Med Biol Eng Comput (2014) 52:131–139
DOI 10.1007/s11517-013-1123-9

ORIGINAL ARTICLE

Automatic feature selection of motor imagery EEG signals using differential evolution and learning automata

Saugat Bhattacharyya · Abhronil Sengupta · Tathagatha Chakraborti · Amit Konar · D. N. Tibarewala

Received: 2 April 2013 / Accepted: 18 October 2013 / Published online: 29 October 2013

Abstract Brain–computer interfacing (BCI) has been the most researched technology in neuroprosthesis in the last two decades. Feature extractors and classifiers play an important role in BCI research for the generation of suitable control signals to drive an assistive device. Due to the high dimensionality of feature vectors in practical BCI systems, implantation of efficient feature selection algorithms has been an integral area of research in the past decade. This article proposes an efficient feature selection technique, realized by means of an evolutionary algorithm, which attempts to overcome some of the shortcomings of several state-of-the-art approaches in this field. The outlined scheme produces a subset of salient features which improves the classification accuracy while maintaining a trade-off with the computational speed of the complete scheme. For this purpose, an efficient memetic algorithm has also been proposed for the optimization purpose. Extensive experimental validations have been conducted on two real-world datasets to establish the efficacy of our approach. We have compared our approach to existing algorithms and have established the superiority of our algorithm to the rest.

Keywords Brain–computer interfacing · Feature selection · Motor imagery · Memetic algorithm · Differential evolution · Learning automata · Power spectral density

S. Bhattacharyya (✉) · A. Sengupta · T. Chakraborti · A. Konar
Department of Electronics and Telecommunication Engineering, Jadavpur University, Kolkata 700032, India
e-mail: saugatbhattacharyya@gmail.com

D. N. Tibarewala
School of Bioscience and Engineering, Jadavpur University, Kolkata 700032, India

1 Introduction

It is well known that intentions for any actions performed by a person originate from the brain [23, 24, 26]. Brain–computer interfacing (BCI) extracts, decodes and translates these intentions into control commands to drive an external device for rehabilitative applications [3, 19, 29]. Other areas of application of BCI include robotics, communication, gaming and virtual reality [13, 15, 16, 32, 44]. *Motor imagery* (movement based) brain signal [12] is one of the most frequently researched fields in BCI. For the acquisition of brain signals, both invasive and non-invasive means have been employed in BCI research, which includes electroencephalography (EEG), magnetoencephalography (MEG), electrocorticography (ECoG), intracortical electrodes and functional magnetic resonance imaging (fMRI) [28]. Among these, the EEG is preferred because it is non-invasive, easily available and portable and has very good temporal resolution [12, 32]. The EEG signals during motor imagery experiments are acquired from C3 and C4 electrode locations [27] (based on the 10–20 electrode system) because they are directly placed above the motor cortex areas of the brain [40].

A general EEG-BCI module contains the following stages: preprocessing of the raw EEG, feature extraction from the EEG and classification of the EEG [22]. Here, time, frequency, time–frequency and nonlinear signal processing methods are employed for feature extraction [12, 32, 37–39] along with linear and nonlinear methods as the classifiers [2, 31]. Sometimes, the features extracted from the EEG by the BCI have high dimensionality [21, 31],

which may result in two major drawbacks: (a) increase in the computational time of the classifier and (b) EEG signals have poor signal-to-noise ratio [31] and are susceptible to the inclusion of features which behave as outliers and therefore reduce the classification accuracy. Thus, during the past few decades, researchers have included a feature selection stage before the classification stage [22]. This stage selects a subset of features from the original feature set having an enhanced discriminative power [39]. Some commonly used feature selection algorithms include sequential forward floating search [14], sequential forward search [20], principal component analysis [1], singular value decomposition [17], independent component analysis [7], curvilinear component analysis, kernel principal component analysis [33]. Existing approaches in feature selection suffer from few major drawbacks, which are as follows: (a) Sometimes, it is seen that even if the variances are good among components, they still have low classification performance. It may be due to the fact that the concerned algorithm failed to remove the redundant features. Determination and removal of redundant features is not possible simply by inspection of the feature set.

(b) Many of the popular feature extraction techniques perform a linear transformation of the original feature set to a vector of low dimensionality for consideration in the classifier stage. (c) The optimal number of reduced features to be considered in the classifier stage after dimension reduction is determined by cumbersome experimental validations. The reduced features in most cases are a linear transformation of the original feature set. Thus, even if the feature set used in the classifier stage is reduced, we must still measure the original features. Here, we have solved the above problems by designing an algorithm to choose an optimal set of features from the original feature set itself. So all the features are not employed in the classification stage. Here, cumbersome experimental validations are avoided and a simple run of the optimizer is sufficient, which also optimizes the classifier performance.

