Original Article

Design of Magnetoencephalography-based Brain–machine Interface Control Methodology through Time-varying Cortical Neural Connectivity and Extreme Learning Machine

Abstract

Introduction: Human-machine interfaces (HMIs) can improve the quality of life for physically disabled users. This study proposes a noninvasive BMI design methodology to control a robot arm using MEG signals acquired during the user's imagined wrist movements in four directions. **Methods:** The BMI uses the partial directed coherence measure and a time-varying multivariate adaptive autoregressive model to extract task-dependent features for mental task discrimination. An extreme learning machine is used to generate a model with the extracted features, which is used to control the robot arm for rehabilitation or assistance tasks for motor-impaired individuals. **Results:** The classification results show that the proposed BMI methodology is a feasible solution with good performance and fast learning speed. **Discussion:** The proposed BMI methodology is a promising solution for rehabilitation or assistance systems for motor-impaired individuals. The BMI provides satisfactory classification performance at a fast learning speed.

Keywords: *Brain–machine interface, extreme learning machine, functional connectivity, magnetoencephalography*

Caglar Uyulan

Department of Mechanical Engineering, Faculty of Engineering and Architecture, İzmir Katip Celebi University, İzmir, Turkey

Introduction

Brain–machine interfaces (BMIs) are built as communication devices that encode brain activity to a machine command signal, not involving muscles. The utilization of BMIs is quite helpful for users having diseases or traumatic injuries which cause muscle control degradation or motor disabilities (termed as a locked-in syndrome). The locked-in syndromes may comprise a wide spectrum of diseases and traumata, i.e., amyotrophic lateral sclerosis, cerebral palsy, muscular dystrophy, multiple sclerosis, brainstem stroke, and brain or spinal cord injury.

Practical applications of brain–computer interfaces (BCIs) cover electrophysiological signals, such as electroencephalography (EEG), electrocorticography, and MEG. BCIs generally control a robotic end system via these aforementioned signals recorded from the synchronized activity of neuron groups inside the brain of subjects. The subjects can be people having different ages, gender, and physical characteristics. The aim here is to establish a user-independent, generic model of the BMI and to search for robust-adaptive algorithms that minimize the variance of classification errors concerning the various kinds of users. The combination of "feature extraction/dimension reduction/feature selection/classification" algorithms with the best classification performance and the least possible error variance concerning users becomes the main model. As a result of this main model obtained, i.e., the robot trajectory control can be realized. The strong hypothesis (model) that is generated, is expected to guarantee the mental task classification of random users by limiting their errors to a lower band. As a result of this research, the BMI can be brought to a level that competes with commercial applications. The design procedure is divided into three basic modules: data acquisition, signal processing, and machine control. The majority of brain waves along the scalp are collected, and the wave information is transferred to a central processing unit (CPU) for processing in

Received : 07-12-2022
Accepted : 11-12-2022
Published : 29-12-2022

Orcid
Caglar Uyulan: {ORCID: 0000-0002-6423-6720}

Address for correspondence:
Dr. Caglar Uyulan, Department of Mechanical Engineering, Faculty of Engineering and Architecture, İzmir Katip Celebi University, İzmir, Turkey.
E-mail: caglaruyulan1@gmail.com

Access this article online
Website: www.jnbsjournal.com
DOI: 10.4103/jnbs.jnbs_35_22
Quick Response Code:

How to cite this article: Uyulan C. Design of magnetoencephalography-based brain–machine interface control methodology through time-varying cortical neural connectivity and extreme learning machine. J Neurobehav Sci 2022;9:96-106.

Ethics committee approval: There is no need for ethics committee approval

the data acquisition process. The CPU filters the signals and runs algorithms that extract features to control the machine. MEG can decrease training time while increasing the robustness of a BMI. As compared to the EEG, MEG has a larger number of sensors. Therefore, the spatial resolution is more. The detection of the frequency information can be above 40 Hz, which is not capable in EEG recordings.[1,2]

BMI architecture to be designed: (1) should adapt to nonstationary brain dynamics, (2) should generate neural signals from general brain states independently of the users, (3) should have a generalizable structure that will allow an easy transition to control applications, and (4) should demonstrate high classification performance.[3-6]

EEG-based BCIs suffer the problem that the recorded signal is nonstationary, that is, the data are subject to covariate shift. The nonstationarity can occur from the transition of the training model without feedback to online use with feedback, the shift of sensors to different brain regions due to head movements in MEG, or variations in mental status over time. These nonstationary cases are valid for a classifier trained on training data showing this mismatch.[7,8]

Since the MEG is a nonstationary signal, many feature extraction methods that originated from time-domain, frequency-domain, and time-frequency domain analysis, are utilized in the literature.[9-12] Time-domain features are extracted directly from the signal and focus on the (averaged) time course. Frequency-domain features highlight the signal power in frequency band components. Time-frequency domain features analyze how the power spectrum changes with respect to time. The short-time Fourier transform and the wavelet transform are the most applicable.[13-16]

Spatial filtering, which extracts signals from multiple sensors to look at the activity localized in a particular brain region, is also applied. Some of the common spatial filtering methods can be listed as: *bipolar montage*, where bipolar channels are evaluated by subtracting the signals from two collocated electrodes;[17] *common average reference*, which subtracts the average value of the full electrode montage from that of the specific channel;[18] *Laplacian method,* which evaluates for each electrode location the second derivative of the instantaneous spatial voltage distribution by combining the value at that location with the values of a set of surrounding electrodes;[19] and *common spatial patterns*, which analyzes multichannel data based on savings from two tasks. The filters are utilized to optimize the variance for one task and minimize it for the other tasks.[20] Other techniques have also been utilized in electrophysiological feature extraction, such as the Kalman filter, fractal dimensions, and entropy.[21-23] The AAR algorithm based on the Kalman filtering approach has also been applied to analyze the electrophysiological signals. The parametric model has an adaptive closed-loop controller with a recursive Bayesian estimator and a linear-square regulator. It estimates states and system parameters from different mental tasks and feeds them back to the optimal controller. The performance of BMIs is enhanced by closed-loop solver adaptation or multiplicative recurrent artificial neural network (ANN) solver coupled with KF. Thanks to this solver, the learned training datasets become more robust and provide a wide variety of neural-kinematic mapping learning.[24-26]

Coming to the classification stage, the use of ANNs is quite common. However, the learning rates of feedforward ANNs, including deep learning applications, are quite low. Among the main reasons for this are the use of slow gradient-based learning algorithms in the training of ANNs and the iterative updating of all parameters of the networks in this way at each stage. ELMs have succeeded in solving this problem by proving that hidden nodes do not necessitate learning and are recursively set. ELM consists of generalized single or several hidden layer feedforward networks. ELMs give importance to feature representations in hidden layers compared to support vector machines (SVMs). These models can demonstrate good generalization performance and learn extremely faster than backpropagation networks. Random input layer weights add to the generalization capabilities of a linear output layer solution as it has nearly orthogonal (weakly correlated) hidden layer properties. If the weight range is constrained, the orthogonal inputs yield a larger solution space volume with these constrained weights. Thanks to the small-weight norms, the system is more stable and robust to noise. The random hidden layer generates weakly correlated hidden layer features, proposing a solution with a low norm and strong generalization performance. Various types of ELM have been proposed in the literature.[27-29]

