Neural Comput & Applic
DOI 10.1007/s00521-016-2738-1

ORIGINAL ARTICLE

# Elitist teaching–learning-based optimization (ETLBO) with higher-order Jordan Pi-sigma neural network: a comparative performance analysis

**Janmenjoy Nayak[1] · Bighnaraj Naik[2] · H. S. Behera[3] · Ajith Abraham[4,5]**

Received: 6 May 2015 / Accepted: 23 November 2016


**Abstract** This paper presents the performance analysis of a newly developed elitist teaching–learning-based optimization algorithm applied with an efficient higher-order Jordan Pi-sigma neural network (JPSNN) for real-world data classification. Teaching–learning-based optimization (TLBO) algorithm is a recent metaheuristic, which is inspired through the teaching and learning process of both teacher and learner. As compared to other algorithms, it is efficient and robust due to its non-controlling parameter adjustments feature. Elitist TLBO is an improved version of TLBO with the addition of elitist solutions, which makes it more efficient. During the experiment, first the TLBO and then ETLBO algorithm are applied with only Pi-sigma neural network and its performance has been compared with other methods such as GA and PSO. Then, the ETLBO algorithm is applied with JPSNN and found better results over other methods. The proposed method has been tested with real-world benchmark datasets considered from UCI machine learning repository, and the performance has been compared with all seven approaches along with other HONN to prove the effectiveness of the method. Simulation results and statistical analysis show the superiority in the performance of the proposed approach as well as prove the potentiality over other existing approaches.

**Keywords** ETLBO · TLBO · JPSNN · PSNN · PSO · GA

✉ Janmenjoy Nayak
mailtojnayak@yahoo.com

[1] Department of Computer Science Engineering, Modern Engineering and Management Studies, Balasore, Odisha, India

[2] Department of Computer Application, Veer Surendra Sai University of Technology, Burla, Sambalpur 768018, Odisha, India

[3] Department of Computer Science Engineering and Information Technology, Veer Surendra Sai University of Technology, Burla, Sambalpur, Odisha 768018, India

[4] Machine Intelligence Research Labs (MIR Labs), 2259, Auburn, WA 98071-2259, USA

[5] IT4Innovations - Center of Excellence, VSB-Technical University of Ostrava, 17. listopadu 2172/15, 708 00 Ostrava, Czech Republic

## 1 Introduction

With the successive development of science and technology, the real-life optimization problems are becoming more complex in nature. The earlier developed traditional optimization algorithms fail to explicate the exact and real solution of the nonlinear and non-differential problems in large search space. The basic limitations to these algorithms are early convergence, use of complicated stochastic functions and higher-order derivatives in solving the equations. During last few decades, some popular optimization algorithms have already shown their effectiveness in solving various real-life problems. In 1992, Holland [1] at University of Michigan and Goldberg (1989) [2] developed the most popular evolutionary algorithm called genetic algorithm. As compared to the gradient search methods, GA performs well at local optima and has lesser chance to trap at local minima positions. Then after, Kennedy and Eberhart [3] developed a stochastic swarm intelligence-based algorithm inspired by the nature of birds called particle swarm optimization (PSO). It is being considered as one of the popular stochastic and heuristic-based search methods till date. Several equivalent variations have also been developed related to PSO such as ant colony optimization (ACO) [4, 5], artificial bee colony optimization (ABC) [6, 7] and fish schooling algorithm [8]. Besides these, some nature-inspired algorithms such as harmony search [9],

Published online: 16 December 2016

gravitational search [10, 11], firefly algorithm [12], and glow swarm algorithm [13, 14], and physical-based algorithms such as electromagnetism-like algorithm (ELA) [15, 16], artificial physics optimization algorithm(APO) [17], big bang–big crunch optimization (BBCO) [18, 19], charged system search (CSS) [20, 21], particle collision algorithm (PCA) [22], and central force optimization (CFO) [23, 24] have been introduced during the last decade. Although these techniques are used to solve many of the complex problems, but still these have some major issues in the convergence criteria, when these are being single handedly applied. This is due to the extensive use of controlling parameters such as population size, environmental conditions and no. of iterations. Therefore, their variations have been developed by integrating some modifications in the parameters or any form of hybridization algorithms to explore their own problem-solving capacity. As any major change in the parameter selection may change the function of the whole algorithm, so hybridization is not the exact solution for solving these complex problems. We have proceeded with this facet in our work. TLBO is a parameter-free natural metaheuristic, inspired by the teaching–learning process of teacher and learner. The basic principle of TLBO relies on the effect of a learner after the teaching of a teacher. A teacher is a person who has greater knowledge than his learners. The teacher is supposed to share his knowledge with the learner in such a way that the learner's outcome must be a reflection of the teaching process of the teacher. If a teacher is giving his best effort to train the learner, then the learner may have a chance to secure good result. However, the learners also share their knowledge among their friends, and increase in knowledge may be possible in that direction.

Although, some major contributions of TLBO have been made in the field of mechanical engineering and electrical engineering, etc., it has not so much widely been used to solve data mining problems. After a rigorous search of all the TLBO papers in various well-known databases such as Science Direct, Springer Link and IEEE, it was found that TLBO is marginally functioning well in various real-life applications such as economic load dispatch, power handling, electric vehicles, sequence planning, robotics and CAD. The detailed literature survey on TLBO in various application areas is illustrated in Sect. 2.

So, a substantial amount of work is needed to be carried out in the prolific areas of data mining such as classification, clustering and forecasting to show the efficiency of this recently developed population-based algorithm. In this work, a novel Elitist teaching–learning-based optimization (ETLBO) algorithm has been incorporated with a higher-order Jordan Pi-sigma neural network for data classification. ETLBO is quite free from the controlling parameters as compared to other algorithms. The rest of the paper is organized as follows. Section 2 reviews some previous literatures based on TLBO and ETLBO. The basic preliminaries such as TLBO, ETLBO, PSNN and JPSNN are briefly explained in Sect. 3. The proposed ETLBO–JPSNN explained in Sects. 4 and 5 gives the detailed ideas about the experimental setup with simulating environment. Section 6 presents the result analysis of the proposed work, and Sect. 7 describes the comparative results of another HONN with the proposed classifier. Statistical analysis of all the classifiers with their comparative results is demonstrated in Sect. 8. Finally, Sect. 9 concludes the work with some future directions.

## 2 Literature survey

In this section, some important literatures of TLBO have been reviewed. In 2011, Rao et al. [25] first developed the concept of TLBO and applied in the field of mechanical design problems. More elaborative descriptions with the global function optimization by the application of TLBO were introduced by Rao et al. [26]. They tested the effectiveness of the TLBO algorithm with the considerations of benchmark functions. Again, Rao and Patel [27] described the TLBO algorithm with some improvements in the existing TLBO and applied for unconstrained optimization problems. In 2012, Rao and Patel [28] introduced the Elitist TLBO for constraint optimization and shown some more improved results on ETLBO over TLBO. For both constrained and unconstrained multiobjective problems, Rao and Waghmare [29] have compared the performance of TLBO with other optimization techniques. Rao and More [30] have used the stochastic TLBO method for design optimization of heat pipe and compared the result of TLBO with niched pareto genetic algorithm, grenade explosion method and generalized external optimization. They found the performance of TLBO better than other methods for the optimization of heat pipes. Estimation of energy consumption in Turkey with the integration of artificial neural network and TLBO algorithm has been achieved by Uzlu et al. [31]. Wang et al. [32] have developed the improved version of the TLBO algorithm with the neighborhood search with the applications of various benchmark functions and artificial neural network. Basu [33] have used TLBO algorithm to solve the multiarea economic dispatch problem in power system with the consideration of different constraints. A new self-tuned TLBO-optimized radial basis function model has been developed by Yang et al. [34] to model the electric vehicle batteries for better efficiency. A multiobjective decomposition-based TLBO algorithm to handle reactive power has been proposed by Medina et al. [35]. Some more diversified application areas of TLBO have been presented as summary of the literature review in Table 1.

From Table 1, it is palpable that TLBO has been applied in several diversified application areas including the fields of power system, optimization problems, pattern recognition, load frequency control, energy systems, engineering designs and machine, but the algorithm has fewer competent applications in the data mining field, specifically in classification, clustering and forecasting-related areas. Inspired by this, in this paper, an attempt has been made to investigate the performance of TLBO with higher-order neural network. Higher-order neural networks are quite efficient than the traditional feed-forward or back-propagation networks. The performance of the algorithm has been analyzed by considering different real-life benchmark datasets and compared with several other methods integrated with HONN. Simulation results of the proposed model divulge that ETLBO–JPSNN performs better than others in terms of classification accuracy.

## 3 Basic preliminaries

In this section, some of the basic preliminaries such as TLBO, ETLBO, Pi-sigma network and Jordan Pi-sigma network have been discussed.

**Table 1** Literature survey on TLBO algorithm

| Reference | Contribution | Area of application | Year |
|---|---|---|---|
| Nayak et al. [36] | MOTLBO | Optimal power flow | 2012 |
| Toğan [37] | TLBO | Engineering design | 2012 |
| Niknam [38] | MOTLBO | Economic dispatch | 2012 |
| Jadhav et al. [39] | MTLBO | Economic load dispatch | 2012 |
| Satapathy et al. [40] | TLBO | ANN | 2012 |
| Zou et al. [41] | TLBO | Multiobjective optimization | 2013 |
| Roy et al. [42] | QOTLBO | Hydro thermal scheduling | 2013 |
| Mandal and Roy [43] | QOTLBO | Power dispatch | 2013 |
| García and Mena [44] | MTLBO | Distributed generation | 2013 |
| Rao and Kalyankar [45] | TLBO | Machining processes | 2013 |
| Roy [46] | TLBO | Scheduling problem | 2013 |
| Roy and Bhui [47] | QOTLBO | Load dispatch | 2013 |
| Singh et al. [48] | TLBO | Power system | 2013 |
| Wang et al. [49] | TLBO | Optimization | 2013 |
| Satapathy et al. [50] | WTLBO | Optimization | 2013 |
| Satapathy et al. [51] | OTLBO | Optimization | 2013 |
| Tuo et al. [52] | HSTL | Optimization | 2013 |
| Xia et al. [53] | STLBO | Sequence planning | 2013 |
| Savsani et al. [54] | TLBO | Robotics | 2013 |
| Wen-Jing et al. [55] | TLBO | Reliability | 2013 |
| Gonzalez-Alvarez et al. [56] | MO-TLBO | Bioinformatics | 2013 |
| Theja et al. [57] | TLBO | Power system | 2013 |
| Sultana and Roy [58] | TLBO | Optimal capacitor placement | 2014 |
| Abarghooee [59] | Gradient-based modified TLBO with black hole | Scheduling of thermal power systems | 2014 |
| Arya and Koshti [60] | TLBO | Load shedding | 2014 |
| Khalghani and Khoob [61] | TLBO | Power quality | 2014 |
| Niu et al. [62] | STLBO | Fuel and solar cell models | 2014 |
| Moghadam and Seifi [63] | Fuzzy-TLBO | Energy loss minimization | 2014 |
| Gonzalez-Alvarez et al. [64] | MTLBO | Biology | 2014 |
| Yammani et al. [65] | MTLBO | Power distribution | 2014 |
| Cheng [66] | TLBO | Temperature calculations | 2014 |
| Sahoo et al. [67] | TLBO | Pattern recognition | 2014 |
| Agrawal et al. [68] | TLBO | Pattern recognition | 2014 |
| Barisal [69] | TLBO | Load frequency control | 2015 |
| Ghasemi et al. [70] | GBTLBO | Power dispatch problem | 2015 |
| Chen et al. [71] | ITLBO | Optimization | 2015 |
| Sahu et al. [72] | TLBO | Power system | 2015 |
| Ghasemi et al. [73] | ITLBO | Power flow | 2015 |
| Chakravarthy et al. [74] | TLBO | Antenna | 2015 |
| Mummareddy and Satapathy [75] | TLBO | Clustering | 2015 |

### 3.1 Teaching–learning-based optimization (TLBO)

It is a new population-based metaheuristic inspired by the teaching and learning process in a classroom environment. The main basis of the algorithm relies on two ideas: (a) the effects of teaching of a teacher upon a student, (b) knowledge gained by the student through the interaction with his or her friends. In this algorithm, a group of students are considered as population, different offered subjects [27] are the design parameters of the algorithm, results of the student are the fitness values, and the teacher is the best solution in the intact population. The algorithm has two consequent phases such as teaching phase and learning phase.

#### 3.1.1 Phase-I (teaching)

The teaching phase simulates the behavior of the student through the teacher. A teacher always tries to give his best in the class to bring all the students up to his own level of knowledge. But practically, it may not possible due to the level of knowledge difference among the students, which can be considered in terms of average, good and best rank. So, for an overall calculation of level of knowledge in the classroom, the mean can be considered which is a random procedure and depends on various external factors.

#### 3.1.2 Phase-II (learning)

The learning phase simulates the behavior of the student through the interaction or discussion of his knowledge with other students or friends in the class. He may acquire some knowledge on a concerned subject from his friends by the method of discussion or interaction. A student can also acquire some new knowledge from his friends if his friends have more expertise than him on the concerned subject.

The algorithm of the TLBO can be realized through the following steps.

**Teacher Phase**

**Step-1.** Initialize the population of students X (candidate solutions) randomly.
**Step-2.** Calculate the mean of each student in the population ($X_{mean}$).
**Step-3.** Compute the fitness of each student in the population and find out the best solution ($X_{teacher}$)
**Step-4.** Generate new population by modifying the solutions in initial population based on best solution (teacher), mean of students in the population (mean) and teaching factor $T_F$.

**for** i=1:1: no.s of weight-sets in the population X

$$T_F = \text{round}(1 + \text{rand}(0,1)(2-1))$$
$$X_i(\text{new}) = X_i(\text{old}) + \text{rand}(1)(X_{\text{teacher}} - T_F * X_{\text{mean}})$$

**End for**

**Learner Phase**

**Step-5.** Update population of student X by comparing fitness of students in old population X and new population $X_{new}$ .

```
for i=1:1: no.s of weight-sets in the population X
        if (fitness of Xi(old) < fitness of Xi (new) )
                Xi= Xi(new)
        Else
                Xi= Xi(old)

        endif
endfor
```

**Step-6.** Randomly select two weight-sets from population and improvise them.

Select i^th and j^th weight-sets $X_i$ and $X_j$ randomly from population.
**If** (fitness of $X_i$ < fitness of $X_j$ )
$X_i(new) = X_i(old) + rand(1)(X_j - X_i)$
**Else**
$X_j(new) = X_j(old) + rand(1)(X_i - X_j)$
**Ifend**

**Step-7.** Check for termination criteria. If reached stop. Else go to step-2.
**Step-8.** Exit

During the teaching phase, the learning quality of the students is being simulated through the teacher. The teacher teaches the students and tries to increase the mean result of the class. The population of students ($X$) is selected randomly and the mean of the population is ($X_{\mathrm{mean}}$) for a particular subject. After computing the fitness of each student in the population, the best solution is termed as ($X_{\mathrm{teacher}}$), who possess the highest knowledge. The teacher may put his best effort for teaching, but the students will acquire the knowledge depending on the quality teaching and the level of students (average, good and best) present in the class. By taking this fact into consideration, the result of teaching quality of the teacher and mean result of the students is represented as: $\mathrm{rand}(1)(X_{\mathrm{teacher}} - T_{\mathrm{F}} \times X_{\mathrm{mean}})$. The value of rand(1) lies in the range [0, 1]. $T_{\mathrm{F}}$ is the teaching factor and is responsible for the modification of the mean value. The value of $T_{\mathrm{F}}$ is described as follows:

$$T_{\mathrm{F}} = \mathrm{round}(1 + \mathrm{rand}(0,1)(2-1)).$$

### 3.2 Elitist teaching–learning-based optimization (ETLBO)

The elitism concept in TLBO was first introduced by Rao et al. [28], and they proposed the Elitist TLBO to solve the constraint optimization problems. Later on, Rajasekhar et al. [76] have introduced the elitism concept in a different manner by integrating opposition-based optimization with TLBO. However, the elitism term is very popular, as it is being frequently used in several population-based evolutionary algorithms. The concept of elitism is the modification of the best solution by replacing the worst solution during the iteration. As in the TLBO algorithm, mean value of the learners is considered, so there may be a possibility of duplicate values after the replacement of elite solution to the worst one. During each generation of the TLBO algorithm, the solutions are modified in both the phases (phase-I and II) and the duplicate solutions are modified in random fashion. Hence, for the Elitist TLBO, we have considered twice of both the population size and no. of population plus the no. of function evaluations required at duplicate value elimination step, i.e., [{$2 \times X \times$ no. of generations} + {No. of function evaluations needed for duplicate value elimination}], where $X$ is the size of the population.

