International Journal of Automation and Computing
17(1), February 2020, 108-122
DOI: 10.1007/s11633-018-1158-3

Accurate Classification of EEG Signals Using Neural Networks Trained by Hybrid Population-physic-based Algorithm

Sajjad Afrakhteh Mohammad-Reza Mosavi Mohammad Khishe Ahmad Ayatollahi

Department of Electrical Engineering, Iran University of Science and Technology, Tehran 16846-13114, Iran

Abstract: A brain-computer interface (BCI) system is one of the most effective ways that translates brain signals into output commands. Different imagery activities can be classified based on the changes in μ and β rhythms and their spatial distributions. Multi-layer perceptron neural networks (MLP-NNs) are commonly used for classification. Training such MLP-NNs has great importance in a way that has attracted many researchers to this field recently. Conventional methods for training NNs, such as gradient descent and recursive methods, have some disadvantages including low accuracy, slow convergence speed and trapping in local minimums. In this paper, in order to overcome these issues, the MLP-NN trained by a hybrid population-physics-based algorithm, the combination of particle swarm optimization and gravitational search algorithm (PSOGSA), is proposed for our classification problem. To show the advantages of using PSOGSA that trains NNs, this algorithm is compared with other meta-heuristic algorithms such as particle swarm optimization (PSO), gravitational search algorithm (GSA) and new versions of PSO. The metrics that are discussed in this paper are the speed of convergence and classification accuracy metrics. The results show that the proposed algorithm in most subjects of encephalography (EEG) dataset has very better or acceptable performance compared to others.

Keywords: Brain-computer interface (BCI), classification, electroencephalography (EEG), gravitational search algorithm (GSA), multi-layer perceptron neural network (MLP-NN), particle swarm optimization.

1 Introduction

The main goal of a brain computer interface (BCI) system is providing a channel for controlling an external device using the electrical activity of the brain[1]. These BCIs can be operated in various applications, such as controlling a wheelchair or neuroprosthesis for disabled individuals, navigation in a virtual environment, and assisting healthy individuals in performing highly demanding tasks or controlling devices such as quadcopters[2,3]. People suffering from amyotrophic lateral sclerosis (ALS) lose their muscle movement degeneratively and at a later stage may become totally paralyzed. Nevertheless, for most ALS patients, their minds are unaffected. BCI addresses this concern by making it possible to translate human thoughts directly to the outside world[4]. Electroencephalography (EEG) has been chosen to capture brainwaves for BCI applications because of its simplicity, cheapness and high temporal resolution[5]. Motor imagery is a mental process of a motor action. It includes preparation for movement, passive observation of action and mental operation of motor representation implicitly or explicitly[6]. The EEG data that is used in this paper is BCI competition IV[7], that is available on its website[1].

Feature extraction is very important to EEG processing in BCI system. In the feature extraction phase, by focusing on dimension reduction in traditional BCI motor imagery, some works such as principle component analysis (PCA) and independent component analysis (ICA) have been done.

The assumption of PCA is that the samples are independent and its main idea is to compute a group of new features arranged in descending order according to their importance from the original ones that are linear combinations[8]. The ICA is similar to the PCA method, except that all components are designed independently[9]. All of these methods have been widely used in visualization and feature extraction, for they are well understood and easy to implement. In 2000, two nonlinear dimensionality reduction methods, isomap and locally linear embedding (LLE) algorithms, were proposed by Tenenbaum et al.[10] Wu et al.[11] proposed EEG feature extraction based on wavelet packet decomposition (WPD). The mean of the coefficients of WPD and wavelet packet energy of special sub-bands were used as the original features. Ramoser et al.[12] proposed common spatial pattern (CSP) filtering for feature extraction that has been proven to be an effect-

Research Article
Manuscript received March 20, 2018; accepted September 3, 2018; published online November 6, 2018
Recommended by Associate Editor Hong Qiao

[1] http://www.bbci.de/competition/iv/desc_1.html.

ive feature extraction algorithm for the binary motor imagery task classification.

The classification phase in BCI system is very important in that its rate indicates one of the metrics for BCI performance. In this literature, we review some of the studies that researchers have performed. Pfurtscheller et al.[13] proposed adaptive auto-regressive (AAR) algorithm for classification of EEG signals. Lemm et al.[14] proposed the probabilistic modeling of sensorimotor μ rhythms for hand movement imaginary classification. Zhou et al.[15] classified the mental task features using linear discriminant analysis (LDA) and support vector machine (SVM). Ma et al.[16] used particle swarm optimization (PSO) algorithms for optimizing the performance of the SVM classifier. Subasi and Erçelebi[17] used conventional neural networks (NNs) for EEG signal classification. This method has some drawbacks such as slow convergence and trapping in local minima.

Multi-layer perceptron neural networks (MLP-NNs) are one of the most widely used NNs in the classification of the BCI mental tasks. The learning process is one of the most important parts of the development of these types of networks, which has been widely considered in recent years. In order to train MLP networks, there are two types of algorithms: deterministic algorithms and stochastic algorithms. At first, the use of deterministic algorithms such as gradient descent was common. Inappropriate classification accuracy, trapping on local optimum and low convergence rates are the drawbacks of traditional methods. In order to overcome these disadvantages, in recent years, the use of heuristic and metaheuristic algorithms has become very common. Compared to full-scale search techniques, heuristic algorithms have accurate and efficient computations to solve multidimensional problems.

Various heuristic algorithms including genetic algorithm (GA)[18], PSO[19], ant colony optimization (ACO)[20], etc. were proposed in many areas. In the context of global optimization, an efficient and effective algorithm must have two important features: exploration and exploitation. Exploration is the ability of an algorithm to search in all feasible spaces for a problem, and exploitation is the convergence ability of an algorithm to the best solution near a good solution. But, there is a very important point, and it is a weakness of convergence that causes falling into local optimum. So, combining heuristic algorithms is an efficient way to solve this problem. The random generation of possible solutions improves dynamically by selecting good solutions based on the capability of exploration and exploitation operations of the evolutionary algorithm.

The main idea is to combine the ability of exploitation of the best solutions in PSO with the ability of exploration of the search space in GSA to enhance the performance of individual optimization algorithms. The implementation of the PSO algorithm is less complicated because the PSO algorithm has fewer parameters for adjustment. So, researchers have paid attention to PSO in many areas such as pattern detection, artificial intelligence and computer engineering. However, this algorithm has a disadvantage in that it converges to local optimum and low convergence rates. To solve this problem, several mechanisms that combined the PSO algorithm with other algorithms were presented, such as genetic algorithm and PSO (GAPSO)[21], PSO and ant colony optimization (PSOACO)[22], PSO-Broyden-Fletcher-Goldfarb-Shanno (BFGS)[23], etc. Mirjalili and Hashim[24] proposed the combination of PSO and gravitational search algorithm (GSA). The main idea of particle swarm optimization and gravitational search algorithm (PSOGSA) is to combine the ability to exploit the best solutions in PSO with the ability to explore the search space in GSA to improve performance than individual optimization algorithms. Sun and Peng[25] used PSOGSA for high dimensional optimization and microarray data clustering. Four artificial data sets and three microarray data sets were tested. The results confirmed that the proposed algorithm possesses better robustness. Aminzadeh and Miri[26] proposed high-speed hybrid PSOGSA algorithm for optimal placement of phasor measurement units.

This paper is trying to train an MLP-NN using PSOGSA algorithm for EEG signal classification. The accuracies and convergence speed of this algorithm is compared to GSA[27], PSO[19], improved PSO (IPSO)[28], modified PSO (MPSO)[29] and time-varying acceleration coefficients PSO (TACPSO)[30]. The results show that the proposed method is superior to others for training MLP-NNs to classify EEG signals. Also, the results of this method are compared to other common machine learning algorithms such as support vector machine (SVM)[31] and K-nearest neighborhood (K-NN)[32]. The results showed that the proposed method for classification has greater performance compared to other methods. The paper is organized as follows: Section 1 is the introduction. Section 2 presents an overview of the BCI system in this problem and describes the pre-processing step in detail. Section 3 describes the dataset used in this paper. Section 4 presents the common spatial pattern (CSP) scheme for motor imagery feature extraction. In Section 5, the proposed method is described. The MLP-NN training process is presented in Section 6. Finally, discussion and results can be found in Section 7 and the result analysis and conclusion are prepared in Sections 8 and 9, respectively.

2 An overview of the BCI system in this problem

The main framework of the EEG-based BCI system is shown in Fig. 1 and includes four parts as follows[33]: input of the BCI system that includes brain measurements, pre-processing on the obtained signal in the previous step, a feature translation process that is decomposed to two

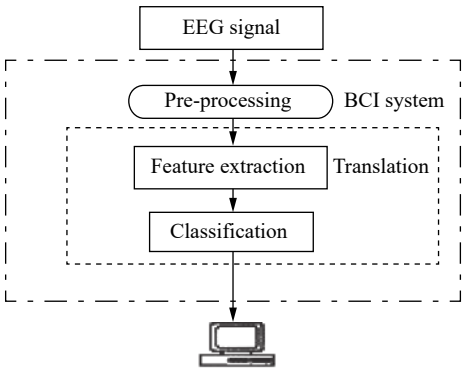


Fig. 1 Framework of BCI system[33]

parts of feature extraction and classification and output of the system that is classified signal for controlling the external device.

