iJRASET
International Journal For Research in Applied Science and Engineering Technology



INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 10 **Issue:** IV **Month of publication:** April 2022

DOI: https://doi.org/10.22214/ijraset.2022.41514

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 10 Issue IV Apr 2022- Available at www.ijraset.com

Optimized Extreme Learning Machine

Roshan Kaloni[1], Tejas R Nayak[2], Mitanshu Sankhe[3], Mr. Govind Wakure[4]
[1, 2, 3]*Student,* [4]*Guide, Department of Information Technology, Manjara Charitable Trust's Rajiv Gandhi institute of technology, Mumbai, Maharashtra–400053, India*

Abstract: ***Extreme Learning Machine (ELM) is a learning method for single-hidden layer feedforward neural network (SLFN) training. The ELM strategy speeds up learning by generating input weights and biases for hidden nodes at random rather than modifying network parameters, making it much faster than the standard gradient-based approach. In this project, an ELM optimized by Hybrid Particle Swarm Optimization approach is presented to optimize the input weights and hidden biases for ELM.***
We will analyze and obtain results for benchmark datasets. The Optimized Extreme Learning Machine algorithm's output is compared to publicly available data. Later we will compare different algorithms and check which one gives better output metrics.
Keywords: ELM, SLFN, PSO, Gradient-based approach, Optimization

I. INTRODUCTION

The amount of data we generate on a daily basis is absolutely astounding. At our present rate, 2.5 quintillion bytes of data are created every day, but that rate is only increasing as the Internet of Things expands (IoT). In just the previous two years, 90 percent of the world's data has been created. And the data's dimensions, such as size and dimensionality, are growing all the time. As a result, it's critical to develop resourceful machine learning approaches that can be utilised to properly comprehend data and discover useful knowledge that can be used to produce new understandings from the vast amount of data available.

Feedforward neural networks are widely employed in a variety of fields because of their ability:

1) To directly approximate complicated nonlinear mappings from input samples
2) To create models for a wide range of natural and man-made phenomena that are challenging to model using traditional parametric methods.

From a mathematical standpoint, feedforward neural networks' approximation capabilities have been studied in two ways: universal approximation on compact input sets and approximation on a finite set. The universal approximation capabilities of standard multi-layer feedforward neural networks have been investigated by a number of researchers. Neural networks are learned in a finite training set in real-world applications. A single-hidden layer feedforward neural network (SLFN) with at most N hidden neurons and virtually any nonlinear activation function can learn N different observations with zero error for function approximation in a finite training set.
Faster learning techniques for neural networks, on the other hand, are lacking. Traditional learning algorithms are typically far slower than what is needed. It should come as no surprise that training neural networks using standard methods might take several hours, days, or even weeks.
Gradient descent-based methods have mostly been employed in various feedforward neural network learning algorithms for decades. However, it is evident that gradient-descent based learning algorithms are either highly slow or easily converge to local minimums due to faulty learning steps. Such learning algorithms also necessitate a large number of iterative learning steps in order to improve learning performance. The output weights (connecting the hidden layer to the output layer) of SLFNs may be analytically computed using a simple generalised inverse operation of the hidden layer output matrices once the input weights and hidden layer biases have been freely set. For single-hidden layer feedforward neural networks, the Extreme Learning Machine (ELM) was developed as an efficient learning algorithm (SLFN). It is characterized by an unplanned initialization of hidden layer weights and a fast-training algorithm which are joined together. In contrast to gradient-based techniques, ELM boosts learning speed by randomly generating weights and biases for hidden nodes. This algorithm's efficiency has been demonstrated to produce superior outcomes, making it particularly desirable for a larger dataset.

©IJRASET: All Rights are Reserved

Although ELM is fast and presents good generalization performance, as the output weights are computed based on the input weights and hidden biases using the Moore Penrose (MP) generalized inverse, it's possible that there's a set of non-optimal input weights and hidden biases, as well as overfitting.

Given that these approaches have the capacity to widen the search space aiming to address a number of optimization problems, nature-inspired population-based methods have been progressively employed to solve a variety of optimization problems to avoid local minima. Among them, the most common are evolutionary algorithms (EAs) such as Genetic Algorithm (GA) and Differential Evolution (DE), Ant Colony Optimization (ACO) and Particle Swarm Optimization (PSO).