### 3.3 Pi-sigma neural network (PSNN)

Shin and Ghosh [77] introduced Pi-sigma neural network (PSNN), in which exponential increase in no. of weight vectors and processing units are reduced. PSNNs are a special type of feed-forward neural networks having an input layer, a single hidden layer of summation units and product units in the output layer. PSNN passes the output in the form of nonlinear function as the product of summation unit in the output layer [78]. By using fewer weight vectors and processing units, these are capable of quick learning which makes them more accurate and tractable than the other networks [79]. The weights connected from the input layer to hidden layer are tailored during the training, and the weights connecting from hidden layer to output layer are fixed to unity. Due to this reason, the complexity of the hidden layer can be dramatically reduced by the number of tunable weights, for which the model can be easily implementable and accelerated [80, 81] (Fig. 1).

Let the input $x = (x_1, \ldots, x_j, \ldots, x_n)^{\mathrm{T}}$ be the n-dimensional input vectors, where additional $B_j$ is the bias unit and $x_j$ denotes the $j$th component of $X$. The weight vectors such that $w_{ij} = (w_{ij1}, w_{ij2}, \ldots, w_{ijn})^{\mathrm{T}}$, $i = 1, 2, \ldots, k$ are summed at a layer of $k$ summing units, where $k$ is the corresponding order of the network. The output at the hidden layer $h_j$ can be computed by Eq. (1).

$$h_j = B_j + \sum w_{ji} x_i \tag{1}$$

where $w_{ij}$ represents the weight from the input to summing unit. $B_j$ is the bias unit of the neural network. As the weight in the hidden layer to output layer is fixed to 1, so the output O can be computed by using Eq. (2).

$$O = f\left(\prod_{j=1}^{k} h_j\right) \tag{2}$$

where $f(\cdot)$ is a suitable activation function. The order of the PSNN can be computed by the exact number of processing neurons in the hidden layer. The structure of the network may be regularly expanded by adding one or more extra summing units in the hidden layer without hampering the structure of PSNN.

### 3.4 Jordan Pi-sigma neural network (JPSNN)

Instead of using the sum of product of the inputs, the product of sum of input units having the linear summation of a single hidden layer and the product of processing units at output layer are used in PSNN. In 1986, Jordan introduced the implementation of JPSNN [82], whose network structure is similar to PSNN. Only it has an additional recurrent link [83] from output layer to input layer. The JPSNN network constitutes with three layers such as input layer, output layer and the hidden layer with the hidden units (Fig. 2).

The weight vectors $x_1(t), \ldots, x_k(t)$ are set at input layer passing toward the hidden layer, and the weights at hidden layer to output layer are set to 1. The tuned weight vectors

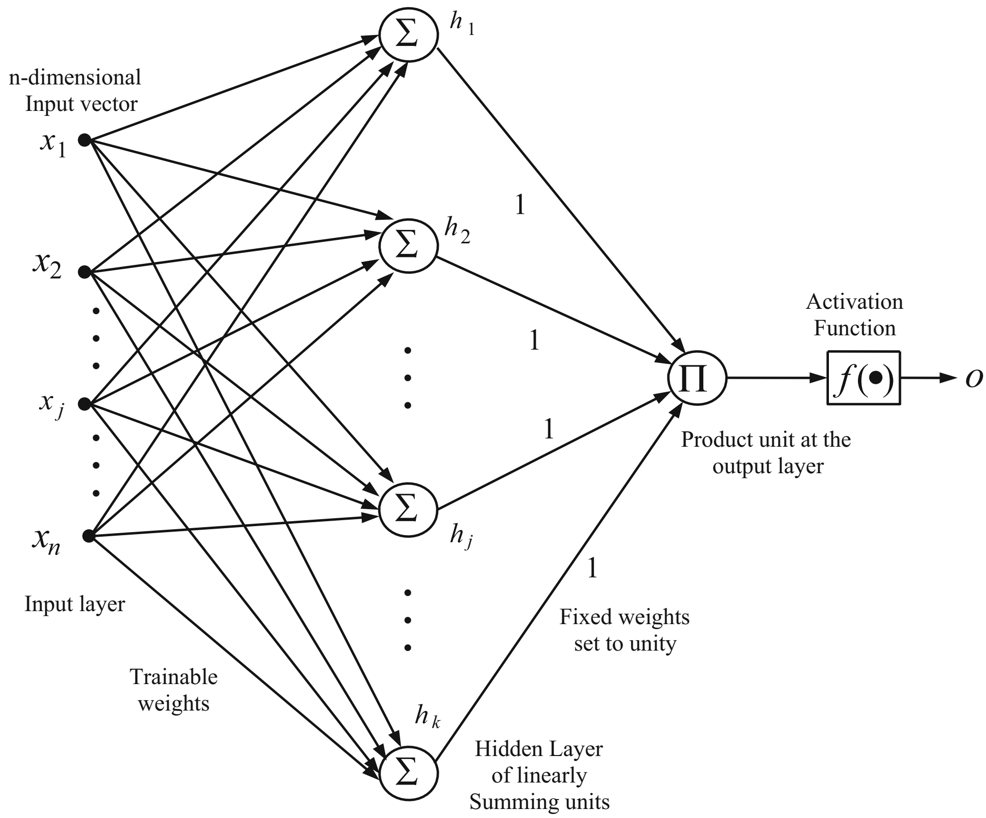



**Fig. 1** Basic structure of a Pi-sigma neural network

are used to test the generalization of new data, and $Z^{-1}$ indicates the operation in time delay.

Suppose at time $t$, $x_k$ $(t)$ is set to the network as $k$th external input. Considering '$n$' number of external inputs, let $w_{ij}$ are the trainable weights, $h_k$ $(t + 1)$ is the summing unit, $O$ $(t + 1)$ indicates the output at time $t + 1$, $f(\cdot)$ indicates the activation function in the network and the number of output is 1.

The overall output [84] at time t is $O(t)$ and is computed as in Eq. (3).

$$O(t) = \begin{cases} x_k(t) & \text{if} \quad 1 \le k \le N \\ 1 & \text{if} \quad k = n + 1 \\ O_k(t) & \text{if} \quad k = n + 2 \end{cases} \tag{3}$$

## 4 Proposed approach

In this section, the Elitist TLBO-based Jordan Pi-sigma neural network has been proposed for classification of real data. The ETLBO algorithm is used with a standard back-propagation-based gradient descent learning for finding the best weight-units for JPSNN network. The main objective is to compare the performance of the proposed method with other methods such as GA–JPSNN, PSO–JPSNN and TLBO–JPSNN. Also, the performance of the PSNN networks with TLBO and ETLBO has been compared with other methods.

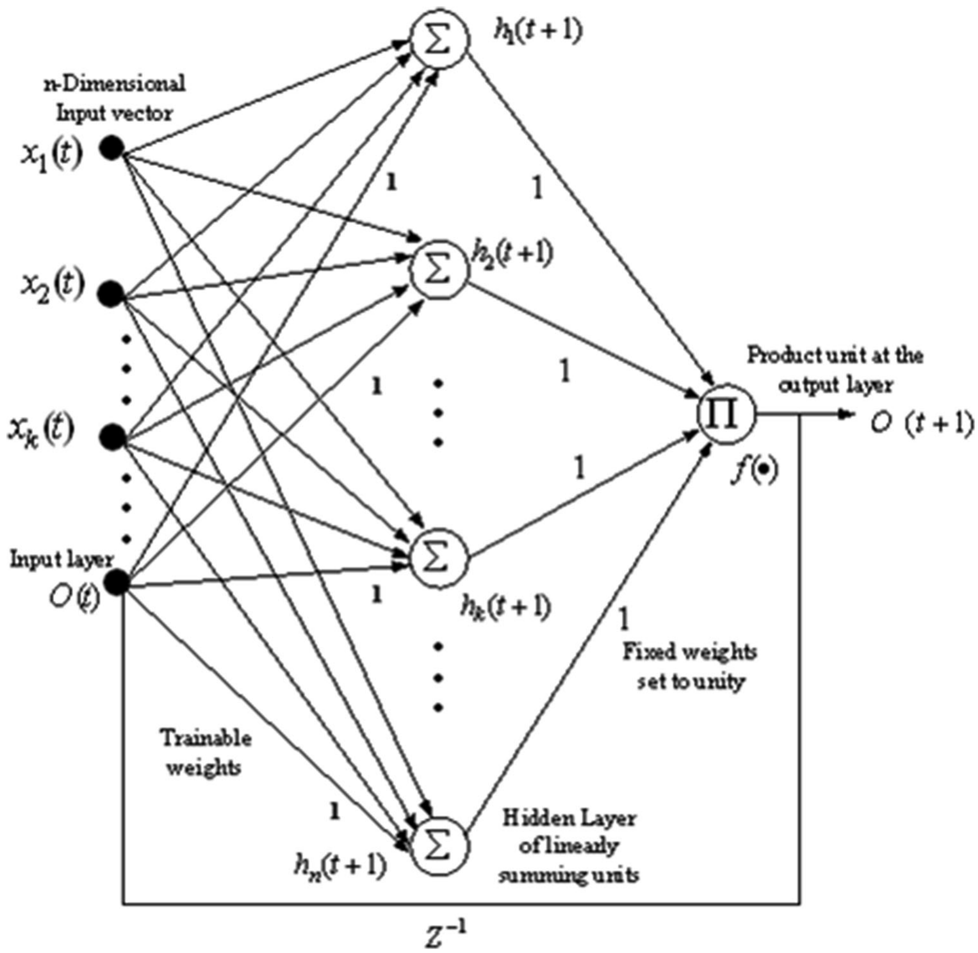
The proposed ETLBO–JPSNN starts with the initial population of the learners $(X)$, initialized with '$n$' no. of weight-units for JPSNN. The weight-units in the population are randomly initialized between the values of $-1$ to 1, and those will act like potential candidate weight-units of JPSNN for classification of an individual dataset. The individual weight-unit in $X$ is computed as in Eq. (4), and the set of weight-units is presented as in Eq. (5).

$$x_i = \left(w_{i,1}, w_{i,2}, \ldots, w_{m\times a\times(2\times k+1)}\right) \tag{4}$$

$$X = (x_1, x_2, \ldots, x_n) \tag{5}$$

where the value of $k$ is to be chosen and $(2 \times k + 1)$ is the no. of functionally expanded values for a single value in input data, '$a$' is the number of attributes in a single input pattern, '$m$' is the number of patterns in the dataset and '$n$' is the number of weight-units in the population. The aim is to prune out optimal weight-set for the JPSNN network for better classification accuracy. The individual weight-unit $W_i$ is set to JPSNN, and the network is trained with a particular dataset. Depending on the obtained output of the network and provided target output, the error of the network is calculated. The working model of the proposed work is illustrated in detailed flow diagram in Fig. 3. For an individual dataset, the root-mean-square error (RMSE)

**Fig. 2** Architecture of JPSNN




for each weight-unit $W_i$ is computed by using output Eq. (6) of the network and given target output (Algorithm 2).

$$\mathrm{RMSE} = \sqrt{\frac{\sum_{i=1}^{n} (O_i - \hat{O}_i)^2}{n}} \tag{6}$$

The network uses supervised learning and has been trained by using a standard back-propagation gradient descent learning [85]. Before training the synaptic weights, the learning method initializes by considering the small random values. As JPSNN uses the adaptive training method, the total error $E$ can be calculated by Eq. (7).

$$E_j(t) = d_j(t) - O_j(t) \tag{7}$$

where $d_j(t)$ is the final desired output at time $(t-1)$. At each of the time $(t-1)$, the output $O_j(t)$ is computed by using Eq. (8).

$$O_j(t) = f\left(\prod_{L=1}^{k} h_L(t)\right) \tag{8}$$

Here, $h_L(t)$ is calculated by using Eq. (9).

$$h_L(t) = \sum_{n=1}^{n} w_{Ln}x_n(t) + w_{L(n)} + w_{L(n+1)}O(t-1) = \sum_{n=1}^{n+1} w_{Ln}Z_n(t-1) \tag{9}$$

where $h_L(t)$ is the activation of '$L$' unit, '$f$' is the Sigmoid activation function between the bounded range of [0, 1]. For each of the nodes in the current layer, repeatedly the overall output error is calculated by using Eq. (10).

$$E_k = {}^{1}/_{m_{\mathrm{TR}}} \sum_{i=1}^{m_{\mathrm{TR}}} O_i - Z_{ki} \tag{10}$$

where $z_{ik}$ is the output of $k$th node with respect to $i$th data value and $m_{\mathrm{TR}}$ is the training sets.