In the pre-processing stage, EEG signals of each channel are sampled at a rate of 100 Hz. Then, these signals are filtered by a band-pass finite impulse response (FIR) filter with zero phase and passband range of 8 Hz to 30 Hz[12]. This frequency band has been selected because firstly, it includes frequency bands of μ (8–13 Hz) and β (14–30 Hz). Secondly, because the frequency of artifacts caused by eye movements and muscle movements is outside of this range, these artifacts and 50 Hz noise power will be removed by this filter[34]. The type of filter that is used in this paper is a third order Butterworth one. The reason for using this type of filter is the smoothness of its response compared to other common filters such as Chebyshev and Bessel. It has the flattest response and slope of 20 dB/dec in each pole.

The EEG signal before and after applying the Butterworth filter is shown in Fig. 2. Other views of the EEG signal that are used in this paper are shown in Figs. 3 and 4 for frequency and time domain that extracted from Matlab software. Because the signal to noise power ratio of the EEG data is very low, detecting events in EEG signals is very difficult. In fact, due to the skull bone tissue, the spatial resolution obtained from the overhead is low. The signal obtained from each EEG channel is the sum of the many signals that originate from under the skull. Thus, detection of useful channels and extraction of important events for separating classes of motor imagery in these channels is not easy work. One of the most successful and widely used methods for feature reduction and feature extraction for motor imagery-based BCI applications is the CSP method. This method was introduced in 1991 by Koles in a two-class problem to diagnose patients from healthy subjects[35]. In this method, signals collected from the surface of the skull is decomposed into two matrices.

One of them indicates the main sources of EEG signals and the other contains weights that show the impact of each brain source. The columns of the weight matrix are called spatial filters. Then, the covariance matrix for each class of motor imagery is obtained using the resulting spatial filters. In fact, each spatial filter is associated with an eigenvector of the covariance matrix of the two-class data. In the CSP method, each temporal sample of EEG channels is considered as a vector and mapped to the new space by the spatial filters. Thus, the useful features are extracted. In the next step, the features should apply to a classifier. Our novelty in this paper is in the classification phase. In this stage, the MLP-NNs trained by meta-heuristic algorithms such as PSO, GSA, and PSOGSA are used for classification. MLP-NNs are one of the most powerful tools for soft computing. With these networks, the non-linear problems can be

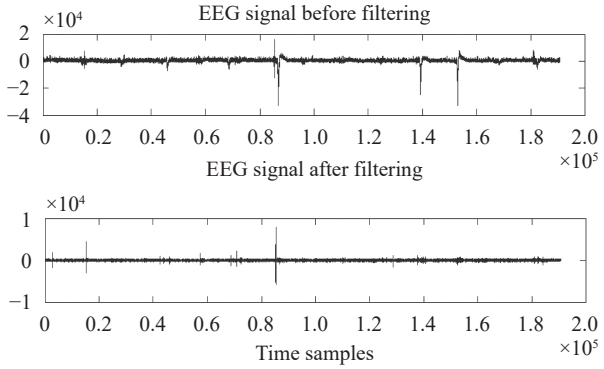


Fig. 2 EEG signals before and after applying the Butterworth filter

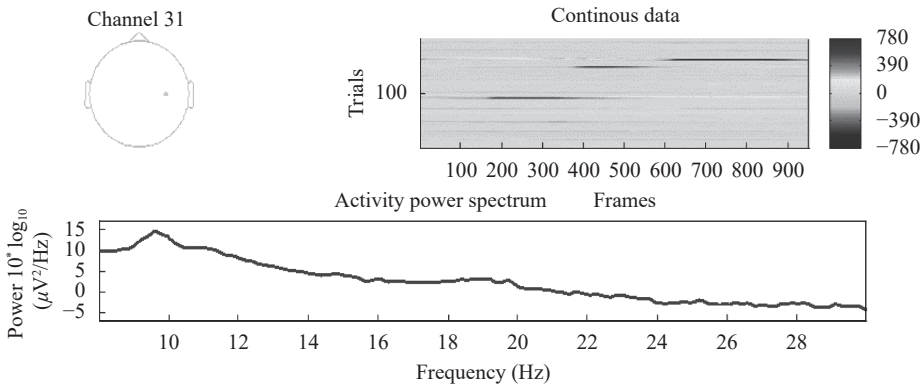


Fig. 3 Power of EEG signal versus frequency for channel 31 in the important frequency range (8–30 Hz)

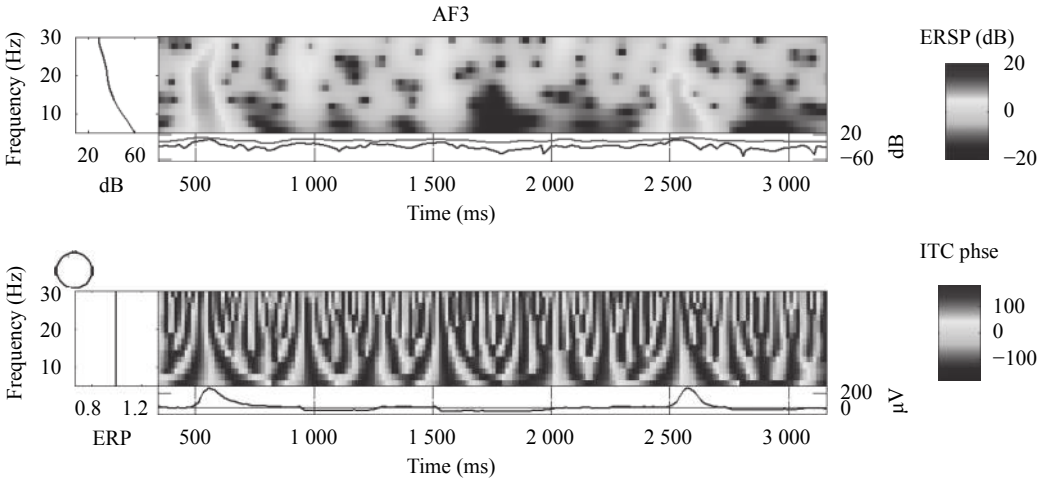


Fig. 4 Time-frequency maps of EEG signal of electrode "AF3" extracted with Matlab software

solved. In general, the MLP-NNs could be used for pattern classification, prediction and approximating the functions[36–42].

3 Dataset

In this paper, the dataset 1 and the calibration data part of BCI competition IV is used. The recording was made using brain Amp magnetic resonance (MR) plus amplifiers and an Ag/AgCl electrode cap. The signals from the 59 EEG positions that were most densely distributed over sensorimotor areas were measured. This dataset has been recorded from 7 subjects. The location of the EEG electrodes is shown in Fig. 5 and is plotted in Matlab software. The dataset belongs to the Berlin BCI group: Berlin Institute of Technology (Machine Learning Laboratory) and Fraunhofer FIRST (Intelligent Data Analysis Group) and Campus Benjamin Franklin of the Charit University Medicine Berlin, Department of Neurology, Neuro-Physics Group. The signals measured belong

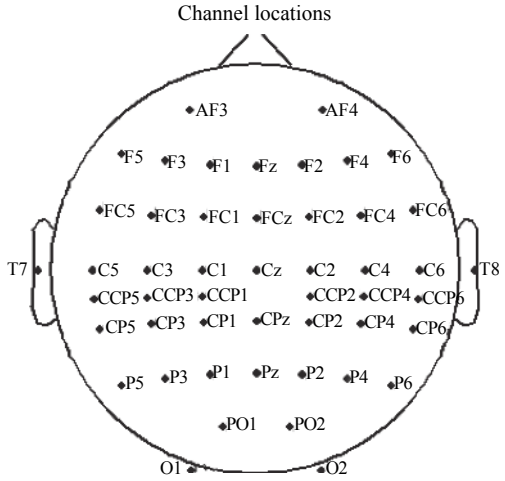


Fig. 5 Placement of EEG electrode for recording data that was extracted with Matlab

to seven healthy subjects named ds1a, ds1b, ds1c, ds1d, ds1e, ds1f and ds1g. The dataset consisted of signals of 200 trials for each subject measured using 59 EEG channel positions of the extended international 10/20 electrode montage system. The subject was provided with a visual cue shown for 4 s for the duration of each trial. Two classes of moto imagery were selected from the three classes "left hand", "right hand" and "foot" for each subject[7].

4 Common spatial pattern method

Before classification, the data that will be the input of the classifier is extracted. In this paper, the CSP method is used for the feature extraction process. So, in this section, this method is introduced and described in detail. The purpose of the CSP method is to obtain filters that are caused the maximum difference in term of variance between two classes of this data by applying them to the EEG signal. The proper spatial filter would provide signals so that they are easy to classify. This method is called the CSP algorithm[43]. We denote CSP filter by

$$S = w^{\mathrm{T}} D \quad or \quad s(t) = W^{\mathrm{T}} d(t) \tag{1}$$

where $W \in \mathbf{R}^{d\times ch}$ is the spatial filter matrix, $S \in \mathbf{R}^{d\times time}$ is the filtered signal matrix. The criterion of CSP is given by maximizing $W^{\mathrm{T}}\Sigma_1 W$ subject to

$$W^{\mathrm{T}}(\Sigma_1 + \Sigma_2)W = I \tag{2}$$

where

$$\Sigma_1 = \frac{D_n D_n^{\mathrm{T}}}{\mathrm{tra}\{D_n D_n^{\mathrm{T}}\}}, \quad D_n \in \{\mathrm{class1}\} \tag{3}$$

$$\Sigma_2 = \frac{D_n D_n^{\mathrm{T}}}{\mathrm{tra}\{D_n D_n^{\mathrm{T}}\}}, \quad D_n \in \{\mathrm{class2}\}. \tag{4}$$