To learn effectively from various data types, ELMs need to be modified to suit the problem. When new samples are added to the dataset and grown, the ELM is retrained, but retraining the network is inefficient because the proportion of new incoming data is small. Therefore, Matias *et al.*[30] proposed an online sequential ELM (OS-ELM). The basic idea behind OS-ELM is to avoid retraining on previous examples using a sequential method. OS-ELM reconstructs settings using new instances sequentially after startup. Huang *et al.*[31] developed an incremental ELM (I-ELM). When a new hidden node is introduced, I-ELM randomly adds nodes to the hidden layer one by one, freezing the output weights of the existing hidden nodes. I-ELM is effective for single-layer feedforward networks (SLFNs) with piecewise continuous activation functions. Rong *et al.*[32] proposed a pruned ELM (P-ELM) algorithm as a systematic and automated strategy for building ELM networks. Using a small number of hidden nodes can cause under/overfitting problems in model classification. Compared to traditional ELM, simulation results showed that P-ELM results in compact network classifiers that produce fast responses and strong prediction accuracy on unseen data. Feng *et al.*[33] proposed an error minimization-based method (EM-ELM) that determines the number of hidden nodes in generalized SLFNs by growing hidden nodes one by one or group by group. As the networks grow, the output weights are gradually changed, significantly reducing the computational burden. The simulation results in sigmoid-type hidden nodes showed that this method can greatly decrease the computational cost of ELM. Nodes are randomly added to the network until they reach a certain error value of ε. A new learning algorithm called an evolutionary ELM (E-ELM) has been developed to optimize input weights and latent biases and determine output weights. Output weights are determined analytically using the MP generalized inverse.[34] The stability and generalization performance of the ELM was investigated. This system consists of two stages. In the first step, a forward iterative algorithm is applied to select hidden nodes from randomly generated candidates at each step and then hidden nodes are added until the stopping criterion is matched. In the second stage, unimportant nodes are removed.[35,36] A kernel-based ELM (KELM) inspired by SVM has been developed and the main kernel function used in ELMs is the radial basis function. KELMs are used to design deep ELMs (DELMs).[37] DELMs utilize KELM as an output layer.[38] Voting-based ELM (V-ELM) has been proposed to improve performance on classification tasks. In V-ELM, not just one network is trained, but also many are trained and then the fittest is chosen according to the majority voting method.[39]

In this study, a signal processing methodology was developed for the subjects who control the movement of a robotic arm using four motor imagery signals related to the following wrist movement states, respectively: right, forward, left, and backward. Section 2 gives a background on the MEG acquisition system and the robot arm, followed by a discussion on the classification and control strategy implemented in this study. In Section 3, the analysis results were presented. Section 4 includes the discussion and conclusive summary part.

Materials and Methods

There is no need for ethics committee approval.

MEG data acquisition and preprocessing

The data set was provided by the Brain Machine Interfacing Initiative, Albert-Ludwigs-University Freiburg, the Bernstein Center for Computational Neuroscience Freiburg, and the Institute of Medical Psychology and Behavioral Neurobiology, the University of Tübingen collected for the BCI Competition IV (https://www.bbci.de/competition/iv/). The data set comprises directionally modulated low-frequency MEG activity that was acquired from two healthy, right-handed subjects. They performed wrist movements in four various directions, i.e., we have four different classes. The task was to move a joystick from a center position toward one of four targets located radially at 90° intervals (four-class center-out paradigm) utilizing exclusively the right hand and wrist.

The MEG device has 10 channels and 625 Hz. sampling rate. The data were band-pass filtered (0.5 to 100 Hz., Butterworth, 3rd order) and resampled at 400 Hz. The data comprise signals from ten MEG channels which were positioned above the motor areas given in Figure 1.

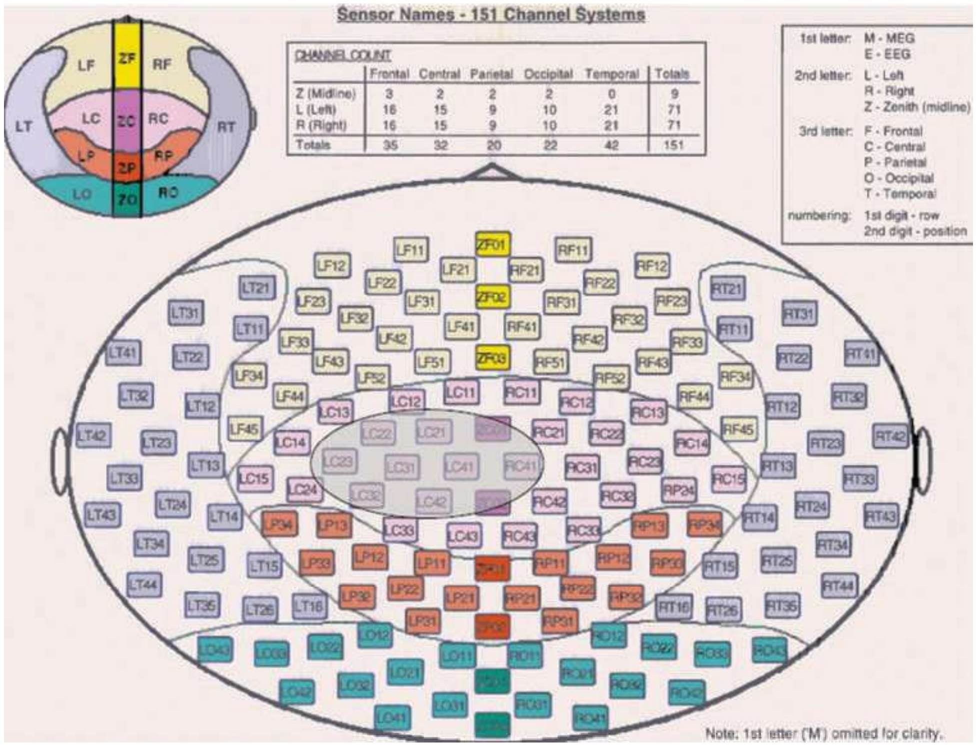
The details about the data acquisition and signal processing processes can be accessed at https://www.bbci.de/competition/iv/. Data were collected from two separate subjects including training sessions during 40 s/trials for each mental task, and testing sessions during 73 trials by executing random tasks. The test data will be used as external validation for assessing the performance of the proposed classifier.

The training data were merged as the size of the matrix for each mental task-based class is 40 (number of trials)×800 (number of sampling data)×10 (number of channels). The train data matrix was reshaped into a two-dimensional form as 32000 (number obtained from multiplication of trials with the sampling data)×(number of channels). The total size of the training data is 128000 × 10.

The testing data including the random sequence of mental tasks were merged as the size of the matrix is 73 (number of trials)×800 (number of sampling data)×10 (number of channels). The test data matrix was reshaped into a two-dimensional form as 58400 (number obtained from multiplication of trials with the sampling data)×10 (number of channels).

