# Classification of Self-paced Finger Movements with EEG Signals Using Neural Network and Evolutionary Approaches

S. R. Liyanage, J. -X. Xu, C. Guan, K. K. Ang, C. S. Zhang and T. H. Lee

*Abstract*— **The dependable operation of brain-computer interfaces (BCI) based on electro electroencephalogram (EEG) signals requires precise classification of multi-channel EEG signals. The design of EEG interpretation and classifiers for BCI are open research questions whose difficulty stems from the need to extract complex spatial and temporal patterns from noisy multidimensional time series obtained from EEG measurements. In this paper we attempt to classify EEG data used in the BCI competition by the combination of pattern classification methods. We use Common Spatial Pattern (CSP) to extract features. A Genetic Algorithm (GA) was applied first to evolve an artificial neural network (ANN) to find the optimum structure of ANN. A Particle Swarm Optimization (PSO) was also attempted to determine the optimal number of hidden neurons complementary to the GA approach. Then the GA was used to evolve the connection weights of the ANN.**

## I. Introduction

Brain–Computer Interface is a direct communication channel between brain and computer or external devices. They create a link between the brain and an output device by bypassing conventional motor output pathways of nerves and muscles. The translation of user intent into device control commands is a major challenge. Success requires the effective interaction of two adaptive controllers: the user's brain, which produces brain activity that encodes intent, and the BCI system, which translates that activity into device control commands. In the literature, many machine learning and pattern classification algorithms have been reported to give excellent results when applied to BCI data in offline analyses.

A typical noninvasive electroencephalogram (EEG) based brain-computer communication devices are composed of three subsystems. Namely: EEG acquisition, EEG signal processing and the output subsystems. The acquired EEG signal can be regarded a complex time series signal that has multiple factors intricately intertwined. Therefore, signal processing and classification methods are essential tools in the development of improved BCI technology.

In this article, we examine the application of evolutionary Neural Networks to the problem of EEG signal classification. The goal of the project was to classify unknown EEG data as associated with either a left finger movement or a right finger movement. The sample data set used had been provided by Fraunhofer FIRST, Intelligent Data Analysis Group and Charité University Medicine Berlin, Campus Benjamin Franklin, Department of Neurology, Neurophysics Group via the BCI competition webpage [1].

## II. Materials and Methods

### A. A Description of the Data Set

This data set had been recorded from a healthy subject during a session with no feedback. In these sessions the subject had sat in a normal chair, with arms relaxed and resting on the table. The fingers had been in the standard typing position at the computer keyboard. The task had been to press one of four assigned keys in a self-chosen order and timing ("self-paced tapping") with either the index or the little finger of either the left or the right hand. The experiment had consisted of three runs of 6 min each. All runs had been conducted in one session with break of a few minutes in between. Typing had been performed at an average speed of 1 key tap per second.

416 epochs of 500 ms EEG had been provided, each ending 130 ms before an actual key press. (This choice of an early endpoint ensured that almost all trials were free of Electromyography activity resulting from muscle movement.) The epochs had been randomly shuffled and split into a training set (316 epochs) which has been labeled "0" for upcoming left hand and "1" for upcoming right-hand movements. Another test set (100 epochs) is also provided for evaluation. EEG had been recorded from 28 scalp positions, mainly covering the primary motor cortices bilaterally.