This problem can be solved by a generalized eigenvalue problem. However, it can also be obtained by two times solving the eigenvalue problem. First, we decom-

pose as

$$\Sigma_1 + \Sigma_2 = UEU^{\mathrm{T}} \tag{5}$$

where U is a set of eigenvectors, and E is a diagonal matrix of eigenvalues. Next, we compute $P = E^{-\frac{1}{2}}U^{\mathrm{T}}$and

$$\hat{\Sigma}_1 = P\,\Sigma_1 P^{\mathrm{T}} \tag{6}$$

$$\hat{\Sigma}_2 = P\Sigma_2 P^{\mathrm{T}}. \tag{7}$$

It should be noted that $\hat{\Sigma}_1 + \hat{\Sigma}_2 = I$. Thus, any orthogonal matrices V satisfy $V^{\mathrm{T}}(\hat{\Sigma}_1 + \hat{\Sigma}_2)\,V = I$. Finally, it is decomposed as

$$\hat{\Sigma}_1 = V\Lambda V^{\mathrm{T}} \tag{8}$$

where V is a set of eigenvectors and Λ is a diagonal matrix of eigenvalues. A set of CSP filters is obtained as

$$W = P^{\mathrm{T}}V. \tag{9}$$

So, we have

$$W^{\mathrm{T}}\Sigma_1 W = \Lambda = \begin{bmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_{ch} \end{bmatrix} \tag{10}$$

$$W^{\mathrm{T}}\Sigma_2 W = I - \Lambda = \begin{bmatrix} 1-\lambda_1 & & \\ & \ddots & \\ & & 1-\lambda_{ch} \end{bmatrix} \tag{11}$$

where $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_{ch}$. Therefore, the first CSP filter w_1 provides maximum variance of class 1, and the last CSP filter w_{ch} provides maximum variance of class 2. We select first and last m filters to use as

$$W_{csp} = (w_1 \ \cdots \ w_m \ \ w_{ch-m+1} \ \cdots \ w_{ch}) \in \mathbf{R}^{2m\times ch}. \tag{12}$$

And the filtered signal matrix is given by

$$s(t) = W_{csp}\,d(t) = (s_1(t) \ \cdots \ s_{2m}(t))^{\mathrm{T}}. \tag{13}$$

So, the feature vector $x = (x_1, x_2, \cdots, x_{2m})^{\mathrm{T}}$ is calculated by

$$x_i = \log\left(\frac{\mathrm{var}(s_i(t))}{\sum_{i=1}^{d}\mathrm{var}(s_i(t))}\right). \tag{14}$$

5 Proposed method

The meta-heuristic algorithms for training MLP-NNs are described as following.

5.1 Particle swarm optimization algorithm

PSO is a population-based stochastic optimization technique that was developed by Eberhart and Kennedy[19] in 1995, inspired by the social behavior of birds. This method uses the number of particles (candidate solutions) in the search space to find the best solution. All particles travel towards the best particles (best solution) that are on their way. PSO is initialized with a group of random particles (solutions) and then searches for optima by updating generations. In every iteration, each particle is updated by following two "best" values. The first one is the best solution (fitness) that has achieved so far (the fitness value is also stored). This value is called "pbest". Another "best" value that is tracked by the particle swarm optimizer is the best value in the population. This best value is a global best and called "gbest". After finding the two best values, the particle updates its velocity and positions with the following (15) and (16):

$$v_i^{t+1} = wv_i^t + c_1 \times rand() \times (pbest_i - \chi_i^t) + c_2 \times rand() \times (gbest - \chi_i^t) \tag{15}$$

$$\chi_i^{t+1} = \chi_i^t + v_i^{t+1} \tag{16}$$

where v_i^t is the particle velocity, χ_i^t is the current position of particle, pbest and gbest are defined as stated before, $rand()$ is a random number in (0, 1) and $c1$ and $c2$ are learning factors. In (15), wv_i^t provides the exploration ability. While the second part of this equation, $c_1 \times rand() \times (pbest_i - \chi_i^t) + c_2 \times rand() \times (gbest - \chi_i^t)$, represents private thinking and cooperation between the particles. The particle's velocities on each dimension are clamped to a maximum velocity $V_{\max}$. If the sum of accelerations would cause the velocity on that dimension to exceed $V_{\max}$, which is a parameter specified by the user, then the velocity on that dimension is limited to $V_{\max}$.

5.2 Gravitational search algorithm

In 2009, Rashedi et al.[27] offered a new heuristic algorithm called GSA. The purpose of this algorithm is finding the best solution in the search space using the physics laws. Newton's theory is the physical theory that the GSA is inspired by which says: "every particle in the universe absorbs another particle with a force that has a direct relationship to the product of their masses and inversely with the square of the distance between the two particles"[44]. GSA algorithm can be considered as a set of agents (candidate solutions) which have masses proportional to their values in the fitness function. During the execution of algorithms, all masses attract each other by the force of attraction between themselves. The higher the mass, the more gravity it will be. So, heavier masses that possibly are nearest masses to the global minimum, absorb other masses proportional to their distance. Ac-

cording to [45], a system is assumed with N factors. The position of each element (mass) that is a candidate solution, is defined as the following equations:

$$X_i = (\chi_i^1, \chi_i^2, \cdots, \chi_i^d, \cdots, \chi_i^n), \quad i = 1, 2, \cdots, N \tag{17}$$

where N and χ_i^d indicate the dimension of the problem and the position of the i-th factor in the d-th dimension, respectively. The algorithm starts its work by putting factors randomly in the search space. During each iteration, the gravitational force is defined as the following equation:

$$F_{ij}^d(t) = G(t) \times \frac{M_{pi}(t) \times M_{aj}(t)}{R_{ij}(t) + \varepsilon} \times (\chi_j^d(t) - \chi_i^d(t)) \tag{18}$$

where $M_{aj}(t)$ indicates the active gravitational mass due to factor j. $M_{Pi}(t)$ is the passive gravitational mass due to the i-th factor, $R_{ij}(t)$ indicates the Euclidean distance between i and j is an expression of two factors. The gravitational constant is calculated as

$$G(t) = G_0 \times \exp(-\alpha \times iter / \max iter) \tag{19}$$

where α is descending coefficient, G_0 is initial gravitational constant, $iter$ is current iteration and $\max iter$ indicates the maximum of iterations. In the d-dimensional problem, general force is calculated by

$$F_i^d(t) = \sum_{j=1, j\neq i}^{N} rand_j \times F_{ij}^d(t). \tag{20}$$

In (20), $rand_j$ indicates a random number between 0 and 1. According to the laws of motion, acceleration of a factor is directly related to resultant force and inversely to its mass. So, the acceleration, velocity, and position of all factors are calculated by

$$a_i^d(t) = \frac{F_i^d(t)}{M_{ii}(t)} \tag{21}$$

$$v_i^d(t+1) = rand_i \times v_i^d(t) + a_i^d(t) \tag{22}$$

$$\chi_i^d(t+1) = \chi_i^d(t) + v_i^d(t+1). \tag{23}$$

The mass of all the factors will be updated using the following equation:

$$m_i(t) = \frac{fit_i(t) - worst(t)}{best(t) - worst(t)} \tag{24}$$

where, in the minimization problem, we have

$$best(t) = \min_{j\in\{1,2,\cdots,N\}} fit_j(t) \tag{25}$$

$$worst(t) = \max_{j\in\{1,2,\cdots,N\}} fit_j(t). \tag{26}$$

The normalization of calculated mass in (24) is done by

$$M_i(t) = \frac{m_i(t)}{\sum_{j=1}^{N} m_j(t)}. \tag{27}$$

The flowchart of this algorithm is shown in Fig. 6. The mass of factors is defined using the evaluation of the fitness function. This means that the factor with the heaviest mass is the most effective factor. From (22) and (23), it can be deduced that the current velocity is defined as a fraction of the last velocity that is added to its acceleration. In addition, the current position is the last position that is added to its current velocity. In all population-based algorithms with social behavior, such as PSO and GSA, two intrinsic characteristics must be considered, the ability of the algorithm to search all parts of the search space and its ability to exploit the best solution. Searching in the entire problem space is called exploration, while converging on the best solution that is near to a good solution, is called exploitation. Population-based algorithms must have these critical features so that they guarantee to find the best solutions. In PSO, the ability of exploration and exploitation is performed using pbest and gbest, respectively. In the GSA algorithm, exploration can be guaranteed by selecting the appropriate value for random parameters G_0 and α[46]. Rashedi conducted a study comparing the GSA algorithm with some known heuristic optimization algorithm such as PSO[27]. The comparison results show that GSA has the ability in optimization problems. However, it has a low search speed in the last iteration[47]. In this paper, to improve this weakness, a combination of GSA and PSO algorithms is suggested, and called PSOGSA.

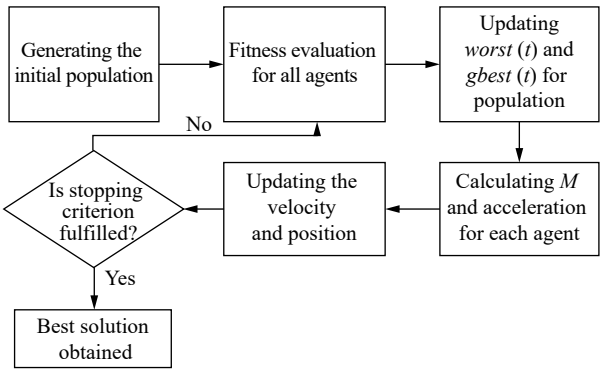


Fig. 6 Flowchart of GSA algorithm

5.3 Combining particle swarm optimization and gravitational search algorithm

The basic idea of PSOGSA is that it combines social thinking ability in the PSO algorithm with local search capabilities in GSA. In order to combine these algori-

thms, the following equation is proposed:

$$V_i(t+1) = w \times V_i(t) + c_1 \times rand \times ac_i(t) + c_2 \times rand \times (gbset - X_i(t)) \quad (28)$$

where $ac_i(t)$ indicates the acceleration and $gbset$ is the best solution that is found so far. Positions in each iteration will be updated as follows:

$$X_i(t+1) = X_i(t) + V_i(t+1). \quad (29)$$

In the PSOGSA algorithm, each agent is assumed as a candidate solution. At first, all agents are initialized randomly. After initializing, the force of gravity, the gravitational constant and the resultant force are calculated using (18)–(20). Then, the acceleration is calculated by (21). The best solution that is found in each iteration, must be updated. After calculating the acceleration and updating the best solution, the velocity of agents can be found using (28). To explain the efficiency of PSOGSA, the following points are mentioned:

1) In the PSOGSA algorithm, the solution quality is considered in the updating method.