Feature extraction

The feature extraction process includes the transformation of the preprocessed signal into a feature matrix by attenuating noise and focusing on important data. The AR_{FIT} package is utilized to find an optimum model order for the time-varying multivariate autoregressive (MVAR) model. The time-invariant parameter and the order of the model can be estimated using AR_{FIT}-package. "Schwarz's Bayesian Criterion" is applied for the order estimation and then the model order is fixed for further analysis. Time-varying partial directed coherences (PDCs) are evaluated through the time-varying MVAR model matched with the signal using an AAR algorithm, which uses linear Kalman filtering for parameter estimation. A surrogate



Sensor Names - 151 Channel Systems

CHANNEL COUNT

	Frontal	Central	Parietal	Occipital	Temporal	Totals
Z (Midline)	3	2	2	2	0	9
L (Left)	16	15	9	10	21	71
R (Right)	16	15	9	10	21	71
Totals	35	32	20	22	42	151

1st letter:	M - MEG E - EEG
2nd letter:	L - Left R - Right Z - Zenith (midline)
3rd letter:	F - Frontal C - Central P - Parietal O - Occipital T - Temporal
numbering:	1st digit - row 2nd digit - position

Note: 1st letter ('M') omitted for clarity.

Figure 1: The MEG channel positions. 'LC21', 'LC22', 'LC23', 'LC31', 'LC32', 'LC41', 'LC42', 'RC41', 'ZC01', 'ZC02' [https://www.bbci.de/competition/iv/]

data method having 50 realizations is performed to choose the most essential quantities of the measure at a 99% confidence level. Surrogates are constructed by randomizing all samples of the signal to remove the causal relations among them.[40,41] The algorithm embeds the linear Kalman filtering to update the MVAR parameters for each time sample.

A d-dimensional time-varying MVAR process is given in Eq. 1.

$$y_k = \sum_{r=1}^{p} A_k^{(r)} y_{k-r} + w_k \tag{1}$$

where p is the model order, $w_k \in R^d$ is a zero-mean white process noise vector, and $A_k^{(r)} \in R^{d\times d}$ is the matrix of autoregressive coefficients at each lag r and time point k=p+1,…,N.

A form of state space equations is built from the MVAR equations by re-organizing all matrix parameters into a state vector of the dynamical system and focusing the nonstationary signal as the measurement is given in Eq. 2.[42,43]

$$\begin{aligned} x_k &= F_{k-1}x_{k-1} + K_k e_k \\ y_k &= H_k x_k + v_k \end{aligned} \tag{2}$$

where x_k is the parameter vector (state vector), k_k is the Kalman gain, H_k is the measurement matrix (observation matrix), e_k is the one-step prediction error, and y_k is the estimated vector.

x_k and H_k are stated in Eq. 3.

$$x_k = \begin{bmatrix} x_{11}(1,k) \\ . \\ . \\ . \\ x_{1m}(p,k) \\ . \\ . \\ . \\ x_{m1}(1,k) \\ . \\ . \\ . \\ x_{mm}(p,k) \end{bmatrix}, \quad H_k = \begin{bmatrix} y^T(k-1) & . & . & . & 0 \\ . & . & . & . & . \\ . & . & . & . & . \\ . & . & . & . & . \\ 0 & . & . & . & y^T(k-1) \end{bmatrix} \tag{3}$$

where $y^T(k\text{-}1)$=[$y^T(k$-1)…$y^T(k$-p))]. The elements of the state vector are estimated via the Kalman filtering approach. The process and observation noise covariance matrices ($w(k)$,$v(k)$)are updated using a specific method given in Eq. 4.[44]

$$\begin{aligned} v(k) &= (1-\zeta)\times v(k-1) + \zeta \times e(k) \\ w(k) &= \left(I \times \zeta \times tr(z(k))\right)/p \end{aligned} \tag{4}$$

where ζ is the update coefficient, $z(k)$ is the a-posteriori correlation matrix. I is the identity matrix, and × implies the matrix product operator.

The update coefficient determines the time resolution and the smoothing of the AR estimates. The $A_k^{(r)}$ matrices given in Eq. 1 are in the following form given in Eq. 5.

$$A_k^{(r)} = \begin{bmatrix} a_{11}(r,k) & . & . & . & a_{1N}(r,k) \\ . & . & . & . & . \\ . & . & . & . & . \\ . & . & . & . & . \\ a_{N1}(r,k) & . & . & . & a_{NN}(r,k) \end{bmatrix} \quad (5)$$

for $r = 1,...,p$ and their elements are predicted utilizing the adaptive method given in Eq. 1-4. Based on that information, adaptive time-varying connectivity measures are defined on the following Z-transform of the MVAR parameters " $A_k^{(r)}$ " to the frequency domain as in Eq. 6.

$$A(k,f) = I - \sum_{r=1}^{p} A_k^{(r)} z^{-r} \Big|_{z=e^{i2\pi f}} \quad (6)$$

PDC, which is a time-varying connectivity measure, is defined in Eq. 7.[45,46]

$$\Pi_{ij}(k,f) \triangleq \frac{\left|A_{ij}(k,f)\right|^2}{\sum_{m=1}^{p}\left|A_{im}(k,f)\right|^2} \quad (7)$$

PDC quantifies the direct influence from time series to time series , after discounting the effect of all the other time series. The square exponents enhance the accuracy and stability of the estimates while the denominator part permits the normalization of outgoing connections by the inflows.[47]

Classification

Extracted features are then defined as input neurons to the classification algorithm. The output layer should include four neurons for the four classes that represent the four wrist movement states. The number of neurons in the input layer changes according to the length of the feature vector. NNs and SVMs play key roles in machine learning and data analysis. However, it is known that there exist some challenging issues with them such as intensive human intervention, slow learning speed, and poor learning scalability. In this article, we use ELM for the classification. ELMs are a kind of feedforward neural network, which does not necessitate gradient-based backpropagation for learning. It utilizes MP generalized inverse to set its weights. ELM not only learns up to tens of thousands faster than NNs and SVMs but also provides unified implementation for regression, binary, and multi-class applications. ELM is efficient for time series, online sequential, and incremental applications. ELM is also efficient for large datasets.

A single-hidden layer feedforward NN is shown in Figure 2.

The algorithm of a single-hidden layer feedforward NN can be listed as multiplication inputs by weights, adding bias,

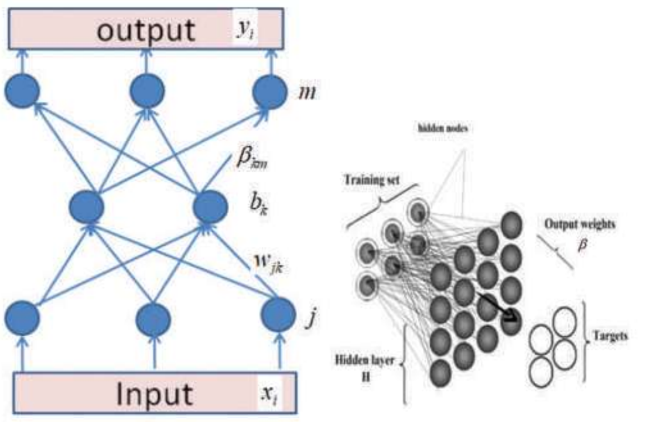


Figure 2: A generic single-hidden layer feedforward neural network architecture[48]

implementing the activation function, repeating the first three steps with a number of layers times, evaluating output, backpropagating, and repeating every step, respectively. The ELM differs from the feedforward NN by removing the repeating steps and replacing the backpropagation step with a matrix inverse operation.