### B. Feature Selection by Common Spatial Patterns (CSP)

The ultimate success of a learning machine relies typically on a proper preprocessing of the data. Furthermore and very

Manuscript received May 10, 2009. Sidath R. Liyanage is with the National University of Singapore Graduate School for Integrative Sciences and Engineering (e-mail: g0801873@nus.edu.sg). Jianxin Xu is with the Department of Electrical & Computer Engineering, National University of Singapore (email: elexujx@nus.edu.sg). C. Guan and K. K. Ang are with the Institute for Infocomm Research, Agency for Science, Technology and Research, Singapore 119613, Singapore (e-mail: ctguan@i2r.a-star.edu.sg; kkang@i2r.a-star.edu.sg).C.S.Zhang is with the Department of Electrical & Electronic Engineering, Nanyang Technological University (email: ecszhang@ntu.edu.sg). T.H.Lee is with the National University of Singapore Graduate School for Integrative Sciences and Engineering and the Department of Electrical & Computer Engineering, National University of Singapore (e-mail: eleleeth@nus.edu.sg).

important in practice, we can discard non-informative dimensions of the data and thus select the features of interest for classification [3].

There are two matrices associated with the first 316 trials. One is a 50×28×316 matrix with all the raw voltage readings from the EEG of the 28 channel sensors over 50ms. The second is a 1×316 matrix of labels for each trial. If the nth entry of this vector is 1, then the subject used a finger of the right hand during the nth trial; otherwise, the subject used a finger of the left hand. To make this massive matrix easier to work with, it is converted to a 316×1400 matrix, where each row contains all the data from that trial. This process creates a 316×1400 matrix containing all the original EEG data, but such a matrix was found to be too large to work with in a computationally efficient manner.

Raw EEG scalp potentials are known to have a poor spatial resolution owing to volume conduction. In a simulation study in [23] only half the contribution to each scalp electrode came from sources within a 3 cm radius. Band-pass filtering is carried out with a low-pass filter applied (cutting off at 40 Hz) to remove the noise in the raw data. After removing the disturbances the spatial filtering can be applied to produce a new time series whose variances are optimal for the classification. Optimal spatial filtering can be achieved using Common Spatial Patterns Analysis (CSP), it is based on simultaneous diagonalization of two covariance matrices.

Common Spatial Patterns (CSP) Analysis is a technique to analyze multichannel data based on recordings from two classes (conditions). CSP yields a data-driven supervised decomposition of the signal parameterized by a matrix, $W \in R^{C\times C}$ ($C$ is the number of channels) that projects the signal, $x(t) \in R^C$ in the original sensor space, to $x_{CSP}(t) \in R^C$ which is the surrogate sensor space, namely $x_{CSP}(t) = W^T x(t)$.

The CSP algorithm was first presented by Koles [24] as a method to extract the abnormal components from EEG, using a set of patterns that are common to both the normal and the abnormal recordings and have a maximally different proportion of the combined variances. Later CSP was used to create features for classification in EEG caused by imagined movements [25]. The first and last few CSP components (the spatial filters that maximize the difference in variance) are used to classify the trials with a high accuracy.

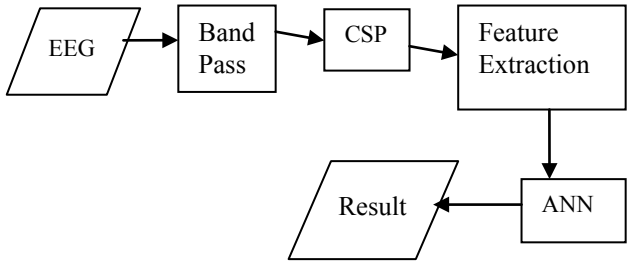


Fig. 1: Flow chart of classification procedure using CSP.

Figure 1 depicts the procedures associated with CSP for optimal spatial filtering and using an Artificial Neural Network for classification.

### *C. Evolving Neural Networks using Genetic Algorithm*

The first approach attempted was to classify the data set using an Evolving Artificial Neural Network (EANN) where the architecture of the ANN itself is evolved. The architecture of an ANN is crucial in the successful application of ANN because the architecture has significant impact on a network's information processing capabilities. Given a learning task, an ANN with only a few connections and linear nodes may not be able to perform the task at all due to its limited capability, while an ANN with a large number of connections and nonlinear nodes may overfit noise in the training data and fail to have good generalization ability. Finding the optimal architecture for an ANN can be formulated as a search problem in the architecture space where each point represents a different ANN structure.