In this project an Optimized Extreme Learning Machine approach is used. Particle swarm optimization method is used to optimize ELM. As a result, the construction of a better ELM model helps ensure that the model produces optimal results. In this context, the particle swarm optimization (PSO) algorithm was utilized in the present study to optimize ELM and a PSO-ELM model was developed.

II. PROBLEM STATEMENT

On the premise that a large number of hidden nodes are selected in the model, extreme learning machines have the advantage of a short training period while maintaining adequate classification and regression performance. The huge number of nodes in the hidden layer slows down the testing performance of ELM while there is no grantee of optimality of the setting of weights on the hidden layer.

Thus, PSO-ELM will be utilized to find optimal weights and biases in ELM and improve the ELM model. In particle swarm optimization algorithm and ELM, the particle swarm optimization algorithm is applied with its powerful performance with the exploration and the exploitation processes. A hybrid optimization mechanism is proposed which combines the discrete-valued PSO with the continuous-valued PSO to optimize the weight selection and the number of hidden nodes to enhance the performance of ELM and helps the algorithm to reach to the optimal or near optimal solution in reasonable time.

The particles in PSO represents weights of ELM. In addition, to accomplish a better accuracy the optimization is requiring to select both the weight's values as well as the number of neurons that are needed in the hidden layer.

The resulting optimized algorithm will be trained and tested on standard datasets (e.g., MNIST, Breast Cancer, Wine-dataset) and compared with other machine learning models (ELM without optimization, Logistics Regression).

III.METHODOLOGY

In this section, we describe our approach how we built the model. In which we first use PSO for weights selection and then applying ELM model to our datasets.

A. Extreme Learning Machine

ELM, proposed by Huang et al., is a single hidden layer feed forward network, where the input weights are chosen randomly and the output weights are calculated analytically. The hidden neuron layer can use sigmoid, sine, gaussian, and hard limiting functions as activation functions. The output neurons have a linear activation function. For feed forward neural networks, ELM has several notable aspects that distinguish it from traditional popular gradient-based learning techniques. Faster learning speed, good generalisation capabilities, and avoidance of issues such local minima are all advantages of ELM.

1) Preliminary Moore-Penrose Generalized Inverse:

The Moore-Penrose generalised inverse is introduced in this section. The resolution of a general linear system

$$Ax = y,$$

where A may be singular and may even not be square, can be made very simple by the use of the Moore-Penrose generalized inverse.

Definition: A matrix G of order n × m is the Moore-Penrose generalized inverse of matrix A of order m × n, if

$$AGA = A,\ GAG = G,\ (AG)^T = AG,\ (GA)^T = GA$$

For the sake of convenience, the Moore-Penrose generalized inverse of matrix A will be denoted by A†.

2) Extreme Learning Machine (ELM):

Given K hidden neurons, an activation function g(x) and N distinct samples (xi, ti), were

$$xi = [xi1, xi2, \ldots, xin]^T \in Rn \text{ and } ti = [ti1, ti2, \ldots, tim]^T \in Rm.$$

In ELM, the input weights and hidden biases are randomly generated by solving the following linear system:

$$H\beta = T$$

where H = {hij} (i = 1, …, N and j = 1, …, K)
is the hidden layer output matrix, hij = g(wj · xi + bj) denotes the output of jth hidden neuron with respect to xi; wj = [wj1, wj2, …, wjn] T is the weight vector connecting jth hidden neuron and input neurons, and bj denotes the bias of jth hidden neuron; wj ·xi denotes the inner product of wj and xi;

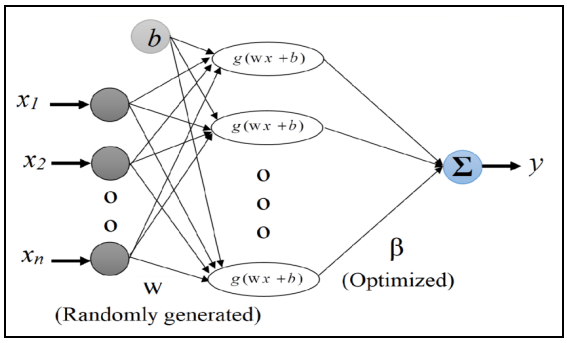


Fig. 1 Extreme Learning Machine

$\beta = [\beta 1, \beta 2, \ldots, \beta K]^T$ is the matrix of output weights and

$\beta j = [\beta j1, \beta j2, \ldots, \beta jm]^T$ (j = 1, …, K)

denotes the weight vector connecting the jth hidden neuron and output neurons;

$T = [t1, t2, \ldots, tn]^T$ is the matrix of targets (desired output).

