International Journal of Applied Metaheuristic Computing
Volume 13 • Issue 1

High Performance Implementation of Neural Networks Learning Using Swarm Optimization Algorithms for EEG Classification Based on Brain Wave Data

Ali Al Bataineh, Electrical and Computer Engineering Department, Norwich University, USA*
Amin Jarrah, Department of Computer Engineering, Yarmouk University, Jordan

ABSTRACT

Electroencephalography (EEG) analysis exploits computer technologies and mathematical signal analysis techniques to derive useful information from EEG signals. EEG analysis aims to help scientists better understand the brain, help physicians make better decisions about patient diagnoses and treatment choices and advance the Brain-Computer Interface (BCI) technology. Over the last years, machine learning has been very much used in EEG analysis. Artificial neural networks (ANNs) are among the most effective machine learning algorithms that perform computing tasks similar to biological neurons in the human brain. Nevertheless, the neural network model's performance might significantly degrade and overfit due to some irrelevant features that negatively influence the model performance. Therefore, feature selection should be the first step of the model design. Furthermore, neural networks performance relies heavily on the algorithm that is employed to train them. Backpropagation is a common technique for training neural networks; however, backpropagation might get stuck into a set of sub-optimal weights and biases from which it cannot escape. Both feature selection and training can be formulated as optimization problems. Thus, proper optimization techniques to use for feature selection and training will contribute to performance improvement. Swarm optimization algorithms are robust and powerful techniques that can be employed to find high-quality solutions to such problems. In this paper, Grey Wolf Optimizer (GWO) and Particle Swarm Optimization (PSO) algorithms are applied for the feature selection and the training of a Feed-forward Neural Network (FFNN) to classify the state of the eye as closed or open based on brain wave data (EEG). The aim is to enhance the FFNN's ability to detect the eye state with high accuracy. The performance of the FFNN in terms of test accuracy, precision, recall, and F1_score is investigated. To better estimate the ability and effectiveness of the proposed model, five classical machine learning classifiers are used: (1) Support Vector Machine; (2) k-Nearest Neighbors; (3) Decision Tree; (4) Gaussian Naive Bayes; (5) Logistic Regression. Experimental results proved that the FFNN model outperforms all other classifiers, particularly when feature selection and training are performed with the GWO algorithm.

KEYWORDS

Artificial Neural Networks (AANs), Brain-Computer Interface (BCI), Electroencephalography (EEG), Grey Wolf Optimizer (GWO), High Performance Computing, Optimized Implementation, Particle Swarm Optimization (PSO)

1. INTRODUCTION

Electroencephalography, or EEG for short, is an electrophysiological process to record the brain's electrical activity using EEG electrodes placed on the scalp (Kosmyna and Lécuyer, 2019). EEG data

DOI: 10.4018/IJAMC.292500

*Corresponding Author

are hard to deal with, as they can be very noisy. Neuroscientists, biomedical engineers, and clinicians often receive years of training to understand and extract meaningful EEG data information. Moreover, the raw recorded data must be processed before it can be viewed by professionals. Raw EEG data is just a discrete-time multivariate (i.e., with multiple dimensions) time-series. The number of EEG channels defines the dimension of each point in the time series. Each time point corresponds to an EEG sample obtained at the same time point. The number of points in the time series depends on the recorded time and the sampling rate (Perronnet et al. 2016). These raw signals are rarely used because they may include DC offsets and drifts, electromagnetic noise, and artefacts that must be filtered. Temporal and spatial filtering is frequently used to remove noise, filter out artefacts, or isolate an enhanced version of the signal of interest. Hereafter, frequency filters like band-pass filters or low-pass are applied on the clean EEG data to isolate the bands of interest and remove those frequencies of no interest as the human brain processes in the P300 evoked response that occurs in the Theta band (4-7 Hz).

Feature extraction is applied to extract important features from the cleaned EEG data (Dry et al. 2020). Before the popularity of deep learning, feature extraction relied on custom methods of the brain's interest process ranging from handcrafted features to more complex technologies, such as linear and nonlinear spatial filtering. The following ranges from general methods such as independent component analysis and principal component analysis, to more specific EEG methods such as CSPs (Blankertz et al. 2008) and (Ang et al. 2008) for energy features and (Rivet et al. 2009) for the temporal ones. The features extracted are usually fitted to the preferences of a specific application, such as finding differences between experimental conditions, distinguishing between a set of predefined categories, predicting behavior, finding anomalies in relation to a normative database. The current state of the art technologies includes Riemannnet based classifiers, filter banks, and adaptive classifiers, used to deal with EEG data challenges with various levels of success (Lotte et al. 2018).

Once features are ready, the processed EEG data can then be inspected visually to recognize anomalies, alterations in mental status, or to examine average responses in a number of people. However, the visual inspection process is tedious, long, and costly. It does not scale accurately and cannot be ported to BCI applications. In contrast, Artificial Intelligence (AI), in particular, Machine Learning (ML), is an ideal approach to automating, expanding, and enhancing the analysis of EEG data. Automated machine learning approaches can be efficiently applied to solve the time series classification problems like EEG. A favorite type of machine learning is supervised learning, which uses a set of examples called training data to learn a model that can predict, classify, or identify EEG patterns based on the extracted features. A generous variety of techniques exist. The most well-known are classification techniques, which classify an EEG pattern into one of a set of predefined classes or regression techniques that transform the EEG pattern into a different signal (for instance, motion direction). Adopted approaches include simple linear techniques (Multiple Linear Regression and LDA for classification), SVM like kernel methods, K-Nearest Neighbors, neural networks, and many more.

Whatever the method, supervised methods must have a labeled training dataset. This dataset is used to train and evaluate the method, normally using some performance metrics such as prediction accuracy. Artificial neural networks (ANNs) are one of the most well-known and widely used methods. ANNs are computation methods inspired by the human brain and widely used to solve a variety of complex problems in many application areas (Bataineh et al. 2018a). The reason is that they are universal and capable of solving any problem if they are properly trained and have enough data. Just like any supervised learning algorithm, in ANNs, the network topology is determined first; then, the input is fed. After that, the error is calculated by comparing the actual output with the predicted output, and then the error is optimized using some optimization algorithm. The most common algorithm to train ANNs is backpropagation since it is flexible and traceable (Al Bataineh, 2019).