The candidate ANN structures were Multilayer Perceptrons (MLP) with varying numbers of hidden neurons in two hidden layers. In the first attempt A genetic algorithm (GA) was used to find the optimal number of hidden neurons for the MLP. Two hidden layers were sufficient to classify the EEG patterns according to the universal approximation theorem [1].

A binary chromosome with 20 bits was used to represent the number of neurons in the two hidden layers. The first 10 bits represent the number of hidden neurons in the first hidden layer and the second 10 bits represent the number of neurons in the second hidden layer. Uniform crossover was employed so that the chromosomes recombine during crossover at one point. A crossover rate of 0.9 was used so that the offspring replace the parents in the next generation with a probability of 0.9. The mutation rate was kept to 0.01. A maximum of 100 neurons for each layer was attempted first. It was found that there exists excessive overfitting in the resulting architecture of the ANN. Later the numbers of neurons in hidden layers were tuned lower according to GA. Fitness measure was the inverse of total misclassifications. Therefore the algorithm tries to minimize classification errors in training data and test data both. By analyzing the output the most appropriate numbers of hidden neurons were determined.

### *D. Particle Swarm Optimization for Evolving Neural Networks*

The PSO algorithm was originally developed to imitate the motion of a flock of birds, or insects [4]. PSO offers rapid optimization of complex multidimensional search spaces, and

is a popular contemporary algorithm for a wide range of search and optimization problems. As a robust stochastic optimization technique based on the movement and intelligence of swarms, PSO applies the concept of social interaction to problem solving. It uses a number of agents (particles) that constitute a swarm moving around in the search space looking for the best solution. Each particle is treated as a point in an N-dimensional space which adjusts its "flying" according to its own flying experience as well as the flying experience of other particles.

The basic concept of PSO lies in accelerating each particle toward its previous best and the group's previous best locations, with a random weighted acceleration at each time step. The algorithm for calculating the next particle position (x) is,

$$v_{t+1} = v_i + C_1 B_1 (p_i - x_i) + C_2 B_2 (p_g - x_i) \quad (1)$$

$$x_{t+1} = x_t + v_{t+1} \quad (2)$$

where constants $C_1$ and $C_2$ determine the balance between the influence of the individual knowledge ($C_1$) and that of the group ($C_2$) (both set initially to 2), $B_1$ and $B_2$ are uniformly distributed random numbers defined by some upper limit, $B_{max}$, that is a parameter of the algorithm, $p_i$ and $p_g$ are the individual's previous best position and the group's previous best position, and $x_i$ is the current position in the dimension considered. For an n-dimensional space the particle velocity is calculated for each dimension, i = 1,2... n, then resolved into a final vector for updating the particle's position [26].

The operations in (1) result in a balanced motion of the particles towards the previously known best points in the space. This balanced motion between the local or global best, *i.e.* between exploration and exploitation, could be considered as broadly equivalent to selection pressure in a GA, although control is more complex in a GA, with other critical parameters such as population size and mutation rate.

The optimal number of hidden neurons was identified using a PSO with thirty particles. The upper limit of the number of hidden neurons was set to 40. The maximum iterations was set to 100. The algorithm converges to the optimal combination of neurons that minimizes the classification errors.

### *E. Pattern Classification*

The goal of pattern classification is to find a rule that assigns an object to one of several possible classes. Minimizing the training error alone does not produce the minimum errors for the test cases most of the time. A more promising approach is to employ an A special class of ANN in which evolution is a fundamental form of adaptation in addition to learning is known as evolutionary artificial neural networks [5]–[9].