2) The agents that are near to a good solution, absorb other agents exploring different parts of the search space.

3) When all of the agents are near to a good solution, they move very slowly. In this case, $gbset$ helps them to exploit the global best solution.

4) In PSOGSA, each agent can see the best solution and desire it.

5) By adjusting c_1 and c_2, global search and local search capabilities can be balanced.

The above-mentioned points make the PSOGSA algorithm powerful for solving a wide range of optimization problems. In the next section, the mechanisms of training MLP-NNs using PSO, GSA and PSOGSA algorithms is described.

6 Training process of multi-layer perceptron neural networks

The ultimate goal of the learning process in NNs is to find the best combination of weights and biases so that the learning of the network leads to the lowest error. Generally, there are three methods for training MLP-NNs using meta-heuristic algorithms. In the first method, the heuristic algorithms are used to find the reasonable combination of the weights and biases of the network, so that this combination provides the minimum error for MLP-NNs. In the second method, meta-heuristic algorithms are used as a way to find an appropriate structure for MLP-NNs in a particular problem. In the last method, evolutionary algorithms (EA) set the parameters of the gradient-based learning algorithm. These parameters are the learning rate and momentum. In this paper, in order to design motor imagery classifiers based on EEG signals, the PSO, GSA, improved PSO (IPSO), MPSO, TACPSO and PSOGSA algorithms are applied to MLP-NNs. The mentioned classifiers are called feed forward NNs trained by PSO (FNNPSO), feed forward NNs trained by GSA (FNNGSA), feed forward NNs trained by IPSO (FNNIPSO), feedforward NNs trained by MPSO (FNNMPSO), feed forward NNs trained by TACPSO (FNNTACPSO) and feed forward NNs trained by PSOGSA (FNNPSOGSA), respectively. The flow chart of training MLP-NNs using PSOGSA (proposed algorithm) is shown in Fig. 7. For designing this classifier, the fitness function of the proposed algorithms is defined in the next sections.

6.1 Fitness function

The three-layer MLP-NN is shown in Fig. 8. In this MLP-NN, we have an input layer, one hidden layer and an output layer. The number of nodes in the input layer is n, the number of hidden and output layer neurons are h and m, respectively. In each course of the training process, the output of each neuron in the hidden layer is obtained by

$$f(s_j) = \frac{1}{(1 + \exp(-s_j))}, \ j = 1, 2, \cdots, h. \quad (30)$$

In (30), we have

$$s_j = \sum_{i=1}^{n} w_{ij} \times \chi_i - \theta_j \quad (31)$$

where w_{ij} indicates connection weight between the i-th node in the input layer and the j-th neuron in the hidden layer, θ_j is the j-th neuron′s bias in the hidden layer and χ_i is the i-th node′s input in the input layer. After calculating the neuron′s output in the hidden layer, the final output is defined as

$$o_k = \sum_{i=1}^{h} w_{kj} \times f(s_j) - \theta_j, \ \ k = 1, 2, \cdots, m \quad (32)$$

where w_{kj} is the connection weight from the j-th neuron in the in the hidden layer to the k-th neuron in the output layer and θ_j is the j-th neuron′s bias in the hidden layer.

Finally, the training error or fitness value is calculated by the following equation:

$$error(k) = \sum_{i=1}^{m} (o_i(k) - d_i(k))^2 \quad (33)$$

$$error = \sum_{k=1}^{q} \frac{error(k)}{q} \quad (34)$$

where q is the number of training samples, $d_i(k)$ indicates the expected output of the i-th input unit when the k-th training sample is used and $o_i(k)$ is the desired output for

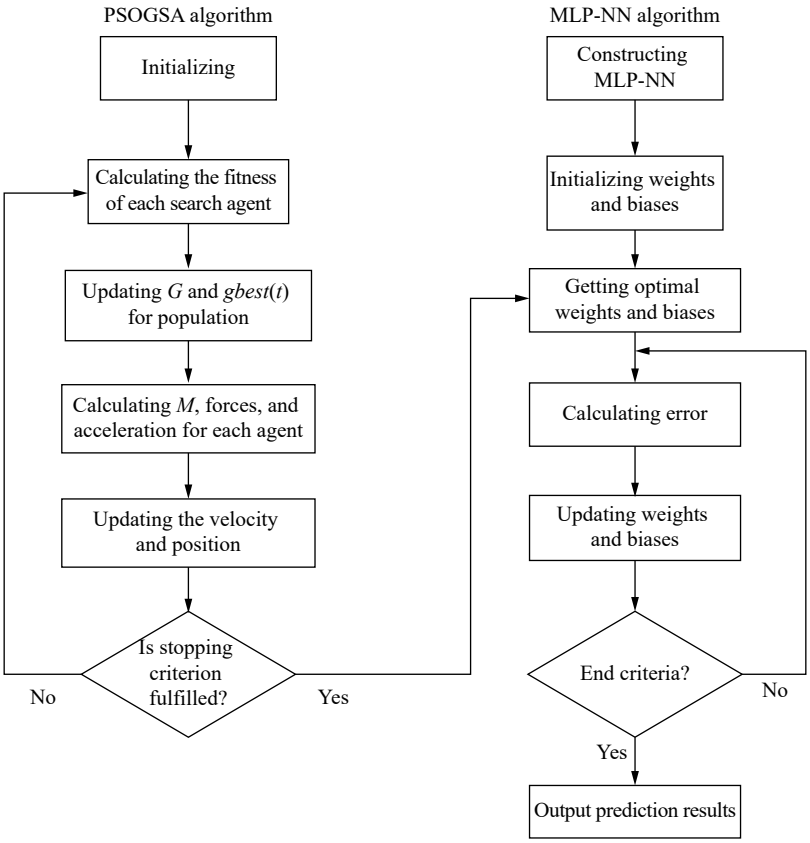


Fig. 7 Flowchart of MLP-NNs trained by PSOGSA algorithm

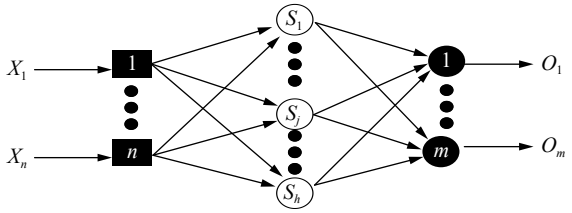


Fig. 8 Three-layer MLP-NN

the k-th training sample. So, the fitness function is defined by

$$FIT(X_i) = error(X_i). \tag{35}$$

6.2 Feed-forward neural networks (FNN's) parameter setting

The input of the FNN is the same as the extracted properties of the feature extraction stage, where the number of extracted features after the CSP is 10. Since there is no established standard for choosing the number of hidden nodes for classification purposes and considering the structure of FNNs, the proposed method in [48] and the following equation is used:

$$H = 2N + 1 \tag{36}$$

where N and H represent the number of inputs and the number of hidden nodes, respectively.

Weights and thresholds of the FNN are also updated using the proposed algorithm.

The number of hidden layer neurons is 15. Also, given that we have two output classes, the number of neurons in the output layer is considered to be 2. Accordingly, because the FNN structure is fully connected, each of the neurons in the hidden layer and the output layer has a threshold, so the number of weights and thresholds are calculated as follows:

$$\#of\ weights = (\#of\ input\ nodes) \times (\#of\ hidden\ neurons) + (\#of\ hidden\ neurons) \times (\#of\ output\ neurons) \tag{37}$$

$$\#of\ Thresholds = (\#of\ hidden\ neurons) + (\#of\ output\ neurons)\,. \tag{38}$$

So, the number of weights and thresholds are (10) × (2×10+1) + (15) × (2) = 240 from (37) and (21) + (2) = 23 from (38), respectively. A vector of P containing these weights and thresholds is considered as a particle, which are actually considered to be optimization variables.