However, the backpropagation algorithm cannot guarantee an optimal solution. In real-world applications, the backpropagation algorithm may converge locally into a set of sub-optimal weights and biases that it cannot escape. As a consequence, the neural network is usually incapable of obtaining a desirable solution to a specific problem. As opposed to, swarm population-based search algorithms such as Particle Swarm Optimization (PSO) and Grey Wolf Optimizer (GWO) are global optimization techniques that can avoid local optima solutions through their essential component, and that is the randomness of the initial solutions. In this paper, a neural network with fixed topology is employed to detect the eye state (open/close) of a person using the EEG eye state dataset. To enhance the detection ability of the neural network model, we first apply feature selection using each algorithm via PSO and GWO optimization algorithms; next, the neural network is trained using each swarm algorithm with the features selected by each algorithm. This work utilized the EEG eye state dataset (Rösler and Suendermann, 2013) to benchmark the performance of the neural network model with each proposed algorithm.

Analyzing EEG data is an excellent approach to investigate many different aspects of human cognition, emotions, and behavior. EEG technology not only helps to study the human brain but also has useful applications in health, emotional and effective monitoring (Dry et al. 2020). It can assist doctors in establishing a medical diagnosis, researchers to learn the brain processes that underlie individual behavior and human to enhance their productivity and wellness. Notwithstanding, EEG data is difficult to interpret. It contains much noise, differs for each individual, and even for the same person, it changes dramatically with time. Recently, machine learning has been significantly utilized to analyze EEG data in an efficient way. Machine learning can automate, extend, and enhance EEG data analysis.

Therefore, this motivates our work to use machine learning, particularly neural networks for EEG data analysis, as they are capable of solving almost any task if a great amount of data available and proper trainer to use to learn the task, which is often accomplished by backpropagation. However, backpropagation depends highly on the initial parameters of the model in which can cause it to be trapped into local minima solutions. For this reason, this work applies alternative approaches to train the neural network model, specifically PSO and GWO, to improve the ability of the model to accurately predict EEG data of a person's eyes state as open or closed. Both algorithms have been extensively proposed to train neural networks by many researchers and have proved their efficiency in obtaining global solutions. This also motivates us to use these algorithms not only for training the neural network but also for feature selection. Feature selection is vital in time series classification problems like EEG as many machine learning applications from medical diagnosis arise from complex relationships between features. Therefore, it is important to identify and remove irrelevant and redundant features that decrease the accuracy of the predictive model.

Evaluating machine learning models on time series classification problems like EEG data is very challenging. Therefore, the proper classifier must be selected that is capable of delivering the best results. This research has the following objectives:

1. Utilizing the particle swarm optimization (PSO) and grey wolf optimizer algorithms (GWO) to select the relevant features which will be fed to the neural network model in an attempt to enhance its performance.
2. Employing each PSO and GWO algorithms to train the neural network mode along with the selected features to further increase the accuracy of the predictive model to detect the eye state based on EEG data. The PSO and GWO algorithms were shown to be successful at obtaining global solutions.

The paper is divided into eight sections as follows: Section 1 provides an introduction. A literature review based on previous research is presented in Section 2. Section 3 presents the basic definition of neural networks, their mathematical model, and the topology used for this research. Swarm-based algorithms GWO and PSO are explained in section 4. The proposed GWO and PSO techniques for feature selection and training are discussed and explained in Section 5. Results are provided in section 6. Finally, sections 7 and 8 present the conclusion of this research and future work, respectively.

2. LITERATURE REVIEW

Multiple machine learning approaches have been widely used to classify EEG data, such as K star, SVM, AR models, ANN, etc. The most commonly used classifier by many authors is the k-nearest neighbor (kNN). Most of the predictive classifiers have not considered the temporal ordering of the observations and predict the eye state based on the current EEG observation. Intuitively, it is not regarded as sufficient, but it was the approach used by Rosler and Suendermann in their paper, who collected the EEG eye state data (Rösler and Suendermann, 2013). Mainly, they evaluated a wide range of classification algorithms in Weka software using 10-fold cross-validation of this problem's framing. They achieved more than 90% accuracy with many algorithms, including instance-based learners such as k-nearest neighbors, IB1 and KStar. Since then, several papers have used the dataset with the same methodology and finding. Wang et al. (2014), proposed EEG eye state detection based on Incremental Attribute Learning (IAL). Results show that IAL can efficiently cope with time series classification problems with proper feature extraction and feature ordering.

(Sahu et al. 2015) investigated the EEG system's eye state identification by finding feature subset selection named Incremental Feature Reordering (IFR). It provides the most non-dominant feature (MND) for Electroencephalography (EEG) signal corpus and creates a reorder set. The removal of MND delivers optimal subset feature, and it improves the classifier accuracy and efficiency. (Li et al. 2009) used Autoregressive (AR) power spectral density estimation method to analyze EEG signals in eyes-open and eyes-closed states. The two states can be discriminated from the topographical distributions of the delta, theta, alpha, and beta power spectrum. The results showed that the AR model's optimum order could be more suitable for estimating different states' EEG power spectrum. (Sulaiman et al. 2011) proposed Relative Energy Ratio (RER), Shannon Entropy (SE), and Spectral Centroids (SC) techniques to extract stress features from EEG signals during the close and open eye states. The combination of RER, SE, and SC methods with the training and testing of k-NN detected and classified the group with the unique stress features with 88.89% accuracy. (Chambayil et al. 2010) presented a new application of training two types of neural networks namely Cascade-forward Backpropagation (CFBP) and Feed-forward Backpropagation (FFBP) classifiers using Kurtosis coefficient, maximum amplitude, and minimum amplitude to detect the eye blink artifact in the EEG signal. The results concluded that the performance of the CFBP network is better compared to the FFBP network in classifying a signal to an eye blink or not.

(Sadatnezhad et al. 2011) implemented a piecewise linear classifier based on XCSF to classify EEG signals of the two mental diseases via Bipolar Mood Disorder (BMD) and Attention Deficit Hyperactivity Disorder (ADHD), 43 patients participated, 21 patients with ADHD and 22 patients with BMD. Their electroencephalogram (EEG) signals are recorded by 22 electrodes in two eyes-open and eyes-closed resting conditions. Experimental results of XCSF-LDA showed a significant improvement of 86.44% accuracy compared to standard XCSF with 78.55% accuracy. (Pour et al. 2008) presented a system that uses the human ability to control a video game on a mobile device using EEG Mu rhythms. The signals were collected using a specially designed electrode cap and equipment and then sent via a Bluetooth connection to a computer that processes it in the real time. The signal is then mapped to two control signals and transmitted through a wireless connection to a mobile gaming device. The human ability to play the video game by manipulating neuronal motor cortex activity in the presence of a visual feedback environment was also investigated. Participants played the video game by using their ideas only with up to 80% accuracy over controlling the target.