The output of the ELM is evaluated as in Eq. 8.

$$f_L(x) = \sum_{i=1}^{L} \beta_i g_i(x) = \sum_{i=1}^{L} \beta_i g\left(w_i * x_j + b_i\right), j = 1,...,N \quad (8)$$

where L is the number of hidden units, N is the number of training samples, β_i is the weight vector between *ith* hidden layer and output, w is the weight vector between the input and hidden layer, $g(.)$ is an activation function, b is a bias vector, and x is an input vector.

β is a special matrix due to the pseudo-inverse operation. Eq. 8 can be represented in a compact form as in Eq. 9.

$T = H\beta$

$$H = \begin{bmatrix} g(w_1 * x_1 + b_1) & . & . & . & g(w_L * x_1 + b_L) \\ . & . & . & . & . \\ . & . & . & . & . \\ . & . & . & . & . \\ g(w_1 * x_N + b_1) & . & . & . & g(w_L * x_N + b_L) \end{bmatrix}_{N\times L}$$

$$\beta = \begin{bmatrix} \beta_1^T \\ . \\ . \\ . \\ \beta_L^T \end{bmatrix}_{L\times m}, T = \begin{bmatrix} t_1^T \\ . \\ . \\ . \\ t_N^T \end{bmatrix}_{N\times m} \quad (9)$$

where m is the number of outputs, H is called the hidden layer output matrix, and T is a training data target matrix.

Since there are no certain rules for choosing the number of hidden neurons, the EM-ELM method was used for finding the optimal configuration.

Machine interface design

JACO robotic arm, which is a generic 6-axis robotic manipulator with a three-fingered hand, can be used for the physical realization and machine interface of the proposed algorithm. The arm has six degrees of freedom in total with a maximum reach of 90 cm radius sphere and a maximum speed of 30 cm/s. It is made of three sensors: force, position, and acceleration. This arm should be suitable for a person with a disability of the upper arm and can be placed in a wheelchair. The upper arm of the robot is made of three links which are similar to the upper limb of the human body, as shown in Figure 3.

An API, which gives freedom of control to users, is provided by the manufacturers. The subject needs to control the movement of the robot arm toward a given target by using four mental commands: Forward (F), Backward (B), Left (L), Right (R), and No Movement command. To end the movement of the robot arm, the subject would generate a "No Movement" command by taking no action. The arm could move on two axes (x and y) and in four directions (forward (+y), backward (−y), left (+x), and right (−x)). The control signals are generated according to the mental commands. To move the arm forward, the subject imagines the forward wrist movement, and performs backward wrist movement to move the arm backward. The subject imagines moving his/her right wrist to move the robotic arm right and imagines moving his/her left wrist to move the robotic arm left. Forward differential kinematics gives the relation between joint velocities and tip velocity. This relation is in matrix form called Jacobian. To form Jacobian, first propagation matrices are created. The propagation matrix given in Eq. 10 produces the relation between sequenced joints.

$$\overline{\overline{V}}_b = \begin{bmatrix} \overrightarrow{w_b} \\ \overrightarrow{V_b} \end{bmatrix} = \begin{bmatrix} I & 0 \\ -\hat{l}_{ab} & I \end{bmatrix} \begin{bmatrix} \overrightarrow{w_a} \\ \overrightarrow{V_a} \end{bmatrix} + \begin{bmatrix} \vec{h} \\ \vec{0} \end{bmatrix} \dot{\theta}$$

$$\overline{\overline{V}}_b = \varphi_{ba} \overline{\overline{V}}_a + \overline{\overline{H}}\dot{\theta} \tag{10}$$

For fixed base manipulators, the robot propagation matrix can be constructed by using these joint relations. This

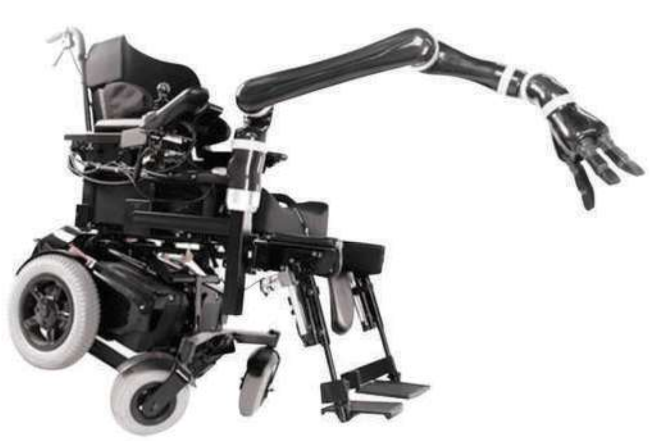


Figure 3: JACO robotic arm mounted on an electric wheelchair[49]

matrix is called φ and gives all joint velocity vectors. To reach tip velocity, Eq. 11 should be evaluated.

$$\overline{\overline{V}}_t = \varphi_t \, \varphi \, H \underline{\dot{\theta}} \tag{11}$$

By using this equation, the robot tip velocity can be calculated concerning joint velocities. Hence, the Jacobian is given in Eq. 12.

$$J = \varphi_t \, \varphi \, H \tag{12}$$

MATLAB Simulink Code is created by using the above equations. First, the Jacobian is implemented, after that according to Rodrigues' formula, the rotation matrices for each joint are built. The coordinate frames for each joint change when the robot moves. For this reason, coordinate frames are updated at each sample time by using joint velocities, sample period, and rotation axes. However, this update mechanism is correct if the joint velocity is constant along the sample period. The selected sample time is 1 ms. The Simulink block diagram of the robot Jacobian is shown in Figure 4.

Inputs are base positions (6 × 1) and Robot Joint Angular Positions (6 × 1), and Outputs are Tip Point Position and Load Center Position. Tip and load center positions are fed to these shapes at the VRML file to confirm the operation of the moving base kinematic code. Only the angular part of the base position is used. These angles are Euler XYZ angles and converted to vector-angle representation and then fed to VRML file. Two common ways to create a dynamical model of a robot are Newton–Euler-based and Euler–Lagrange-based dynamical modeling. The Newton–Euler method, which is a vectorial approach, was explained in this article for dynamical modeling.

The dynamic model stated in Eq. 13 describes the relationship between joint forces and joint accelerations. The forces acting on a joint are the sequenced joint force and linear and angular inertia of the joint.

$$\overrightarrow{\tau_k} = \overrightarrow{\tau_{k+1}} + \overrightarrow{l_{k,k+1}} \times \overrightarrow{f_{k+1}} + \overrightarrow{l_{k,c}} \times \dot{\overrightarrow{v_k}} \, m_k + \frac{d}{dt}(I_k \overrightarrow{w_k})$$

$$\overrightarrow{f_k} = \overrightarrow{f_{k+1}} + m_k \frac{d}{dt}(\overrightarrow{v_k} + \overrightarrow{w_k} \times \overrightarrow{l_{k,c}}) \tag{13}$$

The compact form of Eq. 13 can be given as in Eq. 14.