The main contribution of this paper lies in the usage of evolutionary approach to the feature selection module, whose best d features are selected from the total feature vector $\mathbf{D}$, where d lies between (0, $\mathbf{D}$]. The algorithm is based on a population of members, which are represented by two components, namely the *Activation Thresholds* and the *Scale Factors*. *Activation Thresholds* determine the corresponding features to be selected, while the *Scale Factors* determine the amount by which a particular feature has to be scaled. On the basis of these two representations, each population member undergoes the evolution process to produce fitter (with respect to the cost function) individuals. Another significant contribution of the paper is the application of our proposed memetic algorithm [34] to optimize the cost function. Memetic algorithms (MA) are population-based search heuristics that integrate the mutual benefits of natural and cultural evolution. Evolutionary algorithms (considering only genetic evolution) often get trapped in local minima. Cultural evolution, on the other hand (on integration with genetic evolution), has proved to be more robust in this respect. This paper employs an intelligent optimization method which uses differential evolution (DE) [30, 36] as the genetic evolution tool and learning automata (LA) [25] for realizing the cultural evolution phase on two different motor imagery EEG datasets. Here, we have employed power spectral estimates for feature extraction and support vector machine for classification.

The rest of the paper is organized as follows. Section 1 describes our proposed framework of feature selection using DE and LA. Section 3 deals with the experiments undertaken to study the performance of our proposed approach. In Sect. 4, we compare our approach with other feature selection approaches. The concluding remarks are given in Sect. 5.

2 The proposed approach

The proposed approach aims to reduce the dimensions of a feature vector based on a synergistic operation between an evolutionary algorithm and a supervised learning classifier. Here, a number of trial vectors are formed with different number of features for the same dataset, which constructs a pseudofeature vector from the original dataset such that each data point consists only of the selected features. Precision of each possible combination of selected features is quantitatively evaluated with the classification accuracy obtained by testing the learning classifier. Then, through the mechanism of mutation and natural selection, eventually, the best solutions start dominating the population, whereas the bad ones are eliminated. Ultimately, the evolution of solutions converges when the fittest solution represents a near-optimal partitioning of the dataset with respect to the employed validity index. In this way, the optimal number of features is selected using our proposed framework. Here, we have employed the use of LA-DE algorithm [34] and SVM for this purpose.

2.1 The LA-DE algorithm

In the proposed framework, the global search mechanism is accomplished by successive generations of DE, while the optimal control parameters for individual members of the population in every generation are provided from a meme pool [43], for given scaling parameter F, which is maintained through the generations. A *meme* is defined as a unit of cultural information [43]. The meme selection process

[43] is controlled by the state transition probability matrix $S_{i,j}$, where the row indices represent the states of the stochastic automata [25] and the column indices represent the actions performed by the automata at a particular state. The rows correspond to the population members ranked in the order of their decreasing *fitness* values, and the columns correspond to uniform quantized values of the control parameter (i.e., F) in a given range, say (0, 2]. In an evolutionary algorithm framework, "*fitness*" signifies the performance of a population member with respect to a cost function (see Sect. 2.3). The principles used in designing the LA-DE algorithm are outlined below.

2.1.1 Initialization

LA-DE algorithm starts with a population of NP D-dimensional parameter vectors, each population vector representing a possible solution vector $\vec{Z}$. The population members are initialized according to a uniform random distribution along every dimension, within the prescribed minimum and maximum bounds: $\vec{Z}_{\min} = \{z_{\min-1}, z_{\min-2}, \ldots, z_{\min-D}\}$ and $\vec{Z}_{\max} = \{z_{\max-1}, z_{\max-2}, \ldots, z_{\max-D}\}$. This ensures that for a reasonable number of vectors, the initial population covers the entire search space uniformly. Hence, we may initialize the jth component of the jth vector at generation $t = 0$ as

$$z_{i,j}(t) = z_{j-\min} + rand_{i,j}(0,1) \times (z_{j-\max} - z_{j-\min}) \tag{1}$$

The state transition probability matrix is initialized with equal values of 0.05 for 20 quantized levels of the parameter F (F_1, $F_2, \ldots, F_{20}$). This is in accordance with the principle of unavailability of a priori information about the environment and assuming all actions to be equally likely at the initial stage.

2.1.2 Adaptive selection of meme

The next step involves the selection of F_j from the meme pool (F_1, $F_2, \ldots, F_{20}$) (for a member of state S_i) such that the cumulative probability of selection of $F = F_j$ through F_{j-1} is greater than a random number r in the range [0, 1] for each population member, i.e.,

$$\sum_{\mathrm{m}=1}^{j-1} p_{s_{i,\mathrm{m}}} < r \leq \sum_{\mathrm{m}=j}^{20} p_{s_{i,\mathrm{m}}} \tag{2}$$

where $p_{s_{i,j}}$is the state transition probability vector at state S_i. Here, Roulette wheel selection [25] has been used for the selection of potentially useful memes. This entails that fitter memes would have higher probabilities of selection, but the memes with poorer fitness also manage to survive and contribute some components in the course of evolution. Thus, this selection mechanism ensures that the diversity of the meme population is effectively maintained.

2.1.3 Differential evolution

Mutation (DE/current-to-best/1) First, DE generates a donor vector $\vec{V}_i(t)$ corresponding to each population member $\vec{Z}_i(t)$ by randomly selecting two other members $\vec{Z}_{rand-1}(t)$ and $\vec{Z}_{rand-2}(t)$, where

$$\vec{V}_i(t) = \vec{Z}_i(t) + F(\vec{Z}_{\text{best}}(t) - \vec{Z}_i(t)) + F(Z_{rand-1}(t) - \vec{Z}_{rand-2}(t)) \tag{3}$$

and F is the scaling factor (selected from the meme pool adaptively), which is used for linear scaling of the difference vectors during the mutation operation and $\vec{Z}_{\text{best}}(t)$is the population member with the best fitness.