Two ANN architectures were adopted in this work. The first type of ANN is an Evolutionary Neural Network, in which the numbers of hidden layer neurons were tuned by GA and PSO. In this first type of EANN the weights of the units were trained using the Backpropagation (BP) algorithm. Weight training in ANN is usually formulated as minimization of an error function, such as the mean square error between target and actual outputs averaged over all examples, by iteratively adjusting connection weights. Most training algorithms, such as BP and conjugate gradient algorithms [10], [11]–[13], are based on gradient descent. There have been some successful applications of BP in various areas [14], but BP has drawbacks due to its use of gradient descent. It often gets trapped in a local minimum of the error function and is incapable of finding a global minimum if the error function is multimodal and/or non-differentiable.

One way to overcome gradient-descent-based training algorithms' shortcomings is to adopt the second type of EANN in which the training process is conducted as the evolution of connection weights in the environment determined by the architecture and the learning task. GA can be used effectively in the evolution to find a near-optimal set of connection weights globally without computing gradient information.

Another limitation of the first type of EANN is that it could lead to overfitting and non-robust behavior. One way to avoid the overfitting problem is to use a relatively simple function that explains most of the data, as a preferable way over a complex one that explains all data. This concept is known as Occam's razor. Therefore in classifying the data set a reduced EANN where the maximum number of neurons in each hidden layer was 10 was developed. The optimal structures were identified using the PSO algorithm and the weights of the selected ANN structures were trained using GA.

The fitness of an ANN can be defined according to different needs. Two important factors which often appear in the fitness (or error) function are the error between target and actual outputs and the complexity of the ANN. Unlike the case in gradient-descent-based training algorithms, the fitness (or error) function does not have to be differentiable or even continuous since the second type of EANN does not depend on gradient information. Because GA can treat large, complex, non-differentiable and multimodal spaces, which are the typical case in the real world, considerable research and application have been conducted on the evolution of connection weights [15].

A typical cycle of the evolution of connection weights is shown in Figure 1 that gives main steps involved in this procedure. The evolution stops when the fitness is greater than a predefined value (i.e., the training error is smaller than a certain value) or the population has converged.

As illustrated in Figure 2, the weights of the ANN were trained using genetic algorithm. The topology of the neural network consisted of 28 neurons in the input layer, and two hidden layers and 2 neurons in the output layer. The maximum number of neurons in the hidden layers was restricted to 10 and the best structures were determined using the PSO algorithm. Then each of the selected architectures were trained using the GA. The weights of the hidden neurons and the biases were represented in a real valued chromosome. As connection weights are represented by real numbers, each individual in the evolving population is a real vector. A population size of 100 was used in training the EANN and a maximum generation size was specified to be 10000.

It is generally very difficult to apply crossover operators in evolving connection weights since they tend to destroy feature detectors found during the evolutionary process. Traditional binary crossover and mutation can no longer be used directly. Special search operators have to be designed. In [16] a large number of tailored genetic operators were defined to incorporate many heuristics about training ANN. The idea was to retain useful feature detectors formed around hidden nodes during evolution. Evolutionary training approach has been proved to be much faster than BP for the problems considered [16].