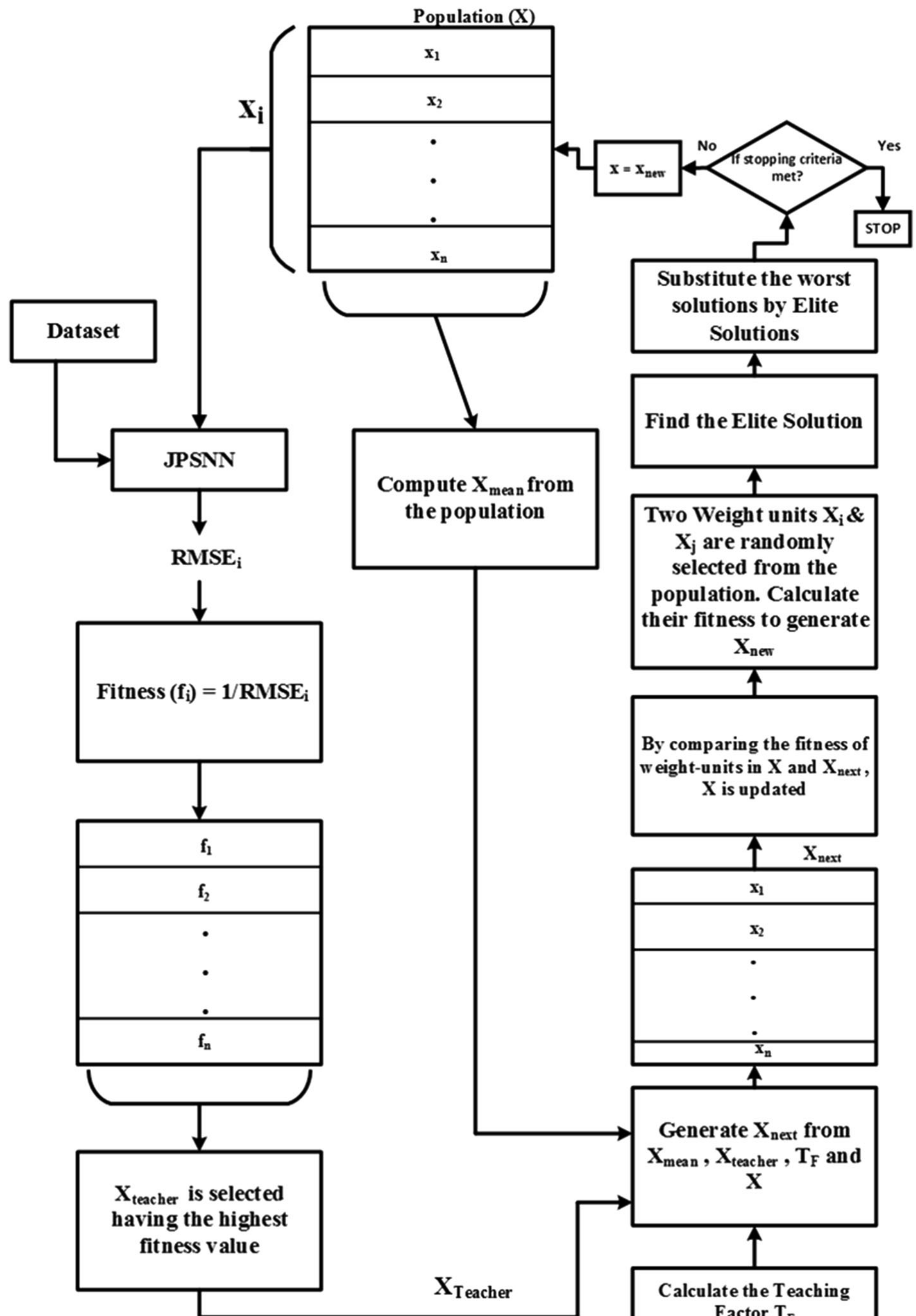
The change in the weight values and each time updation of weight-units are calculated as in Eqs. (11) and (12), respectively.

$$\Delta w_j = \eta\left(\prod_{j\neq 1}^{m} h_{ji}\right)x_k \tag{11}$$

$h_{ji}$ is the summing layer output and $\eta$ is the learning rate.

$$w_i = w_i + \Delta w_i \tag{12}$$

**Fig. 3** Detailed working model of the proposed ETLBO–JPSNN




To accelerate the convergence of errors, an extra term $\alpha$ (momentum) is added and the weight-unit values are calculated by Eq. (13).

$$w_i = w_i + \alpha \Delta w_i \tag{13}$$

The accuracy of classification is computed as in Eq. (14)

$$\text{Accuracy} = \frac{\sum_{i=1}^{n} \sum_{\substack{j=1,\\ i==j}}^{m} \text{cm}_{i,j}}{\sum_{i=1}^{n} \sum_{j=1}^{m} \text{cm}_{i,j}} \times 100\% \tag{14}$$

where cm is confusion matrix.

After evaluating the fitness values for each weight-unit in the population, the units having maximum fitness values are selected as Teacher ($\mathbf{x_{teacher}}$). Then, in the population, the mean of the weight-sets ($\mathbf{X_{mean}}$) is computed by calculating the mean of all the weight-units, and among them, the elitist solution is selected. After calculating the teaching factor ($\mathbf{T_F}$), the next population $\mathbf{X_{next}}$ is generated from $\mathbf{X}$, $\mathbf{X_{mean}}$, $\mathbf{x_{teacher}}$ and $\mathbf{T_F}$. Then, the weight-units in initial population $\mathbf{X}$ are updated by comparing the fitness of weight-units in $\mathbf{X}$ and $\mathbf{X_{next}}$. This process has continued till the execution of maximum no. of iterations or significant increase in the fitness of weight-units in the population. The complete flow of the components in the working model of the proposed ETLBO–JPSNN is depicted in Fig. 3.

**Algorithm – 1 ETLBO-JPSNN Learning Model**

INPUT: Dataset with target vector 't' , initial population of weight-units 'X', Bias B.

OUTPUT: JPSNN with optimized weight-set 'w'.

1. Initialize the population of learners as 'n' no. of weight-units .
X = $\{x_1, x_2, \ldots, x_n\}$, where each $x_i$ is a randomly initialized potential weight-unit of JPSNN network as $x_i = \{w_{i,1}, w_{i,2}, \ldots, w_{i,n}\}$.
2. Compute the mean of all the weight-units ($x_{mean}$) in the population X.
3. Find elitist solutions in the population.
4. Calculate the fitness of all the weight-units in 'X' by using algorithm-2 and select the weight-set having maximum fitness as best weight-unit ($x_{teacher}$).
5. Generate the next population $X_{next}$ by using weight-units in the old population X, $X_{mean}$ , $x_{teacher}$ and $T_F$.
    **for** i=1:1: no.s of weight-units in the population X
        $T_F = round(1 + rand(0,1)(2-1))$
        $x_i = x_i + rand(1)(x_{teacher} - T_F * X_{mean})$
        $X_{next}(i) = x_i$
    **endfor**
6. By comparing the fitness of weight-unts in X and $X_{next}$, Update the population of weight-units.
    **for** i=1:1: nos of weight-units in the population X
        **if** (X(i) < $X_{next}$(i) )
            X(i)= $X_{next}$(i)
        **endif**
    **endfor**
7. Select randomly two weight-units in the population and improvise them.
    **for k** =1:1: no.s of weight-units in the population
        Select i$^{th}$ and j$^{th}$ weight-sets $x_i$ and $x_j$ randomly from the population .
        Calculate the fitness of $x_i$ by using algorithm-2 as $F_i$= **Fitness-From-Training (x,w,t,B).**
        Calculate the fitness of $x_j$ by using algorithm-2 as $F_j$= **Fitness-From-Training (x,w,t,B).**
        **If** ($F_i$< $F_j$ )
        $X_{new}(i) = x_i + rand(1)(x_j - x_i)$
        **Else**
        $X_{new}(j) = x_j + rand(1)(x_i - x_j)$
        **Ifend**
    **Endfor**
8. Substitute the worst solutions with elite solutions in the population.
9. Check for termination criteria:
    **if** (maximum no. of generation reached **OR** 95% of weight-units in the population are similar )
        then goto step-10.
    **else** goto step-2
    **endif**
10. **Exit**

***Algorithm – 2: Fitness From Training Procedure***

1. **FUNCTION** F= **Fitness-From-Training** (x, w, t, B)
2. **FOR** i = 1 to n, n is the length of the dataset
3. Compute the output at the hidden layer by using (9)
4. Compute the output of the network by using (8).
5. Calculate the error term by using eq. (7) and compute the fitness F(i)=1/RMSE.
6. **END FOR**
7. Compute root mean square error (RMSE) by using eq. (6) from target value and output.
8. The weight changes by using the BP-GDL algorithm can be computed by using (11).
9. Update the weight by using eq. (12).
10. The weight value can be calculated after adding the momentum term by using eq. (13).
11. **IF** the stopping criteria like training error or maximum no. of epochs are satisfied, then Stop.
    **ELSE** repeat the step from 2 to 11.
12. **END**

In Algorithm 1, $T_F$ is not a TLBO algorithmic parameter and rather is a function to decide the mean, so its value changes (either 1 or 2) and is randomly decided by the expression $T_F = \text{round}(1 + \text{rand}(0,1)(2-1))$ with the chance in equal probability. As both 'rand and $T_F$' are not TLBO algorithmic-specific parameters, so their values are not to be tuned as in case of the mutation and crossover parameters in GA, the value of the inertia weight in PSO, etc. Both 'rand and $T_F$' are generated randomly during the run of the program, and TLBO does not require controlling any major parameters. This property of TLBO makes it more popular than the other evolutionary population-based algorithms.

## 5 Experimental setup

In this section, the simulation environments, dataset used for the experiment and other experimental details have been illustrated.

### 5.1 Simulation environment

The proposed approach has been designed to correctly classify the data having large number of feature sets and various class labels. A vast comparative analysis among all the classifiers has been done in two phases. In the first phase, both the TLBO and ETLBO have been applied to only PSNN and are compared with GA and PSO based methods. In the next phase, the same TLBO and ETLBO have been applied to JPSNN and also been compared with GA and PSO. The proposed ETLBO–JPSNN along with TLBO–JPSNN, PSO–JPSNN, GA–JPSNN and ETLBO–PSNN, TLBO–PSNN, PSO–PSNN, GA–PSNN methods have been implemented by using MATLAB 9.0 on a system with an Intel Core 2 Duo CPU T5800, 2 GHz processor, 2 GB RAM and Microsoft Windows-2007 OS.

### 5.2 Parameter settings

The quality of each learner is represented through its corresponding fitness value with elite solutions. The lists of parameters set for both JPSNN and ETLBO during the experiment are given in Table 2.

### 5.3 Dataset information

The benchmark datasets (Table 3) used for classification are originated from UCI machine learning repository [86] and processed by KEEL software [87]. The first column in the table shows the corresponding name of the datasets. The other information such as number of attributes and number of class labels have been indicated in the rest of some columns. The detail descriptions about all these dataset can be obtained at 'http://archive.ics.uci.edu/ml/' and 'http://keel.es/'.

**Table 2** Parameter settings

| JPSNN parameters | ETLBO parameters |
|---|---|
| Initialization of weight vector except output layer: values between $-1$ and 1 | $T_F = 1$ or 2 (with equal probability) |
| Initialization of weight vector at output layer: 1 | Population size = 40 |
| Number of epochs: 500 | Number of generations = 100 |
| – | Stopping criteria: maximum number of iteration |

**Table 3** Dataset information

| Dataset | Number of pattern | Number of features/attributes | Number of classes |
|---|---|---|---|
| Heart | 256 | 14 | 02 |
| Hepatitis | 155 | 19 | 02 |
| Pima | 768 | 09 | 02 |
| Ecoli | 336 | 07 | 08 |
| Vehicle | 846 | 18 | 04 |
| Balance | 625 | 04 | 03 |
| Hayesroth | 160 | 05 | 03 |
| New Thyroid | 215 | 06 | 03 |
| Wine | 178 | 14 | 03 |
| Dermatology | 256 | 34 | 06 |
| Parkinson | 196 | 23 | 02 |
| Ionosphere | 351 | 33 | 02 |
| Coil2000 | 9822 | 85 | 02 |
| SpectF Heart | 267 | 44 | 02 |
| Spambase | 4597 | 57 | 02 |

The brief description about the datasets used for the experimental analysis is as follows:

*Heart dataset* This dataset is related to the human heart, and its attributes are age, sex, chest pain type, resting blood pressure, etc. It comprises of 256 patterns, 14 no. of attributes and 2 class labels. It is of multivariate type and has no missing values

*Hepatitis dataset* This dataset is used as the information about the hepatitis patients. It has 155 patterns, 19 no. of attributes and 2 no. of classes. It has no missing values

*Pima dataset* This dataset is a collection of females more than 21 years old of Pima Indian Heritage. It consists of 768 patterns, 9 no. of attributes and 2 class labels. There are no missing values for this dataset

*Ecoli dataset* This dataset is used to predict the localization site of proteins by employing some measures about the cell such as cytoplasm and lipoprotein. It has 336 patterns, 07 no. of attributes and 08 no. of classes. It has no missing values

*Vehicle dataset* This dataset is used to classify a given silhouette as one of four types of vehicle, using a set of features extracted from the silhouette. It is based on the vehicle classification having 846 patterns, 18 no. of attributes and 4 no. of classes. It has no missing values

*Balance dataset* This dataset is used to model the psychological experimental results. The balance scale may shift to left or right or to be balanced. It is based on the balance-scale measurements having 625 patterns, 4 no. of attributes and 3 classes. It has no missing values

*Hayesroth dataset* This dataset consists of 5 numerically valued attributes such as name, hobby and age. It has 160 no. of patterns, 5 no. of attributes and 3 class labels having no missing values

*New Thyroid dataset* This dataset is used to classify the patient's thyroid condition as normal, hypo and hyper. The dataset consists of 215 patterns,6 no. of attributes and 3 class labels. It has no missing attributes

*Wine dataset* These are the resultant of the chemical analysis wines grown in the same region in Italy. The characteristics of the dataset are multivariate type and its attributes are alcohol, malic acid, etc. It consists of 178 patterns, 14 no. of attributes and 3 no. of classes. It has no missing values

*Dermatology dataset* This dataset is used to detect the type of erythemato-squamous disease. It has 256 patterns, 34 no. of attributes and 02 no. of classes

*Parkinson dataset* This is used to distinguish the healthy peoples from those who are affected with the Parkinson's diseases. It has 196 patterns, 23 no. of attributes and 02 no. of classes

*Ionosphere dataset* This dataset is used to test the good or bad signals. It has 351 patterns, 33 no. of attributes and 02 no. of classes

*Coil2000 dataset* This is a real-world dataset, which contains information on customers of an insurance company and was used in the CoIL 2000 Challenge. It has 9822 patterns, 85 no. of attributes and 02 no. of classes

*SpectF Heart dataset* This dataset is used to the diagnosis of cardiac Single Proton Emission Computed Tomography (SPECT) images. It has 267 patterns, 44 no. of attributes and 02 no. of classes. It has no missing values

*Spambase dataset* It describes the information about 4597 e-mail messages. The task of this dataset is to determine whether a given e-mail is spam (class 1) or not (class 2), depending on its contents. It has 4597 patterns, 57 no. of attributes and 02 no. of classes

### 5.4 Cross-validation

The cross-validation [88] is a statistical technique, which is used to estimate the generalized performance of the learned model from the data. The comparison between the learning algorithms are made by dividing dataset into two segments: training set and testing set. In *k*-fold cross-validation (Mosteller and Turkey [89]), the data are partitioned into *k* equally or nearly equal-sized fragments on which training and validation are performed in such a way that, in each test, different fold of the data is used for training and validation.