$$\overline{\overline{F}}_k = \varphi^T_{(k+1,k)} \overline{\overline{F}}_{k+1} + M_k \dot{\overline{\overline{V}}}_k + \overline{\overline{b}}_k$$

$$M_k = \begin{bmatrix} I_k & m_k \, \hat{l}_{k,c} \\ -m_k \, \hat{l}_{k,c} & m_k \, I \end{bmatrix} \quad \dot{\overline{\overline{V}}}_k = \varphi_{k,k-1} \dot{\overline{\overline{V}}}_{k-1} + \overline{\overline{H}}_k \ddot{\theta}_k + \overline{\overline{a}}_k$$

$$\overline{\overline{b}}_k = \begin{bmatrix} \overrightarrow{w_k} \times I_k \overrightarrow{w_k} \\ m_k \overrightarrow{w_k} \times (\overrightarrow{w_k} \times \overrightarrow{l_{k,c}}) \end{bmatrix} \quad \overline{\overline{a}}_k = \begin{bmatrix} \overrightarrow{w_{k-1}} \times \overrightarrow{w_k} \\ \overrightarrow{w_{k-1}} \times (\overrightarrow{w_{k-1}} \times \overrightarrow{l_{k-1,k}}) \end{bmatrix} \tag{14}$$

If the robot joint dynamic equations are put together, Eq. 15 is obtained.

$$\underline{F} = \varphi^T (M(\underline{\dot{V}}) + \underline{b} + \varphi_t^{\,T} \overline{\overline{F}}_k) \tag{15}$$

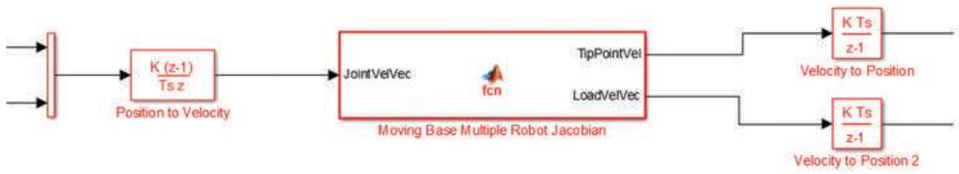


Figure 4: Simulink block diagram of the robot Jacobian

The above equation includes all forces acting on the robot joints. To reach forces that make the robot joint accelerate, the force vector $\underline{F}$ is multiplicated with the rotation axis matrix as in Eq. 16.

$$\tau = H^T \underline{F} = \mu\ddot{\theta} + C + J^T F_t \tag{16}$$

Based on MATLAB kinematic code for moving base-robot systems, a dynamic model is built. The torque expressions are related to Jacobian terms. Because of that dynamic model code can be easily improved by starting with kinematic code. The first addition to kinematic code is creating μ the matrix according to base velocities, base accelerations, and fixed base dual robot "μ" The fixed base dual robot "μ" requires robot joint masses, inertia tensors, position vector of the center of mass, and joint angular velocity vectors. Masses, inertia tensors, and center of gravities concerning joint frames are found by investigating the SolidWorks drawing of the robot arm. Joint angular velocity vectors are provided from the previous iteration with zero initial points. After evaluating the fixed base "μ" the moving base features are added to the "μ" and a new "μ" is created. These additions are base mass matrix which depends on base mass, inertia tensor, the center of gravity, and base-robot mass matrices. C matrix is created based on H, φ, a vector, b vector, and mass matrix. After calculating these matrices, the forward dynamics of the robot with a moving base are simulated. Above torque, expression is prepared as torque input and acceleration output given in Figure 5 and Eq. 17.

$$\ddot{\theta} = \mu^{-1}(\tau - C - J^T F_t) \tag{17}$$

The inputs are Tip Forces (12 × 1), Base Forces (6 × 1), Joint Torques (12 × 1), and Joint Angular Velocity Vectors (12 × 1), respectively. The outputs are Accelerations (18 × 1), Tip Point Velocities (12 × 1), and Load Velocity (6 × 1), respectively.

Results

After applying the adaptive AR-based PDC feature extraction method to the train data matrix for each class, the time-variant estimated MVAR parameters were obtained. The size of the parameter vector for each class is 32000 × 200.The rows represent time in terms of data samples, and the columns show the parameter values estimated over samples. The most significant values of the adaptive PDC measures at a 99% level of significance after applying the surrogate data method are given in Figure 6 for each class of wrist movement states.

The adaptive PDC results show the connectivity between channels. It can be inferred from Figure 6 that the most significant direct coupling activities emerge from channel 3 to channels 1, 2, 4, and 5, respectively. The model parameters are accurately tracked and each class has different patterns.

The adaptive AR-based PDC feature matrices are concatenated and the feature train matrix is built, and the obtained train matrix is fed into the ELM. The size of the feature train matrix is 128000 × 200.

A principle component analysis-based dimension reduction algorithm, which is an orthogonal transformation constructing the relevant features, is applied to the feature train matrix because the number of extracted features is too much. The first four components having the most variance in the feature data are selected. After the dimensionality reduction process, the size of the feature train matrix becomes 128000 × 4.

The obtained data are randomly divided into training, testing, and validation sets. Every time the system is executed, samples were used for each task (70400 samples [55%] are used for training, 25,600 samples are used for validation [20%], and the remaining 32000 trials were used for the test [25%]). ELM has trained for 100 epochs (iterations) by incrementing the number of neurons in the hidden layer (hidden node). For each hidden node, the classification accuracy was evaluated in terms of the "root mean squared error" criteria. Figure 7 represents the performance curve sensitivity concerning the hidden node.

The rmse error approaches the equilibrium state while increasing the hidden nodes. Therefore, the number of hidden neurons is chosen as 100. The training ratio is selected using a trial and error process as 0.7. The activation function of the output layer is chosen as "sigmoid."

The confusion matrices are given in Figure 8.

Test data are used for the external validation process. The size of the reduced feature test matrix is 58400 × 4. After feeding the test data to the trained ELM model, the external validation accuracy of the ELM is found as 84.88%.