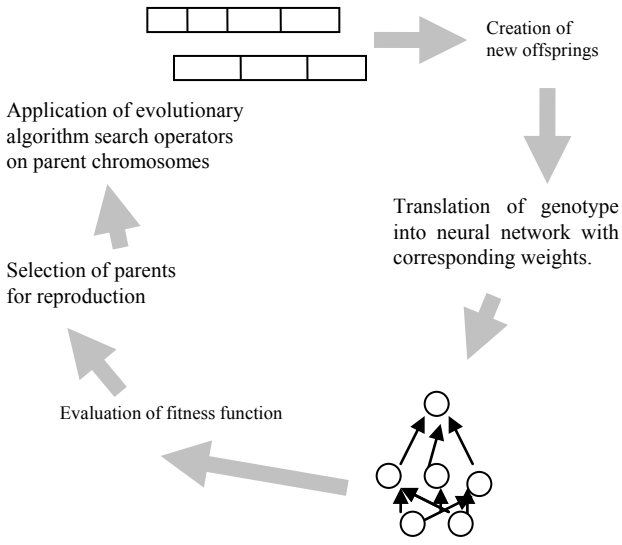


Fig.2: Typical cycle of the evolution of connection weights

The typical cycle of evolving the connection weights using GA begins with decoding each individual (genotype) in the current generation into a set of connection weights. Each ANN is evaluated by computing its total mean square error between actual and target outputs. The fitness of an individual is determined by the error. The higher the error, lower the fitness. Parents are selected for reproduction based on their fitness and crossover and mutation are applied on the parent chromosomes. The generated offspring form the next generation.

## III. RESULTS

The results from the first type of EANN with a maximum number of 100 hidden units in each of the two layers are shown in Figures 3-5. Figure 3 depicts how the minimum errors had fluctuated in successive generations. Figure 4 depicts how the average errors had changed during the generations. Figure 5 shows the best solutions found during successive generations of the algorithm. Table 1 shows some of the results obtained from the last population of PSO-based approach for evolving the number of hidden units. The computational cost incurred on a 2.53GHz Processor for the first EANN was 147682.89 Seconds. The PSO-based approach incurred a cost of 105682. 41 Seconds. This is a 28% reduction in the execution time.

The results obtained by using a reduced number of hidden neurons trained by GA are given in Table 2. The computational cost incurred in training the weights of the ANN was in the range of 72 hours in average.

The accuracy of results obtained by applying the band filter for 0-40Hz, CSP for Optimal Spatial filtering and EANN for classification was 78% for the test set.

Comparing the above results, we can identify that the performance of the method using CSP with EANN is on par with the best result achieved in the BCI Competition which had used a combination of common subspace decomposition and Fisher discriminant for feature selection and a Perceptron neural network for classification. The ANN with a smaller number of hidden neurons has performed marginally better than the ANN with a large number of hidden neurons in the hidden layers.

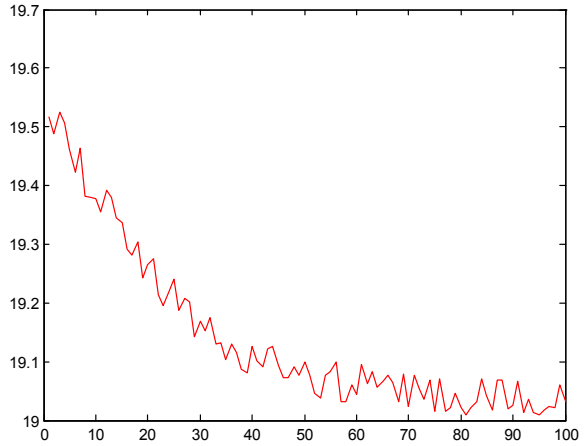


Fig. 3: Minimum errors in each generation.

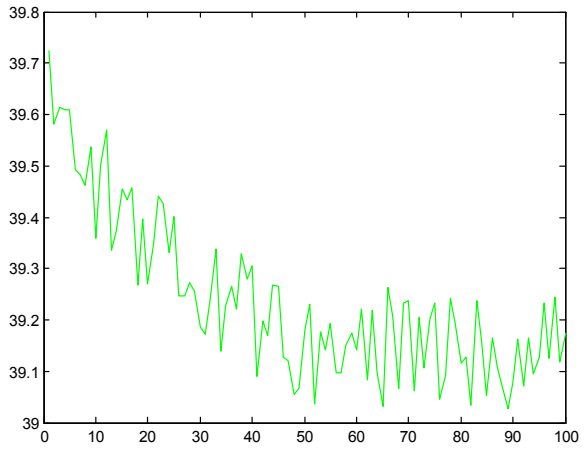


Fig. 4: Average errors in each generation.