The output weights (linking the hidden layer to the output layer) are determined by finding the least-square solution to the given linear system. The minimum norm least-square (LS) solution to the linear system is

$$\hat{\beta} = \mathbf{H}^{\dagger}\mathbf{T}$$

where H† is the Moore Penrose generalized inverse of matrix H. The minimum norm Least Squares solution is unique and has the smallest norm among all the Least Squares solutions.

The standard ELM algorithm is as follows

a) Generate randomly the input weights wi and biases bi;
b) Compute the hidden-layer output matrix H;
c) Calculate the output weights matrix as $\hat{\beta} = \mathbf{H}^{\dagger}\mathbf{T}$

B. Particle Swarm Optimization (PSO)

Particle swarm optimization (PSO) is a population-based stochastic optimization technique created by Dr. Eberhart and Dr. Kennedy in 1995 and inspired by social behaviour such as bird flocking and fish schooling.

PSO has a lot in common with evolutionary computing techniques like Genetic Algorithms (GA). The system starts with a population of random solutions and then uses updating generations to look for optima. Particles, which are potential solutions in PSO, travel through the issue space following the current optimum particles. The next parts will delve deeper into the subject.

The particle swarm notion began as a simple social system simulation. The goal was to graphically mimic the choreography of a bird block or a fish school. The particle swarm model, on the other hand, was discovered to be a good optimizer.



Fig. 2 Particle Swarm Optimization

PSO, as previously said, simulates flocking behaviours in birds. Consider the following scenario: a flock of birds is seeking for food in an area at random. In the area being searched, there is only one food item. The birds are all in the dark about where the food is. They do, however, know how far the meal has travelled in each iteration. So, what's the best way to track down the food? The most efficient method is to follow the bird that is closest to the meal.

PSO took what it had learned from the scenario and applied it to the optimization challenges. Each solution in PSO is a "bird" in the search space. It's known as "particle." All particles have fitness values that are evaluated by the fitness function in order to be optimised, as well as velocities that direct the particles' flight.

The particles follow the current optimum particles through the problem space. PSO starts with a set of random particles (solutions) and then updates generations to look for optima. Each particle is updated in each iteration by comparing two "best" values. The first (fitness) is currently the most effective option. (Also saved is the fitness value.) pbest is the name given to this value. The best value obtained so far by any particle in the population is another "best" value recorded by the particle swarm optimizer. This best value is referred to as gbest, which stands for "global best." The best value is a local best and is denoted lbest when a particle takes part of the population as its topological neighbours.

After finding the two best values, the particle updates its velocity and positions with following equations

v[ ] = v[ ] + c1 * rand( ) * (pbest[ ] - present[ ]) + c2 * rand( ) * (gbest[ ] - present[ ])

present[ ] = present[ ] + v[ ]

v[ ] is the particle velocity, present[ ] is the current particle (solution). pbest[ ] and gbest[ ] are defined as stated before. rand ( ) is a random number between (0,1). c1, c2 are learning factors. Usually, c1 = c2 = 2.

The pseudo code of the procedure is as follows:

```
For each particle
    Initialize particle
END

Do
    For each particle
        Calculate fitness value
        If the fitness value is better than the best fitness value (pBest) in history
            set current value as the new pBest
    End

    Choose the particle with the best fitness value of all the particles as the gBest
    For each particle
        Calculate particle velocity according equation (a)
        Update particle position according equation (b)
    End
While maximum iterations or minimum error criteria is not attained
```

Particles' velocities on each dimension are clamped to a maximum velocity Vmax. If the sum of accelerations would cause the velocity on that dimension to exceed Vmax, which is a parameter specified by the user. Then the velocity on that dimension is limited to Vmax.