Crossover Following the generation of the donor vectors, crossover operation is performed to increase the potential diversity of the population. There are two types of crossover (recombination) schemes: binomial and exponential [6, 9]. In the proposed realization, we have used binomial crossover, where the D components are changed whenever a randomly generated number, in the range [0, 1] following a binomial distribution, is less than or equal to the crossover ratio. Here, a trial vector $\vec{U}_i(t)$ is generated for each pair of $\vec{V}_i(t)$ and $\vec{Z}_i(t)$ by (4).

$$u_{i,j}(t) = \begin{cases} v_{i,j}(t), & \text{if } rand_{i,j} \leq Cr \text{ or } j = j_{rand} (j_{rand} \in [1, D]) \\ z_{i,j}(t), & \text{otherwise} \end{cases} \tag{4}$$

where $rand_{i,j}(0,1)$is a uniformly distributed random number lying in [0, 1] and Cr is the crossover ratio.

Selection The next step of DE decides whether the trial vector $\vec{U}_i(t)$or the target vector $\vec{Z}_i(t)$ is selected for the next generation, according to their fitness. The selection process is

$$\vec{Z}_i(t+1) = \vec{U}_i(t) \text{ if } f(\vec{U}_i(t)) \leq f(\vec{Z}_i(t)) = \vec{Z}_i(t) \text{ if } f(\vec{U}_i(t)) > f(\vec{Z}_i(t)) \tag{5}$$

where $f(\vec{x})$is the fitness.

2.1.4 Update of state transition probability matrix

Let a member at state S_{m} on selection of F_j from the meme pool produce a trial vector after mutation and crossover. If the fitness of the trial vector increases, then the state transition probabilities are updated according to (6); otherwise, it is updated according to (7). Here, the linear reinforcement scheme [34] is employed for the updation process, which is *If* $f(\vec{x}_t) > f(\vec{x}_{t-1})$,

$$\begin{cases} p_j(t+1) &= (1-a).p_j(t) \quad \forall j \neq i \text{ for state } S_m \\ p_i(t+1) &= p_i(t) + a.(1-p_i(t)) \end{cases} \tag{6}$$

Otherwise,

$$\begin{cases} p_j(t+1) &= \frac{b}{e-1} + (1-b).p_j(t) \quad \forall j \neq i \\ p_i(t+1) &= (1-b).p_i(t) \end{cases} \tag{7}$$

where $a \in [0, 1]$ is the reward response, $b \in [0, 1]$ is the penalty response, and e is the number of actions of the automata process.

2.1.5 State assignment

Following the updation of the state transition probability matrix, we sort the population members in decreasing order of their fitness and assign their corresponding states.

2.1.6 Convergence

After each evolution, we repeat from step 2 (see Sect. 2.4) until the termination conditions are satisfied. The algorithm is stopped if the maximum number of generations (gen_max) is reached or the cost function falls below a predefined level.

The overall schematic of the proposed LA-DE algorithm for feature selection is shown in Fig. 1.

2.2 Solution representation and fitness evaluation

In order to judge the quality of the proposed feature selection method, we partition the entire dataset into three mutually exclusive partitions $\varphi 1$, $\varphi 2$ and $\varphi 3$ for training, validation and testing, respectively. $\varphi 1$ is employed to train the classifier for each population member, and $\varphi 2$ is used for fitness function calculation. Finally, the optimal set of selected features, obtained as a result of optimization, is tested on $\varphi 3$. For this purpose, k-fold cross-validation technique [2] is employed and the features are normalized in the range [0, 10] before partitioning to reduce ambiguity. If CA is the classification accuracy for the validation set $\varphi 2$, then the fitness of the jth population member will be given by

$$f_j = 1/CA \tag{8}$$

Thus, minimization of this cost equation ensures population members with better validation-stage classification accuracy to be selected during the evolution phase.

For the total set of D features, we represent each member of the population (of NP members) participating in evolution as a 2D vector $P_j = (w_{1j}, \ldots, w_{2pj})$ where the ith component of the vector, w_{ij}, $(w_{ij} \in [0, 1], i = 1, 2, \ldots, D; j = 1, 2, \ldots, \text{NP})$ represents the *Activation Thresholds* for the respective features. We state that if a particular component is greater than 0.5, then the corresponding feature *feat* is considered for validation and testing in the classifier stage. On the other hand, the ith component of the vector $w_{ij}(w_{ij} \in [0, 1], i = D+1, D+2, \ldots, 2D; j = 1, 2, \ldots, N)$ represents the *Scaling Factors* for the respective features. *Activation Thresholds* and *Scaling Factors* determine which features are to be included in the final set S_j of optimal features. The selection of the jth feature and jth population member is made as follows:

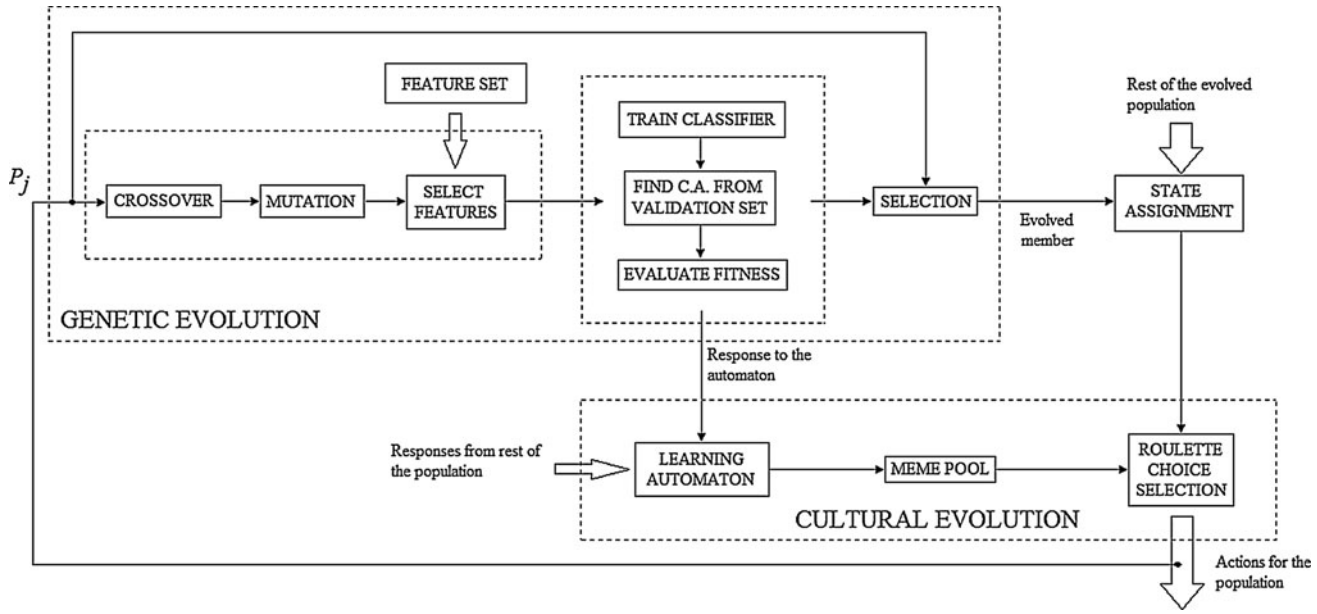

Fig. 1 The proposed feature selection scheme

$$\begin{aligned} &If\ (w_{ij} > 0.5) \\ &\text{feat}_i \leftarrow w_{i+D_j} \times \text{feat}_i \\ &\text{feat}_i \in S_j \\ &\text{else} \\ &\text{ignore feat}_i \end{aligned} \tag{9}$$

Thus, an *Activation Weight* of greater than 0.5 selects a feature for training and validation, and in that case, the feature is multiplied by the corresponding *Scaling Factor*. This is done to enhance the discriminating power of the classifier. Linear scaling of features before classification enables the classifier to discriminate more efficiently along the feature axis with larger scale factors. In this way, the weights not only determine which feature to be selected but also the importance of the selected features. As an example, we consider the weight vector P_j of the jth population member in Fig. 2. Taking $D = 6$, i.e., a total of 6 features, the values of the weights shown indicate that the first, second and third features are to be selected for training and validation after multiplication with appropriate scaling factors, while the rest are ignored. Thus, the set of selected features for the jth population member becomes $S_j \equiv \{0.68 \times f_1,\ 0.76 \times f_2,\ 0.44 \times f_3\}$.

2.3 Final feature selection

The optimization continues until a stipulated number of generations have reached or an acceptable rate of classification is obtained. After termination, the population member with the best fitness is selected for testing $\varphi 3$. The weights of this vector determine the final selection of features. Once again, the chosen features are subjected to linear scaling by their corresponding weights. This ensures that throughout the process of training, validation and testing, the features are multiplied by the same scaling factors, and hence, there is no scope of any misrepresentation of data.

2.4 Pseudocode

The pseudocode for the complete algorithm is given here.

- Step I: Initialize a set of NP vectors each with $2D$ components initialized randomly between 0 and 1.
- Step II: Select features according to rule (8) for every population member.
- Step III: Train classifier on $\varphi 1$ with selected feature set.
- Step IV: Validate on $\varphi 2$ and calculate fitness of population members.
- Step V: Update population members according to the evolutionary algorithm guided by the fitness values calculated above.
- Step VI: if $\text{gen} < \text{gen}_{\text{MAX}}$, goto Step II, else select member with the best fitness to get the final set of features. Here, gen_{MAX}denotes the maximum number of generations.

3 Experiments and results

This section provides details on the experiments undertaken to examine the performance of our proposed feature selection algorithm. For this purpose, the power spectral density (PSD) estimates of two separate motor imagery datasets are employed as features. These features are fed to our proposed LA-DE feature selection algorithm to reduce the dimensions of the feature vector. Support vector machines (SVM) are employed to optimize the results of the LA-DE and validate its performance from the recognition accuracy of the given EEG datasets.

3.1 Dataset I: dataset IVa from BCI competition III

This dataset comprises EEG recordings of five healthy subjects (namely, *aa*, *al*, *av*, *aw* and *ay*) with a sampling frequency of 100 Hz. Details on the motor imagery experiments performed by the subjects are given in [12]. As indicated in [12], the visual cues for the two motor imageries the subject should perform: right hand (Class 1) and right foot (Class 2) are time-locked within 3.5 s. In this study, the signals obtained from C3 and C4 electrodes are filtered using Laplace filters [12] to reduce the effects from the neighboring electrodes.