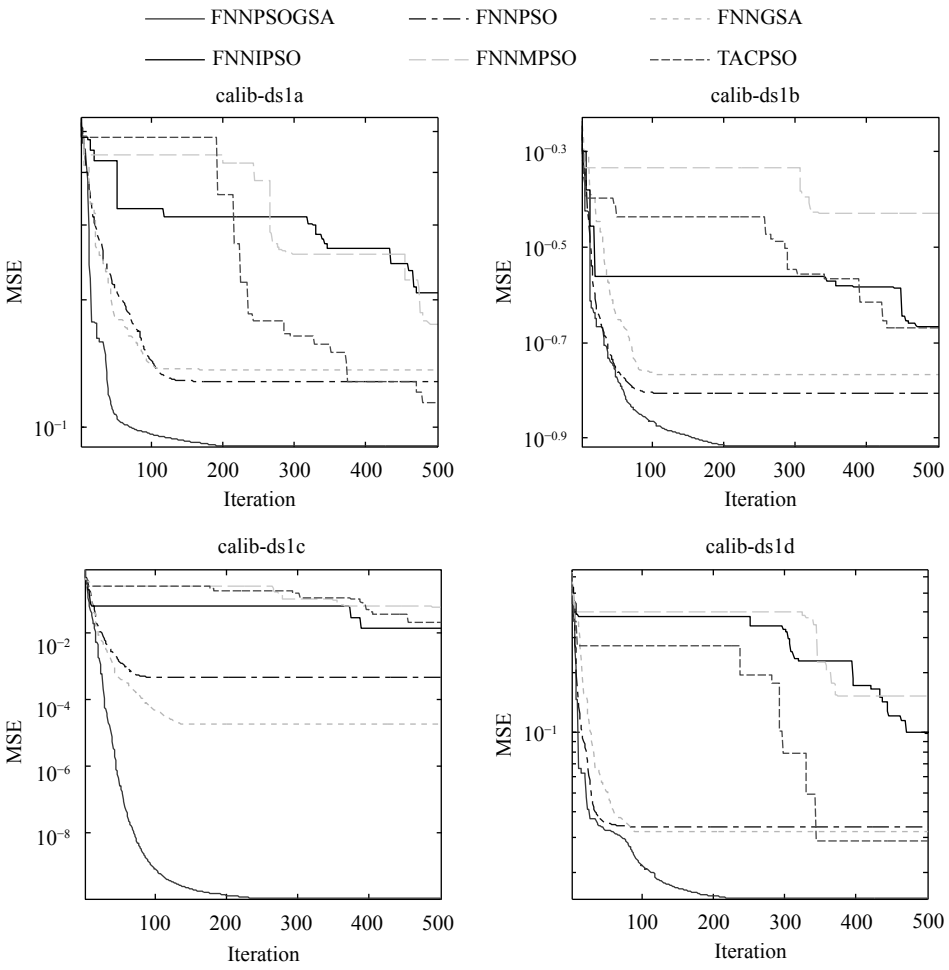
7 Results

In this paper, our main goal is the classification of two motor imagery (right and foot) based on the EEG signal

for the aforementioned dataset in Section 3. At first, in pre-processing, the EEG signal is filtered using a 3rd order band-pass filter. Then, in feature extraction, the CSP filter (with m=5) is used. According to what was said in describing the CSP method, the number of features is equal to $2m$ ($2m$=10). From all trials in EEG data, 70 percent of them are considered. In our data, the number of trials is 200.

So, the output of the feature extraction stage is a 140×10 matrix with 140 labels. Now, this matrix must be applied to the classifier. Our main idea in this paper was in the classifier stage. In this step, an artificial MLP-NN was proposed for classification. In other words, the MLP-NN was trained with the PSOGSA algorithm. The total number of particle is 140, the dimension is 30, $c1$ and $c2$ are 2.0, max and min inertia weight are 0.9 and 0.5, respectively.

The convergence curves for all classifiers are shown in Fig. 9. The convergence curves show the averages of 20 independent runs over the course of 500 iterations. Fig. 9 shows that FNNPSOGSA has the fastest convergence speed and best classification accuracy on the EEG dataset. These numerical results are shown in Table 1. These results show the superiority of the proposed algorithm to other algorithms in term of classification rate and convergence speed. As it can be seen, the average classification rates using FNNPSOGSA classifier for ds1a, ds1b, ds1c, ds1d, ds1e, ds1f and ds1g are 93.392 9%, 94.357 1%, 100%, 99.5%, 99.857 1%, 99.642 9% and 98%, respectively which is very suitable for BCI application. It can be observed that the FNNGSA classifier, doesn′t have good performance compared to FNNPSO. However, it is very strong due to its searching ability compared to other evolutionary algorithms. This disadvantage of GSA is due to a lower speed and lower classification accuracy. The proposed classifier, FNNPSOGSA, is better than others in term of classification accuracy and speed of convergence. To compare the performance of all algorithms, the results are collected over 20 independent runs. In order to see whether the results of PSOGSA differ from PSO, GSA, IPSO, MPSO, and TACPSO in a statistically significant way, a non-parametric statistical test, Wilcoxon's rank-sum test is used. The average (AVE), standard deviation (STD) and P-value of all algorithms are reported in Table 1 for all subjects. As it can be seen for ds1a, ds1b, ds1d, ds1e, ds1f and ds1g, the FNNPSOGSA is the best classifier in terms of classification accuracy and convergence speed. In ds1c, the classification rate of all classifi-



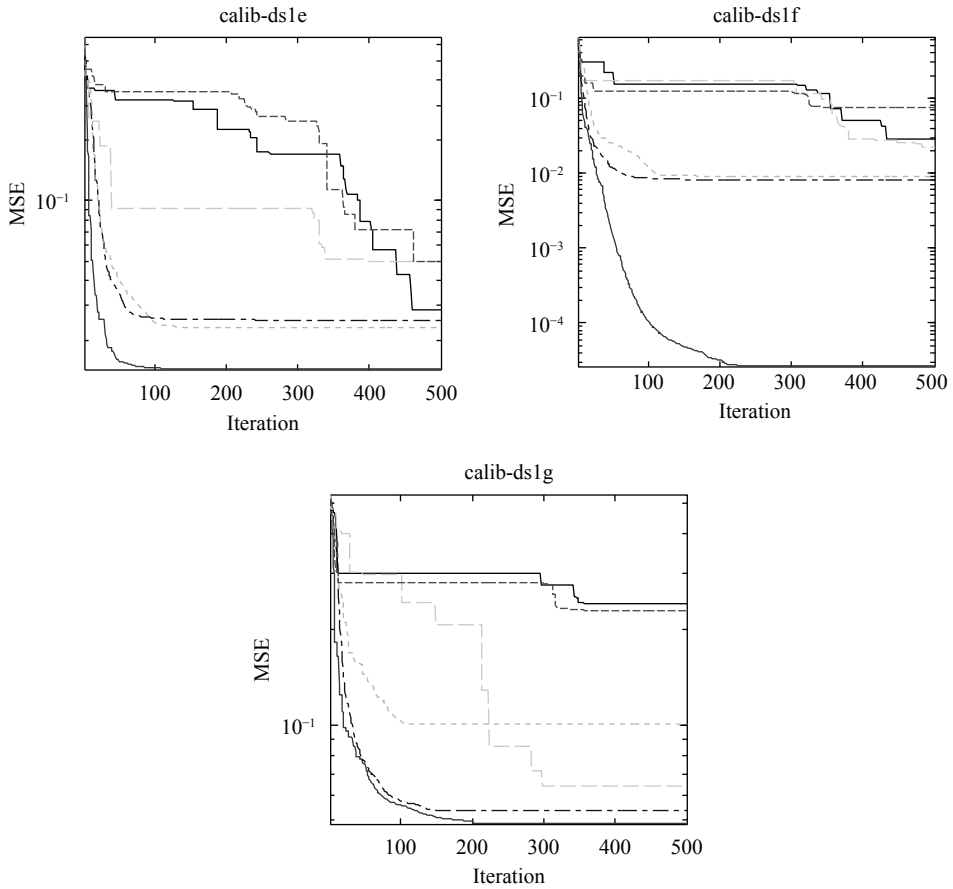


Fig. 9 Comparison of MSE for the proposed algorithm to others when applying them on MLP-NN

ers is almost equal. But, the convergence speed of the FNNPSOGSA classifier is the best one. For doing a good comparison, it is desirable that the values of mean square error (MSE) for all runs are close to the mean value. So, the large value of STD indicates the high dispersion around the mean value, that this is not desirable. The *P*-value calculation is done for each algorithm to show how strong the proposed algorithm is compared to each algorithm. If this amount is less than 0.05 for each algorithm, the proposed algorithm will have a significant advantage. As can be seen in Table 1, the amounts of the *P*-value for each algorithm are much less than 0.05. So, the superiority of the proposed algorithm is very significant. These results for ds1a are presented in Table 2. Moreover, the classification accuracy of FNNPSOGSA is compared with some popular machine learning classifiers such as SVM and KNN in terms of classification rate. Fig. 10 shows this comparison and it can be seen that the classification accuracy of the proposed algorithm, FNNPSOGSA, is much better than others. All of the simulation and results were done using the Matlab software.

For a good comparison, the FNNPSOGSA classifier performance is compared with some benchmark methods that test this dataset in similar conditions. Table 3 shows the motor imagery classification accuracy obtained per subject. As seen, the FNNPSOGSA approach reaches the best accuracy in comparison with the method given in [49], that includes common spatio-time-frequency patterns to design the time windows for the motor imagery task. The motor imagery classification procedure described in [50] also involves the empirical mode decomposition (EMD)-based common spatial pattern (CSP) preprocessing. The proposed adaptive frequency band selection together with the developed method of feature extraction are not sufficient, causing a low classification performance with a high standard deviation value. The approach given in [51] is based on a robust learning method that extracts spatio-spectral features for discriminating multiple EEG tasks. The achieved motor imagery classification has the lowest performance among the comparative approaches. Another technique given in [52] is based on feature relevance analysis within the motor imagery classification framework. This method reached to a 92.86% classification accuracy that is good performance, but the proposed method in this paper, FNNPSOGSA, has the highest performance compared to the other techniques in term of classification accuracy.