C. PSO-ELM

In ELM, as the computation of output weights is based on the input weight and hidden biases, their choice greatly influences the performance of ELM. The issues with random generation of parameters have been discussed in the above chapters. In order to ensure better generalization ability of ELM, this paper proposes chaotic PSO algorithm-based selection of input weight and biases.

The detailed steps of the proposed method are as follows:

1) First, the swarm is randomly generated. Each particle in the swarm is composed of a set of input weights and hidden biases. All components in the particles are randomly initialized within the range of [−1,1].

2) Second, for each particle, the corresponding output weights are computed. Then the fitness of each particle is evaluated. In this work the fitness of each particle is based on the mean square error of the training set.

3) Third, with the fitness of all the particles, the Pbesti and gbesti of the swarm are updated.

4) Fourth, each particle updates its position and velocity accordingly, and a new population is generated.

5) Finally, the above optimization process is repeated until maximum iteration.

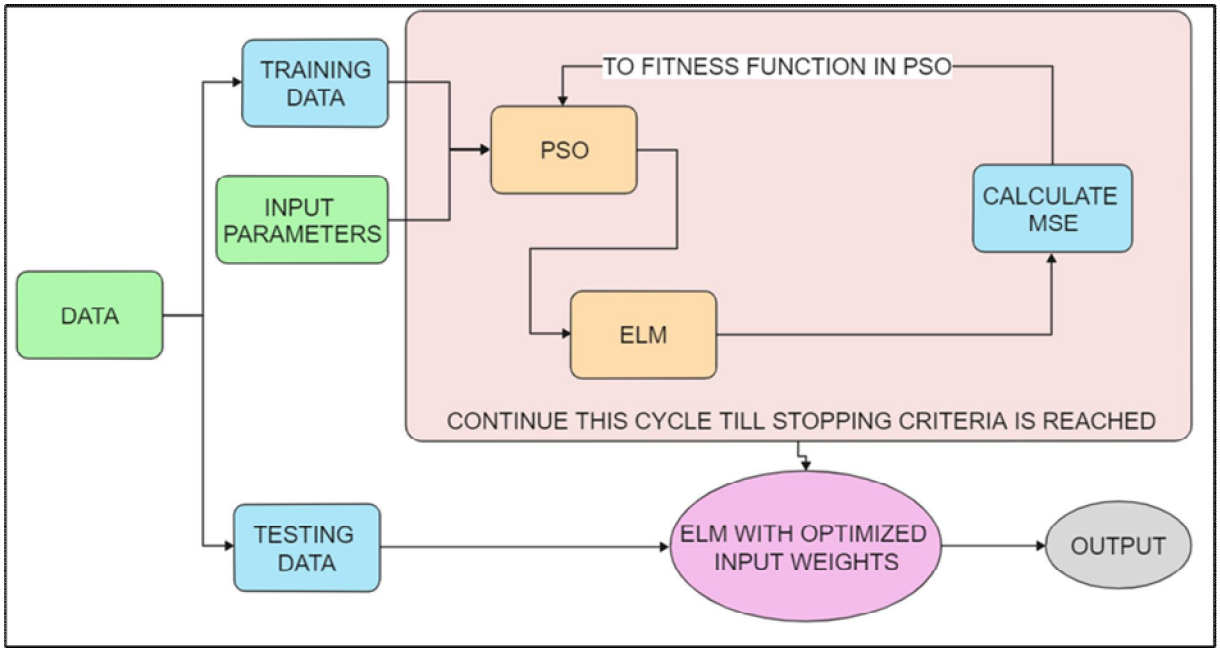


Fig. 3 Block Diagram of PSO-ELM

IV. PERFORMANCE EVALUATION AND DISCUSSIONS

A. Dataset Description

In our experiments, three standards classification datasets from scikit-learn library were used to determine the performance of the model. The dataset which we used are the wine dataset and mnist digit dataset. These datasets present different number of classes and difficulties. All attributes have been normalized into the range [0, 1]. The ELM activation function used was the sigmoid function. Each dataset was divided in training, testing, validation sets. The training, validation and testing sets were randomly generated at each trail of simulations.