Table 1: Numbers of neurons and their corresponding errors for the first type of EANN.

| No of Neurons in layer 1 | No of Neurons in layer 2 | Error % |
|---|---|---|
| 29 | 88 | 27 |
| 74 | 89 | 31 |
| 44 | 36 | 26 |
| 77 | 81 | 25 |
| 42 | 37 | 24 |

Table 2: Numbers of neurons and their corresponding errors for the second type of EANN.

| No of Neurons in layer 1 | No of Neurons in layer 2 | Error % |
|---|---|---|
| 7 | 6 | 23 |
| 4 | 7 | 24 |
| 8 | 7 | 22 |
| 6 | 7 | 24 |
| 5 | 7 | 23 |

## IV. Discussions

Using EEG to detect motions is a highly challenging issue. In this particular application, the success rate is not that high due to the difficulty in EEG classification [17]. The performance of 22% misclassifications is in the vicinity of the best performances achieved in the BCI Competition [2]. The winning entry from Tsinghua University, Beijing had achieved an error rate of 16% and the entry from University of Toronto with an error rate of 19% had been placed second. Our results show that there is room for further improvement as we are doing. Future work leading to enhanced classification methods and more accurate preprocessing techniques are expected to improve the results.

One of the problems faced by evolutionary training of ANN is the permutation problem [18], [19], also known as the competing convention problem. It is caused by the many-to-one mapping from the representation (genotype) to the actual ANN (phenotype) since two ANNs that order their hidden nodes differently in their chromosomes could still be equivalent functionally. In general, any permutation of the hidden nodes will produce functionally equivalent ANNs with different chromosome representations. The permutation problem makes crossover operator very inefficient and ineffective in producing good offspring. However, empirical results have shown that the adverse effects from the permutation problem can be alleviated with a high enough selection pressure in selecting the parents [18].

Evolutionary training can be slow for some problems in comparison with fast variants of BP [20] and conjugate gradient algorithms [13], [21]. However, GA is generally much less sensitive to initial conditions of training. They always search for a globally optimal solution, while a gradient descent algorithm can only find a local optimum in a neighborhood of the initial solution.

## References

[1] Haykin, Simon (1998). Neural Networks: A Comprehensive Foundation, 2, Prentice Hall. ISBN 0132733501

[2] P. Sajda, A. Gerson, K.-R. Müller, B. Blankertz, and L. Parra, "A data analysis competition to evaluate machine learning algorithms for use in brain-computer interfaces," IEEE Trans. Neural Syst. Rehab. Eng., vol. 11, pp. 184–185, June 2003.