3. NEURAL NETWORKS

Machine learning is essential for very complex tasks that humans cannot directly code. Some tasks are impractical and even impossible for humans to explicitly resolve and code all of the nuances (Keneni, 2019). So, alternately, a large amount of data is provided to the machine learning algorithm and let the algorithm work on it by examining that data and scanning for a model that fulfills what the programmers set it to achieve (Bataineh et al. 2019a). Within the general machine learning literature, there is a successful class of models called neural networks. Neural networks are algorithms that have transformed the machine learning field. Biological neurons in the human brain inspire them, and the current so-called deep neural networks have shown to work exceptionally well. Neural Networks are themselves nonlinear universal function approximations, which is why they can be applied literally to any complex machine learning problem where the problem is about learning a mapping from the input to output space. There are different topologies of neural networks: Multi-layer perceptron also called Feed-forward neural network (Rosenblatt, 1958), Convolutional neural network (Lecun et al. 1998), Recurrent neural network (Elman, 1990), Autoencoders (Rumelhart et al. 1985), and Generative adversarial network (Goodfellow et al. 2014). For this research, we are employing the feed-forward neural network (FFNN) for classifying the eye state based on EEG data. Feed-forward neural networks (FNNNs) have been employed extensively to tackle supervised learning problems. FFNN is such a type in which information is fed in a forward direction on a layer-by-layer basis. FFNN contains one or more hidden layers. It has one input layer, which accepts signals or data from the external world, at least one middle or hidden layer of neurons between input and output to detect features, and one output layer of neurons accepting the features from the hidden layers to specify the output pattern. The popularity of FFNNs (or MLPs) grew with the invention of the backpropagation method in 1986 (Rumelhart et al. 1986). The backpropagation approach uses gradient descent to optimize the weights and biases of the neural network. The algorithm has two phases. In the first phase, the network input layer receives a training input pattern then propagates the input pattern through each layer in the network until the output layer generates the output pattern, next, the difference between the generated output pattern and the desired output (error) is computed. In the second phase, the computed error is propagated backward from the output layer to the input layer through the network, and the initial weights and biases are adjusted while the error is being propagated. It iteratively computes the values of weight using a learning algorithm such as a gradient descent algorithm (Bataineh et al. 2018b). Every neuron in the hidden and output layers determines its output by computing the net weighted input based on Eqn. (1).

$$Output = \sum_{i=0}^{n} w_i x_i + b \tag{1}$$

where n is the number of inputs, and b is the bias value applied to the neuron, then this input value is passed through soft activation or transfer function (sigmoid) based on Eqn. (2).

$$sigmoid\left(x\right) = \frac{1}{1 + e^{-X}} \tag{2}$$

Training FFNNs using backpropagation does not always guarantee an optimal solution as might get trapped into local minima solutions. Also, data features that we apply to train FFNNs models significantly impact the performance that can be achieved. Irrelevant features can negatively influence the model performance. Feature selection is crucial and should be the first step of the model design. Therefore, this research proposes PSO and GWO as alternatives to backpropagation to train FFNN for the EEG classification problem. These algorithms proved their efficiency in discovering a global solution when the search space is large and challenging (Bataineh and Kaur, 2019) Furthermore, unlike gradient descent based backpropagation, there is no need for the cost function to be differentiable. They are a straightforward implementation.

In the next sections, the GWO and PSO algorithms are employed to select optimal features and train the FFNN. The proposed topology of the FFNN in this work consists of three layers; one input layer contains neurons that correspond to the number of features selected by each swarm algorithm, one hidden layer of 100 neurons, and one output layer which includes one neuron corresponding to the eye state (open/close).

4. SWARM- BASED OPTIMIZATION ALGORITHMS

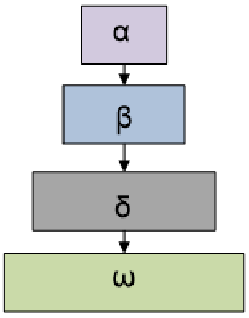
Recently, Optimization algorithms that mimic the swarming behavior have reached immense popularity. The main judgment for their risen applications can be associated with its unique benefits over preceding evolutionary and conventional approaches. Self-organizing abilities are popular with swarm algorithms; all entities that embody the solutions imitate the flocking behaviors within the populations and camps of diverse animals, mammals, and birds. Their application is directed at achieving exploration and exploitation as the swarm proceeds (Mairaj et al. 2019). A popular evolutionary algorithm, such as a Genetic algorithm (GA) uses genetic operators such as crossover, mutation and selection (Whitley, 1994). However, swarm-based algorithms are quite different. In general, in these algorithms, all living entities are turning towards their prey or source of food, then information is shared with others if one of the agents finds the food source. There are different varieties of these algorithms, e.g., some mimic the behaviors of particles, wolves, whales, etc. These algorithms can be utilized for identifying optimal solutions to optimization problems. For example, the global optimization problems are often seen as synonymous to minimization problems, but these problems can be easily converted to maximization problems by negating the function. In such functions, the optimal solution may be a set of points within the available points in the search space, instead of a single minimum or maximum value. There is a possibility of a multitude of optimal solutions that are dependent on the domain of the search space. The purpose of utilizing any global optimization algorithm is to identify the globally optimal or sub-optimal solutions. The following sub-sections present a detailed discussion on these two swarm algorithms GWO and PSO, respectively.

4.1. Grey Wolf Optimizer (GWO)

Grey wolf optimizer (GWO) is a swarm-based optimization technique proposed by (Mirjalili et al. 2014). GWO mimics the leadership hierarchy of the grey wolves and their hunting mechanism. It belongs to the Canidae family and most of them favor living in a pack. They have a strict social ruling hierarchy; the leader (male or female) is called Alpha (α). The alpha is responsible for making decisions about hunting, sleeping place, time to wake, and so on. The pack should follow the orders of the leader (alpha). The Betas (β) are the second level in the hierarchy, which is the advisor for the alpha in making decisions and discipliner for the pack. The lowest ranking grey wolf is the Omegas (ω), which have to submit to all other dominant wolves. The wolf that is not Alpha or Beta or Omega is called Delta (δ). Delta wolves report to alpha and beta but dominate omegas. Figure 1 shows the hierarchy of the grey wolf; the dominance decreases from top-down.

The grey wolf optimizer mathematical model was designed and implemented by acknowledging the fittest solution as the alpha (α). Consequently, the best second and third solutions are called (β) and delta (δ) respectively. The rest of the candidate solutions are named omega (ω). In the GWO algorithm, the hunting is guided via α, β, and δ. The ω wolves follow the tracking of these three wolves. The wolves encircle the prey during the hunting and update their positions around α, β, or δ based on Eqns. (3) and (4).

