IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 21 June 2024, accepted 7 July 2024, date of publication 18 July 2024, date of current version 20 August 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3430838

RESEARCH ARTICLE

EEG Motor Imagery Classification by Feature Extracted Deep 1D-CNN and Semi-Deep Fine-Tuning

MOHAMAD TAGHIZADEH[1], FATEMEH VAEZ[2], AND MIAD FAEZIPOUR[3], (Senior Member, IEEE)

[1]Department of Electrical Engineering, School of Electrical Engineering, Iran University of Science and Technology, Tehran 16846-13114, Iran
[2]Department of Electrical Engineering, School of Electrical Engineering, Shahrekord University, Shahrekord 88186-34141, Iran
[3]School of Engineering Technology, Electrical and Computer Engineering Technology, Purdue University, West Lafayette, IN 47907, USA

Corresponding author: Mohamad Taghizadeh (Mohamad.Taghizadeh.Official@gmail.com)

ABSTRACT The main goal of this paper is to introduce a Motor Imagery (MI) classification system for electroencephalography (EEG) that is extremely precise. To achieve this goal, we propose using a feature-extracted deep one-dimension (1D) convolutional neural network (CNN) which provides a model that can be further improved through hyperheuristic multi-objective evolutionary search. We can improve the classification performance by training this deep CNN model with feature-extracted data from the Physionet MI dataset. We also present a semi-deep fine-tuning approach that can yield improvements with just four epochs. Our findings using the Physionet MI dataset illustrate that the approach we suggest surpasses most contemporary techniques used for classifying EEG signals. Our system is computationally efficient and can be trained using reliable EEG data for individual patients, allowing for accurate classification of their EEG records. Because of its straightforward and parameter-independent characteristics, our system is versatile and can be utilized with any EEG dataset.

INDEX TERMS Brain-computer interface, classification, deep learning, electroencephalogram, transfer learning.

I. INTRODUCTION

Brain-Computer Interfaces (BCI) are novel technologies that are advancing in their development for real-world usage. Engaging with electronic technologies establishes communication between the brain and external environments [1]. Over the past few decades, BCI technology has seen progress due to the development of machine learning algorithms, resulting in better classification accuracy and performance. BCIs have vast applications in augmenting, and treating cognitive [2], [3] and sensory-motor impairments [4], [5], as well as recreational purposes [6], [7], [8].

The electroencephalogram (EEG) is a frequently utilized brain signal in BCIs due to its noninvasive nature and superior temporal resolution [9]. Different neurophysiological patterns have been utilized in brain-computer interfaces (BCIs), such as motor imagery (MI), visual evoked potentials (VEP), readiness potentials, and P300. Motor imagery based BCIs and VEPs have received greater focus than others. It has been shown that MI, unlike many other types of BCI, does not require external stimuli and can be used in a self-directed manner closer to natural control [10], [11], [12], [13].

In macaque V1, researchers used scalp EEG and multiscale recordings of Multiunit Activity (MUA) and Local Field Potential (LFP) to describe how suppression affects the temporal frequency (TF) axis and alters Steady-State Visual Evoked Potential (SSVEP) responses. The modulation varies depending on the two driving stimuli's relative temporal frequency and orientation. Although this technique is efficient, it necessitates individuals to gaze at a monitor and is prone to significant external influences [14]. In contrast, MI involves imagining specific movement that leads to

The associate editor coordinating the review of this manuscript and approving it for publication was Md Kafiul Islam.

the desynchronization of neurons in the contralateral and ipsilateral sensorimotor cortex. Although this approach requires training, it can be applied to various situations.

The use of EEG-based BCI for motor imagery has great potential in the field of medical applications, such as stroke rehabilitation, wheelchair control, prostheses control, exoskeleton control, cursor control, speller, and thought-to-text conversion. This promising technology is highly in demand due to its potential to improve the lives of individuals with motor disabilities and can have significant medical implications [15].

In addition to its usage in machine learning, Deep Learning (DL) has been applied in computer vision and natural language processing, surpassing traditional machine learning techniques. Machine learning approaches that rely on assumptions, such as support vector machines (SVMs), linear discriminant analysis (LDAs), random forests, and transformer-based models [16], [17], are all based on the extraction of relevant features. The extraction of features may therefore exclude some essential features. Using DL models, features that are difficult to extract via traditional methods may be learned.

Deep neural networks can learn thanks to their ability to discover complex feature representations from raw data. Neuro-engineering researchers are increasingly interested in using DL for BCI development because it eliminates the need for domain-specific expertise in manual feature extraction, which is required in traditional BCIs [18]. EEG-based BCIs have recently been developed using DL approaches [1]. Artificial neural networks organized in layers form the basis for DL [19]. Each layer uses various nonlinear units for feature extraction and nonlinear signal mapping. Deep learning methods are employed in feature extraction and classification in MI EEG-based BCI systems [20]. In various brain activity research and clinical investigations, wavelet transform coefficient matrices categorize event-related potential (ERP) and EEG signals. The performance of wavelet-based classifiers can be significantly enhanced by utilizing the cone of influence (COI) of the continuous wavelet transform (CWT), as proposed in [21].

In MI-based EEG classification tasks, convolutional neural networks (CNNs) have been successfully used in various DL paradigms. Convolutional neural networks (CNNs or ConvNets) are a type of artificial neural network (ANNs) that are primarily employed for analyzing visual images [22]. In CNNs, a set of convolutional kernels or jointly weighted filters are employed to slide across the input features, producing a translation-equivalent output referred to as feature maps. Each convolutional layer consists of multiple convolutional filters (or kernels). Convolutional layers are activated and trained by sliding a filter over a spatially organized input layer and performing a convolution operation on points on the vertices of the grid. The two essential characteristics of a convolutional layer are its parameter sharing and sparse interaction. We can concentrate on local features in the spatial domain by employing smaller filters, resulting in sparse interaction. Furthermore, the filter's connection weights are multiplied by the number of locations visited in the input pattern, which allows the convolutional layer to be trained for multiple locations through parameter sharing.