8 Discussions

PSO, GSA and PSOGSA have several parameters which should be initialized. For PSO, we use these set-

Table 1 Comparison of classification accuracy and mean square error (MSE) for the proposed algorithm to others

	ds1a			ds1b		
Algorithm	MSE (AVE±STD)	*P*-value	Accuracy (%)	MSE (AVE±STD)	*P*-value	Accuracy (%)
FNNGSA	0.1356±0.0119	6.79E–08	86.1071	0.1770±0.0150	1.8267E–04	87.0714
FNNPSO	0.1004±0.0096	0.0051	91.7500	0.1331±0.0167	0.0036	90.0000
FNNIPSO	0.1716±0.0417	6.80E–08	87.5000	0.2730±0.0430	1.8267E–04	76.4286
FNNMPSO	0.1943±0.0458	6.80E–08	84.6786	0.2635±0.0489	1.8267E–04	79.4286
FNNTACPSO	0.1909±0.0673	6.79E–08	84.7143	0.2630±0.0646	1.8267E–04	77.7143
FNNPSOGSA	**0.0914±0.0084**	**N/A**	**93.3929**	**0.098 ±0.0188**	**N/A**	**94.3571**
	ds1c			ds1d		
Algorithm	MSE (AVE±STD)	*P*-value	Accuracy (%)	MSE (AVE±STD)	*P*-value	Accuracy (%)
FNNGSA	5.1898E–05±4.2781E–05	1.83E–04	100.0000	0.0309±0.0023	1.8267E–04	96.6429
FNNPSO	0.0026±0.0026	1.82E–04	99.9286	0.0271± 0.0065	5.8284E–04	98.0000
FNNIPSO	0.0300±0.0183	0.0028	97.5000	0.0726± 0.0528	1.7861E–04	94.1429
FNNMPSO	0.0199±0.0201	0.0399	98.1429	0.0638± 0.0724	1.7265E–04	95.0714
FNNTACPSO	0.0204±0.0263	0.0257	98.0714	0.0454± 0.0222	1.7761E–04	96.7857
FNNPSOGSA	**1.2041E–08±2.5627E–08**	**N/A**	**100**	**0.0108± 0.0056**	**N/A**	**99.5000**
	ds1e			ds1f		
Algorithm	MSE (AVE±STD)	*P*-value	Accuracy (%)	MSE (AVE±STD)	*P*-value	Accuracy (%)
FNNGSA	0.0226± 0.0050	1.82E–04	98.9286	0.0090± 0.0022	0.0173	92.2857
FNNPSO	0.0214±0.0085	4.39E–04	96.5714	0.0084± 0.0032	0.0340	99.4286
FNNIPSO	0.0492±0.0225	2.46E–04	87.5000	0.0264± 0.0073	3.1307E–04	98.5000
FNNMPSO	0.0522 ±0.0188	1.81E–04	96.2857	0.0336± 0.0244	0.0022	97.4286
FNNTACPSO	0.0568±0.0339	1.81E–04	96.0000	0.0487± 0.0290	3.2813E–04	97.0000
FNNPSOGSA	**0.0036±0.0043**	**N/A**	**99.8571**	**0.0052± 0.0047**	**N/A**	**99.6429**

ds1g			
Algorithm	MSE (AVE±STD)	*P*-value	Accuracy (%)
FNNGSA	0.0910± 0.0068	1.82E–04	93.7857
FNNPSO	0.0598±0.0104	4.39E–04	96.5714
FNNIPSO	0.0883±0.0420	2.46E–04	96.5000
FNNMPSO	0.1186±0.0451	1.81E–04	90.9286
FNNTACPSO	0.0800±0.0040	1.81E–04	94.7857
FNNPSOGSA	**0.0445±0.0043**	**N/A**	**98.0000**

tings: population size=30, c_1=2, c_2=2, w is decreased linearly from 0.9 to 0.2, maximum iteration=500, and stopping criteria is maximum iteration. For GSA and PSOGSA, we use these settings: population size=30, c'_1=0.5, c'_2=1.5, w= random number in [0,1], $G0$=1, α=20, maximum iteration is 500, and stopping criteria is maximum iteration. The MSE and comparison convergence curves are used to compare the aforementioned algorithms. To perform a fairly good comparison, all algorithms stop when the maximum number of replications reaches 500. Ultimately, the convergence of the results will be examined for a comprehensive comparison. Statistically, the PSOGSA algorithm is sufficiently capable of preventing trapping at local minima that this fact is confirmed by statistical results in Table 1, and its convergence track can be seen in Fig. 9. The better performance of the PSOGSA algorithm is due to the GSA algorithm's search for all of the problem space and the high convergence rate to achieve a general optimum with the PSO algorithm. In general, it can be seen that FNNGSA does not perform well compared to other classifiers. This weakness is due to the low velocity of the search for gravity algorithm that affects the exploitation of FNNGSA. The learning algorithms used in MLP-NNs require not only

Table 2 Statistical comparison results among all algorithms for EEG data set (ds1a)

	GSA		PSO		IPSO		MPSO		TACPSO		PSOGSA	
Number	MSE	Accuracy	MSE	Accuracy	MSE	Accuracy	MSE	Accuracy	MSE	Accuracy	MSE	Accuracy
Run1	0.13897	90.714	0.11603	89.286	0.14613	91.429	0.17181	81.429	0.16429	91.429	0.08	94.286
Run2	0.145	77.857	0.09913	91.429	0.16534	86.429	0.19035	88.571	0.12857	93.571	0.09761	93.571
Run3	0.13797	90	0.09501	92.857	0.21912	83.571	0.20698	87.143	0.22595	80.714	0.09593	92.143
Run4	0.12382	91.429	0.09645	92.143	0.22997	77.143	0.15358	89.286	0.32703	61.429	0.08107	95
Run5	0.11884	90.714	0.09835	92.857	0.16148	90	0.22143	88.571	0.36935	52.143	0.0744	93.571
Run6	0.12444	71.429	0.09818	92.143	0.15421	91.429	0.12854	92.857	0.17257	87.143	0.09764	93.571
Run7	0.13387	77.143	0.10541	89.286	0.22249	75.714	0.14418	90	0.25626	77.143	0.08773	92.143
Run8	0.14007	90.714	0.08362	93.571	0.17608	85	0.17701	86.429	0.21524	81.429	0.08839	92.857
Run9	0.13692	91.429	0.11364	90.714	0.14955	90	0.22835	88.571	0.16498	85	0.10035	92.857
Run10	0.12278	92.143	0.10314	90.714	0.14286	92.857	0.23694	82.857	0.17145	90	0.08282	95
Run11	0.16181	87.143	0.10166	91.429	0.13935	90.714	0.25575	79.286	0.23571	81.429	0.09555	92.143
Run12	0.15601	87.143	0.11869	90	0.17143	90.714	0.16775	90	0.2	89.286	0.08047	95.714
Run13	0.15428	52.143	0.08239	94.286	0.14961	87.143	0.22653	82.143	0.13547	91.429	0.08667	94.286
Run14	0.13092	91.429	0.09656	93.571	0.12857	93.571	0.1436	87.857	0.17143	90.714	0.09663	94.286
Run15	0.13634	90.714	0.10403	92.143	0.15	92.143	0.17926	88.571	0.21321	80.714	0.09	92.143
Run16	0.14156	90	0.10239	92.143	0.15005	91.429	0.31447	50.714	0.11429	94.286	0.0978	92.143
Run17	0.1222	89.286	0.10514	91.429	0.24475	81.429	0.17481	80	0.15	92.143	0.09938	93.571
Run18	0.13083	90	0.0869	92.857	0.15	91.429	0.2243	80	0.15	92.143	0.10579	92.857
Run19	0.12383	89.286	0.09428	93.571	0.11429	94.286	0.19754	86.429	0.12461	91.429	0.09718	92.143
Run20	0.13084	91.429	0.10642	88.571	0.26625	73.571	0.14286	92.857	0.12778	90.714	0.09164	93.571
Mean	**0.1356**	**86.1071**	**0.1004**	**91.7500**	**0.1716**	**87.5000**	**0.1943**	**84.6786**	**0.1909**	**84.7143**	**0.0914**	**93.3929**
STD	**0.0119**	**89.7869**	**0.0096**	**1.5968**	**0.0417**	**6.1902**	**0.0458**	**9.0067**	**0.0673**	**10.8979**	**0.0084**	**1.1084**
***P*-value**	**6.79E–08**	**9.13E–08**	**0.0051**	**0.0016**	**6.80E–08**	**1.04E–05**	**6.80E–08**	**5.34E–07**	**6.79E–08**	**5.78E–06**	**N/A**	**N/A**

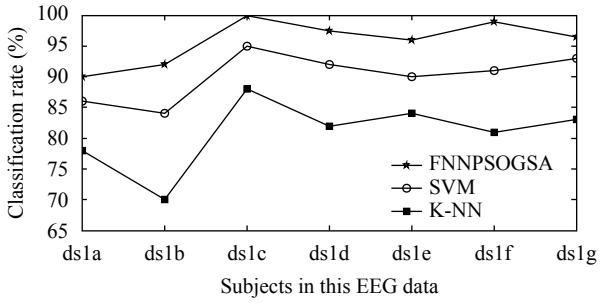


Fig. 10 Comparison of MSE for the proposed algorithm to others when applying them on MLP-NN

the ability to search robustly, but also the ability to operate accurately. According to the results of classification by MLP-NNs trained by EAs, it can be concluded that FNNPSO performance is better than FNNGSA. This superiority is due to the ability of the PSO algorithm to exploit more precisely. But this algorithm still has the problem of being trapped in the local minimum. This means that the FNNPSO performs without stability. The results obtained by FNNPSOGSA prove that this algorithm has both advantages of strong exploitation and good search capability. In other words, the PSO and the GSA have been successfully used and show very good performance in MLP-NN training, which the results confirm. This means that FNNPSOGSA is capable of solving the problem of trapping at the local minimum and has a high convergence rate.