3.2 Dataset II: experimental data

This dataset was constructed from the EEG acquired from nine subjects during a left–right motor imagery experiment conducted at our laboratory. EEG signals were acquired using a 19

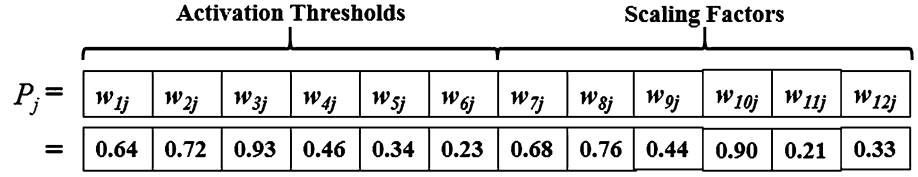

Fig. 2 An example of the vector representation of a particular population member for a dataset with 6 features

channel NeuroWin (manufactured by NASAN) amplifier, with a sampling frequency of 250 Hz. Signals from C3 and C4 electrodes were selected to extract relevant information on the different movement. Prior to the start of the experiment, the subjects were given a small introduction about the research work and stages of the experiment involved. The subjects performed the experiments on a single day, consisting of 3 separate sessions with 15-min relaxation in between.

The subjects perform the motor imagery tasks based on the visual stimuli shown to them during each experimental session. Each session comprises 30 trials; thus, for every subject, a total of 90 trials are obtained. The subjects imagine moving their right and the left hand when right and left arrows are displayed on screen. In each session, a blank screen was displayed in the first 10 s. In the tenth second, a fixation cross "+" was displayed on the screen, which indicates the beginning of a trial. From the twelfth second onwards, the visual stimuli are displayed for three seconds to indicate which arm to move. Next, a blank screen is shown for 2 s during which the subject can relax. This stage also reduces the effect of the previous motor imagery performed by the subject on the current one. Figure 3 gives a generic structure of the visual stimuli.

First, the acquired EEG is band-pass-filtered between 8 and 35 Hz using an IIR elliptical filter of order 14, to remove the noise acquired from the amplifier and the environment. The elliptical filter was selected because the filter has a sharper roll-off as compared to the other filters, requires a small filter order and can independently adjust both the passband and stop-band ripples, as per the user's wish. The passband attenuation and stop-band attenuation are experimentally determined to be 1 and 50 dB, respectively. Then from each trial, the average of the two seconds of data acquired during the fixation cross period is subtracted from the three seconds of the movement stimuli data to remove the effect of background EEG from the motor imagery data. Finally, three seconds of the motor imagery data from each trial is selected for feature extraction.

3.3 Feature extraction: power spectral density estimates

Spectrum estimation describes the power distribution contained in a signal over frequency based on a finite set of data. For our study, we have used the Welch's periodogram for the spectral estimation of the EEG data. Here, the data segments are overlapped and windowed prior to the calculation of the periodogram. The overlapping and windowing of the data segments leads to a decrease in the variance and more control over the bias/resolution properties of the calculated PSD, respectively. These characteristics make this method highly suitable for the analysis of a non-stationary signal [5].

For our study, we have prepared the original feature vector from PSD estimates using the Welch's method. Here, a Hamming window of size 125 and 50 for datasets I and II, respectively, and 50 % overlap was used to obtain the frequency distribution of the extracted filtered EEG over 128 frequency points, for both the electrodes C3 and C4. Thus, the total size of the feature vector is 128 features $\times 2$ electrodes.

3.4 Feature selection and classification results

The features extracted from the previous section are fed as inputs to our proposed feature selection. According to our algorithm, the best minimum number of features is selected, which would yield the best result, i.e., classification accuracy. Thus, the classifiers used in this study have two functions: First, it is used to optimize the LA-DE feature selection, and second, it is used to validate the selection of the features on a test dataset. For this purpose, we have selected support vector machines (SVM) [35, 42] with linear kernel as the classifier. SVM has earned popularity in recent years because it has good recognition ability at high computational speed as compared to other standard classifiers.

Experiments undertaken further reveal that the parameters in the LA-DE algorithm that give the best performance are as follows: population size = 50, scaling factor = 0.5, crossover ratio = 0.9, maximum number of generations = 10,000, stopping criteria = 10 number of same fitness and reward/penalty rate = 0.01. Tables 1 and 2 give the recognition accuracy of the classifiers before and after feature selection for both datasets I and II, respectively. For this purpose, the dataset is partitioned into training set and test set using k-fold cross-validation technique [41]. Here, k is selected as 10 and the accuracies

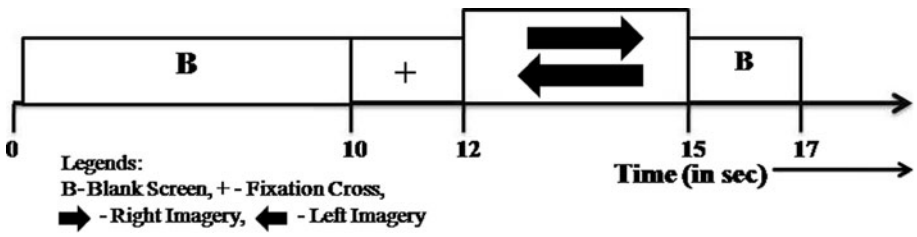

Fig. 3 Timing scheme diagram of the visual cue

Table 1 Classification accuracies (in %) for dataset I (S.D.—standard deviation)

Subjects	aa	al	av	aw	ay	Mean	S.D.
SVM	88.24	95.45	77.78	100	100	92.29	9.43
LA-DE-SVM	97.06	100	100	100	100	99.41	1.31

for each subject are measured over 10 runs. As observed from Tables 1 and 2, the accuracies of the dataset after employing the LA-DE algorithm have significantly improved.