In this paper, all the datasets used for classification are prepared for cross-validation by using five fold cross-validation technique. The datasets have been prepared by splitting into fivefold, out of which fourfold are used for training and onefold is used for testing. For example (Table 4), the 'Hayesroth-5-1tra.dat' and 'Hayesroth-5-1tst.dat' data are a pair of datasets sample of Hayesroth dataset which is used for training and testing phase for a single run, respectively. As fivefold cross-validation is employed, the Hayesroth dataset contains five such pair of dataset sample for training and testing the algorithms. All

**Table 4** Fivefold cross-validated Hayesroth dataset

| Dataset | Data files | Number of pattern | Task | Number of pattern in class-1 | Number of pattern in class-2 | Number of pattern in class-3 |
|---|---|---|---|---|---|---|
| Hayesroth | hayesroth-5-1trn.dat | 128 | Training | 52 | 51 | 25 |
| | hayesroth-5-1tst.dat | 32 | Testing | 13 | 13 | 06 |
| | hayesroth-5-2trn.dat | 128 | Training | 52 | 51 | 25 |
| | hayesroth-5-2tst.dat | 32 | Testing | 13 | 13 | 06 |
| | hayesroth-5-3trn.dat | 128 | Training | 52 | 51 | 25 |
| | hayesroth-5-3tst.dat | 32 | Testing | 13 | 13 | 06 |
| | hayesroth-5-4trn.dat | 128 | Training | 52 | 51 | 25 |
| | hayesroth-5-4tst.dat | 32 | Testing | 13 | 13 | 06 |
| | hayesroth-5-5trn.dat | 128 | Training | 52 | 52 | 24 |
| | hayesroth-5-5tst.dat | 32 | Testing | 13 | 12 | 07 |

other datasets are prepared for fivefold cross-validation in the same manner and collected from KEEL dataset repository.

## 6 Result analysis and discussion

In this study, to investigate the efficiency of the proposed algorithm, a rigorous performance comparison has been made between the proposed method and other methods. First, the comparison has been made for the Pi-sigma network. With PSNN, four optimization techniques such as ETLBO, TLBO, PSO and GA have been applied to find out the improvements in the weight-set of the network. Every time, the changes in the weight-sets are measured and the process has been continued till there are any significant changes in the weight-set, i.e., closer to the target value. Then, the same procedure has been followed for Jordan pi-sigma network. The average classification accuracy results of all the datasets are indicated in Tables 5 and 6. In all the 11 cases, the performance results of both ETLBO–PSNN and ETLBO–JPSNN are quite promising. In some cases such as Pima, Balance and Wine datasets, the training results of TLBO–PSNN are a little better than the results of ETLBO–PSNN. This is due to the evaluations of repeated same elitist solutions after the calculations of mean values in the population. In case of worst solutions in the population, the elitist value can be replaced. But in the case of repeated duplicate solutions, for each time, the elitist value can be replaced with lesser chance of repetition. So, in some cases of only TLBO, the best value in the population may also be treated as the Elitist value in the ETLBO. However, in most of the cases, ETLBO

**Table 5** Performance comparison between ETLBO–PSNN, TLBO–PSNN and other models

| Dataset | Average classification accuracy (%) | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | ETLBO–PSNN | | TLBO–PSNN | | PSO–PSNN | | GA–PSNN | |
| | Train | Test | Train | Test | Train | Test | Train | Test |
| Heart | **96.008** | **95.864** | 95.256 | 95.398 | 90.231 | 91.148 | 89.203 | 90.128 |
| Hepatitis | **93.635** | **93.828** | 93.324 | 93.637 | 82.021 | 82.018 | 80.071 | 79.058 |
| Pima | **96.312** | **96.463** | 96.365 | 96.338 | 91.273 | 91.315 | 90.244 | 89.382 |
| Ecoli | **95.238** | **95.009** | 94.834 | 94.768 | 91.003 | 91.018 | 90.365 | 90.333 |
| Vehicle | **98.725** | **98.776** | 96.368 | 96.277 | 91.607 | 91.552 | 90.333 | 90.398 |
| Balance | **98.907** | **98.623** | 98.935 | 98.942 | 95.206 | 95.094 | 94.129 | 93.795 |
| Hayesroth | **97.398** | **97.346** | 96.663 | 95.703 | 91.292 | 90.278 | 90.200 | 90.234 |
| New Thyroid | **98.016** | **98.272** | 97.098 | 97.137 | 94.365 | 94.320 | 94.310 | 94.096 |
| Wine | **96.139** | **96.743** | 96.274 | 96.200 | 94.449 | 94.317 | 91.324 | 92.236 |
| Dermatology | **98.625** | **98.473** | 98.205 | 98.213 | 95.604 | 95.263 | 95.827 | 95.318 |
| Parkinson | **97.676** | **97.438** | 96.467 | 96.306 | 94.362 | 94.824 | 91.769 | 92.383 |

The bold faced results are the obtained results of the proposed method

**Table 6** Performance comparison between ETLBO–JPSNN, TLBO–JPSNN and other models

| Dataset | Average classification accuracy (%) | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | ETLBO–JPSNN | | TLBO–JPSNN | | PSO–JPSNN | | GA–JPSNN | |
| | Train | Test | Train | Test | Train | Test | Train | Test |
| Heart | **96.679** | **96.368** | 95.763 | 95.438 | 92.349 | 92.304 | 90.581 | 90.312 |
| Hepatitis | **93.324** | **93.638** | 93.408 | 93.762 | 84.297 | 84.328 | 81.634 | 81.547 |
| Pima | **97.389** | **97.437** | 96.008 | 96.037 | 92.964 | 92.819 | 91.461 | 90.258 |
| Ecoli | **96.003** | **96.186** | 95.076 | 95.029 | 92.897 | 92.653 | 90.873 | 90.537 |
| Vehicle | **98.968** | **98.824** | 96.999 | 96.462 | 93.089 | 93.319 | 91.007 | 90.658 |
| Balance | **98.906** | **98.867** | 98.914 | 98.598 | 96.079 | 96.563 | 94.862 | 94.633 |
| Hayesroth | **98.067** | **98.637** | 96.333 | 96.295 | 92.005 | 92.293 | 90.769 | 90.258 |
| New Thyroid | **99.114** | **98.903** | 97.863 | 97.632 | 95.463 | 94.949 | 94.627 | 94.004 |
| Wine | **98.753** | **98.999** | 96.780 | 96.465 | 95.392 | 95.327 | 93.507 | 93.004 |
| Dermatology | **98.579** | **98.769** | 98.506 | 98.362 | 95.570 | 95.237 | 95.916 | 95.283 |
| Parkinson | **98.653** | **98.529** | 97.008 | 96.982 | 95.336 | 95.650 | 92.438 | 92.164 |

The bold faced results are the obtained results of the proposed method

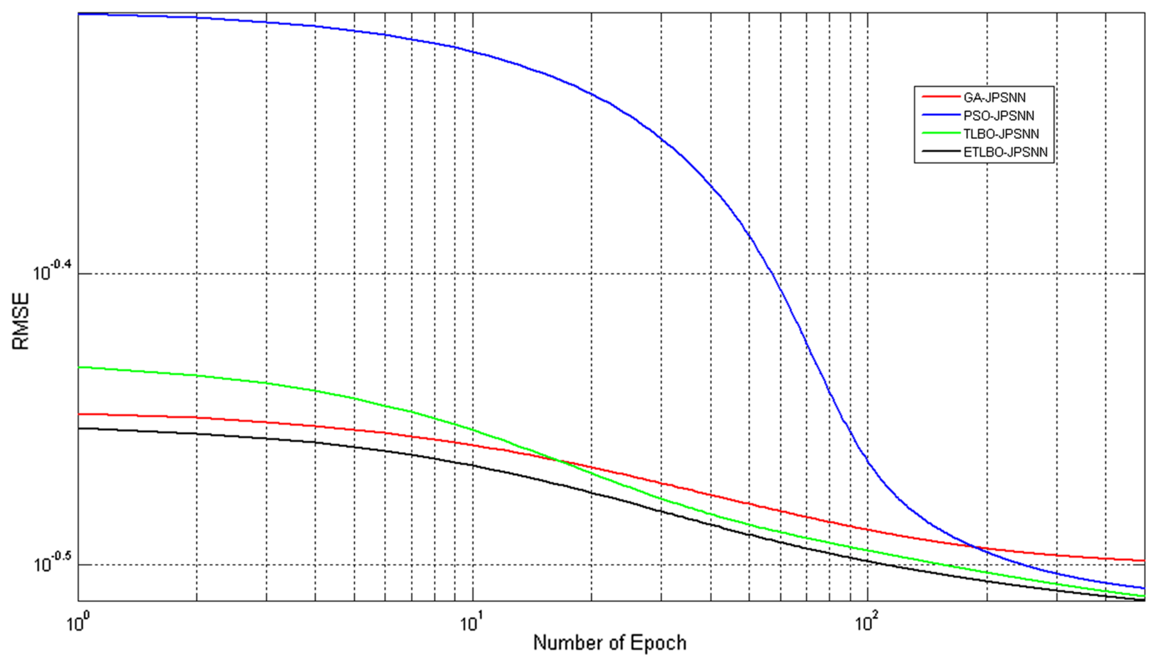



**Fig. 4** Performance of ETLBO–JPSNN (RMSE vs. number of epochs) on Heart dataset

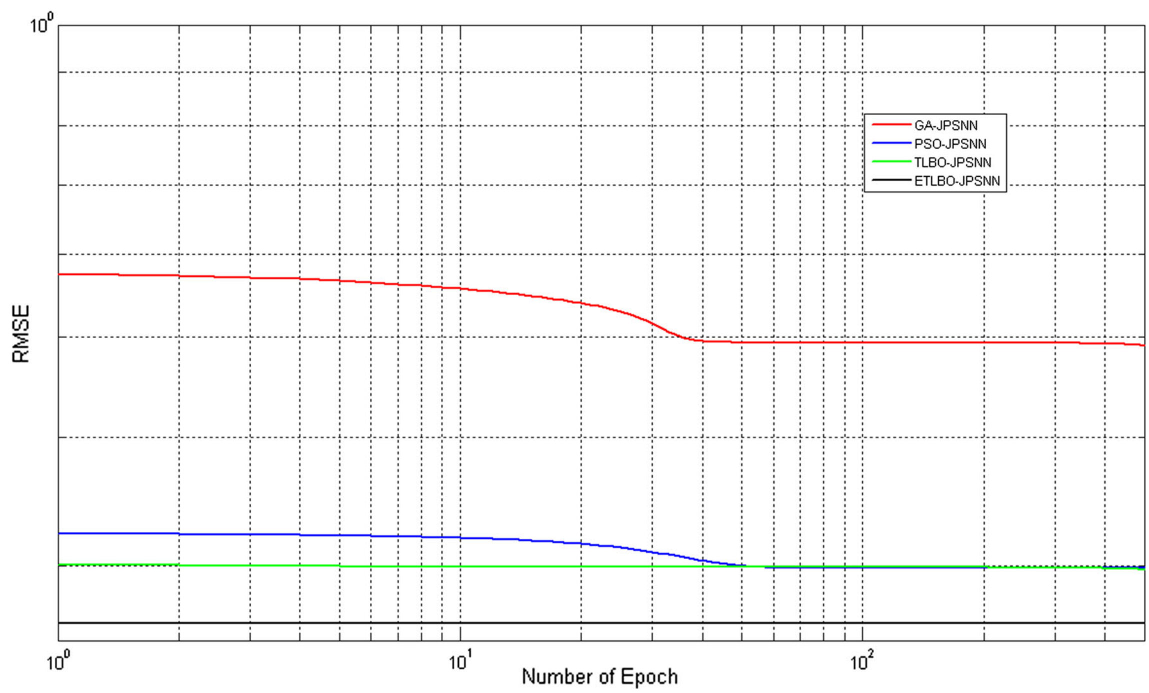



**Fig. 5** Performance of ETLBO–JPSNN (RMSE vs. number of epochs) on Ecoli dataset

performs better than TLBO with the elitism property. Some marginal better classification accuracies have been obtained in case of larger population size datasets such as Pima, Vehicle and Balance. For one instance, the training and testing results of Pima dataset for TLBO–JPSNN are 96.008 and 96.037, respectively. Compared to that, the ETLBO–JPSNN performs with better classification accuracies such as 97.389 and 97.437. So, ETLBO performs quite better than TLBO in case of large population sizes. The performance of the proposed ETLBO–JPSNN along with other approaches for Heart, Ecoli, New Thyroid, Wine and Parkinson's datasets have been shown in terms of RMSE and number of epochs in Figs. 4, 5, 6, 7 and 8. The simulation results in all the cases demonstrate the superiority on the performance of ETLBO–JPSNN as compared to other techniques.