Figure 1. Hierarchy of the grey wolf



$$D = \left| \vec{C}.\overrightarrow{X_p}\left(t\right) - \vec{X}\left(t\right) \right| \tag{3}$$

$$\vec{X}\left(t+1\right) = \overrightarrow{X_p}\left(t\right) - \vec{A}\ .\vec{D} \tag{4}$$

where t indicates the current iteration, $\vec{A}$ and $\vec{C}$ are coefficient vectors, $\overrightarrow{X_p}$ is the position vector of prey and $\vec{X}$ indicates position vector of grey wolf. The coefficient vectors $\vec{A}$ and $\vec{C}$ are calculated based on Eqns. (5) and (6).

$$\vec{A} = 2\vec{a}\ .\overrightarrow{r_1} - \vec{a} \tag{5}$$

$$\vec{C} = 2.\overrightarrow{r_2} \tag{6}$$

The components of $\vec{a}$ are linearly decreased from 2 to 0 throughout the iterations and $\overrightarrow{r_1}$, $\overrightarrow{r_2}$ are random vectors in the interval [0, 1]. The grey wolves finish the hunting by attacking the prey when it stops moving. To mathematically model this behavior of approaching the prey, the value of $\vec{a}$ is decreased in each iteration and the fluctuation range of $\vec{A}$ is also reduced by $\vec{a}$. The wolf can relocate itself around the prey with the stated equations (1) and (2) where the random parameters of $\vec{A}$ and $\vec{C}$ let the wolves relocate to any position in the continuous space around the prey. This will make the grey wolves identifying the location of prey and encircle it. The alpha usually leads the hunt. New beta and delta emerge in each iteration as all the other wolves update their positions. We assume that the alpha (best candidate solution), beta, and delta have better knowledge about the potential location of prey. This information is used to simulate the hunting behavior of grey wolves mathematically. Therefore, we save the first three best solutions obtained so far and update the positions of the other search agents including the omegas according to the position of the best search agent. This was performed by proposing and implementing Eqns. (7) to (13).

$$\overrightarrow{\mathrm{D}_{\pm}} = \left|\overrightarrow{\mathrm{C}_1}.\overrightarrow{\mathrm{X}_{\pm}} - \vec{\mathrm{X}}\right| \tag{7}$$

$$\overrightarrow{D_{\beta}} = \left|\overrightarrow{C_2}.\overrightarrow{X_{\beta}} - \vec{X}\right| \tag{8}$$

$$\overrightarrow{\mathrm{D}_{,}} = \left|\overrightarrow{\mathrm{C}_3}.\overrightarrow{\mathrm{X}_{,}} - \vec{\mathrm{X}}\right| \tag{9}$$

$$\overrightarrow{\mathrm{X}_1} = \overrightarrow{\mathrm{X}_{\pm}} - \overrightarrow{\mathrm{A}_1}.\left(\overrightarrow{\mathrm{D}_{\pm}}\right) \tag{10}$$

$$\overrightarrow{\mathrm{X}_2} = \overrightarrow{\mathrm{X}_{2}} - \overrightarrow{\mathrm{A}_2}.\left(\overrightarrow{\mathrm{D}_{2}}\right) \tag{11}$$

$$\overrightarrow{\mathrm{X}_3} = \overrightarrow{\mathrm{X}_{,}} - \overrightarrow{\mathrm{A}_3}.\left(\overrightarrow{\mathrm{D}_{,}}\right) \tag{12}$$

$$\vec{\mathrm{X}}\left(\mathrm{t}+1\right) = \frac{\overrightarrow{\mathrm{X}_1} + \overrightarrow{\mathrm{X}_2} + \overrightarrow{\mathrm{X}_3}}{3} \tag{13}$$

4.2. Particle Swarm Optimization (PSO)

Particle Swarm Optimization (PSO) is a population-based stochastic optimization technique proposed by Eberhart and Kennedy in 1995 (Kennedy and Eberhart, 1995). It is formed on the coordinated behavior of groups such as flocks of birds and schools of fish. Each particle has a virtual position that represents a potential solution to a minimization problem. In the case of a neural network, a particle's position represents the values for the neural network's weights and biases. The goal is to find a position (weights and biases) to minimize the difference between the network's predicted output and the true. PSO employs a set of particles, which is an iterative process (Bataineh et al. 2019b). In every iteration, each particle moves to a new position, hoping that it will be a better solution to the problem. The movement of a particle is based on the current speed and direction (velocity) of the particle, the best position detected by the particle at any moment, and the best position detected by any particle in the swarm (McCaffrey, 2013). The velocity and position updates can be computed based on Eqns. (14) and (15).

$$v\left(t+1\right) = \left(w^{*}v\left(t\right)\right) + (c_1 r_1{}^{*}\left(p\left(t\right) - x\left(t\right)\right) + (c_2 r_2{}^{*}\left(g\left(t\right) - x\left(t\right)\right) \tag{14}$$

$$x\left(t+1\right)=x\left(t\right)+v\left(t+1\right) \quad (15)$$

where $v(t+1)$ is the velocity of a particle at time $t+1$, w is a constant called inertia weight, $v(t)$ is the current velocity at time t, c_1 is a constant called the cognitive (personal) weight, c_2 is a constant called the social or global weight. The r_1 and r_2 are random variables in the range [0, 1), the $p(t)$ vector value is the particle's best position found so far. The $x(t)$ vector value is the particle's current position. The $g(t)$ vector value is the best-known position found by any particle in the swarm so far. Once the new velocity, $v(t+1)$ has been determined, it is used to compute the new particle position $x(t+1)$.

5. FEATURE SELECTION AND TRAINING FFNN USING PSO AND GWO

Feature selection is known as input selection, is one of the main tasks in machine learning. It is the process of identifying the optimal features that are suitable for a classification task and removing less irrelevant and redundant features that do not contribute or decrease the accuracy of the predictive model (Li, 2006) (Bataineh, 2019). By applying feature selection, the number of features is reduced, resulting in dimensionality reduction in data (Al Bataineh, 2020). This can prevent the curse of dimensionality and may actually speed up the classification. Feature selection is formulated as a combinational optimization problem. The objective function to optimize is the generalization performance of the predictive model (e.g., neural network), represented by the error on the selected features of a dataset. However, obtaining the best set of features is a challenging task and has exponential computational growth, especially if there are a large number of features belong to the dataset. For instance, there are 14 features in the EEG eye dataset; therefore, there are almost 2^{14} solutions. So, using a grid or exhaustive selection of features methods to find the optimal features is not an effective approach and impracticable. Therefore, we require intelligent techniques that enable the selection of features in practice. PSO and GWO are some of the most advanced algorithms for feature selection. They are more efficient, and they can guide the feature selection process faster since they do not search for the whole search space (Al-Tashi, 2019).