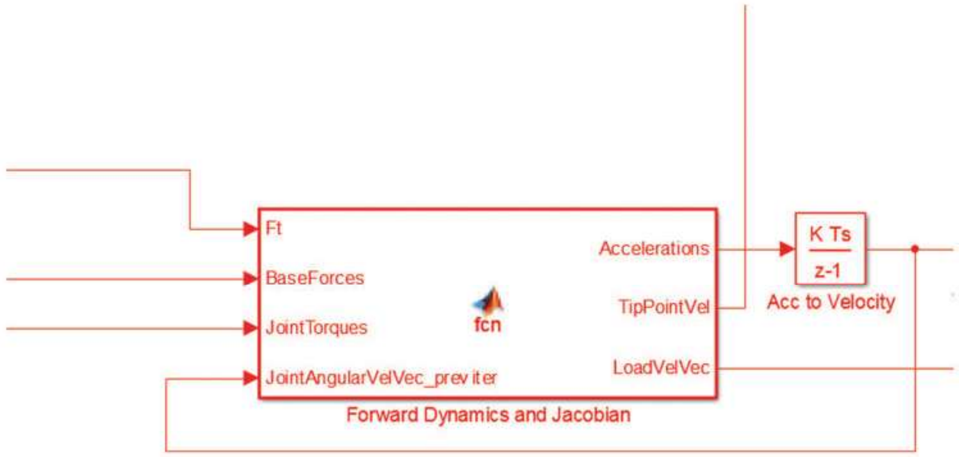


Figure 5: Simulink diagram of the forward dynamics

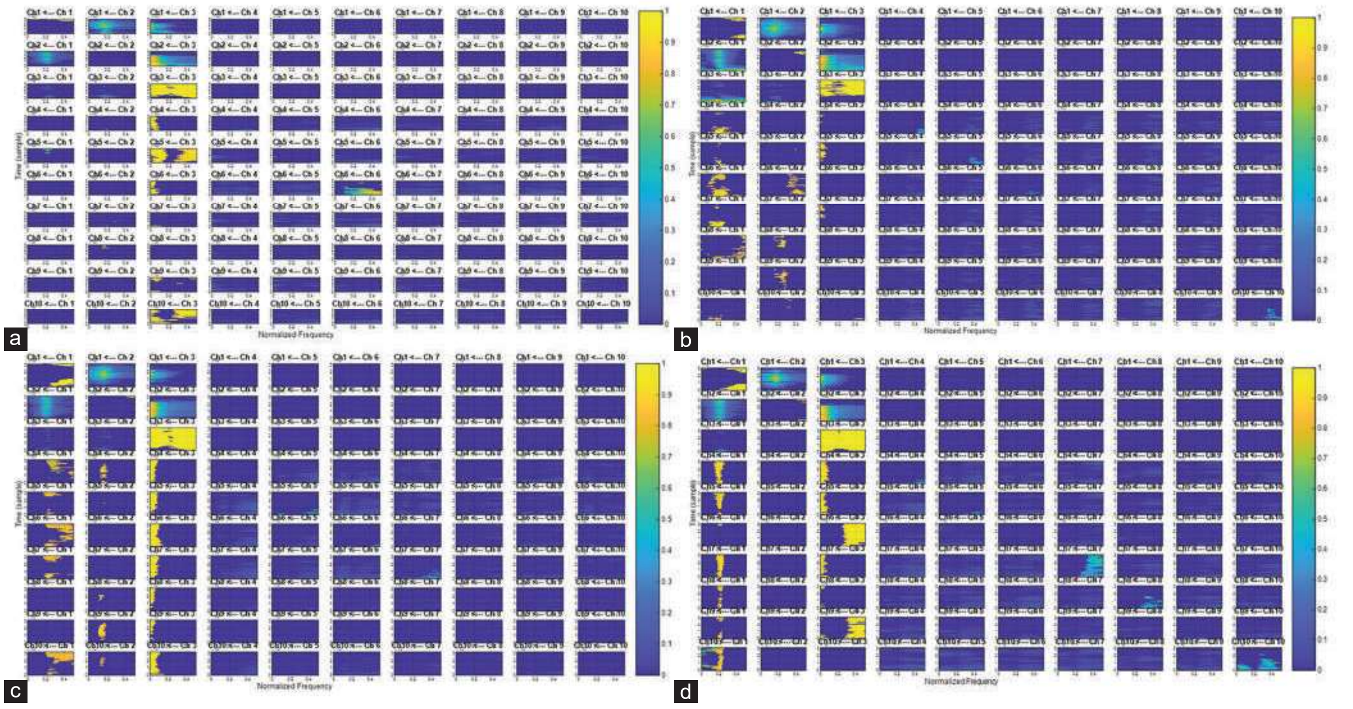


Figure 6: PDC for the simulated model utilizing the Kalman filtering approach. The x-axis represents normalized frequency ([0 0.5] corresponding to [0 0 F_s/2 and the y-axis represents the time direction in terms of data samples. Wrist movement states: (a) right, (b) forward, (c) left, and (d) backward. PDC: Partial directed coherence

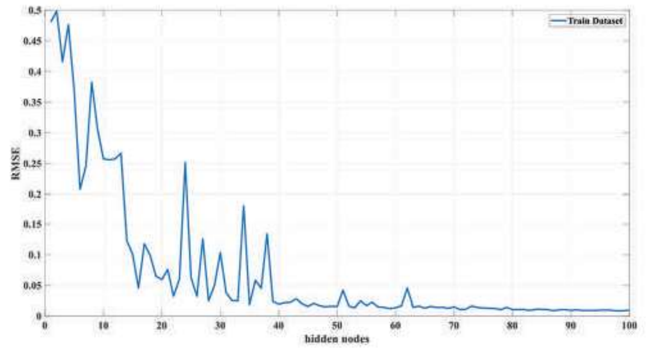


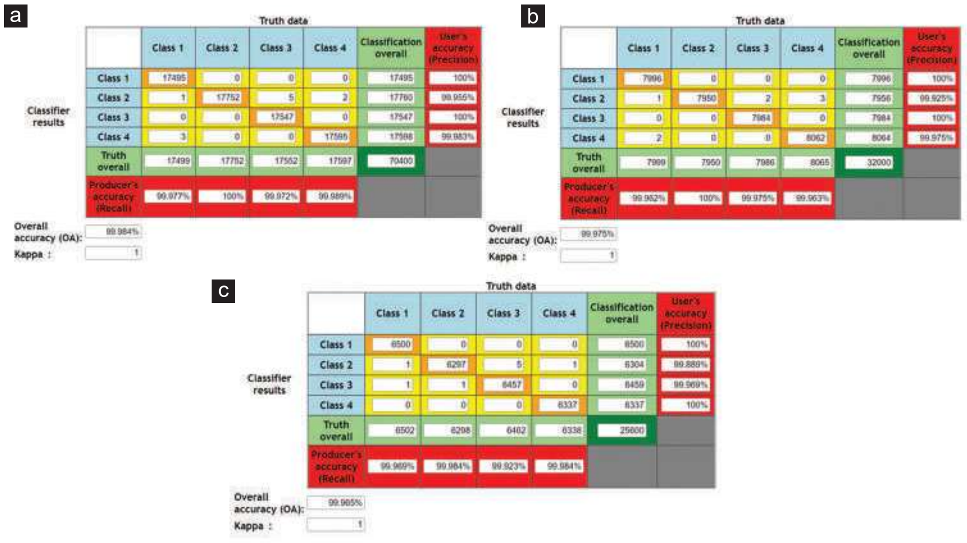
Figure 7: Hidden node selecting criteria

A comparative analysis was conducted over the external validation results using various classification methods and given in Table 1.

According to the external validation results, the ELM outperforms the other machine learning models, and also the computational speed is extremely fast.