B. Metrics Performance

In this paper, the metrics which we have used to evaluate the performance of PSO-ELM is mean squared error (MSE) over the validation set and it is also used as fitness function in PSO. The testing accuracy refers to the percentage of correctness produced by the trained SLFNs on the testing set. Thus, it is calculated using the equation

$$\text{Testing accuracy} = 100 * (nc / nT)$$

where nT is the size of the testing set and nc is the number of correct classifications. The MSE over the validation set is calculated using equation

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(f_i - y_i)^2$$

Where N is the number of datapoints, f_i the value returned by the model and y_i the actual value for datapoint i.

The same MSE metric is used as a fitness function to find the optimal weight for the model. The target error and hidden layer neurons used were 1e-6 and 85 respectively. The PSO was executed for 500 iterations because we want to provide enough time to exchange of information between particles and get the optimal weights for our model.

C. Results

In this part of the paper, we present the results obtained. As shown in the table I and table II, First the three datasets are trained, tested and validated on the plain ELM model which represents the good accuracy score in mnist dataset whereas it does not perform well in the wine dataset as well as in the breast cancer dataset. It is happening because of the randomly generating weights and hidden biases in the ELM model. So, to overcome this issue the ELM algorithm is optimized with the PSO. Then after optimizing the ELM algorithm with the PSO optimization technique, we were getting the optimal weights for our model and the testing accuracy score for both the model has shown a significant improvement. We have also verified this by comparing it with other classification algorithm viz. Logistics Regression which one of the best classification algorithms. We have observed that our model is performing better than the ELM and Logistic Regression. The validation metrics for training and testing which is representing using MSE also show better results than the other two algorithm.

Accuracy						
Data sets	Logistics Regression		ELM		PSOELM	
	Training	Testing	Training	Testing	Training	Testing
Wine	0.992481	0.944444	1.0	0.933333	1.0	0.972222
Breast Cancer	0.962637	0.894737	1.0	0.850877	1.0	0.912281
Mnist Digit	1.0	0.951111	0.960334	0.938889	0.994433	0.980556

Table I Accuracy scores

Loss (MSE)						
Data sets	Logistics Regression		ELM		PSOELM	
	Training	Testing	Training	Testing	Training	Testing
Wine	0.007519	1.666667	0.00161	0.935968	0.001862	0.013727
Breast Cancer	0.037363	0.105263	0.0	0.069244	0.000746	0.008092
Mnist Digit	0.0	1.153333	0.003952	0.00534	0.001917	0.003724

Table II Loss (MSE) scores

These results can also be clearly represents using the charts as shown in the fig. 4, fig. 5, fig. 6 and fig.7.