In order to implement features selection using PSO or GWO, the first step is to create and initialize the individuals in the population. In GWO, these individuals are called gray wolves where in PSO; they are called particles. As these algorithms are stochastic optimization techniques, the individuals' genes are usually initialized at random. For instance, let us consider the proposed neural network with 14 features for the EEG eye state dataset. If we create a population of 10 individuals, we will have 10 different neural networks with random features. Every individual (particle or gray wolf) in the population will be represented by 14 binary genes, as shown in Figure 2. If the gene value is set to 1, then that corresponding feature is included in the neural network model; however, if the gene value is set to 0, then the corresponding feature will be dropped and not include in the neural network. Next, we need to assign a fitness value to each individual in the population. Here, the fitness value is calculated based on some objective function and that is Mean Squared Error (MSE). MSE is the average of the squared differences between the predicted and actual values shown in Eqn. (16).

$$MSE=\frac{1}{n}\sum\nolimits_{i=1}^{n}(Y_i-Y_i^{\sim}) \quad (16)$$

where n is the number of training samples in the dataset, Y_i is the predicted output, and $Y_i^{\sim}$ is the actual output. Minimum MSE implies better fitness.

Note that calculating the fitness value (error) indicates training the neural network with the training data for that corresponding individual with the selected features. For example, if we have 10

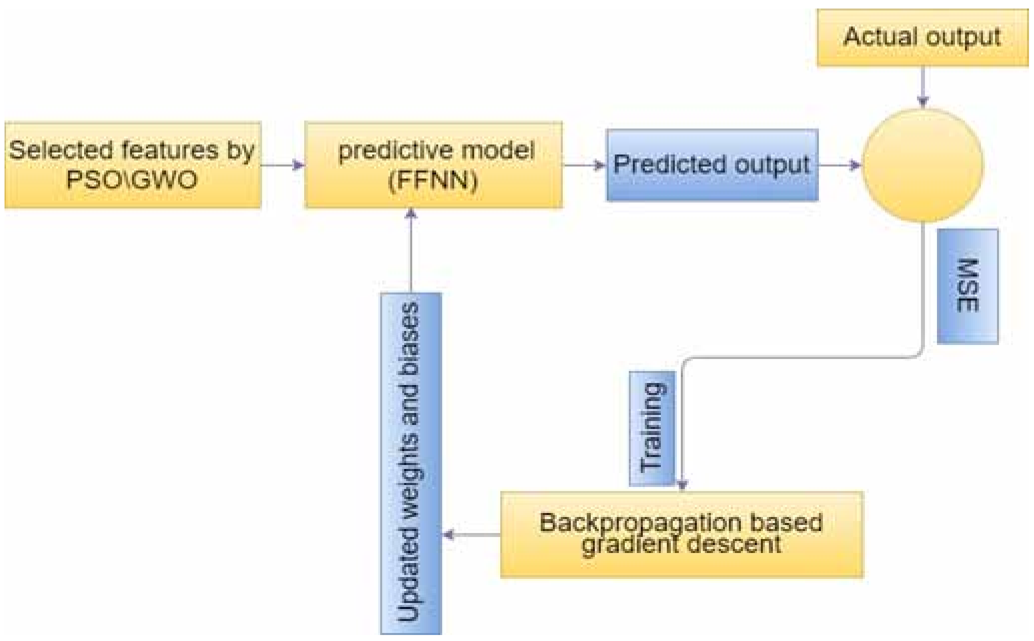
Figure 2. n-dimensional binary mask vector

1	1	1	1	0	1	0	1	1	1	1	0	0	1

particles in the case of PSO, then in each generation (swarm), we will train 10 different neural networks. It is good to mention that in this phase, we are employing a backpropagation algorithm for training the predictive model or the neural network, as shown in Figure 3. Finally, in GWO, the best position or the binary vector obtained (selected features) is represented by $\vec{X}\left(t+1\right)$, whereas, in PSO, the best global position or the binary vector (selected features) is represented by X_{id}. After finding the best features by each algorithm that are suitable for the classification task, the next step is to train the neural network model using PSO and GWO instead of backpropagation to further improve the classifier performance. Each algorithm will train the neural network model with its selected features. Fig. 4 illustrates the complete procedure of the neural network training process with the proposed algorithms.

To use swarm-based algorithms in training FFNNs, we almost followed the same steps applied for the features selection. The only difference is the representation of the problem. In the features

Figure 3. Schematic diagram of GWO/PSO-based FFNN for feature selection

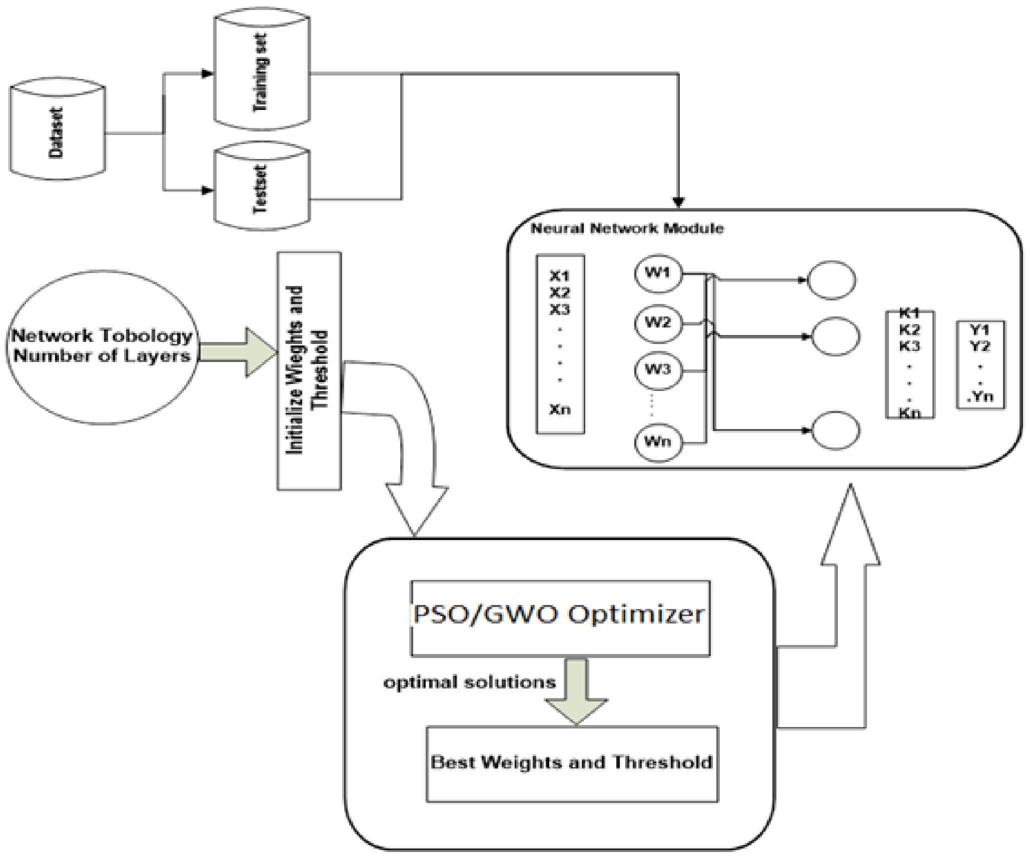

Actual output
Selected features by PSO\GWO
predictive model (FFNN)
Predicted output
MSE
Training
Updated weights and biases
Backpropagation based gradient descent

section problem, we used the binary version of the swarm-based algorithms. However, for training the FFNN model, we will be using the standard versions of these algorithms. The FFNN uses labeled data and a trainer (such as GWO or PSO) to find a set of parameters (weights and bias) values to generate outputs that closely match the known outputs. Here, both GWO and PSO accept the parameters in the form of a vector as shown in Eqn. (17).