Discussion and Conclusive Summary

The obtained results clarify that by applying the proposed methodology, control signals from an individual's MEG signals can control the robotic arm. Time-varying cortical neural connectivity features are



a

Truth data

Classifier results	Class 1	Class 2	Class 3	Class 4	Classification overall	User's accuracy (Precision)
Class 1	17495	0	0	0	17495	100%
Class 2	1	17752	5	2	17760	99.955%
Class 3	0	0	17547	0	17547	100%
Class 4	3	0	0	17595	17598	99.983%
Truth overall	17499	17752	17552	17597	70400	
Producer's accuracy (Recall)	99.977%	100%	99.972%	99.989%		

Overall accuracy (OA): 99.984%
Kappa : 1

b

Truth data

Classifier results	Class 1	Class 2	Class 3	Class 4	Classification overall	User's accuracy (Precision)
Class 1	7996	0	0	0	7996	100%
Class 2	1	7950	2	3	7956	99.925%
Class 3	0	0	7984	0	7984	100%
Class 4	2	0	0	8062	8064	99.975%
Truth overall	7999	7950	7986	8065	32000	
Producer's accuracy (Recall)	99.962%	100%	99.975%	99.963%		

Overall accuracy (OA): 99.975%
Kappa : 1

c

Truth data

Classifier results	Class 1	Class 2	Class 3	Class 4	Classification overall	User's accuracy (Precision)
Class 1	6500	0	0	0	6500	100%
Class 2	1	6297	5	1	6304	99.889%
Class 3	1	1	6457	0	6459	99.969%
Class 4	0	0	0	6337	6337	100%
Truth overall	6502	6298	6462	6338	25600	
Producer's accuracy (Recall)	99.969%	99.984%	99.923%	99.984%		

Overall accuracy (OA): 99.985%
Kappa : 1

Figure 8: Confusion matrices: (a) training, (b) test, (c) validation. Class 1: right, Class 2: forward, Class 3: left, Class 4: backward

Table 1: Comparison of external validation classification results

Classifier	Sensitivity (%)	Precision (%)	Specificity (%)	F-measure (%)	Accuracy (%)
SVM	83.6	82.88	82.71	83.25	83.17
ELM	85.12	83.73	83.62	84.95	84.88
NB	75.45	77.03	76.74	76.23	76.08
k-NN	78.54	67.93	76.79	78.23	77.71
ANN	79.78	79.23	76.79	79.50	78.43

SVM: Support vector machine, ELM: Extreme learning machine, NB: Naive Bayesian, k-NN: k-nearest neighbor, ANN: Artificial neural network

strong biomarkers for characterizing the MEG patterns, which originated from wrist movements. According to the external validation classification results, ELM outperforms the other classifiers by achieving the highest accuracy. The accuracy of which these control signals are extracted from the MEG signals is extensively measured using the wide industry-employed receiver operator characteristic curves. After the signal processing part is taken care of, it is necessary to establish the dynamic model of the robot, create and simulate the solid model, eliminate the errors caused by the system dynamics in the robot joints, and synchronize with the brain signals. MEG-based BCI systems are capable to be robustly utilized as a control device through the proposed framework.

Patient informed consent

There is no need for patient informed consent.

Ethics committee approval

There is no need for ethics committee approval.

Conflicts of interest

There are no conflicts of interest to declare.

Financial support and sponsorship

No funding was received.

Author contribution subject and rate:

- Caglar Uyulan (100%): Design the research, data collection, and analyses and wrote the whole manuscript.

References

1. Mellinger J, Schalk G, Braun C, Preissl H, Rosenstiel W, Birbaumer N, *et al*. An MEG-based brain-computer interface (BCI). Neuroimage 2007;36:581-93. [Doi: 10.1016/j. neuroimage. 2007.03.019].
2. David G, Corral GH, Zentina LM. Using EEG / MEG data of cognitive processes in brain-computer interfaces. AIP Conf Proc 2008;1032:7. [Doi: 10.1063/1.2979300].
3. Çağlayan O, Arslan RB. Robotic arm control with brain computer interface using P300 and SSVEP. Biomed Eng 2013. p. 141-4. [Doi: 10.2316/p. 2013.791-082].
4. NandikollaV, Medina Portilla DA. Teleoperation robot control of a hybrid EEG-based BCI arm manipulator using Ros. J Rob 2022;2022:14. [Doi: 10.1155/2022/5335523].
5. Qasim M, Ismael OY. Shared control of a robot arm using BCI and computer vision. J Eur Syst Automat 2022;55:139-46. [Doi: 10.18280/jesa. 550115].
6. Musk E, Neuralink. An integrated brain-machine interface platform with thousands of channels. J Med Internet Res 2019;21:e16194. [Doi: 10.2196/16194].
7. Spüler M, Rosenstiel W, Bogdan M. Adaptive SVM-Based classification increases performance of a meg-based brain-computer interface (BCI). In: Artificial Neural Networks and Machine Learning – ICANN. Berlin, Heidelberg: Springer;

2012;7552:669-76. [Doi: 10.1007/978-3-642-33269-2_84].