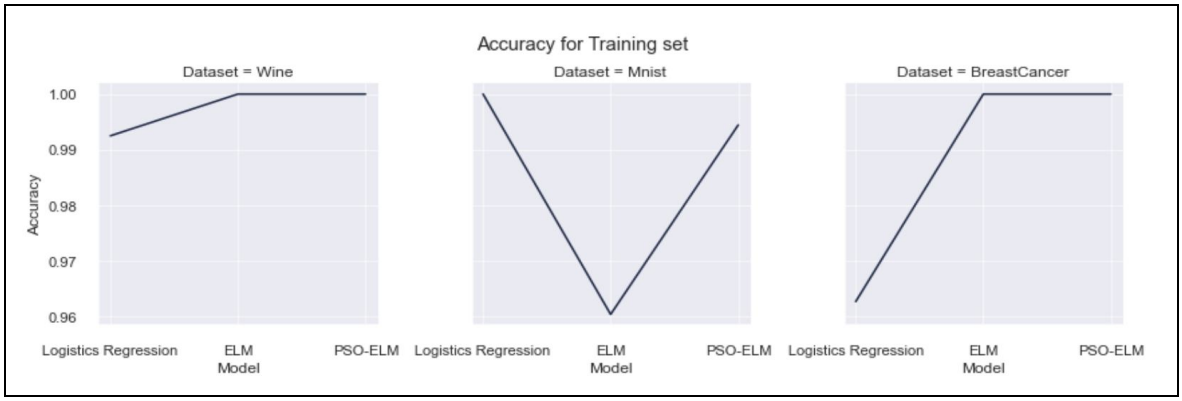


Fig. 4 Accuracy for training datasets

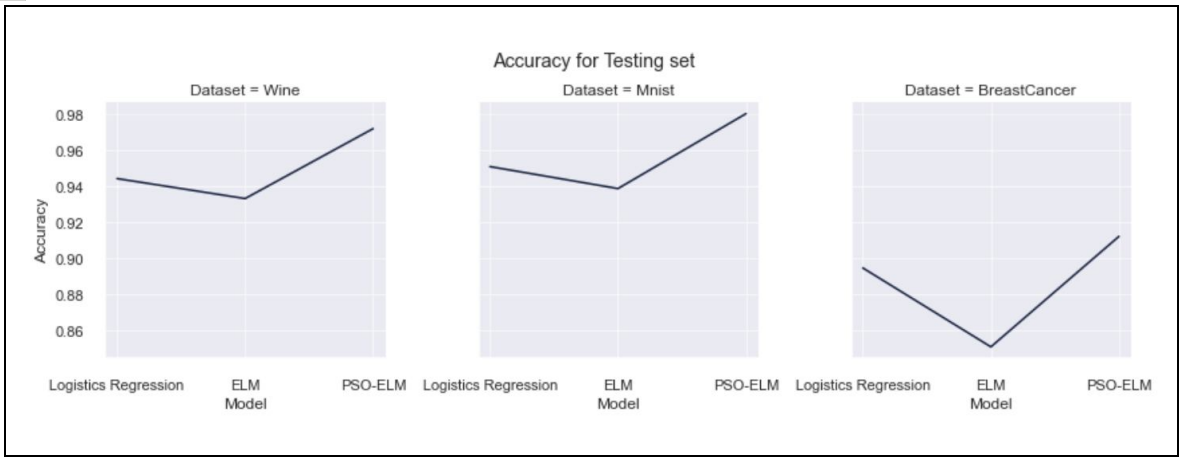


Fig. 5 Accuracy for testing datasets

As shown by the line chart in figures 4 and 5, show that the accuracy score of PSO-ELM is always better and greater than the other two algorithms for training and testing datasets.