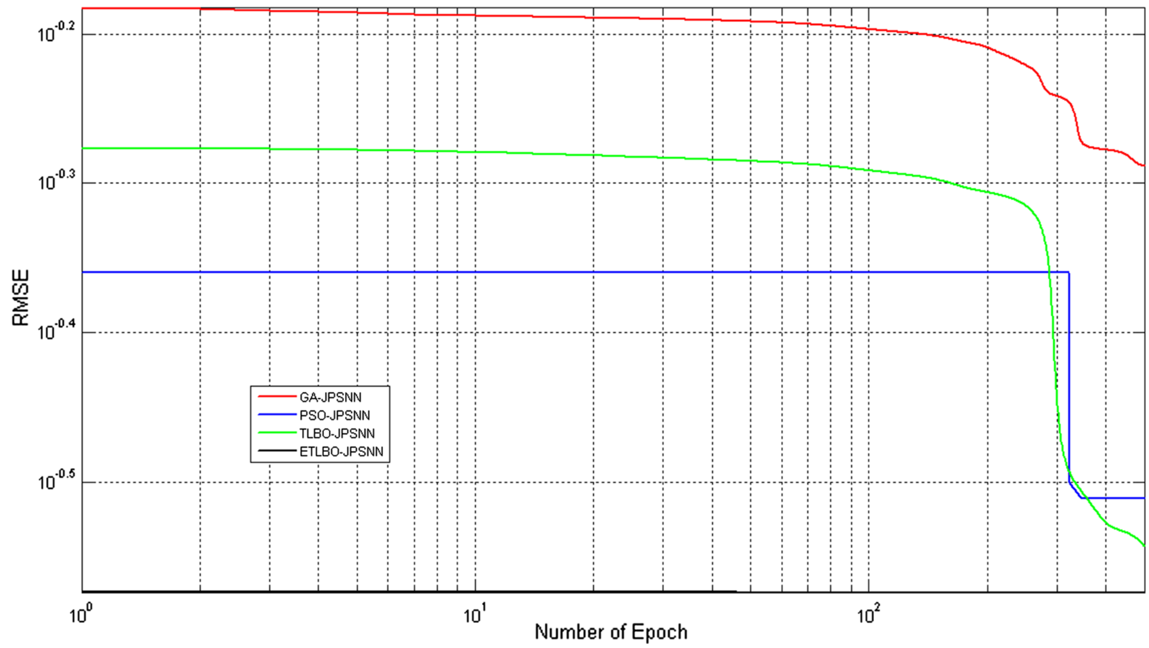



**Fig. 6** Performance of ETLBO–JPSNN (RMSE vs. number of epochs) on New Thyroid dataset

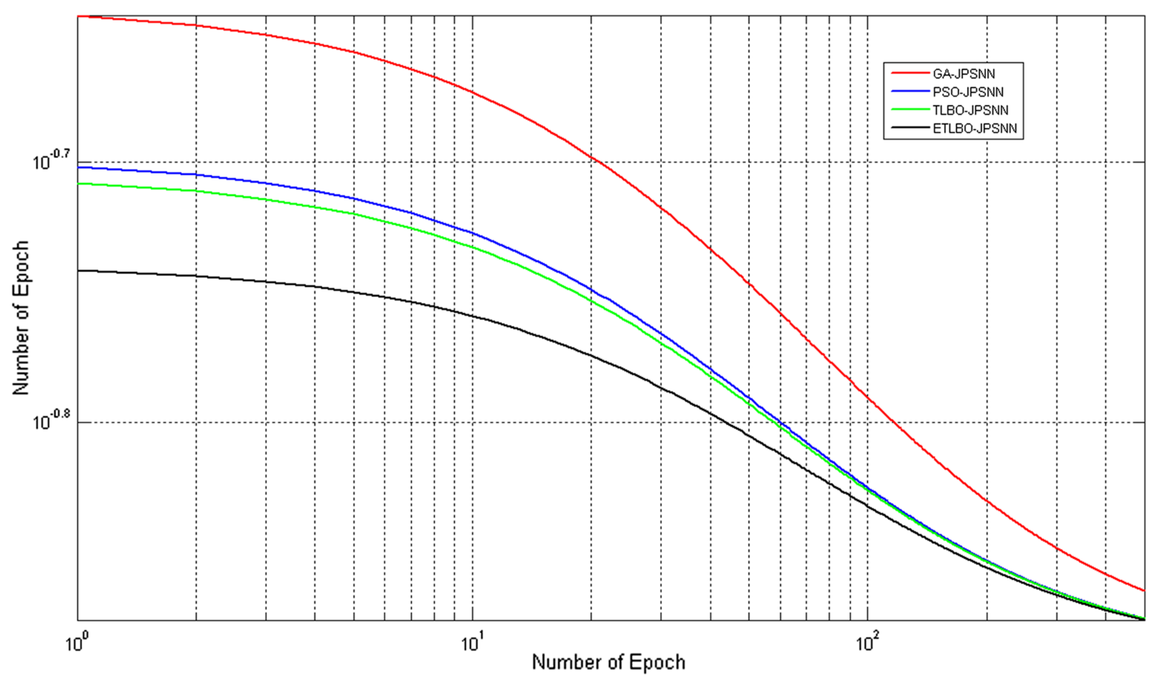



**Fig. 7** Performance of ETLBO–JPSNN (RMSE vs. number of epochs) on Wine dataset

## 7 Comparison with other neural network classifiers

In this section, performance comparisons of the proposed method have been made with another higher-order neural network such as functional link artificial neural network (FLANN). FLANN [90, 91] is a class of higher-order neural networks that makes use of higher combination of its inputs. In FLANN, the dimension of input pattern increases artificially through the functional expansion, and then, the extended and transformed input data are used to train the feed-forward network. During functional expansion, various mathematical functions, such as sine, cosine and log, are used to transform an original input pattern to its extended version. The number of input terms during functional expansion depends upon the number of

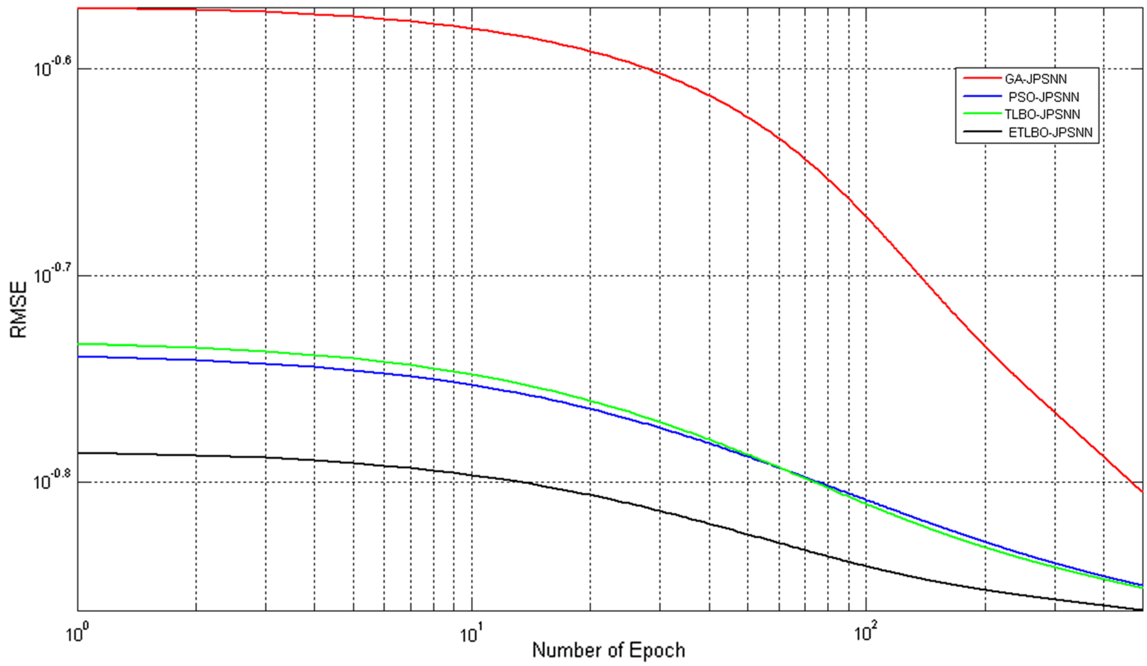



**Fig. 8** Performance of ETLBO–JPSNN (RMSE vs. number of epochs) on Parkinson dataset

attributes of an input pattern. The functionally expanded values for dataset $x$ can be generated by using Eq. (15), where $x_i(j)$ stands for $j$th attribute value of $i$th pattern and '$x$' is a dataset in a form of matrix of order $m \times n$. Here, '$m$' is the no of pattern and '$n$' is the no. of attribute of each pattern.

$$\varphi(x_i(j)) = \{x_i(j), \cos \Pi x_i(j), \sin \Pi x_i(j), \cos 2\Pi x_i(j), \sin 2\Pi x_i(j), \ldots, \cos n\Pi x_i(j), \sin n\Pi x_i(j) \tag{15}$$

Here, $2n + 1$ number of functionally expanded values are generated for an input attribute value $x_i(j)$ of a pattern $x_i$. So, $(n \times (2n + 1))$ number of expanded values are generated for a single input pattern $x_i$. In Eq. (15), value of '$i$' can be ranged from '1' to '$n$' and value of '$j$' can be ranged from '1' to '$m$', where '$m$' and '$n$' are number of input patterns and no. of attribute values of each input pattern except class label, respectively. Therefore, the complete set of functionally expanded values for dataset $x$ is represented using Eq. (16).

$$\varphi = \Big\{\{\varphi(x_1(1)), \varphi(x_1(2)), \ldots, \varphi(x_1(n))\}^{\mathrm{T}}, \{\varphi(x_2(1)), \varphi(x_2(2)), \ldots, \varphi(x_2(n))\}^{\mathrm{T}} \ldots \{\varphi(x_m(1)), \varphi(x_m(2)), \ldots, \varphi(x_m(n))\}^{\mathrm{T}}\Big\}. \tag{16}$$

The weights of FLANN set randomly prior to the above functionally expanded values '$\varphi$' are the input to FLANN classifier. Total $n \times (2n + 1)$ number of weights are set for each individual pattern, as each input pattern is transformed to $n \times (2n + 1)$ number of functionally expanded values. Random initialization of weight-set for each individual pattern can be visualized as in Eq. (17).

$$W_i = \{w_{i,1}, w_{i,2}, \ldots, w_{i,2n+1}\}, \quad \text{for } i = 1, 2, \ldots, n \tag{17}$$

Here, $w_i$ is the randomly initialized weight vector for a single input value. Hence, initialization of set of weight for input patterns of dataset '$x$' can be viewed as a weight vector $W = \{W_1, W_2, \ldots, W_m\}^{\mathrm{T}}$, where $W_i$ is the set of weight for $i$th pattern in the dataset '$x$'. The dataset '$x$' is supplied to FLANN in terms of functionally expanded values '$\varphi$' and the net output is obtained as follows. First, value of $S$ is calculated as $S = \varphi * W = \{s_1, s_2, \ldots, s_m\}$. Then, the net output $Y$ is computed as $Y = f(S) = \{f(s_1), f(s_2), \ldots, f(s_m)\} = \{y_1, y_2, \ldots, y_m\} = \{\tan h(s_1), \tan h(s_2), \ldots, \tan h(s_m)\}$. Here, $\tan h$ is used as activation function and net output $y_i$ is for input pattern $x_i$.Based on net output $y_i$ and given target value $t_i$, error of FLANN is calculated and gradient descent learning (GDL) method is adapted to adjust weight values of FLANN. The details of design procedure of GDL-based FLANN may be found from recently published related works [92–94].

Initially, the population of weight-sets $\mathbf{X}$ (population of students) is initialized with '$n$' no. of weight-sets for FLANN. Each weight-set in the population $\mathbf{X}$ is a vector of weights initialized randomly between $-1$ and 1, which are the potential candidate weight-sets of FLANN model of a particular dataset. Each weight-set $x_i$ is set to FLANN individually, and the FLANN model is trained with a particular dataset. The corresponding values of RMSE and fitness have been computed as same to the PSNN and JPSNN networks.

After evaluation of fitness values for each weight-set in $X$, the weight-set with maximum fitness is selected as Teacher ($\mathbf{x}_{\mathbf{teacher}}$). From the population of $X$, the mean of the weight-sets ($\mathbf{X}_{\mathbf{mean}}$) is computed by calculating the mean of all the weight-sets in $X$. After the calculation of teaching factor ($\mathbf{T}_{\mathbf{F}}$), the next population $\mathbf{X}_{\mathbf{next}}$ is generated from $\mathbf{X}$, $\mathbf{X}_{\mathbf{mean}}$, $\mathbf{x}_{\mathbf{teacher}}$ and $\mathbf{T}_{\mathbf{F}}$. Then, the weight-sets in initial population $\mathbf{X}$ are updated by comparing fitness of weight-sets in $\mathbf{X}$ and $\mathbf{X}_{\mathbf{next}}$. The resultant population of weight-sets $\mathbf{X}$. goes through improvisation steps in which two weight-sets are randomly selected from the population $\mathbf{X}$ and best among them are chosen as weight-set for next generation $\mathbf{X}_{\mathbf{next}}$ by comparing their fitness, thereby giving more chances to migrate better weight-sets for next generation. These processes are continued until maximum iteration is reached or increase in fitness of weight-sets in $\mathbf{X}$ is not significant.

### 7.1 Parameter setup

The following parameters have been set during the experiment of TLBO–FLANN and ETLBO–FLANN.

| FLANN parameters | TLBO parameters |
|---|---|
| For FLANN, the learning parameter '$\mu$' is set to 0.13 in gradient descent learning by testing the models in the range 0–3<br>For functional expansion in FLANN, the value of n is set to 5, thereby each value in the input pattern is expanded to 11 number of functionally expanded input values<br>The number of functionally input value increases hugely if larger value of n is selected and the small value of n is unable to handle nonlinear nature of real-world datasets | For TLBO, we have set the common algorithmic parameters by testing the model by considering the suggested values (population size = 40; number of generations = 100; stopping criteria = maximum number of generation) |

### 7.2 Experimental results

In this section, the performance of FLANN classifier in contrast to the proposed ETLBO–JPSNN has been examined in order to know the improvement in weight-sets in the population as well as the classification accuracy obtained by these algorithms in various iterations. Here, all the previously used datasets (Sect. 6) have been tested by using TLBO–FLANN and ETLBO–FLANN. Moreover, we have considered four more high-dimensional datasets (Ionosphere, Coil2000, Spectf Heart and Spambase) to prove the effectiveness of the proposed method over the other neural network classifiers. Table 7 describes the comparison of average classification accuracies of all the considered datasets. From the table, it is clear that the proposed ETLBO–JPSNN not only performs better for all the first described 11 data sets, but also it has better classification accuracies in case of next four

**Table 7** Performance comparison between ETLBO–JPSNN with FLANN and other classifiers

| Dataset | Average classification accuracy (%) | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | ETLBO–JPSNN | | TLBO–JPSNN | | ETLBO–FLANN | | TLBO–FLANN | |
| | Train | Test | Train | Test | Train | Test | Train | Test |
| Heart | **96.679** | **96.368** | 95.763 | 95.438 | 92.534 | 86.34 | 89.826 | 79.891 |
| Hepatitis | **93.324** | **93.638** | 93.408 | 93.762 | 86.538 | 81.292 | 82.577 | 76.29 |
| Pima | **97.389** | **97.437** | 96.008 | 96.037 | 86.294 | 84.33 | 81.0 | 80.794 |
| Ecoli | **96.003** | **96.186** | 95.076 | 95.029 | 94.294 | 91.373 | 92.212 | 84.236 |
| Vehicle | **98.968** | **98.824** | 96.999 | 96.462 | 94.841 | 91.529 | 94.075 | 90.342 |
| Balance | **98.906** | **98.867** | 98.914 | 98.598 | 94.522 | 90.843 | 92.362 | 88.613 |
| Hayesroth | **98.067** | **98.637** | 96.333 | 96.295 | 93.547 | 89.643 | 91.825 | 85.523 |
| New Thyroid | **99.114** | **98.903** | 97.863 | 97.632 | 95.329 | 86.541 | 94.413 | 79.26 |
| Wine | **98.753** | **98.999** | 96.780 | 96.465 | 97.99 | 95.848 | 97.915 | 95.622 |
| Dermatology | **98.579** | **98.769** | 98.506 | 98.362 | 97.661 | 95.24 | 97.138 | 94.55 |
| Parkinson | **98.653** | **98.529** | 97.008 | 96.982 | 93.576 | 92.290 | 93.488 | 92.244 |
| Ionosphere | 94.856 | 93.582 | 93.491 | 93.992 | 92.861 | 90.247 | 91.483 | 90.738 |
| Coil2000 | 86.217 | 85.694 | 84.621 | 81.379 | 81.073 | 79.644 | 78.462 | 78.646 |
| SpectF heart | 84.348 | 83.988 | 81.457 | 80.431 | 78.647 | 78.254 | 76.249 | 75.892 |
| Spambase | 89.653 | 86.246 | 85.549 | 85.177 | 83.461 | 81.279 | 82.593 | 80.438 |