[3] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," Journal of Machine Learning Research, vol. 3, pp. 1157–1182, Mar. 2003.
[4] Kennedy J, Eberhart RC (1995) Particle swarm optimization. In: Proceedings of the IEEE international conference on neural networks, Piscataway, NJ, pp 1942–1948
[5] X. Yao, "Evolution of connectionist networks," in Preprints Int. Symp. AI, Reasoning & Creativity, Queensland, Australia, Griffith Univ., 1991, pp. 49–52.
[6] X. Yao, "A review of evolutionary artificial neural networks," Int. J. Intell. Syst., vol. 8, no. 4, pp. 539–567, 1993.
[7] X. Yao, "Evolutionary artificial neural networks," Int. J. Neural Syst., vol. 4, no. 3, pp. 203–222, 1993.
[8] X. Yao, "The evolution of connectionist networks," in Artificial Intelligence and Creativity, T. Dartnall, Ed. Dordrecht, The Netherlands: Kluwer, pp. 233–243, 1994.
[9] X. Yao, "Evolutionary artificial neural networks," in Encyclopedia of Computer Science and Technology, vol. 33, A. Kent and J. G. Williams, Eds. New York: Marcel Dekker, 1995, pp. 137–170.
[10] J. Hertz, A. Krogh, and R. Palmer, Introduction to the Theory of Neural Computation. Reading, MA: Addison-Wesley, 1991.
[11] D. R. Hush and B. G. Horne, "Progress in supervised neural networks," IEEE Signal Processing Mag., vol. 10, pp. 8–39, Jan. 1993.
[12] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," in Parallel Distributed Processing: Explorations in the Microstructures of Cognition, vol. I, D. E. Rumelhart and J. L. McClelland, Eds. Cambridge, MA: MIT Press, 1986, pp. 318–362.
[13] M. F. Møller, "A scaled conjugate gradient algorithm for fast supervised learning," Neural Networks, vol. 6, no. 4, pp. 525–533, 1993.
[14] K. J. Lang, A. H. Waibel, and G. E. Hinton, "A time-delay neural network architecture for isolated word recognition," Neural Networks, vol. 3, no. 1, pp. 33–43, 1990.
[15] P. Zhang, Y. Sankai, and M. Ohta, "Hybrid adaptive learning control of nonlinear system," in Proc. 1995 American Control Conf. Part 4 (of 6), pp. 2744–2748.
[16] D. Montana and L. Davis, "Training feedforward neural networks using genetic algorithms," in Proc. 11th Int. Joint Conf. Artificial Intelligence. San Mateo, CA: Morgan Kaufmann, 1989, pp. 762–767.
[17] Y. Wang, Z. Zhang, Y. Li, X. Gao, S. Gao, and F. Yang, BCI Competition 2003—Data Set IV:An Algorithm Based on CSSD and FDA for Classifying Single-Trial EEG, in IEEE Trans. Biomed. Eng., vol. 51, pp. 1081–1086, June 2004.
[18] R. K. Belew, J. McInerney, and N. N. Schraudolph, "Evolving networks: Using genetic algorithm with connectionist learning," Comput. Sci. Eng. Dep. (C-014), , Univ. of California, San Diego, Tech. Rep. CS90-174 (revised), Feb. 1991.
[19] P. J. B. Hancock, "Genetic algorithms and permutation problems: A comparison of recombination operators for neural net structure specification," in Proc. Int. Workshop Combinations of Genetic Algorithms and Neural Networks (COGANN-92), D. Whitley and J. D. Schaffer, Eds. Los Alamitos, CA: IEEE Computer Soc., 1992, pp. 108–122.
[20] S. E. Fahlman, "Faster-learning variations on back-propagation: An empirical study," in Proc. 1988 Connectionist Models Summer School, D. S. Touretzky, G. E. Hinton, and T. J. Sejnowski, Eds. San Mateo, CA: Morgan Kaufmann, 1988, pp. 38–51.
[21] E. M. Johansson, F. U. Dowla, and D. M. Goodman, "Backpropagation learning for multilayer feed-forward neural networks using the conjugate gradient method," Int. J. Neural Syst., vol. 2, no. 4, pp. 291–301, 1991.
[22] P.L. Nunez, R. Srinivasan, A.F. Westdorp, R.S. Wijesinghe, D.M. Tucker, R.B. Silberstein, and P.J. Cadusch, "EEG coherency I: Statistics, reference electrode, volume conduction, Laplacians, cortical imaging, and interpretation at multiple scales," Electroencephalogr. Clin. Neurophysiol., vol. 103, no. 5, pp. 499–515, 1997.
[23] Z. J. Koles. The quantitative extraction and topographic mapping of the abnormal components in the clinical eeg. Electroencephalography and Clinical Neurophysiology, 79(6):440–447, December 1991.
[24] H.Ramoser, J. Muller-Gerking, and G. Pfurtscheller. Optimal spatial filtering of single trial EEG during imagined hand movement. IEEE Trans. on Neural Systems and Rehabilitation, 8(4):441–446, 2000.
[25] A. Banks, J. Vincent and C. Anyakoha, A Review of Particle Swarm Optimization. Part 1: Background and Development, Natural Computing Springer Netherlands, Vol. 6, No. 4, pp. 467-484, 2007.