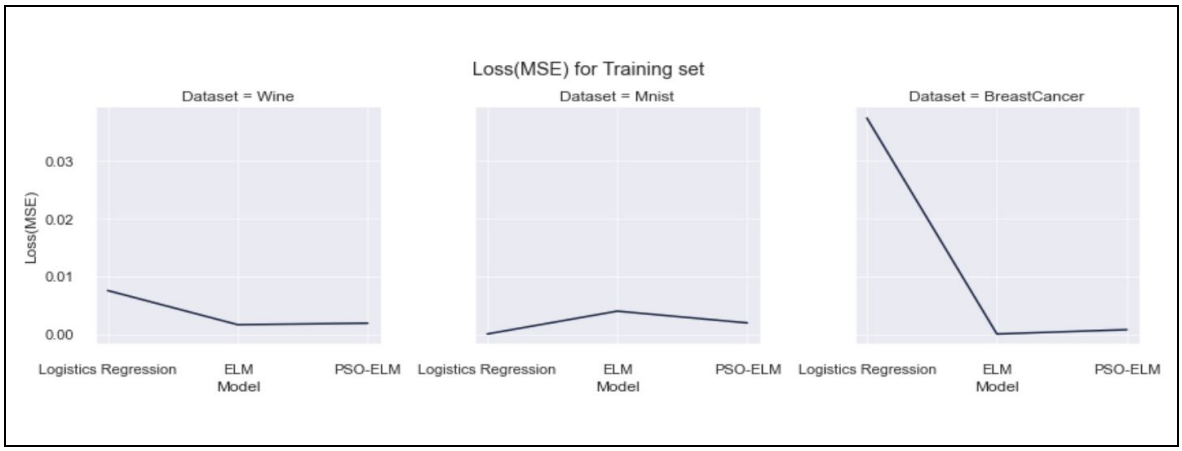


Fig. 6 MSE for training datasets

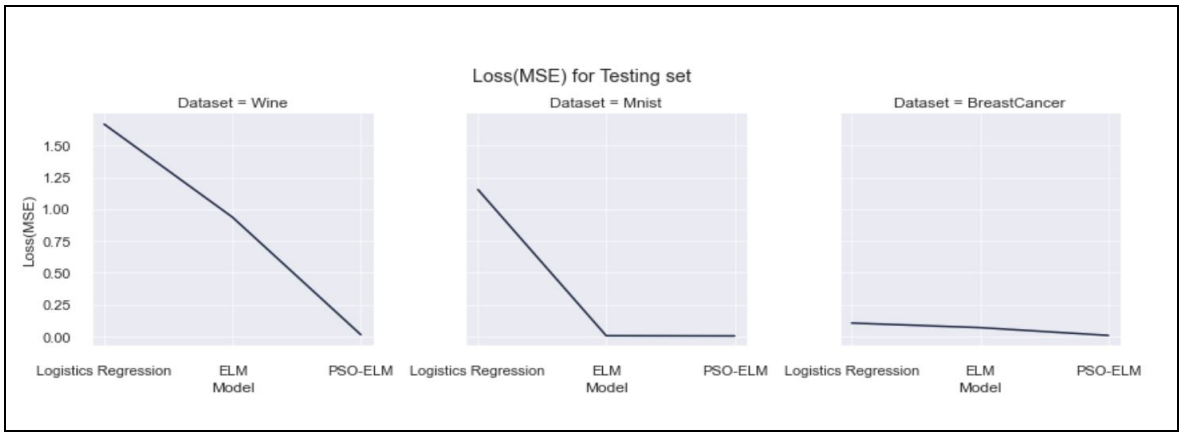


Fig.7 MSE for testing datasets

To ensure that our model outperforms other algorithms, we calculated the mse for validation alongside the accuracy score, which means that if the mse score of the model is close to zero, the model is performing better, without a doubt. Figures 6 and 7 show instances where the mse score for both the training and testing sets is always close to zero.