The bold faced results are the obtained results of the proposed method

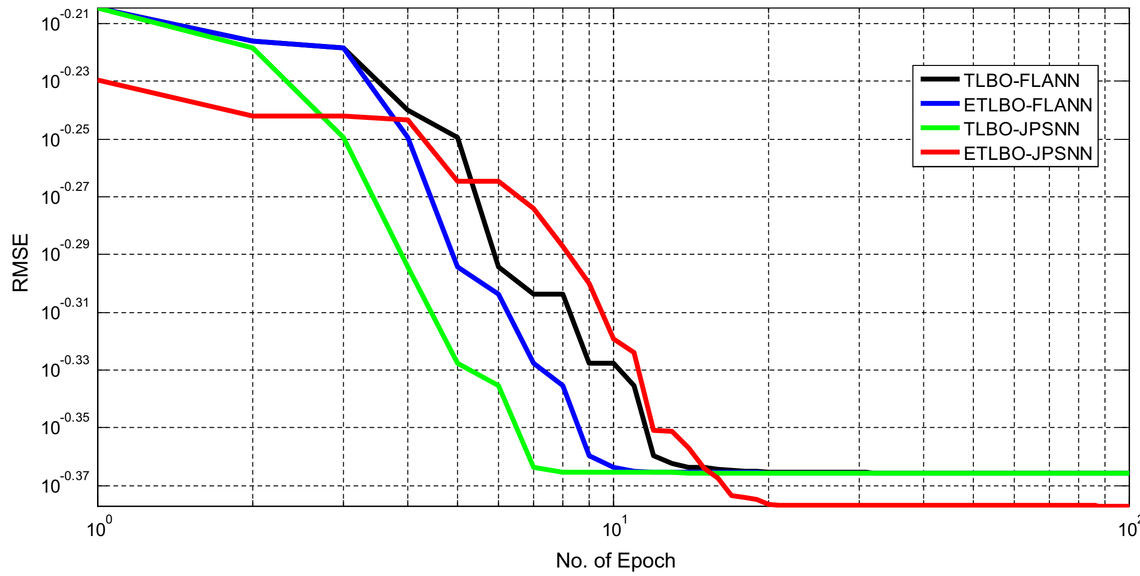



**Fig. 9** Performance comparison of ETLBO–JPSNN with other HONN on Ionosphere dataset

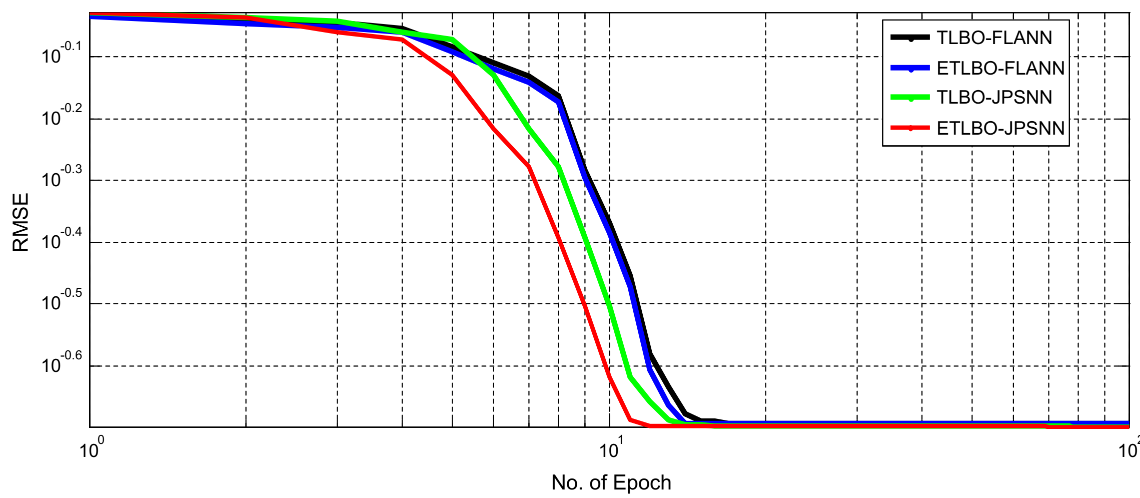



**Fig. 10** Performance comparison of ETLBO–JPSNN with other HONN on Coil2000 dataset

high-dimensional data sets. As compared to TLBO–FLANN and ETLBO–FLANN, the proposed method seems to produce good results in all the considered datasets. The changes in the RMSE values in different iterations observed in these high-dimensional datasets are illustrated in Figs. 9, 10, 11 and 12. The figures clearly demonstrate better results of the proposed method over different FLANN classifiers. Moreover, the average classification accuracies shown in Table 7 have been compared with the work presented in [95]. The authors in [95] have considered five data sets and experimented with the FLANN model which is optimized with differential evolution algorithm. They found the classification accuracy for Wine, Heart and Pima as 93.10, 86.57 and 79.20, respectively. For these three datasets, the classification accuracies of the proposed ETLBO–JPSNN are quite better as compared to them.

## 8 Statistical analysis and performance measures

Statistical analysis tools are used to investigate the comparison among improved performance of a proposed approach over any existing methods and help to analyze the nature of data. The performance in the improvement of the proposed algorithm (from existing if any) or a completely new algorithm should be statistically significant either in terms of classification accuracy, error measures or any other criteria in classification problems. Various statistical tests and their analytical measures along with different

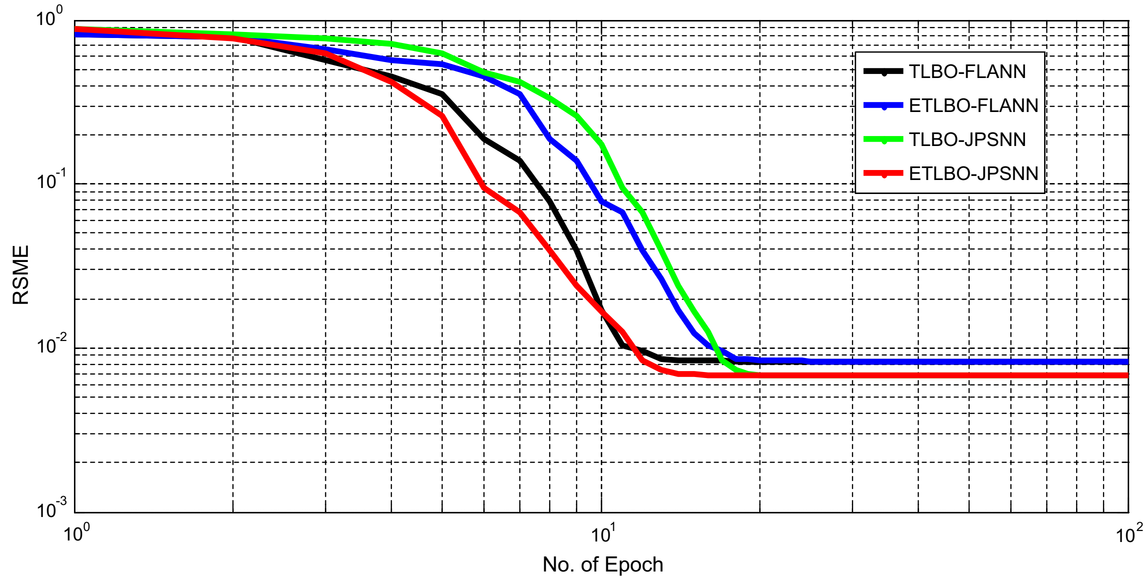



**Fig. 11** Performance comparison of ETLBO–JPSNN with other HONN on SpectF Heart dataset

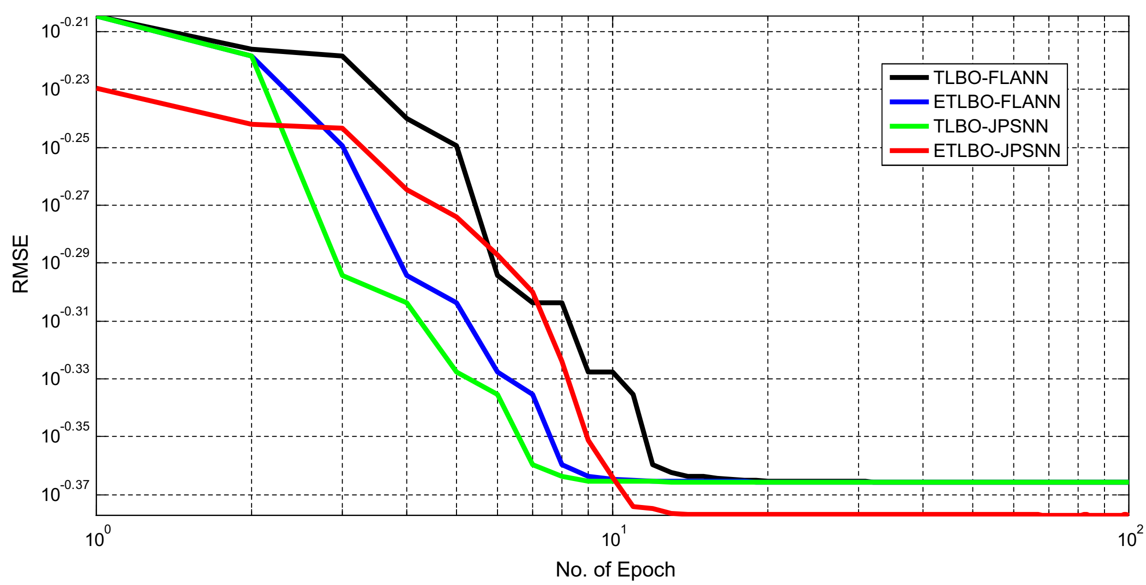



**Fig. 12** Performance comparison of ETLBO–JPSNN with other HONN on Spambase dataset

experimental validations have been reviewed by Damsar [96]. We have considered the statistical tests such as ANOVA, Friedman test, Tukey test, Dunnett test and post hoc test to measure the statistical correctness of the proposed algorithm with the other existing algorithms.

### 8.1 ANOVA

The main objective of one-way ANOVA (Fisher [97]) is to test the null hypothesis and to estimate the variability in the performance of the models. The sum variability is divided into the variability among the classifiers, variability between the datasets and the residual (error) variability by ANOVA [98]. We can reject the null hypothesis and get some difference among the classifiers, based on some marginal better variability of the between classifier compared with error variability. The test has been carried out using one-way ANOVA in Duncan's multiple test range with 95% confidence interval, 0.05 significant level and linear polynomial contrast, and the result is indicated in Fig. 13, and the result of Tukey and Duncan tests is shown in Fig. 14.

**Descriptives**

Sample

| | N | Mean | Std. Deviation | Std. Error | 95% Confidence Interval for Mean Lower Bound | 95% Confidence Interval for Mean Upper Bound | Minimum | Maximum |
|---|---|---|---|---|---|---|---|---|
| ETLBO-JPSNN | 11 | 97.7045 | 1.72915 | .52136 | 96.5429 | 98.8662 | 93.48 | 99.00 |
| ETLBO-PSNN | 11 | 96.9736 | 1.62275 | .48928 | 95.8835 | 98.0638 | 93.73 | 98.76 |
| TLBO-JPSNN | 11 | 96.5291 | 1.49427 | .45054 | 95.5252 | 97.5330 | 93.58 | 98.75 |
| PSO-JPSNN | 11 | 93.2173 | 3.31133 | .99840 | 90.9927 | 95.4419 | 84.31 | 96.32 |
| GA-JPSNN | 11 | 91.3736 | 3.74796 | 1.13005 | 88.8557 | 93.8915 | 81.59 | 95.59 |
| Total | 55 | 95.1596 | 3.48521 | .46995 | 94.2175 | 96.1018 | 81.59 | 99.00 |

**ANOVA**

Sample

| | | | Sum of Squares | df | Mean Square | F | Sig. |
|---|---|---|---|---|---|---|---|
| Between Groups | (Combined) | | 327.241 | 4 | 81.810 | 12.445 | .000 |
| | Linear Term | Contrast | 296.512 | 1 | 296.512 | 45.106 | .000 |
| | | Deviation | 30.728 | 3 | 10.243 | 1.558 | .211 |
| Within Groups | | | 328.682 | 50 | 6.574 | | |
| Total | | | 655.923 | 54 | | | |

**Fig. 13** ANOVA results with 95% confidence interval

**Homogeneous**

**Sample**

| | Algorithm | N | Subset for alpha = 0.05: 1 | 2 |
|---|---|---|---|---|
| Tukey HSD[a] | GA-JPSNN | 11 | 91.3736 | |
| | PSO-JPSNN | 11 | 93.2173 | |
| | TLBO-JPSNN | 11 | | 96.5291 |
| | ETLBO-PSNN | 11 | | 96.9736 |
| | ETLBO-JPSNN | 11 | | 97.7045 |
| | Sig. | | .451 | .818 |
| Duncan[a] | GA-JPSNN | 11 | 91.3736 | |
| | PSO-JPSNN | 11 | 93.2173 | |
| | TLBO-JPSNN | 11 | | 96.5291 |
| | ETLBO-PSNN | 11 | | 96.9736 |
| | ETLBO-JPSNN | 11 | | 97.7045 |
| | Sig. | | .098 | .317 |

Means for groups in homogeneous subsets are displayed.

a. Uses Harmonic Mean Sample Size = 11.000.

**Fig. 14** Results of Tukey and Duncan test

### 8.2 Tukey and Dunnett test

Post hoc test is used to reject the null hypothesis in ANOVA. For the comparison of the performance of all classifiers with each other, Tukey's test (Tukey [99]) and for comparisons of all classifiers with the proposed classifier, the Dunnett test (Dunnett [100]) have been used. ETLBO–JPSNN acts like the control group and it is being compared with all other groups such as TLBO–JPSNN, PSO–JPSNN and GA–JPSNN. The results of post hoc (Tukey test and Dunnett test) tests are illustrated in Fig. 15. In Tukey test, by considering one group as the control group remaining are compared against that group and this process has continued for all the others. During the test, it is found that the mean difference between the classifier variability is larger than the error variability in all the considered cases which may lead to the rejection of null hypothesis. Hence, the resulting performance values of all the statistical tests show that the ETLBO–JPSNN performs better than the other models.

### 8.3 Friedman test

In this paper, to calculate the differences among multiple test classifiers, Friedman test (Milton Friedman [101, 102]) has been used. Certain ranks has been assigned to each of the classifier's values in each rows, such that the best performed algorithm will have the chance of getting highest rank followed by others and the measured dependent variable must be ordinal. For the similar cases, the average ranks may be calculated by using Eq. (18) in each of the columns.

$$R_j = {}^{1}\!/_{N} \sum_{i}^{1} r_i^j \qquad (18)$$

where $r_i^j$ is the rank of the $j$th classifiers and $N$ is the number of datasets. Table 8 shows the assigned ranks (shown in brackets) of each classifiers on different cross-validated datasets. The average accuracy values (train + test) in Table 6 of all the 11 datasets have been considered for ranking purpose in all the cases. Based on the assigned rank values, the average values $\{R_1 = 1.18, R_2 = 2, R_3 = 2.81, R_4 = 4.09, R_5 = 4.90\}$ have been calculated for all the five algorithms.

Let us consider the null hypothesis, 'H: All the classifiers are in same rank and hence they are equivalent,' all the algorithms are same and so that, the ranks will be equal. Based on the ranks $R_j$ of the classifiers, the Friedman statistics $X_F^2$ is computed by using Eq. (19).