$$V = \left\{ w_{11}, w_{12}, \ldots, w_{ij}, b_1, b_2, \ldots, b_j, \ w_{11}, w_{12}, \ldots, w_{jk}, \ b_k \right\} \tag{17}$$

where w_{ij} shows the connection weight from the i_{th} node in the input layer to the j_{th} node in the hidden layer, b_j is the bias of the j_{th} hidden node, w_{jk} is the connection weight from the j_{th} node in the hidden layer to the k_{th} node in the output layer, and b_k is the bias of the k_{th} output node.

Figure 4. The complete procedure of the neural network training process with the PSO/GWO optimizer



The length of the vector is based on the FFNN topology. For instance, the FFNN topology for solving the EEG eye state dataset with 10 selected features is 10-100-1. Therefore, the length vector is (10 * 100) + (100 * 1) + (100 + 1) = 1201 parameter; 1100 elements represent the weights of the FFNN, and the remaining 101 elements represent the biases of the neurons in the FFNN. Here, the vector is simply a particle or grey wolf that represents a candidate neural network with its own weights and bias. The fitness of a given vector is evaluated by just plugging its weights and biases values in the cost function, such as the MSE cost function. In other words, both GWO and PSO algorithms keep iterating for a specific number of generations (iterations) defined by the designer to find a set of weights and biases that minimizes the MSE function. In PSO, the best position vector X_{id} found by the swarm will be the optimal weight and bias parameters of the neural network, whereas, in GWO, the best position $\vec{X}\left(t+1\right)$ will be the optimal weight and bias parameters of the neural network.

6. RESULTS AND DISCUSSION

In order to simulate and validate the proposed work, the EEG eye state dataset by (Rösler and Suendermann, 2013) was adopted and applied. The EEG recording was performed for one person for 117 seconds while the subject was opened and closed their eyes; they were recorded through a video camera. The open/closed state is recorded for each time step in the EEG trace manually. EEG was recorded using Emotiv EEG Neuroheadset, resulting in 14 traces. A total of 14,980 observations were obtained within 117 seconds (128 observations per second). In other words, the dataset contains 14,980 instances. Each instance comprises of 14 features describing the values of the various electrodes (AF3,F7,F3,FC5,T7,P,O1,O2,P8,T8,FC6,F4,F8,AF4) and an eye-state class (either 0 for open, or 1 for closed). Completely closed eyes were categorized as closed; however, open, or partially open eyes were categorized as open. For this research, we have removed a total of 676 instances that are four standard deviations or more from the mean. These are called outliers and removing them helps better understand the relationship between the EEG traces and the state of the eyes open/closed. Therefore, the instances of the dataset correspond to the eye open are now 7,855 (54.92%), and the instances to the eye closed state are 6,449 (45.08%). 80% of the data was used as training and the remaining 20% for testing the model. This work is divided into two major parts; firstly, the feature selection is applied then training of the topology of the FFNN using PSO and GWO, respectively. It is assumed that the optimization process starts with generating random weights and biases in the range of [-5,5]. Other assumptions for the GWO and PSO are given in Table 1. An attempt was made for increasing the population size and the number of iteration for each algorithm; however, no performance improvement was observed.

6.1 Performance Evaluation

Table 1. The initial parameters of the algorithms

Algorithm	Parameter	Value
GWO	A Population Size (grey wolfs) Maximum number of iterations	linearly decreased from 2 to 0 20 200
PSO	Population Size (Swarms) Maximum number of iterations cognitive (or local) weight (C1) social (or global) weight (C2) momentum or inertia weight (w)	20 200 0.5 0.3 0.9

In general, any machine learning approach contains the following steps for evaluation purpose:

- Prepare the dataset to get the data into a suitable format for the predictive model (classifier) and remove any anomalies or outliers. These outliers can negatively impact the predictive model's performance.
- Split the data into training/testing sets.
- Instantiate and train the predictive model on training split; afterward, a prediction is made with the classifier.

- Finally, evaluate the performance, which indicates how accurate the classifier is.

Mainly, there are three types of problems in machine learning: classification, regression, and clustering. Depending on the problem on hand, we apply particular metrics to estimate the model's performance (Al Bataineh, 2021). We have used four different metrics for this research: accuracy, precision, recall, and F1_Score. Before listing the metrics, there are four essential terms to define:

- True Positives (TP): The model predicted positive and the actual output is positive.
- True Negatives (TN): The model predicted negative and the actual output is negative.
- False Positives (FP): The model predicted positive and the actual output is negative.
- False Negatives (FN): The model predicted negative and the actual output is positive.

1. Accuracy: The ratio between the correctly classified instances and the total number of instances based on Eqn. (18).

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \times 100\% \tag{18}$$

2. Precision: It is an excellent indicator of the model performance when the class distribution is imbalanced. The EEG eye state is an imbalanced dataset in which the instances belong to the open state is more frequent than the close state. Mathematically, precision is defined based on Eqn. (19).

$$Precision = \frac{TP}{TP + FP} \times 100\% \tag{19}$$

3. Recall: It is another well-known metric defined as the fraction of samples from a class that is correctly predicted by the model based on Eqn. (20).

$$Recall = \frac{TP}{TP + FN} \times 100\% \tag{20}$$

4. F1-score: It is a metric that combines both precision and recall as there are numerous applications in which precision and recall are necessary. Mathematically, it is the harmonic mean of recall and precision, which can be defined based on Eqn. (21).

$$F1\_Score = 2 * \frac{Precision * Recall}{Precision + Recall} \times 100\% \tag{21}$$

6.2 Experimental Results for PSO and GWO

PSO and GWO were applied for training the proposed neural network with the features that were selected by the algorithm itself in the first phase as discussed in section 5. PSO selected all features except the feature labeled O2 where GWO selected all features except the feature labeled T7. We also use all features for purpose of comparison. The results are shown in Table 2. We can indicate that the neural network performance of detecting the eye state with PSO and GWO as a trainer is better with the feature O2 and T7 being removed. This means that PSO and GWO are a good algorithm for training neural networks and was also able to discover the optimal features that maximize the neural network model's ability to classify patterns. Note that the accuracy achieved with backpropagation on the PSO selected features was about 89%. Note that the accuracy achieved with backpropagation on the GWO selected features was about 89.9%. From a statistical point, the GWO algorithm outperformed the PSO in training the neural network and the selection of the best features that give the best results. The reason for that is the high local optima avoidance of this algorithm.