4 Performance analysis

In this section, we examine the performance of the LA-DE algorithm with the following competitor algorithms: particle swarm optimization (PSO) [18], genetic algorithm (GA) [8] and kernelized principal component analysis (kPCA) [4] using linear SVM as the classifier. The parameters selected for PSO and GA are similar to the one selected for LA-DE in Sect. 3.4. For kPCA, we have reduced the dimensions of the algorithm to half of its original length, i.e., 128. The performance of the algorithms is measured by the following parameters: (1) recognition accuracy (Acc.), (2) computational time (C.T) and (3) features selected (F.S). They are defined below for ready reference.

Recognition Accuracy (Acc.): It is the ratio of the number of test data correctly classified by the trained classifier to the total number of test data. It is expressed in percentage (%).

Computational Time (C.T): It is the total time taken by the feature selection algorithm to produce the best result. It is expressed in seconds (sec).

Features Selected (F.S): It is the number of features selected after using the feature selection algorithms on the original datasets.

Each performance measure was calculated on MATLAB 7.9 environment run on a computer with the following specifications: Intel Core 2 Duo 1.19 GHz, 3.2-GB RAM and Windows platform. Tables 3 and 4 provide the results for the comparison of LA-DE with its competitors. As observed from Table 3, LA-DE yields the best result in terms of accuracy, but PSO gives the best result in terms of features selected. But from Table 4, it is observed that LA-DE yields the best performance in terms of both accuracy and features selected. From both the tables, it is noted that kPCA requires the minimum amount of time, but the accuracy obtained is very poor. Maintaining a trade-off between the accuracy, computational time and features selected, it is noted from Tables 3 and 4 that LA-DE gives the best optimal result.

The performance measures of LA-DE are further validated by means of Friedman's test [10, 11] performed on both datasets I and II. The null hypothesis here states that all the algorithms are equivalent, so their ranks R_m should be equal. The Friedman statistic

$$\chi_F^2 = \frac{12N}{k(k+1)}\left[\sum_{m=1}^{k} R_m^2 - \frac{k(k+1)^2}{4}\right] \tag{14}$$

is distributed accordingly to χ_F^2 with $K-1$ degrees of freedom, where k is the number of algorithms to be compared and N is the number of parameters used for comparison. In this study, we have selected the mean of all the three parameters for both the datasets; thus, $k = 4$ and $N = 6$, and Table 5 is the ranking table prepared from Tables 3 and 4.

From Table 5, we calculate the value of R_j, which is further used in Eq. (14) to get $\chi_F^2 = 22.5 > \chi_{4,0.095}^2 = 9.488$. So, the null hypothesis, claiming that all the algorithms are equivalent, is wrong, and therefore, the performances of the algorithms are determined by their ranks only. It is clear from the table that the rank of LA-DE is 1.5, claiming that LA-DE yields better results than its competitors.

5 Conclusion

The paper proposes a novel feature selection technique based on differential evolution and learning automata. Experiments have been performed on two datasets using power spectral density for feature extraction and SVM as the pattern classifier. Comparisons have been performed with GA, PSO and kPCA, and the results indicate that the proposed approach yields better result. A major advantage of the proposed algorithm is that the optimal number of relevant features is a direct outcome of the algorithm and does not have to be experimentally determined, which

Table 2 Classification accuracies (in %) for dataset II (S.D.—standard deviation)

Subjects	1	2	3	4	5	6	7	8	9	Mean	S.D.
SVM	71.63	62.80	67.12	54.55	49.89	62.16	66.87	69.99	34.10	59.90	11.98
LA-DE-SVM	100	88.89	87.50	88.89	77.78	88.89	100	90.00	70.00	87.99	9.51

Table 3 Comparison of the performance measures of LA-DE with other algorithms for dataset I

Subjects	LA-DE			PSO			GA			kPCA		
	Acc.	C.T.	F.S.	Acc.	C.T.	F.S.	Acc.	C.T.	F.S.	Acc.	C.T.	F.S.
aa	97.06	312	134	94.11	454	112	90.29	512	134	82.35	121	128
al	100	305	119	100	472	125	95.55	520	145	97.78	120	128
av	100	325	131	94.45	472	123	92.09	534	156	66.67	120	128
aw	100	305	117	100	472	122	100	520	144	100	121	128
ay	100	311	128	100	472	120	100	509	130	100	120	128
Mean	**99.41**	**311.6**	**126**	**97.71**	**468.4**	**120**	**95.59**	**519**	**142**	**89.36**	**120.4**	**128**

Average of the performance measures shown in bold

Table 4 Comparison of the performance measures of LA-DE with other algorithms for dataset II