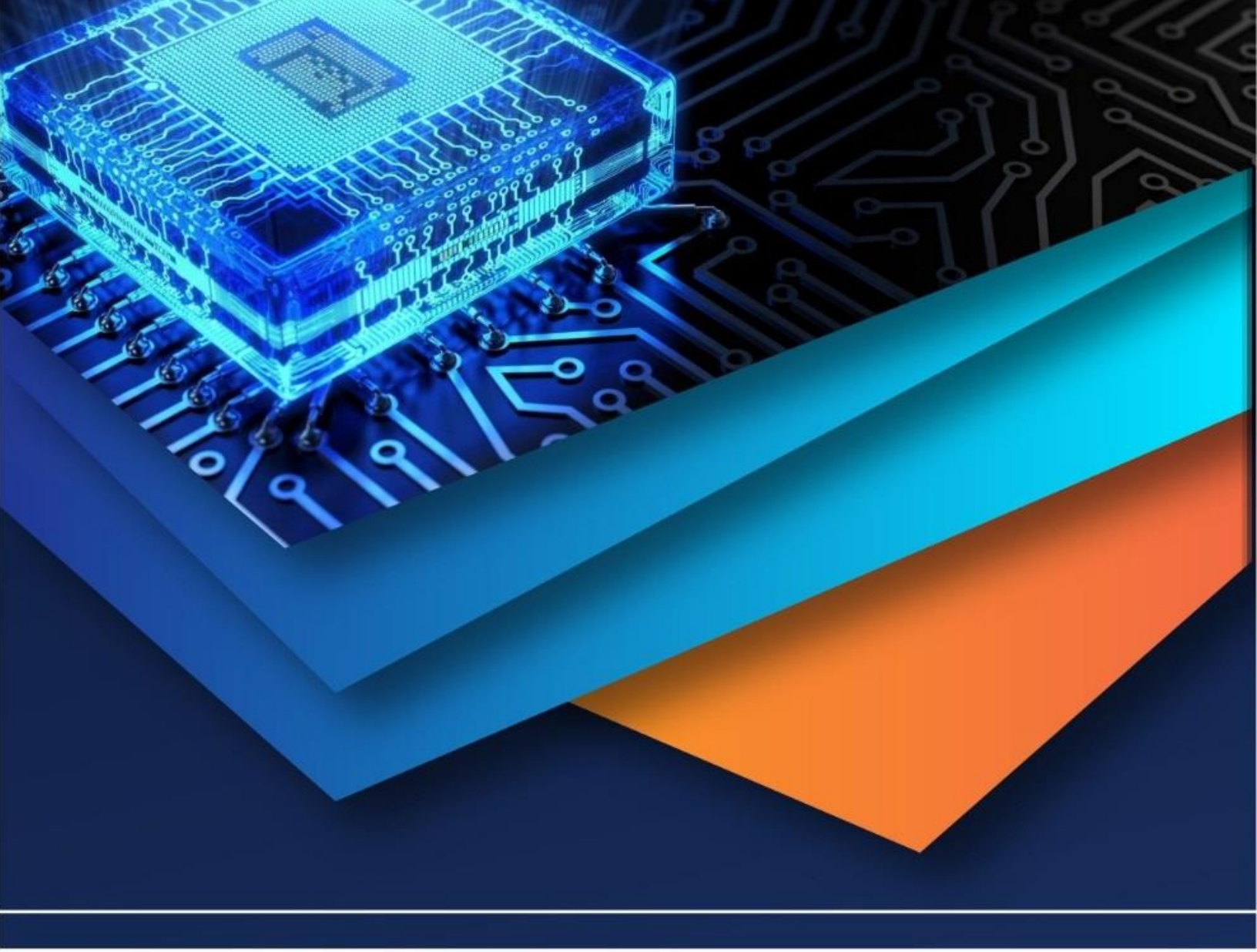
V. CONCLUSIONS

In this paper, we present a particle swarm optimization technique that incorporates an extreme learning machine algorithm. The PSO-ELM model generates input weights and hidden biases using the PSO method and calculates output weights using MP generalized inverse with mse (mean squared error) as a fitness function. The best parameter selection will avoid the problem of random input while simultaneously improving prediction accuracy. The accuracy was used to assess the performance of PSO-ELM. In most situations, the PSO-ELM outperformed the other approaches in terms of classification accuracy. The suggested method's feasibility and effectiveness are validated using PSO-ELM testing on standard datasets.

REFERENCES

[1] Guang-Bin Huang, Qin-Yu Zhu, Chee-Kheong Siew, “Extreme learning machine: Theory and applications”. School of Electrical and Electronic Engineering, Nanyang Technological University, Nanyang Avenue, Singapore 639798, Singapore.
[2] Guang-Bin Huang, Qin-Yu Zhu, and Chee-Kheong Siew “Extreme Learning Machine: A New Learning Scheme of Feedforward Neural Networks”. School of Electrical and Electronic Engineering Nanyang Technological University.
[3] Mustafa Abdul Salam, Hossam M. Zawbaa, E. Emary, Kareem Kamal A.Ghany, B. Parv., “A hybrid dragonfly algorithm with extreme learning machine for prediction”
[4] Nilesh Rathod and Sunil B. Wankhade, “Improving Extreme Learning Machine Algorithm Through Optimization Technique”. Advanced Computing Technologies and Applications Proceedings of 2nd International Conference on Advanced Computing Technologies and Applications— ICACTA 2020.
[5] J. Kennedy and R. Eberhart, “Particle Swarm Optimization,” Proceedings of IEEE International Conference on Neural Networks, pp.1942-1948, 1995
[6] Shaowei Chen, Yue Shang, Minhua Wu, “Application of PSO-ELM in Electronic System Fault Diagnosis”. School of Electronic Information, Northwestern Polytechnical University Xi’an, China,710129.
[7] R. Ahila, V. Sadasivam, K. Manimala Department of Computer Science and Engineering, Dr. Sivanthi, Adiatanar College of Engineering, Tiruchendur, Tamil Nadu, India, “An integrated PSO for parameter determination and feature selection of ELM and its application in classification of power system disturbances”. Department of Computer Science and Engineering, Manonmanium Sundaranar University, Tirunelveli, Tamilnadu, India.
[8] Anupam Sagnik, and Padmini Pani. “Flood Forecasting Using a Hybrid Extreme Learning Machine-Particle Swarm Optimization Algorithm (ELM-PSO) Model - Modeling Earth Systems and Environment.” SpringerLink, link.springer.com, 7 Nov. 2019.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089 (24*7 Support on Whatsapp)