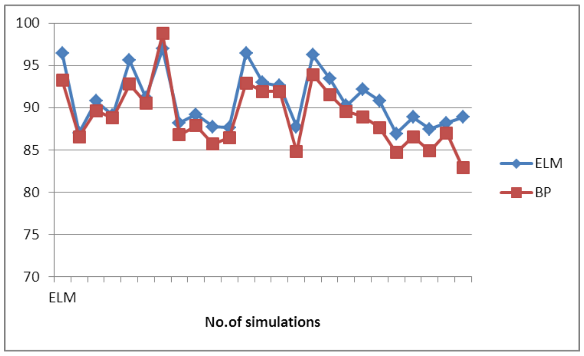

Figure4:Performance comparison of Haberman

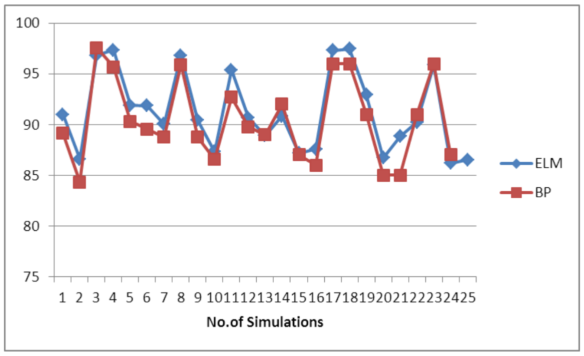

Figure 5:Performance comparison of Glass

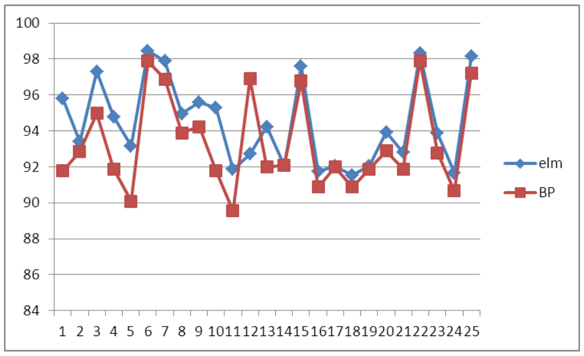

Figure 6:Performance comparison of Bupaliver disorder

VI. CONCLUSION

ELM is a learning mechanism for the generalized single layer feedforward neural network, where learning is made without iterative tuning. The bottleneck observed in the use of BP algorithm can be overcome by use of ELM which shows faster convergence to global minima rather than local minima. The importance of ELM is that the hidden layer of the generalized single layer fed forward neural networks should not be tuned. ELM shows similar or better generalization performance for regression and binary class classification cases and much better performance for multiclass classification cases. ELM has better scalability and runs at much faster learning speed than traditional approaches.

VII. References

[1] Guang-Bin Huang,Qin-Yu Zhu,Chee-KheongSiew,"Extreme learning machine: Theory and applications"Neurocomputing,Volume70, Issues 1–3, , Pages 489–501,December 2006.

[2] Guang-Bin Huang,Hongming Zhou, Xiaojian Ding, and RuiZhangExtreme Learning Machine for Regression and Multiclass Classification,IEEE Transactions on Systems, Man and Cybernetics-Part B:Cybernetics,Vol-42,No.2,April 2012

[3] G.-B. Huang, H.A. Babri"Upper bounds on the number of hidden neurons in feedforward networks with arbitrary bounded nonlinear activation functions",IEEE Trans. Neural Networks, 9 (1) pp. 224–229(1998)

[4] N.-Y. Liang, et al., "A Fast and Accurate On-line Sequential Learning Algorithm for feeedforwardNetworks,"IEEE Transactions on Neural Networks, vol. 17, no. 6, pp. 1411-1423, 2006.

[5] Q.-Y. Zhu, et al., "Evolutionary Extreme Learning Machine", Pattern Recognition, vol. 38, no. 10, pp.1759-1763, 2005.

[6] R. Zhang, et al., "Multi-Category Classification Using Extreme Learning Machine for Microarray GeneExpression Cancer Diagnosis," IEEE/ACM Transactions on Computational Biology and Bioinformatics (inpress), 2006.

[7] 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. Inf. Theory 44 (2) 525–536 (1998).