9 Conclusions

This paper has proposed the PSOGSA to train NNs for motor imagery classification. The CSP method has used to extract the feature of an EEG signal. Using CSP, the data can be reduced from 59 dimensions to 10 dimensions. Then, these features are classified using an MLP-NN whose parameters are trained by PSOGSA algorithm. The classification accuracy of this classifier was compared with MLP-NN classifiers trained by other metaheuristic algorithms such as GSA, PSO, IPSO, MPSO, and TACPSO. In addition to classification accuracy, another parameter that demonstrates the performance of the BCI system is the speed. In this problem, the convergence speed indicated this parameter. It observed that the GSA algorithm is good for its global search capability, but it has the low convergence speed in the last itera-

Table 3 Comparison the proposed method with other works in term of classification accuracy (average accuracy ± standard deviation)

Subject	Higashi and Tanaka[49]	He et al.[50]	Zhang et al.[51]	Álvarez-Meza et al.[52]	FNNPSOGSA (proposed)
ds1a	92.30±02.50	67.70±02.20	77.20±00.03	91.50±01.20	93.3929±01.1084
ds1b	90.60±07.20	70.70±01.20	70.80±0.02	96.50±03.37	94.3571±0.9421
ds1c	–	83.90±01.30	–	91.50±04.74	100±1.0710
ds1d	–	93.00±01.20	–	87.00±06.32	99.5000±01.7021
ds1e	–	93.20±01.20	–	91.50±07.47	99.8571±01.0178
ds1f	93.30±03.60	–	76.80±0.03	98.50±02.42	99.6429±0.9987
ds1g	94.10±04.10	–	80.00±0.03	93.50±07.09	98.0000±1.1270
Mean± STD	92.58±01.51	81.70±12.06	76.20±03.87	92.86±03.77	**97.8214±2.7890**

tions. On the other hand, due to the unique advantage of PSO in exploitation phase, the combination of both are used, and called "PSOGSA". The results showed that the classification accuracy of MLP-NN trained by PSOGSA is better than the other discussed algorithms and has the best convergence. Statistical results showed that the performance of proposed method, FNNPSOGSA, is very good. In the future work, it would be interesting to investigate the performance of PSOGSA in training other types of NNs such as radial basis function and recurrent NNs.

References

[1] A. S. Aghaei, M. S. Mahanta, K. N. Plataniotis. Separable common spatio-spectral patterns for motor imagery BCI systems. *IEEE Transactions on Biomedical Engineering*, vol. 63, no. 1, pp. 15–29, 2016. DOI: 10.1109/TBME.2015.2487738.

[2] D. D. Huang, K. Qian, D. Y. Fei, W. C. Jia, X. D. Chen, O. Bai. Electroencephalography (EEG)-based brain-computer interface (BCI): A 2-D virtual wheelchair control based on event-related desynchronization/synchronization and state control. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 20, no. 3, pp. 379–388, 2012. DOI: 10.1109/TNSRE.2012.2190299.

[3] K. LaFleur, K. Cassady, A. Doud, K. Shades, E. Rogin, B. He. Quadcopter control in three-dimensional space using a noninvasive motor imagery based brain-computer interface. *Journal of Neural Engineering*, vol. 10, no. 4, Article number 046003, 2013. DOI: 10.1088/1741-2560/10/4/046003/meta.

[4] J. R. Wolpaw, N. Birbaumer, D. J. McFarland, G. Pfurtscheller, T. M. Vaughan. Brain-computer interfaces for communication and control. *Clinical Neurophysiology*, vol. 113, no. 6, pp. 767–791, 2002. DOI: 10.1016/S1388-2457(02)00057-3.

[5] T. M. Vaughan. Guest editorial brain-computer interface technology: A review of the second international meeting. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 11, no. 2, pp. 94–109, 2003. DOI: 10.1109/TNSRE.2003.814799.

[6] C. R. Hema, M. P. Paulraj, S. Yaacob, A. H. Adom, R. Nagarajan. EEG motor imagery classification of hand movements for a brain machine interface. *Biomedical Soft Computing and Human Sciences*, vol. 14, no. 2, pp. 49–56, 2009. DOI: 10.24466/ijbschs.14.2_49.

[7] B. Blankertz, G. Dornhege, M. Krauledat, K. R. Muller, G. Curio. The non-invasive Berlin brain-computer interface: Fast acquisition of effective performance in untrained subjects. *NeuroImage*, vol. 37, no. 2, pp. 539–550, 2007. DOI: 10.1016/j.neuroimage.2007.01.051.

[8] X. Y. Yu, P. Chum, K. B. Sim. Analysis the effect of PCA for feature reduction in non-stationary EEG based motor imagery of BCI system. *Optik-International Journal for Light and Electron Optics*, vol. 125, no. 3, pp. 1498–1502, 2014. DOI: 10.1016/j.ijleo.2013.09.013.

[9] R. N. Vigário. Extraction of ocular artefacts from EEG using independent component analysis. *Electroencephalography and Clinical Neurophysiology*, vol. 103, no. 3, pp. 395–404, 1997. DOI: 10.1016/S0013-4694(97)00042-8.

[10] J. B. Tenenbaum, V. de Silva, J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, vol. 290, no. 5500, pp. 2319–2323, 2000. DOI: 10.1126/science.290.5500.2319.

[11] T. Wu, G. Z. Yan, B. H. Yang, H. Sun. EEG feature extraction based on wavelet packet decomposition for brain computer interface. *Measurement*, vol. 41, no. 6, pp. 618–625, 2008. DOI: 10.1016/j.measurement.2007.07.007.

[12] H. Ramoser, J. Muller-Gerking, G. Pfurtscheller. Optimal spatial filtering of single trial EEG during imagined hand movement. *IEEE Transactions on Rehabilitation Engineering*, vol. 8, no. 4, pp. 441–446, 2000. DOI: 10.1109/86.895946.

[13] G. Pfurtscheller, C. Neuper, A. Schlogl, K. Lugger. Separability of EEG signals recorded during right and left motor imagery using adaptive autoregressive parameters. *IEEE Transactions on Rehabilitation Engineering*, vol. 6, no. 3, pp. 316–325, 1998. DOI: 10.1109/86.712230.

[14] S. Lemm, C. Schafer, G. Curio. BCI competition 2003-data set III: Probabilistic modeling of sensorimotor mu rhythms for classification of imaginary hand movements. *IEEE Transactions on Biomedical Engineering*, vol. 51, no. 6, pp. 1077–1080, 2004. DOI: 10.1109/TBME.2004.827076.

[15] S. M. Zhou, J. Q. Gan, F. Sepulveda. Classifying mental tasks based on features of higher-order statistics from EEG signals in brain-computer interface. *Information Sciences*, vol. 178, no. 6, pp. 1629–1640, 2008. DOI: 10.1016/j.ins.2007.11.012.

[16] Y. L. Ma, X. H. Ding, Q. S. She, Z. Z. Luo, T. Potter, Y. C. Zhang. Classification of motor imagery EEG signals with support vector machines and particle swarm optimization.

Computational and Mathematical Methods in Medicine, vol. 2016, Article number 4941235, 2016. DOI: 10.1155/2016/4941235.

[17] A. Subasi, E. Erçelebi. Classification of EEG signals using neural network and logistic regression. *Computer Methods and Programs in Biomedicine*, vol. 78, no. 2, pp. 87–99, 2005. DOI: 10.1016/j.cmpb.2004.10.009.

[18] D. Whitley. A genetic algorithm tutorial. *Statistics and Computing*, vol. 4, no. 2, pp. 65–85, 1994. DOI: 10.1007/BF00175354.

[19] J. Kennedy, R. Eberhart. Particle swarm optimization. In *Proceedings of International Conference on Neural Networks*, Perth, Australia, pp. 1942–1948, 1995. DOI: 10.1109/ICNN.1995.488968.

[20] M. Dorigo, V. Maniezzo, A. Colorni. Ant system: Optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 26, no. 1, pp. 29–41, 1996. DOI: 10.1109/3477.484436.

[21] X. H. Shi, Y. C. Liang, H. P. Lee, C. Lu, L. M. Wang. An improved GA and a novel PSO-GA-based hybrid algorithm. *Information Processing Letters*, vol. 93, no. 5, pp. 255–261, 2005. DOI: 10.1016/j.ipl.2004.11.003.

[22] A. J. Ouyang, Y. Q. Zhou. An improved PSO-ACO algorithm for solving large-scale TSP. *Advanced Materials Research*, vol. 143–144, pp. 1154–1158, 2011. DOI: 10.4028/www.scientific.net/AMR.143-144.1154.

[23] S. T. Li, M. K. Tan, I. W. Tsang, J. T. Y. Kwok. A hybrid PSO-BFGS strategy for global optimization of multimodal functions. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 41, no. 4, pp. 1003–1014, 2011. DOI: 10.1109/TSMCB.2010.2103055.

[24] S. Mirjalili, S. Z. M. Hashim. A new hybrid PSOGSA algorithm for function optimization. In *Proceedings of International Conference on Computer and Information Application*, Tianjin, China, pp. 374–377, 2010. DOI: 10.1109/ICCIA.2010.6141614.

[25] S. Q. Sun, Q. K. Peng. A hybrid PSO-GSA strategy for high-dimensional optimization and microarray data clustering. In *Proceedings of IEEE International Conference on Information and Automation*, Hailar, China, pp. 41–46, 2014. DOI: 10.1109/ICInfA.2014.6932623.