**Post Hoc**

**Multiple Comparisons**

Dependent Variable:Sample

| | (I) Algorithm | (J) Algorithm | Mean Difference (I-J) | Std. Error | Sig. | 95% Confidence Interval Lower Bound | 95% Confidence Interval Upper Bound |
|---|---|---|---|---|---|---|---|
| Tukey HSD | ETLBO-JPSNN | ETLBO-PSNN | .73091 | 1.09326 | .962 | -2.3628 | 3.8246 |
| | | TLBO-JPSNN | 1.17545 | 1.09326 | .818 | -1.9182 | 4.2692 |
| | | PSO-JPSNN | 4.48727* | 1.09326 | .001 | 1.3936 | 7.5810 |
| | | GA-JPSNN | 6.33091* | 1.09326 | .000 | 3.2372 | 9.4246 |
| | ETLBO-PSNN | ETLBO-JPSNN | -.73091 | 1.09326 | .962 | -3.8246 | 2.3628 |
| | | TLBO-JPSNN | .44455 | 1.09326 | .994 | -2.6492 | 3.5382 |
| | | PSO-JPSNN | 3.75636* | 1.09326 | .010 | .6627 | 6.8501 |
| | | GA-JPSNN | 5.60000* | 1.09326 | .000 | 2.5063 | 8.6937 |
| | TLBO-JPSNN | ETLBO-JPSNN | -1.17545 | 1.09326 | .818 | -4.2692 | 1.9182 |
| | | ETLBO-PSNN | -.44455 | 1.09326 | .994 | -3.5382 | 2.6492 |
| | | PSO-JPSNN | 3.31182* | 1.09326 | .030 | .2181 | 6.4055 |
| | | GA-JPSNN | 5.15545* | 1.09326 | .000 | 2.0618 | 8.2492 |
| | PSO-JPSNN | ETLBO-JPSNN | -4.48727* | 1.09326 | .001 | -7.5810 | -1.3936 |
| | | ETLBO-PSNN | -3.75636* | 1.09326 | .010 | -6.8501 | -.6627 |
| | | TLBO-JPSNN | -3.31182* | 1.09326 | .030 | -6.4055 | -.2181 |
| | | GA-JPSNN | 1.84364 | 1.09326 | .451 | -1.2501 | 4.9373 |
| | GA-JPSNN | ETLBO-JPSNN | -6.33091* | 1.09326 | .000 | -9.4246 | -3.2372 |
| | | ETLBO-PSNN | -5.60000* | 1.09326 | .000 | -8.6937 | -2.5063 |
| | | TLBO-JPSNN | -5.15545* | 1.09326 | .000 | -8.2492 | -2.0618 |
| | | PSO-JPSNN | -1.84364 | 1.09326 | .451 | -4.9373 | 1.2501 |
| Dunnett t (2-sided) a | ETLBO-JPSNN | GA-JPSNN | 6.33091* | 1.09326 | .000 | 3.5736 | 9.0882 |
| | ETLBO-PSNN | GA-JPSNN | 5.60000* | 1.09326 | .000 | 2.8427 | 8.3573 |
| | TLBO-JPSNN | GA-JPSNN | 5.15545* | 1.09326 | .000 | 2.3982 | 7.9128 |
| | PSO-JPSNN | GA-JPSNN | 1.84364 | 1.09326 | .279 | -.9137 | 4.6009 |

*. The mean difference is significant at the 0.05 level.

a. Dunnett t-tests treat one group as a control, and compare all other groups against it.

**Fig. 15** Results of post hoc test and Dunnett test

**Table 8** Assigned Friedman's rank to all the classifiers

| Dataset | Average classification accuracy [train + test] (%) | | | | |
|---|---|---|---|---|---|
| | ETLBO–JPSNN | ETLBO–PSNN | TLBO–JPSNN | PSO–JPSNN | GA–JPSNN |
| Heart | **96.52 (1)** | 95.93 **(2)** | 95.60 **(3)** | 92.32 **(4)** | 90.44 **(5)** |
| Hepatitis | **93.48 (3)** | 93.73 **(1)** | 93.58 **(2)** | 84.31 **(4)** | 81.59 **(5)** |
| Pima | **97.41 (1)** | 96.38 **(2)** | 96.02 **(3)** | 92.89 **(4)** | 90.85 **(5)** |
| Ecoli | **96.09 (1)** | 95.12 **(2)** | 95.05 **(3)** | 92.77 **(4)** | 90.70 **(5)** |
| Vehicle | **98.89 (1)** | 98.75 **(2)** | 96.73 **(3)** | 93.20 **(4)** | 90.83 **(5)** |
| Balance | **98.88 (1)** | 98.76 **(2)** | 98.75 **(3)** | 96.32 **(4)** | 94.74 **(5)** |
| Hayesroth | **98.35 (1)** | 97.37 **(2)** | 96.31 **(3)** | 92.14 **(4)** | 90.51 **(5)** |
| New Thyroid | **99.00 (1)** | 98.14 **(2)** | 97.74 **(3)** | 95.20 **(4)** | 94.31 **(5)** |
| Wine | **98.87 (1)** | 96.44 **(3)** | 96.62 **(2)** | 95.35 **(4)** | 93.25 **(5)** |
| Dermatology | **98.67 (1)** | 98.54 **(2)** | 98.43 **(3)** | 95.40 **(5)** | 95.59 **(4)** |
| Parkinson | **98.59 (1)** | 97.55 **(2)** | 96.99 **(3)** | 95.49 **(4)** | 92.30 **(5)** |
| Average | **1.18** | **2** | **2.81** | **4.09** | **4.90** |

The bold faced results are the obtained results of the proposed method

$$X_F^2 = {}^{12N}\!/_{m(m+1)} \left[ \sum_j^1 R_j^2 - \frac{m(m+1)^2}{4} \right] \tag{19}$$

where $X_F^2$ is the Friedman statistics and is distributed with $(m-1)$ degree of freedom. The values of $N$ and $m$ are considered as integer values. Iman and Davenport [103]

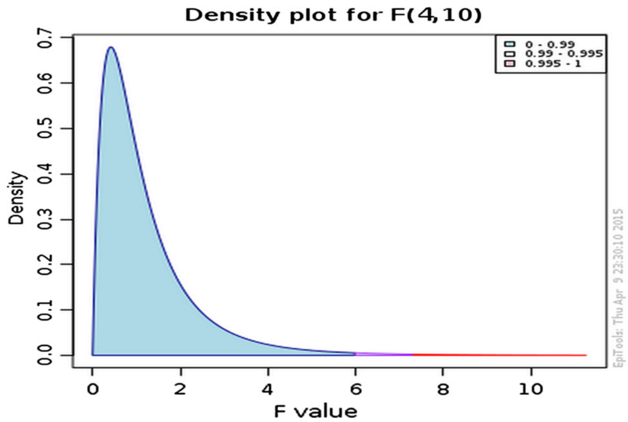



**Fig. 16** Density plot

distributed the statistics with $(m-1)$, $(m-1)$ $(N-1)$ degree of freedom as per $F$-distribution and developed a better performed Friedman statistics shown in Eq. (20).

$$F_{\mathrm{F}} = {}^{(N-1)X_{\mathrm{F}}^2}\!/_{N(m-1)-X_{\mathrm{F}}^2} \tag{20}$$

The value of $N$ (no. of datasets) is 11 and $X_{\mathrm{F}}^2$ is 36.68 under the $(m-1)$ degree of freedom. As per F-distribution, the $F_{\mathrm{F}}$ is computed as 50.10 by placing the values of $X_{\mathrm{F}}^2$, $N$, $m$ in Eq. (18). The $F_{\mathrm{F}}$ value is calculated with the $(m-1)$, $(m-1)$ $(N-1)$ degree of freedom, i.e., $(5-1)$, $(5-1)$ $(11-1)$ degree of freedom, and the crucial value can be obtained as 5.99 by appropriately selecting the value of $\alpha$ as 0.01. As per the above calculations, the critical value is less than the $F_{\mathrm{F}}$ statics, so, the null hypothesis is rejected. Hence, we can proceed for the post hoc analytical test. The density plot with the $F$ value and critical value is shown in Fig. 16.

After rejection of the null hypothesis, the post hoc test has been carried out by using the Holm procedure (Garcia et al. [104]; Luengo et al. [105]) to compute the performance of each of the classifiers against the other classifiers depending on the $z$ value and $p$ value. The $z$ value is calculated by using Eq. (21), and accordingly, the $p$ value is computed from the normal distribution table.

$$Z = \frac{(R_i - R_j)}{\sqrt{{}^{m(m+1)}\!/_{6N}}} \tag{21}$$

where $z$ indicates the $z$ score value. $R_i$ and $R_j$ are the average rank of $i$th and $j$th classifier, respectively. The number of classifiers is $m$, and $N$ is the number of datasets, respectively. The classifiers ETLBO–PSNN, TLBO–JPSNN, PSO–JPSNN and GA–JPSNN are compared with ETLBO–JPSNN based on $z$ value, $p$ value and $\alpha/(m-i)$, where '$i$' is the classifier's number as described in Table 9.

By using the Holm test, when we compare the value of $\mathrm{p_i}$ with $\alpha/(m-i)$, it is observed that, the null hypothesis can be rejected as $p_i$ is less than $\alpha/(m-i)$ in all the three cases. Hence, it is proved that all the null hypotheses are rejected. On the other side, Hochberg procedure works with the same procedure, but the largest $p$ value will be compared with $\alpha$ and the next $p$ value with $\alpha/2$ and so on. In this case also, the null hypothesis is rejected. Hence, the proposed classifier 'ETLBO–JPSNN' is statistically significant and performs quite well on cross-validated datasets and outperforms the other explained classifiers.

## 9 Conclusion and future directions

All the optimization techniques have their own objective function strategy for efficiently optimizing the functions or variables. Basically, they depend on tuning or adjustments of the algorithmic parameters, and accordingly, their performance can be measured. Teaching–learning-based optimization is quite new and effective for solving real-world optimization problems. The key feature of this algorithm is that, it need not depend on any strict controlling parameters. In this paper, an efficient elitist TLBO with higher-order Jordan Pi-sigma neural network has been proposed for solving the classification problems in data mining. In the beginning, first the algorithm has been applied to only Pi-sigma neural network along with its earlier version of TLBO algorithm. In the next phase, both TLBO and ETLBO have been applied with JPSNN to show the classification efficiency with a less error rate than the other models. As indicated in the result table, the proposed method is able to classify the nonlinear data with a better classification accuracy as compared to TLBO, PSO and GA. However, the results of both TLBO–JPSNN and

**Table 9** Result of Holm and Hochberg procedure

| $i$ | Classifiers | $z$ values | $p$ values | $\alpha/(m-i)$ |
|---|---|---|---|---|
| 1 | **ETLBO–JPSNN: GA–JPSNN** | 5.55 | 1.432362e−8 | 0.0025 |
| 2 | **ETLBO–JPSNN: PSO–JPSNN** | 4.34 | 0.000007 | 0.0033 |
| 3 | **ETLBO–JPSNN: TLBO–JPSNN** | 2.43 | 0.007549 | 0.005 |
| 4 | **ETLBO–JPSNN: ETLBO–PSNN** | 1.22 | 0.111232 | 0.01 |

The bold faced results are the obtained results of the proposed method

ETLBO–JPSNN are quite closer. But in maximum cases (especially in the datasets having large population size like Pima, Vehicle, Balance), we have achieved better results with ETLBO. After a rigorous experimental analysis and statistical analysis, it is found that the proposed ETLBO–JPSNN model is steady, effective, valid and quite promising for future research in other application domains.

In the near future, the performance of ETLBO will be tested in some other applications of data mining such as clustering and prediction. However, a deep focus will be made on other improved version of TLBO, such as OL-TLBO (opposition learning-based TLBO) in various data mining applications.

**Acknowledgements** This work is supported by Department of Science and Technology (DST), Ministry of Science and Technology, New Delhi, Government of India, under Grants No. DST/INSPIRE Fellowship/2013/585. The author would like to thank to the editor and the reviewers for their valuable comments and suggestions that helped to improve the content of the paper in a large extent.

## References

1. Holland JH (1992) Genetic algorithms. Scientific American, New York, pp 66–72
2. Goldberg DE (1989) Genetic algorithms in search, optimization, and machine learning. Addison-Wesley, Reading
3. Kennedy J, Eberhart R (1995) Particle swarm optimization. In: Proceedings of the 1995 IEEE international conference on neural networks vol 4, pp 1942–1948
4. Alatas B, Akın E (2005) FCACO: fuzzy classification rules mining algorithm with ant colony optimization. In: ICNC 2005. Lecture notes in computer science, vol 3612. Springer, Berlin, pp 787–797
5. Dorigo M, Maziezzo V, Colorni A (1996) The ant system: optimization by a colony of cooperating ants. IEEE Trans Syst Man Cybern B 26(1):29–41
6. Alatas B (2010) Chaotic bee colony algorithms for global numerical optimization. Expert Syst Appl 37(8):5682–5687
7. Karaboga D, Basturk B (2007) A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. J Global Optim 39(3):459–471
8. Li XL (2003) New intelligent optimization-artificial fish swarm algorithm. PhD thesis, Zhejiang University, China
9. Geem ZW, Kim JH, Loganathan GV (2001) A new heuristic optimization algorithm: harmony search. Simulation 76:60–70
10. Yin M, Hu Y, Yang F, Li X, Gu W (2011) A novel hybrid K-harmonic means and gravitational search algorithm approach for clustering. Expert Syst Appl 38(8):9319–9324
11. Rashedi E, Nezamabadi-pour H, Saryazdi S (2009) GSA: a gravitational search algorithm. Inf Sci 179(13):2232–2248
12. Yang XS (2009) Firefly algorithms for multimodal optimization. In: Stochastic algorithms: foundations and applications SAGA 2009. Lecture notes in computer sciences, vol 5792, pp 169–178
13. Krishnanand KN, Ghose D (2006) Glowworm swarm based optimization algorithm for multimodal functions with collective robotics applications. Multiagent Grid Syst 2(3):209–222
14. Wu B, Qian C, Ni W, Fan S (2012) The improvement of glowworm swarm for continuous optimization problems. Expert Syst Appl 39(7):6335–6342
15. Jamili A, Shafia MA, Tavakkoli-Moghaddam R (2011) A hybridization of simulated annealing and electro magnetism-like mechanism for a periodic job shop scheduling problem. Expert Syst Appl 38(5):5895–5901
16. Birbil SI, Fang SC (2003) An electromagnetism-like mechanism for global optimization. J Global Optim 25:263–282
17. Xie L, Zeng J, Cui Z (2009) General framework of artificial physics optimization algorithm. In: IEEE nature & biologically inspired computing, pp 1321–1326
18. Alatas B (2011) Uniform big bang-chaotic big crunch optimization. Commun Nonlinear Sci Numer Simul 16(9):3696–3703
19. Erol OK, Eksin I (2006) A new optimization method: big bang-big crunch. Adv Eng Softw 37:106–111
20. Kaveh A, Laknejadi K (2011) A novel hybrid charge system search and particle swarm optimization method for multi-objective optimization. Expert Syst Appl 38(12):15475–15488
21. Kaveh A, Talatahari S (2010) A novel heuristic optimization method: charged system search. Acta Mech 213(3–4):267–289
22. Sacco WF, de Oliveira CRE (2005) A new stochastic optimization algorithm based on a particle collision metaheuristic. In: 6th world congresses of structural and multidisciplinary optimization
23. Formato RA (2007) Central force optimization: a new metaheuristic with applications in applied electromagnetics. Prog Electromagn Res PIER 77:425–491
24. Green RC, Wang L, Alam M (2012) Training neural networks using central force optimization and particle swarm optimization: insights and comparisons. Expert Syst Appl 39(1):555–563
25. Rao RV, Savsani VJ, Vakharia DP (2011) Teaching–learning-based optimization: a novel method for constrained mechanical design optimization problems. Comput Aided Des 43:303–315
26. Rao RV, Savsani VJ, Vakharia DP (2012) Teaching–learning-based optimization: an optimization method for continuous non-linear large scale problems. Inf Sci 183:1–15
27. Rao RV, Vivek P (2013) An improved teaching–learning-based optimization algorithm for solving unconstrained optimization problems. Scientia Iranica D 20(3):710–720
28. Venkata Rao R, Patel V (2012) An elitist teaching–learning-based optimization algorithm for solving complex constrained optimization problems. Int J Ind Eng Comput 3:535–560
29. Rao RV, Waghmare GG (2013) A comparative study of a teaching–learning-based optimization algorithm on multi-objective unconstrained and constrained functions. J King Saud Univ Comput Inf Sci. doi:10.1016/j.jksuci.2013.12.004
30. Rao RV, More KC (2015) Optimal design of the heat pipe using TLBO (teaching–learning-based optimization) algorithm. Energy 80:535–544
31. Ergun U, Murat K, Adem A, Tayfun D (2014) Estimates of energy consumption in Turkey using neural networks with the teaching–learning-based optimization algorithm. Energy 75:295–303
32. Wang L, Zou F, Hei X, Yang D, Chen D, Jiang Q (2014) An improved teaching–learning-based optimization with neighborhood search for applications of ANN. Neurocomputing 143:231–247
33. Basu M (2014) Teaching–learning-based optimization algorithm for multi-area economic dispatch. Energy 68:21–28
34. Yang Z, Li K, Foley A, Zhang C (2014) A new self-learning TLBO algorithm for RBF neural modelling of batteries in electric vehicles. In: IEEE congress on evolutionary computation (CEC), pp 2685–2691. doi:10.1109/CEC.2014.6900428
35. Medina MA, Coello Coello CA, Ramirez JM (2013) Reactive power handling by a multi-objective teaching learning optimizer based on decomposition. IEEE Trans Power Syst 28(4):3629–3637. doi:10.1109/TPWRS.2013.2272196