Subjects	LA-DE			PSO			GA			kPCA		
	Acc.	C.T.	F.S	Acc.	C.T	F.S	Acc.	C.T.	F.S	Acc.	C.T	F.S
1	100	234	129	100	344	127	99.8	370	130	78.89	75	128
2	88.89	225	123	87.50	350	132	85.55	370	130	60.00	76	128
3	87.50	234	134	75.00	350	130	80.00	378	135	70.00	75	128
4	88.89	219	114	80.00	344	127	80.00	376	134	65.12	75	128
5	77.78	220	119	77.78	356	138	77.78	378	135	66.67	77	128
6	88.89	220	121	87.50	360	143	85.00	379	137	66.67	75	128
7	100	226	127	88.89	321	109	90.00	361	121	72.22	74	128
8	90.00	225	123	88.89	344	127	83.33	384	143	66.67	73	128
9	70.00	228	126	66.67	363	152	60.00	390	150	60.90	75	128
Mean	**87.99**	**225.67**	**124**	**83.58**	**348**	**132**	**82.38**	**376.22**	**135**	**67.46**	**75**	**128**

Average of the performance measures shown in bold

Table 5 Ranking table for performance comparison using Friedman's test

		LA-DE	PSO	GA	kPCA
Dataset I	Acc	1	2	3	4
	C.T	2	3	4	1
	F.S	2	1	4	3
Dataset II	Acc	1	2	3	4
	C.T	2	3	4	1
	F.S	1	2	4	3
Avg. Rank		1.5	2	4	3

saves the time of the user. Also, this algorithm allows the user to employ other classifiers in place of the ones discussed in this paper. The only disadvantage of this algorithm is that the performance of the optimizer may degrade due to stagnation. However, this can be easily overcome by running the optimizer a number of times (say 10) for the same cost function and then taking the best results. This does not affect the computational time as the optimizer works in an offline environment. Further study in this direction aims to optimize the feature extraction and classification techniques to be implemented in the online classification of the EEG data for BCI research and thus to ultimately develop a complete stand-alone system for an EEG-driven neuroprosthetic control for rehabilitation purpose.

Acknowledgments I would like to thank University Grants Commission, India; University of Potential Excellence Programme (Phase II) in Cognitive Science; Jadavpur University; and Council of Scientific and Industrial Research, India.

References

1. Abdi H, Williams LJ (2010) Principal component analysis. Wiley Interdiscip Rev Comput Stat 2:433–459
2. Alpaydin E (2004) Introduction to machine learning (adaptive computation and machine learning). The MIT Press, Cambridge, MA
3. Birbaumer N, Murguialday AR, Cohen L (2008) Brain–computer interface in paralysis. Curr Opin Neurol 21(6):634–638
4. Cao LJ, Chua KS, Chong WK, Lee HP, Gu QM (2003) A comparison of PCA, KPCA and ICA for dimensionality reduction in support vector machine. Neurocomputing 55(1–2):321–336
5. Childers DG (1978) Modern spectrum analysis. IEEE Press, New York