8. Rezaei S, Tavakolian K, Nasrabadi AM, Setarehdan SK. Different classification techniques considering brain computer interface applications. J Neural Eng 2006;3:139-44. [Doi: 10.1088/1741-2560/3/2/008].
9. Corsi MC, Chavez M, Schwartz D, Hugueville L, Khambhati AN, Bassett DS. Integrating EEG and MEG signals to improve motor imagery classification in brain-computer interface. Int J Neural Syst 2019;29:12. [Doi: 10.1142/s0129065718500144].
10. Sabra NI, Abdel Wahed M. The use of meg-based brain computer interface for classification of wrist movements in four different directions. In: 2011 28th National Radio Science Conference (NRSC). Cairo, Egypt: IEEE; 2011. [Doi: 10.1109/nrsc. 2011.5873644].
11. Daliri MR. A hybrid method for the decoding of spatial attention using the meg brain signals. Biomed Signal Proc Control 2014;10:308-12. [Doi: 10.1016/j.bspc. 2012.12.005].
12. Scherer R, Vidaurre C. Motor imagery based brain-computer interfaces. Smart Wheelchairs Brain Comput Interfaces 2018. p. 171-95. [Doi: 10.1016/b978-0-12-812892-3.00008-x].
13. Uyulan C, Erguzel TT. Analysis of time – Frequency EEG feature extraction methods for mental task classification. Int J Comput Intell Syst 2017;10:1280. [Doi: 10.2991/ijcis. 10.1.87].
14. Uyulan C, Erguzel T. Comparison of wavelet families for mental task classification. J Neurobehav Sci 2016;3:59. [Doi: 10.5455/jnbs. 1454666348].
15. Sun L, Feng ZR. Classification of Imagery Motor EEG data with wavelet denoising and features selection. In: 2016 International Conference on Wavelet Analysis and Pattern Recognition (ICWAPR). Jeju, Korea (South): IEEE; 2016. [Doi: 10.1109/icwapr. 2016.7731641].
16. Murugappan M, Murugappan S Human emotion recognition through short time electroencephalogram (EEG) signals using fast fourier transform (FFT). In: 2013 IEEE 9th International Colloquium on Signal Processing and Its Applications. Kuala Lumpur, Malaysia: IEEE; 2013. [Doi: 10.1109/cspa. 2013.6530058].
17. Zaveri HP, Duckrow RB, Spencer SS. On the use of bipolar montages for time-series analysis of intracranial electroencephalograms. Clin Neurophysiol 2006;117:2102-8. [Doi: 10.1016/j.clinph. 2006.05.032].
18. Ludwig KA, Miriani RM, Langhals NB, Joseph MD, Anderson DJ, Kipke DR. Using a common average reference to improve cortical neuron recordings from microelectrode arrays. J Neurophysiol 2009;101:1679-89.
19. Mourino J, del R Millan J, Cincotti F, Chiappa S, Jane R, Babiloni F. Spatial filtering in the training process of a brain computer interface. In: 2001 Conference Proceedings of the 23rd Annual International Conference of the IEEE Engineering in Medicine and Biology Society. Istanbul, Turkey: IEEE; 2001. [Doi: 10.1109/iembs. 2001.1019016].
20. Saha PK, Rahman MA, Alam MK, Ferdowsi A, Mollah MN. Common spatial pattern in frequency domain for feature extraction and classification of multichannel EEG signals. SN Comput Sci 2021;2:11. [Doi: 10.1007/s42979-021-00586-9].
21. Uyulan C, Ergüzel TT, Tarhan N. Entropy-based feature extraction technique in conjunction with Wavelet packet transform for multi-mental task classification. Biomed Eng/Biomed Technik 2019;64:529-42. [Doi: 10.1515/bmt-2018-0105].
22. Güçlü U, Güçlütürk Y, Loo CK. Evaluation of fractal dimension estimation methods for feature extraction in motor imagery based brain computer interface. Procedia Comput Sci 2011;3:589-94. [Doi: 10.1016/j.procs. 2010.12.098].
23. Khadijah Nik AN, Yeon-Mo Y. Applying Kalman filter in EEG-based brain computer interface for motor imagery classification. In: 2013 International Conference on ICT Convergence (ICTC). Jeju, Korea (South): IEEE; 2013. [Doi: 10.1109/ictc. 2013.6675451].
24. Nicolas-Alonso LF, Gomez-Gil J. Brain computer interfaces, a review. Sensors (Basel) 2012;12:1211-79.
25. Madi MK, Karameh FN. Hybrid cubature kalman filtering for identifying nonlinear models from sampled recording: Estimation of neuronal dynamics. PLoS One 2017;12:e0181513.
26. Ma X. The research of brain-computer interface based on AAR parameters and neural networks classifier. In: Proceedings of 2011 International Conference on Computer Science and Network Technology. Harbin, China: IEEE; 2011. [Doi: 10.1109/iccsnt. 2011.6182491].
27. Nilesh R, Sunil W. Improving extreme learning machine through optimization a review. In: 2021 7th International Conference on Advanced Computing and Communication Systems (ICACCS). Coimbatore, India: IEEE; 2021. [Doi: 10.1109/icaccs51430.2021.9442007].
28. Lefebvre G, Cumin J. Recognizing human actions based on extreme learning machines. In: Proceedings of the 11th Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications. Italy: Roma; 2016. [Doi: 10.5220/0005675004780483].
29. Mukherjee H, Das S, Ghosh S, Obaidullah SM, Santosh KC, Das N, *et al.* A study on the extreme learning machine and its applications. In: Document Processing Using Machine Learning. ImprintChapman and Hall/CRC; 2019. p. 43-52. [Doi: 10.1201/9780429277573-4].
30. Matias T, Souza F, Araújo R, Gonçalves N, Barreto JP. On-line sequential extreme learning machine based on recursive partial least squares. J Process Control 2015;27:15-21. [Doi: 10.1016/j.jprocont. 2015.01.004].
31. Huang GB, Chen L, Siew CK. Universal approximation using incremental constructive feedforward networks with random hidden nodes. IEEE Trans Neural Netw 2006;17:879-92.
32. Rong HJ, Ong YS, Tan AH, Zhu Z. A fast pruned-extreme learning machine for classification problem. Neurocomputing 2008;72:359-66. [Doi: 10.1016/j.neucom. 2008.01.005].
33. Feng G, Huang GB, Lin Q, Gay R. Error minimized extreme learning machine with growth of hidden nodes and incremental learning. IEEE Trans Neural Netw 2009;20:1352-7. [Doi: 10.1109/tnn. 2009.2024147].
34. Silva DN, Pacifico LD, Ludermir TB. An evolutionary extreme learning machine based on group search optimization. In: 2011 IEEE Congress of Evolutionary Computation (CEC). New Orleans, LA, USA: IEEE; 2011. [Doi: 10.1109/cec. 2011.5949670].
35. Lan Y, Soh YC, Huang GB. Two-stage extreme learning machine for regression. Neurocomputing 2010;73:3028-38. [Doi: 10.1016/j.neucom. 2010.07.012].
36. Deng W, Zheng Q, Chen L. Regularized extreme learning machine. In: 2009 IEEE Symposium on Computational Intelligence and Data Mining. Nashville, TN, USA: 2009 IEEE Symposium on Computational Intelligence and Data Mining; 2009. [Doi: 10.1109/cidm. 2009.4938676].
37. X-jian D, X-guang L, Xu X. An optimization method of extreme learning machine for regression. In: Proceedings of the 31st Annual ACM Symposium on Applied Computing. 2016. p. 891-3 [Doi: 10.1145/2851613.2851882].
38. Ding S, Zhang N, Xu X, Guo L, Zhang J. Deep extreme learning machine and its application in EEG classification. Math Probl

Eng 2015;2015:1-11. [Doi: 10.1155/2015/129021].

39. Stosic D, Stosic D, Ludermir T. Voting based Q-generalized extreme learning machine. Neurocomputing 2016;174:1021-30. [Doi: 10.1016/j.neucom. 2015.10.028].
40. Omidvarnia AH, Mesbah M, Khlif MS, O'Toole JM, Colditz PB, Boashash B. Kalman filter-based time-varying cortical connectivity analysis of newborn EEG. Annu Int Conf IEEE Eng Med Biol Soc 2011;2011:1423-6.
41. Winterhalder M, Schelter B, Hesse W, Schwab K, Leistritz L, Klan D, *et al.* Comparison of linear signal processing techniques to infer directed interactions in multivariate neural systems. Signal Process 2005;85:2137-60. [Doi: 10.1016/j.sigpro. 2005.07.011].
42. Arnold M, Milner XH, Witte H, Bauer R, Braun C. Adaptive AR modeling of nonstationary time series by means of kalman filtering. IEEE Trans on Biomed Eng 1998;45:553-62. [Doi: 10.1109/10.668741].
43. Wang F, Balakrishnan V. Robust kalman filters for linear time-varying systems with stochastic parametric uncertainties. IEEE Trans Signal Process 2002;50:803-13. [Doi: 10.1109/78.992124].
44. Pagnotta MF, Plomp G, Pascucci D. A regularized and smoothed general linear kalman filter for more accurate estimation of time-varying directed connectivity(). Annu Int Conf IEEE Eng Med Biol Soc 2019;2019:611-5.
45. Pascucci D, Rubega M, Plomp G. Modeling time-varying brain networks with a self-tuning optimized kalman filter. PLoS Comput Biol 2020;16:e1007566.
46. Ghumare EG, Schrooten M, Vandenberghe R, Dupont P. A time-varying connectivity analysis from distributed EEG sources: A simulation study. Brain Topogr 2018;31:721-37.
47. UyulanC, de la Salle S, Erguzel TT, Lynn E, Blier P, Knott V, *et al.* Depression diagnosis modeling with advanced computational methods: Frequency-domain emvar and deep learning. Clin EEG Neurosci 2021;53:24-36. [Doi: 10.1177/15500594211018545].
48. Hu J, Zhang J, Zhang C, Wang J. A new deep neural network based on a stack of single-hidden-layer feedforward neural networks with randomly fixed hidden neurons. Neurocomputing 2016;171:63-72. [Doi: 10.1016/j.neucom. 2015.06.017].
49. Paul I, Ghosh S, Konar A. Voice Command decoding for position control of Jaco robot arm using a type-2 fuzzy classifier. In: 2020 IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT). Bangalore, India: IEEE; 2020. [Doi: 10.1109/conecct50063.2020.9198684].