36. Nayak MR, Nayak CK, Rout PK (2012) Application of multi-objective teaching learning based optimization algorithm to optimal power flow problem. Proc Technol 6:255–264
37. Toğan V (2012) Design of planar steel frames using teaching-learning based optimization. Eng Struct 34:225–232
38. Niknam T, Golestaneh F, Sadeghi MS (2012) $\theta$-multiobjective teaching–learning-based optimization for dynamic economic emission dispatch. IEEE Syst J 6(2):341–352. doi:10.1109/JSYST.2012.2183276
39. Jadhav HT, Chawla D, Roy R (2012) Modified teaching–learning based algorithm for economic load dispatch incorporating wind power. In: 11th international conference on environment and electrical engineering (EEEIC), pp 397–402. doi:10.1109/EEEIC.2012.6221410
40. Satapathy SC, Naik A, Parvathi K (2012) Teaching learning based optimization for neural networks learning enhancement. Lect Notes Comput Sci 7677:761–769
41. Zou F, Wang L, Hei X, Chen D, Wang B (2013) Multi-objective optimization using teaching–learning-based optimization algorithm. Eng Appl Artif Intell 26:1291–1300
42. Kumar RP, Aditi S, Kumar PD (2013) Optimal short-term hydro-thermal scheduling using quasi-oppositional teaching learning based optimization. Eng Appl Artif Intell 26:2516–2524
43. Mandal B, Roy PK (2013) Optimal reactive power dispatch using quasi-oppositional teaching learning based optimization. Electr Power Energy Syst 53:123–134
44. García JAM, Mena AJG (2013) Optimal distributed generation location and size using a modified teaching–learning based optimization algorithm. Electr Power Energy Syst 50:65–75
45. Venkata RR, Kalyankar VD (2013) Parameter optimization of modern machining processes using teaching–learning-based optimization algorithm. Eng Appl Artif Intell 26:524–531
46. Roy PK (2013) Teaching learning based optimization for short-term hydrothermal scheduling problem considering valve point effect and prohibited discharge constraint. Electr Power Energy Syst 53:10–19
47. Roy PK, Bhui S (2013) Multi-objective quasi-oppositional teaching learning based optimization for economic emission load dispatch problem. Electr Power Energy Syst 53(2013):937–948
48. Singh M, Panigrahi BK, Abhyankar AR (2013) Optimal coordination of directional over-current relays using teaching–learning-based optimization (TLBO) algorithm. Electr Power Energy Syst 50:33–41
49. Wang K-L, Wang H-B, Yu L-X, Ma X-Y, Xue Y-S (2013) Toward teaching-learning-based optimization algorithm for dealing with real-parameter optimization problems. In: Proceedings of the 2nd international conference on computer science and electronics engineering (ICCSEE 2013), pp 0606–0609
50. Satapathy SC, Naik A, Parvathi K (2013) Weighted teaching–learning-based optimization for global function optimization. Appl Math 4:429–439
51. Satapathy SC, Naik A, Parvathi K (2013) A teaching learning based optimization based on orthogonal design for solving global optimization problems. Springer Plus 2:130
52. Tuo S, Yong L, Zhou T (2013) An improved harmony search based on teaching–learning strategy for unconstrained optimization problems. Math Problems Eng. doi:10.1155/2013/413565
53. Kai X, Gao L, Wang L, Li W, Chao K-M (2013) A simplified teaching–learning-based optimization algorithm for disassembly sequence planning. In: IEEE 10th international conference on e-business engineering (ICEBE), pp 393–398. doi:10.1109/ICEBE.2013.60
54. Savsani P, Jhala RL, Savsani VJ (2013) Optimized trajectory planning of a robotic arm using teaching learning based optimization (TLBO) and artificial bee colony (ABC) optimization techniques In: IEEE international systems conference (SysCon), pp 381–386. doi:10.1109/SysCon.2013.6549910
55. Gao W-J, Xing B, Marwala T (2013) Teaching–learning-based optimization approach for enhancing remanufacturability pre-evaluation system's reliability. In: IEEE symposium on swarm intelligence (SIS), pp 235–239. doi:10.1109/SIS.2013.6615184
56. Gonzalez-Alvarez DL, Vega-Rodriguez MA, Gomez-Pulido JA, Sanchez-Perez JM (2012) Multiobjective teaching–learning-based optimization (MO-TLBO) for motif finding. In: IEEE 13th international symposium on computational intelligence and informatics (CINTI), pp 141–146. doi:10.1109/CINTI.2012.6496749
57. Theja BS, Rajasekhar A, Abraham A (2013) An optimal design of coordinated PI based PSS with TCSC controller using modified teaching learning based optimization. In: World Congress on nature and biologically inspired computing (NaBIC), pp 99–106. doi:10.1109/NaBIC.2013.6617845
58. Sultana S, Roy PK (2014) Optimal capacitor placement in radial distribution systems using teaching learning based optimization. Electr Power Energy Syst 54:387–398
59. Rasoul A-A, Taher N, Farhad B, Mohsen Z (2014) Short-term scheduling of thermal power systems using hybrid gradient based modified teaching–learning optimizer with black hole algorithm. Electr Power Syst Res 108:16–34
60. Arya LD, Koshti A (2014) Anticipatory load shedding for line overload alleviation using Teaching learning based optimization (TLBO). Electr Power Energy Syst 63:862–877
61. Reza KM, Hassan KM (2014) A novel self-tuning control method based on regulated bi-objective emotional learning controller's structure with TLBO algorithm to control DVR compensator. Appl Soft Comput 24:912–922
62. Niu Q, Zhang H, Li K (2014) An improved TLBO with elite strategy for parameters identification of PEM fuel cell and solar cell models. Int J Hydrogen Energy 39(2014):3837–3854
63. Moghadam A, Seifi AR (2014) Fuzzy-TLBO optimal reactive power control variables planning for energy loss minimization. Energy Convers Manag 77:208–215
64. Gonzalez-Alvarez DL, Vega-Rodriguez MA, Rubio-Largo A (2014) Finding patterns in protein sequences by using a hybrid multiobjective teaching learning based optimization algorithm. Issue: 99. doi:10.1109/TCBB.2014.2369043
65. Yammani C, Sowjanya G, Maheswarapu S, Matam SK (2014) Optimal placement and sizing of DER's with load models using a modified teaching learning based optimization algorithm. In: International conference on green computing communication and electrical engineering (ICGCCEE). doi:10.1109/ICGCCEE.2014.6922306
66. Cheng Y-H (2014) Estimation of teaching–learning-based optimization primer design using regression analysis for different melting temperature calculations. IEEE Trans Nano Biosci. doi:10.1109/TNB.2014.2352351
67. Sahoo S, Murty SB, Krishna KM (2015) Character recognition using teaching learning based optimization. Adv Intell Syst Comput 327:737–744
68. Agrawal S, Sharma S, Silakari S (2014) Teaching learning based optimization (TLBO) based improved iris recognition system. Adv Intell Syst Comput 330:735–740
69. Barisal AK (2015) Comparative performance analysis of teaching learning based optimization for automatic load frequency control of multi-source power systems. Electr Power Energy Syst 66:67–77
70. Mojtaba G, Mahdi T, Sahand G, Jamshid A, Abbas A (2015) Solving optimal reactive power dispatch problem using a novel

teaching–learning-based optimization algorithm. Eng Appl Artif Intell 39:100–108
71. Debao C, Feng Z, Zheng L, Jiangtao W, Suwen L (2015) An improved teaching–learning-based optimization algorithm for solving global optimization problem. Inf Sci 297:171–190
72. Kumar SB, Swagat P, Kumar MP, Sidhartha P (2015) Teaching–learning based optimization algorithm based fuzzy-PID controller for automatic generation control of multi-area power system. Appl Soft Comput 27:240–249
73. Mojtaba G, Sahand G, Mohsen G, Ebrahim A (2015) An improved teaching–learning-based optimization algorithm using Lévy mutation strategy for non-smooth optimal power flow. Electr Power Energy Syst 65:375–384
74. Chakravarthy VVSSS, Naveen Babu K, Suresh S, Chaya Devi P, Mallikarjuna RP (2015) Linear array optimization using teaching learning based optimization. Adv Intell Syst Comput 338:183–190
75. Kumar MP, Chandra SS (2015) An hybrid approach for data clustering using K-means and teaching learning based optimization. Adv Intell Syst Comput 338:165–171
76. Rajasekhar A, Rani R, Ramya K, Abraham A (2012) Elitist teaching learning opposition based algorithm for global optimization. In: IEEE international conference on systems, man, and cybernetics (SMC), pp 1124–1129
77. Shin Y, Ghosh J (1991) The Pi-sigma networks : an efficient higher order neural network for pattern classification and function approximation. In: Proceedings of international joint conference on neural networks, Seattle, Washington, vol 1, pp 13–18
78. Nayak J, Naik B, Behera HS (2015) A novel chemical reaction optimization based higher order neural network (CRO-HONN) for nonlinear classification. Ain Shams Eng J. doi:10.1016/j.asej.2014.12.013
79. Nayak J, Naik B, Behera HS (2014) A hybrid PSO–GA based Pi sigma neural network (PSNN) with standard back propagation gradient descent learning for classification. In: IEEE international conference on control, instrumentation, communication and computational technologies (ICCICCT), 2014, pp 878–885. doi:10.1109/ICCICCT.2014.6993082
80. Ghosh J, Shin Y (1992) Efficient higher-order neural networks for classification and function approximation. Int J Neural Syst 3:323–350
81. Shin Y, Ghosh J (1991) Realization of boolean functions using binary pi-sigma networks. In: Dagli CH, Kumara SRT, Shin YC (eds) Intelligent engineering systems through artificial neural networks. ASME Press, New York, pp 205–210
82. Jordan MI (1986) Attractor dynamics and parallelism in a connectionist sequential machine. In: Proceedings of the eighth conference of the cognitive science society, New Jersey, USA
83. Ghazali R, Husaini NA, Ismail LH, Samsuddin NA (2012) An application of Jordan Pi-sigma neural network for the prediction of temperature time series signal. Recurr Neural Netw Soft Comput 13:275–290
84. Nayak J, Kanungo DP, Naik B, Behera HS (2014) A higher order evolutionary Jordan Pi-sigma neural network with gradient descent learning for classification. In: IEEE international conference on high performance computing and applications (ICHPCA), pp 1–6. doi:10.1109/ICHPCA.2014.7045328
85. Rumelhart DE, Hinton GE, Williams RJ (1986) Learning representations by back-propagating errors. Nature 323(9):533–536
86. Bache K, Lichman M (2013) UCI machine learning repository [http://archive.ics.uci.edu/ml]. University of California, School of Information and Computer Science, Irvine
87. Alcala-Fdez J, Fernandez A, Luengo J, Derrac J, García S, Sánchez L, Herrera F (2011) keel data-mining software tool: data set repository. Integration of algorithms and experimental analysis framework. J Mult Valued Log Soft Comput 17(2–3):255–287
88. Larson S (1931) The shrinkage of the coefficient of multiple correlation. J Educ Psychol 22:45–55
89. Mosteller F, Turkey JW (1968) Data analysis, including statistics. In: Handbook of Social Psychology, vol 2, pp 80–203
90. Pao YH (1989) Adaptive pattern recognition and neural networks. Addison-Wesley, Reading
91. Pao YH, Takefuji Y (1992) Functional-link net computing: theory, system architecture, and functionalities. Computer 25:76–79
92. Naik B, Nayak J, Behera HS (2015) An efficient FLANN model with CRO-based gradient descent learning for classification. Int J Bus Inf Syst 21(1):73–116
93. Naik B, Nayak J, Behera HS (2015) A global-best harmony search based gradient descent learning FLANN (GbHS-GDL-FLANN) for data classification. Egypt Inf J (in Press)
94. Naik B, Nayak J, Behera HS, Abraham A (2015) A self adaptive harmony search based functional link higher order ANN for non-linear data classification. Neurocomputing. doi:10.1016/j.neucom.2015.11.051
95. Dash CSK et al. (2015) Towards crafting an improved functional link artificial neural network based on differential evolution and feature selection. Informatica 39(2):195–208
96. Demšar J (2006) Statistical comparisons of classifiers over multiple data sets. J Mach Learn Res 7:1–30
97. Fisher RA (1959) Statistical methods and scientific inference, 2nd edn. Hafner Publishing Co., New York
98. Nayak J, Naik B, Behera HS, Abraham A (2015) Particle swarm optimization based higher order neural network for classification. Smart Innov Syst Technol 31:401–414
99. Tukey JW (1949) Comparing individual means in the analysis of variance. Biometrics 5:99–114
100. Dunnett CW (1980) A multiple comparison procedure for comparing several treatments with a control. J Am Stat Assoc 50:1096–1121
101. Friedman MA (1937) The use of ranks to avoid the assumption of normality implicit in the analysis of variance. J Am Stat Assoc 32:675–701
102. Friedman MA (1940) Comparison of alternative tests of significance for the problem of m rankings. Ann Math Stat 11:86–92
103. Iman RL, Davenport JM (1980) Approximations of the critical region of the Friedman statistic. Commun Stat 9:571–595
104. Garcia S, Fernandez A, Luengo J, Herrera F (2010) Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: experimental analysis of power. Inf Sci 180:2044–2064
105. Luengo J, Garcia S, Herrera F (2009) A study on the use of statistical tests for experimentation with neural networks: analysis of parametric test conditions and non-parametric tests. Expert Syst Appl 36:7798–7808