[26] H. Aminzadeh, M. Miri. Optimal placement of phasor measurement units to obtain network observability using a hybrid PSO-GSA algorithm. *Australian Journal of Electrical and Electronics Engineering*, vol. 12, no. 4, pp. 342–349, 2015. DOI: 10.1080/1448837X.2015.1092929.

[27] E. Rashedi, H. Nezamabadi-Pour, S. Saryazdi. GSA: A gravitational search algorithm. *Information Sciences*, vol. 179, no. 13, pp. 2232–2248, 2009. DOI: 10.1016/j.ins.2009.03.004.

[28] Y. Jiang, T. S. Hu, C. C. Huang, X. N. Wu. An improved particle swarm optimization algorithm. *Applied Mathematics and Computation*, vol. 193, no. 1, pp. 231–239, 2007. DOI: 10.1016/j.amc.2007.03.047.

[29] T. Geetha, M. Sathya. Modified particle swarm optimization (MPSO) algorithm for web service selection (WSS) problem. In *Proceedings of International Conference on Data Science & Engineering*, Cochin, India, pp. 113–116, 2012. DOI: 10.1109/ICDSE.2012.6281954.

[30] G. Q. Bao, K. F. Mao. Particle swarm optimization algorithm with asymmetric time varying acceleration coefficients. In *Proceedings of IEEE International Conference on Robotics and Biomimetics*, Guilin, China, pp. 2134–2139, 2009. DOI: 10.1109/ROBIO.2009.5420504.

[31] C. Cortes, V. Vapnik. Support-vector networks. *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995. DOI: 10.1007/BF00994018.

[32] N. S. Altman. An introduction to kernel and nearest-neighbor nonparametric regression. *The American Statistician*, vol. 46, no. 3, pp. 175–185, 1992. DOI: 10.1080/00031305.1992.10475879.

[33] M. Wessel. Pioneering Research into Brain Computer Interfaces, Master dissertation, Delft University of Technology, the Netherlands, 2006.

[34] H. Jasper, W. Penfield. Electrocorticograms in man: Effect of voluntary movement upon the electrical activity of the precentral gyrus. *Archiv fur Psychiatrie und Nervenkrankheiten*, no. 1–2, pp. 163–174, 1949. DOI: 10.1007/BF01062488.

[35] Z. J. Koles. The quantitative extraction and topographic mapping of the abnormal components in the clinical EEG. *Electroencephalography and Clinical Neurophysiology*, vol. 79, no. 6, pp. 440–447, 1991. DOI: 10.1016/0013-4694(91)90163-X.

[36] V. Abedifar, M. Eshghi, S. Mirjalili, S. M. Mirjalili. An optimized virtual network mapping using PSO in cloud computing. In *Proceedings of the 21st Iranian Conference on Electrical Engineering*, Mashhad, Iran, 2013. DOI: 10.1109/IranianCEE.2013.6599723.

[37] L. S. Nguyen, D. Frauendorfer, M. S. Mast, D. Gatica-Perez. Hire me: Computational inference of hirability in employment interviews based on nonverbal behavior. *IEEE Transactions on Multimedia*, vol. 16, no. 4, pp. 1018–1031, 2014. DOI: 10.1109/TMM.2014.2307169.

[38] P. Auer, H. Burgsteiner, W. Maass. A learning rule for very simple universal approximators consisting of a single layer of perceptrons. *Neural Networks*, vol. 21, no. 5, pp. 786–795, 2008. DOI: 10.1016/j.neunet.2007.12.036.

[39] M. R. Mosavi, M. Khishe. Training a feed-forward neural network using particle swarm optimizer with autonomous groups for sonar target classification. *Journal of Circuits, Systems and Computers*, vol. 26, no. 11, Article number 1750185, 2017. DOI: 10.1142/S0218126617501857.

[40] M. Khishe, M. R. Mosavi, M. Kaveh. Improved migration models of biogeography-based optimization for sonar dataset classification by using neural network. *Applied Acoustics*, vol. 118, pp. 15–29, 2017. DOI: 10.1016/j.apacoust.2016.11.012.

[41] M. R. Mosavi, M. Khishe, A. Ghamgosar. Classification of sonar data set using neural network trained by gray wolf optimization. *Neural Network World*, vol. 26, no. 4, pp. 393–415, 2016. DOI: 10.14311/NNW.2016.26.023.

[42] M. R. Mosavi, M. Khishe, M. Akbarisani. Neural network trained by biogeography-based optimizer with chaos for sonar data set classification. *Wireless Personal Communications*, vol. 95, no. 4, pp. 4623–4642, 2017. DOI: 10.1007/s11277-017-4110-x.

[43] J. Müller-Gerking, G. Pfurtscheller, H. Flyvbjerg. Designing optimal spatial filters for single-trial EEG classification in a movement task. *Clinical Neurophysiology*, vol. 110, no. 5, pp. 787–798, 1999. DOI: 10.1016/S1388-2457(98)00038-8.

[44] J. Yan. The Attract Force Equation of Energy. *American Journal of Modern Physics*, vol. 3, no. 6, pp. 224–226, 2014. DOI: 10.11648/j.ajmp.20140306.13.

[45] A. A. Abarghouei, A. Ghanizadeh, S. M. Shamsuddin. Advances of soft computing methods in edge detection. *International Journal of Advances in Soft Computing and Its Applications*, vol. 1, no. 2, pp. 162–203, 2009.

[46] E. Rashedi, H. Nezamabadi-Pour, S. Saryazdi. BGSA: Binary gravitational search algorithm. *Natural Computing*, vol. 9, no. 3, pp. 727–745, 2010. DOI: 10.1007/s11047-009-9175-3.

[47] S. Mirjalili, S. M. Mirjalili, A. Lewis. Let a biogeography-based optimizer train your multi-layer perceptron. *Information Sciences*, vol. 269, pp. 188–209, 2014. DOI: 10.1016/j.ins.2014.01.038.

[48] D. H. Wolpert, W. G. Macready. No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 67–82, 1997. DOI: 10.1109/4235.585893.

[49] H. Higashi, T. Tanaka. Common spatio-time-frequency patterns for motor imagery-based brain machine interfaces. *Computational Intelligence and Neuroscience*, vol. 2013, Article number 537218, 2013. DOI: 10.1155/2013/537218.

[50] W. He, P. F. Wei, L. P. Wang, Y. X. Zou. A novel emd-based common spatial pattern for motor imagery brain-computer interface. In *Proceedings of IEEE-EMBS International Conference on Biomedical and Health Informatics*, Hong Kong, China, pp. 216–219, 2012. DOI: 10.1109/BHI.2012.6211549.

[51] H. H. Zhang, C. T. Guan, K. K. Ang, C. C. Wang, Z. Y. Chin. BCI competition IV - data set I: Learning discriminative patterns for self-paced EEG-based motor imagery detection. *Frontiers in Neuroscience*, vol. 6, Article number 7, 2012. DOI: 10.3389/fnins.2012.00007. DOI: 10.3389/fnins.2012.00007.

[52] A. M. Alvarez-Meza, L. F. Velasquez-Martinez, G. Castellanos-Dominguez. Time-series discrimination using feature relevance analysis in motor imagery classification. *Neurocomputing*, vol. 151, pp. 122–129, 2015. DOI: 10.1016/j.neucom.2014.07.077.

Sajjad Afrakhteh received the B. Sc. degree in electrical engineering from Shiraz University of Science and Technology, Iran in 2015, and the M. Sc. degree in digital electronic from Iran University of Science and Technology (IUST), Iran in 2017. He is currently a Ph.D. degree candidate in electronic engineering, IUST, Iran.

His research interests include system and circuit design, signal processing, image processing, artificial neural networks and meta-heuristic algorithms.

E-mail: s _afrakhteh94@elec.iust.ac.ir
ORCID iD: 0000-0002-4184-9436

Mohammad-Reza Mosavi received the B. Sc., M. Sc. and Ph. D. degrees in electronic engineering from Iran University of Science and Technology, Iran in 1997, 1998, and 2004, respectively. He is currently faculty member (full professor) of the Department of Electrical Engineering, IUST, Iran. He is the author of more than 330 scientific publications in journals and international conferences.

His research interests include circuits and systems design.

E-mail: m _mosavi@iust.ac.ir (Corresponding author)
ORCID iD: 0000-0002-2389-644X

Mohammad Khishe received the B. Sc. degree in maritime electronic and communication engineering from Imam Khomeini Maritime Sciences University, Iran in 2007, the M. Sc. and Ph. D. degrees in electronic engineering from Islamic Azad University, Qazvin branch and Iran University of Science and Technology, Tehran in 2012 and 2017, respectively. Since 2011, he has been a faculty member of Maritime Electrical and Electronic Engineering at Imam Khomeini Maritime Sciences University, Iran.

His research interests include digital signal processing, artificial neural networks, meta-heuristic algorithms, sonar and radar signal processing, and field-programmable gate array (FPGA) design.

E-mail: m _khishe@elec.iust.ac.ir
ORCID ID: 0000-0002-1024-8822

Ahmad Ayatollahi received the B. Sc. degree in electrical engineering from Iran University of Science and Technology, Iran in 1976, and the M. Sc. and Ph.D. degrees in instrumentation engineering and electrical engineering from University of Manchester Institute of Science and Technology (UMIST), UK in 1985 and 1989, respectively. He is currently faculty member (associate professor) of the College of Electrical Engineering, Iran University of Science and Technology, Tehran.

His research interests include medical image and signal processing, design of electronic circuits, power electronics, and ultrasound.

E-mail: Ayatollahi@iust.ac.ir