Table 2. Performance Evaluation of PSO and GWO

Performance Parameter	PSO with all features	PSO with selected features	GWO with all features	GWO with selected features
Accuracy	95.00%	96.40%	95.89%	97.67%
Precision	94.19%	96.86%	95.91%	97.41%
Recall	94.61%	94.84%	94.90%	97.34%
F1_Score	94.40%	95.84%	95.40%	97.37%

6.3 Performance Comparison

This research has also implemented five classical machine learning classifiers for the purpose of comparison. The five algorithms are Support Vector Machine, k-Nearest Neighbors, Decision Tree, Gaussian Naive Bayes, and Logistic Regression. Each algorithm used its default parameters setting in the Sklearn library. Every algorithm was trained with a max number of iterations equal to 300. Data also was split into 80% for training and 20% to test the models. The test accuracy is used to compare the performance in this section. Table 3 shows the results for the five algorithms. The test accuracy is calculated for each classifier that is trained with all features, GWO features, and PSO features. When comparing the results in Table 3 with Table 2, experimental results show that the feed-forward neural network outperforms all other classifiers when the training and features selection is employed using the GWO algorithm. This indicates that neural networks are leading-edge predictive models if proper training is applied. Furthermore, this research shows that GWO is an excellent algorithm to select relevant features and to train neural networks, which will contribute to enhancing the performance of the classifier.

Table 3. Performance Evaluation of the 5 Machine learning Classifiers

Algorithm	Accuracy with all feature	Accuracy with GWO features	Accuracy with PSO features
Support Vector Machine	90.91%	91.22%	90.10%
k-Nearest Neighbors	94.75%	95.63%	94.79%
Decision Tree	81.75%	82.03%	81.92%
Gaussian Naive Bayes	64.17%	64.41%	63.99%
Logistic Regression	60.39%	63.47%	63.36%

7. CONCLUSION

In this study, we have proposed a predictive model represented by a feed-forward neural network (FFNN) to classify the eye state (open or closed) of a person based on EEG signals. First, we proposed two different swarm algorithms via Grey Wolf Optimizer (GWO) and Particle Swarm Optimization (PSO) for finding the most relevant features that can contribute to or increase the accuracy of the neural network model. Second, we investigated training the FFNN model using GWO and PSO instead of backpropagation on the same features selected through each algorithm. Finally, we compared our approach with five different machine learning algorithms, namely, Support Vector Machine, k-Nearest Neighbors, Decision Tree, Gaussian Naive Bayes, and Logistic Regression. Experimental results have shown that the FFNN-GWO trainer with GWO selected features outperforms all algorithms achieving an accuracy of 97.67%, followed by FFNN-PSO trainer with the PSO selected features achieving an accuracy of 96.40%. Therefore, this research concludes that feature selection and training of FFNN for time series classification problems like the EEG eye state problem can obtain better results when features selection and training of the FFNN model are accomplished using GWO algorithm.

8. FUTURE WORK

As illustrated in this article, the findings from the application of neural networks to develop machine learning approaches to EEG analysis and classification showed promising results. The research thereof has the potential to diverge into many potential future directions, including real-time EEG eye state classification and other EEG analysis tasks such as monitoring of mental and cognitive states, which greatly contribute to avoiding hazardous and possibly dangerous conditions in everyday life. Another possible future work is utilizing a popular field of machine learning called deep learning. Deep learning makes use of neural networks that contain several layers. In fact, deep learning has immensely revolutionized machine learning in many possible areas (e.g., computer vision, reinforcement learning, speech recognition.) by providing imprecise and flexible predictive models that can directly work with raw data and learn transformations. Deep learning models use vast amounts of data to learn features directly and capture data structure in an efficient manner that can then be conveyed and/or adapted to various tasks. This comprehensive learning ability lends itself well to the requirements of EEG analysis, in which multiple interrelated processes are fundamental and, until recently, were particularly intended for each distinct task.

REFERENCES

Al Bataineh, A. (2019). A comparative analysis of nonlinear machine learning algorithms for breast cancer detection. *International Journal of Machine Learning and Computing*, *9*(3), 248–254. doi:10.18178/ijmlc.2019.9.3.794

Al Bataineh, A., & Kaur, D. (2021). Immunocomputing-Based Approach for Optimizing the Topologies of LSTM Networks. *IEEE Access: Practical Innovations, Open Solutions*, *9*, 78993–79004. doi:10.1109/ACCESS.2021.3084131

Al Bataineh, A., Mairaj, A., & Kaur, D. (2020). *Autoencoder based semi-supervised anomaly detection in turbofan engines*. Academic Press.

Al-Tashi, Q., Kadir, S. J. A., Rais, H. M., Mirjalili, S., & Alhussian, H. (2019). Binary optimization using hybrid grey wolf optimization for feature selection. *IEEE Access: Practical Innovations, Open Solutions*, *7*, 39496–39508.

Ang, K. K., Chin, Z. Y., Zhang, H., & Guan, C. (2008). Filter bank common spatial pattern (FBCSP) in brain-computer interface. *IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*, 2390–2397.

Bataineh, A., & Kaur, A. (2018a). A comparative study of different curve fitting algorithms in artificial neural network using housing dataset. *NAECON 2018-IEEE National Aerospace and Electronics Conference*, 174–178.

Bataineh, A., & Kaur, A. (2019a). Optimal Convolutional Neural Network Architecture Design Using Clonal Selection Algorithm. *International Journal of Machine Learning and Computing*, *9*(6), 9–9. doi:10.18178/ijmlc.2019.9.6.874

Bataineh, A., Kaur, A., & Jarrah, D. (2018b). Enhancing the parallelization of backpropagation neural network algorithm for implementation on fpga platform. *NAECON 2018-IEEE National Aerospace and Electronics Conference*, 192–196.

Bataineh, A. A., & Kaur, D. (2019). Immuno-Computing-based Neural Learning for Data Classifica- tion. *International Journal of Advanced Computer Science and Applications*, *10*(6). Advance online publication. doi:10.14569/IJACSA.2019.0100632

Bataineh, A. S. A. (2019). A gradient boosting regression based approach for energy consumption prediction in buildings. *Advances in Energy Research*, *6*(2), 91–101.