6. Chou C-H, Su M-C, Lai E (2004) A new cluster validity measure and its application to image compression. Pattern Anal Appl 7(2):205–220
7. Comon P (1994) Independent component analysis, a new concept? Signal Process 36(3):287–314
8. Corralejo R, Hornero R, Alvarez D (2011) Feature selection using a genetic algorithm in a motor imagery-based Brain Computer Interface. In: Proceedings of the 2011 annual international conference of the IEEE engineering in medicine and biology society, EMBC, Boston, MA, pp 7703–7706. doi:10.1109/IEMBS.2011.6091898
9. Das S, Abraham A, Konar A (2008) Automatic clustering using an improved differential evolution algorithm. IEEE Trans SMC Part A Syst Hum 38(1):218–237
10. Demšar J (2006) Statistical comparisons of classifiers over multiple data sets. J Mach Learn Res 7:1–30
11. Dietterich TG (1998) Approximate statistical tests for comparing supervised classification learning algorithms. Neural Comput 10(7):1895–1923
12. Dornhege G (2007) Toward brain–computer interfacing. MIT Press, Cambridge, MA
13. Dornhege G, Blankertz B, Curio G, Muller K (2004) Boosting bit rates in noninvasive EEG single-trial classifications by feature combination and multiclass paradigms. IEEE Trans Biomed Eng 51(6):993–1002
14. Dyson M, Sepulveda F, Gan JQ (2008) Mental task classification against the idle state: a preliminary investigation. In: Proceedings of the 30th annual international conference of the IEEE engineering in medicine and biology society, Vancouver, BC, pp 4473–4477. doi:10.1109/IEMBS.2008.4650206
15. Finke A, Knoblauch A, Koesling H, Ritter H (2011) A hybrid brain interface for a humanoid robot assistant. In: Proceedings of the 2011 annual international conference of the IEEE engineering in medicine and biology society, Boston, MA, pp 7421–7424. doi:10.1109/IEMBS.2011.6091728
16. Galan F, Nuttin M, Lew E, Ferrez PW, Vanacker G, Philips J, del Millan JR (2008) A brain-actuated wheelchair: asynchronous and non-invasive Brain–computer interfaces for continuous control of robots. Clin Neurophysiol 119(9):2159–2169
17. Hansen PC (1986) The truncated SVD as a method for regularization. Stanford University, Stanford, CA
18. Hu Z, Chen G, Chen C, Xu H, Zhang J (2010) A new EEG feature selection method for self-paced brain–computer interface. In: Proceedings of the 10th international conference on intelligent systems design and applications, Cairo. doi:10.1109/ISDA.2010.5687156
19. Kauhanen L, Jylanki P, Lehtonen J, Rantanen P, Alaranta H, Sams M (2007) EEG-based brain–computer interface for tetraplegics. Intell Neurosci 2007:1–11
20. Lin H, Zhuliang Y, Zhenghui G, Yuanqing Li (2009) Bhattacharyya bound based channel selection for classification of motor imageries in EEG signals. In: Proceedings of the Chinese control and decision conference, Guilin, pp 2353–2356. doi:10.1109/CCDC.2009.5192711
21. Lotte F, Congedo M, Lecuyer A, Lamarche F, Arnaldi B (2007) A review of classification algorithms for EEG-based brain–computer interfaces. J Neural Eng 4(2):R1–R13
22. Mason SG, Birch GE (2003) A general framework for brain–computer interface design. IEEE Trans Neural Syst Rehab Eng 11(1):70–85
23. Michele MT, Robert RL, Rudiger RR, Jose Del RJRM (2010) Towards natural non-invasive hand neuroprostheses for daily living. In: Proceedings of the 2010 annual international conference of the IEEE engineering in medicine and biology society, Beunos Aires, pp 126–129. doi:10.1109/IEMBS.2010.5627178
24. Mueller-Putz G, Scherer R, Pfurtscheller G, Neuper C (2010) Temporal coding of brain patterns for direct limb control in humans. Front Neurosci 4(34):1–11
25. Narendra KS, Thathachar M (1974) Learning automata—a survey. IEEE Trans SMC SMC-4(4):323–334
26. Nijholt A, Tan D (2008) Brain–computer interfacing for intelligent systems. IEEE Intel Syst 23(3):72–79
27. Pfurtscheller G, Neuper C, Schlogl A, Lugger K (1998) Separability of EEG signals recorded during right and left motor imagery using adaptive autoregressive parameters. IEEE Trans Rehab Eng 6(3):316–325
28. Pfurtscheller G, Neuper C, Muller GR, Obermaier B, Krausz G, Schlogl A, Scherer R, Graimann B, Keinrath C, Skliris D, Wortz M, Supp G, Schrank C (2003) Graz-BCI: state of the art and clinical applications. IEEE Trans Neural Syst Rehab Eng 11(2):1–4
29. Prasad G, Herman P, Coyle D, McDonough S, Crosbie J (2010) Applying a brain–computer interface to support motor imagery practice in people with stroke for upper limb recovery: a feasibility study. J Neuroeng Rehabil 7(1):1–17
30. Price K, Storn RM, Lampinen JA (2005) Differential evolution: a practical approach to global optimization (Natural Computing Series). Springer-Verlag New York, Inc
31. Rakotomamonjy A, Guigue V, Mallet G, Alvarado V (2005) Ensemble of SVMs for improving brain computer interface p300 speller performances. In: Duch W, Kacprzyk J, Oja E, Zadrozny S (eds) 15th international conference on Artificial Neural Networks: Biological Inspirations (ICANN), LNCS 3696, Springer, Berlin Heidelberg, pp 45–50. doi:10.1007/11550822_8
32. Sanei S, Chambers JA (2008) EEG signal processing. Wiley, West Sussex, England
33. Scholkopf B, Smola A, Muller K-R (1998) Nonlinear component analysis as a kernel eigenvalue problem. Neural Comput 10(5):1299–1319
34. Sengupta A, Chakraborti T, Konar A, Eunjin K, Nagar AK (2012) An adaptive memetic algorithm using a synergy of differential evolution and learning automata. In: Proceedings of the 2012 IEEE congress on evolutionary computation (CEC), Brisbane, QLD, pp 1–8. doi:10.1109/CEC.2012.6256574
35. Shawe-Taylor J, Sun S (2011) A review of optimization methodologies in support vector machines. Neurocomputing 74(17):3609–3618
36. Storn R, Price K (1997) Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces. J Global Optim 11(4):341–359
37. Sun S (2008) The extreme energy ratio criterion for EEG feature extraction. In: Kurkova V, Neruda R, Koutnik J (eds) international conference on artificial neural networks (ICANN), LNCS 5164, Springer, Berlin Heidelberg, pp 919–928. doi:10.1007/978-3-540-87559-8_95
38. Sun S (2010) Extreme energy difference for feature extraction of EEG signals. Expert Syst Appl 37(6):4350–4357
39. Sun S, Zhang C (2006) Adaptive feature extraction for eeg signal classification. J Med Bio Eng Com 44(10):931–935
40. Tamraz JC, Comair YG (2006) Central region and motor cortex. In: Atlas of regional anatomy of the brain using MRI. Springer, Berlin, pp 117–138
41. Theodoridis S, Koutroumbas K (2006) Pattern recognition. Elsevier Science, Amsterdam
42. Vapnik VN (1995) The nature of statistical learning theory. Springer, New York
43. Yew-Soon O, Meng-Hiot L, Xianshun C (2010) Memetic computation-past, present & future. IEEE Comput Intell Mag 5(2):24–31
44. Yongwook C, Jaeseung J, Sungho J (2012) Toward brain-actuated humanoid robots: asynchronous direct control using an EEG-based BCI. IEEE Trans Robot 28(5):1131–1144