Blankertz, B., Tomioka, R., Lemm, S., Kawanabe, M., & Muller, K. (2008). Optimizing Spatial filters for Robust EEG Single-Trial Analysis. *IEEE Signal Processing Magazine*, *25*(1), 41–56. doi:10.1109/MSP.2008.4408441

Chambayil, B., Singla, R., & Jha, R. (2010). EEG eye blink classification using neural network. *Proceedings of the World Congress on Engineering*, *1*, 2–5.

Elman, J. L. (1990). Finding Structure in Time. *Cognitive Science*, *14*(2), 179–211.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., & Ozair, S. (2014). Generative adversarial nets. Advances in Neural Information Processing Systems, 2672–2680.

Keneni, B. M., Kaur, D., Al Bataineh, A., Devabhaktuni, V. K., Javaid, A. Y., Zaientz, J. D., & Marinier, R. P. (2019). Evolving rule-based explainable artificial intelligence for unmanned aerial vehicles. *IEEE Access: Practical Innovations, Open Solutions*, *7*, 17001–17016. doi:10.1109/ACCESS.2019.2893141

Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization. *Proceedings of ICNN'95- International Conference on Neural Networks*, *4*, 1942–1948. doi:10.1109/ICNN.1995.488968

Kosmyna, N., & Lécuyer, A. (2019). A conceptual space for EEG-based brain-computer interfaces. *PLoS One*, *14*(1), e0210145–e0210145. doi:10.1371/journal.pone.0210145 PMID:30605482

Lecun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*(11), 2278–2324. doi:10.1109/5.726791

Li, L., Xiao, L., & Chen, L. (2009). Differences of EEG between eyes-open and eyes-closed states based on autoregressive method. *Journal of Electronic Science and Technology*, *7*(2), 175–179.

Li, T. S. (2006). Feature selection for classification by using a GA-based neural network approach. *Journal of the Chinese Institute of Industrial Engineers*, *23*(1), 55–64. doi:10.1080/10170660609508996

Lotte, F., Bougrain, L., Cichocki, A., Clerc, M., Congedo, M., Rakotomamonjy, A., & Yger, F. (2018). A review of classification algorithms for EEG-based brain–computer interfaces: A 10 year update. *Journal of Neural Engineering*, *15*(3), 031005–031005. doi:10.1088/1741-2552/aab2f2 PMID:29488902

Mairaj, A., Bataineh, A., Kaur, D., & Javaid, A. (2019). Identifying the Optimal Solutions of Bo- hachevsky Test Function Using Swarming Algorithms. *Proceedings on the International Conference on Artificial Intelligence (ICAI)*, 109–115.

McCaffrey, J. (2013). Neural Network Training Using Particle Swarm Optimization. *Visual Studio Magazine*. https://visualstudiomagazine.com/articles/2013/12/01/neural-network-training-using-particle-swarm-optimization.aspx

Mirjalili, S. (2015). How effective is the Grey Wolf optimizer in training multi-layer perceptrons. *Applied Intelligence*, *43*(1), 150–161. doi:10.1007/s10489-014-0645-7

Mirjalili, S., Mirjalili, S. M., & Lewis, A. (2014). Grey Wolf Optimizer. *Advances in Engineering Software*, *69*, 46–61. doi:10.1016/j.advengsoft.2013.12.007

Pour, P. A., Gulrez, T., Alzoubi, O., Gargiulo, G., & Calvo, R. A. (2008). Brain-computer inter- face: Next generation thought controlled distributed video game development platform. *2008 IEEE Symposium On Computational Intelligence and Games*, 251–257. doi:10.1109/CIG.2008.5035647

Rivet, B., Souloumiac, A., Attina, V., & Gibert, G. (2009). xDAWN Algorithm to Enhance Evoked Potentials: Application to Brain–Computer Interface. *IEEE Transactions on Biomedical Engineering*, *56*(8), 2035–2043. doi:10.1109/TBME.2009.2012869 PMID:19174332

Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, *65*(6), 386–408. doi:10.1037/h0042519 PMID:13602029

Rösler, O., & Suendermann, D. (2013). A first step towards eye state prediction using EEG. *Proc. of the AIHLS*.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back- propagating errors. *Nature*, *323*(6088), 533–536. doi:10.1038/323533a0

Sadatnezhad, K., Boostani, R., & Ghanizadeh, A. (2011). Classification of BMD and ADHD patients using their EEG signals. *Expert Systems with Applications*, *38*(3), 1956–1963. doi:10.1016/j.eswa.2010.07.128

Sahu, M., Nagwani, N. K., Verma, S., & Shirke, S. (2015). An incremental feature reordering (IFR) algorithm to classify eye state identification using EEG. Information Systems Design and Intelligent Applications, 803–811.

Sulaiman, N., Taib, M. N., Lias, S., Murat, Z. H., Aris, S. A., & Hamid, N. H. A. (2011). Novel methods for stress features identification using EEG signals. *International Journal of Simulation: Systems*, *Science and Technology*, *12*(1), 27–33.

Wang, T., Guan, S. U., Man, K. L., & Ting, T. O. (2014). Time series classification for EEG eye state identification based on incremental attribute learning. *2014 International Symposium on Computer, Consumer and Control*, 158–161. doi:10.1109/IS3C.2014.52

Whitley, D. (1994). A genetic algorithm tutorial. *Statistics and Computing*, *4*(2), 65–85. doi:10.1007/BF00175354

Ali Al Bataineh earned a B.Sc. degree in computer engineering from Yarmouk University, Jordan, in 2010, and an M.Sc. degree in computer engineering from the University of Bridgeport, CT, USA, in 2016, and a Ph.D. degree in electrical and computer engineering from the University of Toledo, OH, USA, in 2021. He is currently an Assistant Professor in the Electrical and Computer Engineering Department, Norwich University, VT, USA. His current research interests include the areas of artificial intelligence, machine learning, metaheuristic optimization, fuzzy logic, embedded systems, and FPGAs.

Amin Jarrah received his MSc and PhD in Computer Engineering from the Jordan University of Science and Technology and The University of Toledo, USA, in 2010 and 2014, respectively. During 2014–2016, he was appointed as an Embedded Software Engineer in Magna Electronics and TATA group companies. In January 2016, he joined the Yarmouk University as an Assistant Professor. Currently, he is an Associate Professor at the Department of Computer Engineering at Yarmouk University, Irbid, Jordan. His research is in the area of real-time embedded hardware implementation, high performance computing, parallel processing using FPGAs, system on chip (SoC), and GPU.