The most widely used technique for extracting distinctive features from EEG data during MI tasks is common spatial patterns (CSP) [23]. CSP develops spatial filters that emphasize the difference between two classes of data. Using these filters to extract spatial features has been shown to significantly improve the performance of brain-computer interfaces based on motor imagery [24]. Several techniques exist to enhance performance utilizing the CSP extension algorithm [23], [24], [25], [26], [27]. The sparse common spatial pattern (SCSP) method, as outlined in [25], enhances classification performance by reducing the number of channels used for the common spatial filter while maintaining classification accuracy. The success of the SCSP method suggests that employing all channels or generating a global CSP may not be the best approach for EEG. The locally generated common spatial pattern (LRCSP) method, detailed in [23], improves performance by forming local regions through the grouping of adjacent channels starting from several central channels.

Moreover, their nonlinear transformations allow deep neural networks to extract low-level features without preprocessing. They can therefore be used for classifying and predicting EEG signals using classification and regression techniques [28], [29], [30], [31], [32]. The discrete wavelet transform was employed in [33] to identify and filter out unwanted frequency components from the raw EEGs.

DL can be used to develop BCI systems for rehabilitation therapy using EEG inputs. Recent research has employed Transformer models based on EEG signals to detect sleep stages, classify imagined speech, and recognize emotions [34], [35], [36], [37], [38], [39]. Dose et al. [40] reported achieving a classification accuracy of 65.73% by utilizing a range of EEG channel setups (from 9 to 64) to categorize raw signals from a 4-task MI dataset. They used a creative per-person performance enhancement approach, and this technique allows for the model to be extended to individuals using transfer learning. Tang et al. [41] proposed a new method using CEMD (Conditional Empirical Mode Decomposition) and 1DMSCNN (One-dimensional Multiscale Convolutional Neural Network) to enhance the classification accuracy of MI EEG signals. The standard EMD algorithm is optimized in their work by using the condition addition strategy to select effective intrinsic modal components (IMFs). For each participant, an average accuracy of 85.83% was achieved. Lun et al. [42] introduced a CNN structure along with the appropriate network architecture and parameters that can classify raw MI-EEG data from symmetric electrodes on the left and right hemispheres of the brain without the need for preprocessing or artificial

feature extraction techniques. The experimental outcomes demonstrate that their models converge effectively on training and testing datasets. The Physionet database serves as the source of data. Their model achieved an accuracy of 97.28%. Chawla et al. described a method for creating classifiers from imbalanced datasets in [43]. The researchers demonstrated that integrating their over-sampling method for the minority class (abnormal) with the under-sampling of the majority class (typical) could improve the performance of the classifier (in the Receiver Operating Characteristic (ROC) space) compared to solely under-sampling the majority class. They also showed that combining minority class oversampling with majority class under sampling can improve classifier performance.

Transfer learning is a machine learning technique that involves reusing a pre-trained model to improve the performance of a related but different task. In deep learning, transfer learning is particularly useful because pre-training a model on a large dataset can help the model learn general features that can be applied to new, smaller datasets. This not only saves time and resources but also improves the accuracy of the model.

Transfer learning involves leveraging knowledge learned from one task or dataset to improve learning on a related task or dataset, even with minimal additional training. This approach can help predict data from a similar dataset (such as one associated with a new individual). A large dataset, such as MI-EGG data from multiple individuals, can be used to train a neural network to learn general patterns and extract features. There are three contexts in which transfer learning approaches are used in BCI. Among the most popular approaches is instance transfer learning [44], [45], [46], which assumes detailed data from a source domain can be transported to a target domain and reused. Second, feature transfer is used to transfer knowledge by identifying the standard features between the source and target domains [47], [48], [49]. The latest transfer learning technique involves transferring a model's parameters from a source domain to a target domain [50], [51].

Existing transfer learning methods can face several challenges related to the number of fine-tuned layers, speed and accuracy. In transfer learning, fine-tuning too many or too few layers can affect the performance of the model. Fine-tuning too many layers can lead to overfitting, while fine-tuning too few layers can result in underfitting. Determining the optimal number of fine-tuned layers is a critical challenge in transfer learning. Transfer learning methods typically require a large amount of data and computational power. This can result in slow training times, which can be a challenge in many applications. Additionally, the accuracy of transfer learning methods may be limited by the quality and quantity of the available data. Moreover, novel techniques for selecting the optimal number of epochs and fine-tuned layers are being developed. In this paper, we propose a method of EEG signal classification which involves using a feature-extracted deep one-dimension (1D) convolutional neural network (CNN) that can be further improved through hyperheuristic multi-objective evolutionary search. By training this deep CNN model with feature-extracted data from the Physionet MI dataset and applying a semi-deep fine-tuning approach, the classification performance can be improved in just four epochs. The results obtained using the Physionet MI dataset demonstrate that this approach outperforms most contemporary techniques used for classifying EEG signals. The system is computationally efficient, can be trained using reliable EEG data for individual patients, and can accurately classify their EEG records. Due to its parameter-independent nature, the system is versatile and can be used with any EEG dataset.

II. METHODS AND MATERIALS

The codes of our 1DCNN and Semi-deep fine-tuning approach are publicly available for download at: https://github.com/MohamadTaghizadeh/EEG-1DCNN.

A. DATASET

The EEG Motor Movement/Imagery Dataset V 1.0.0 [52] was the data source used in this study. The dataset records contain EEG data from 64 scalp electrodes while the subjects perform four tasks:

I. The participant performs hand movements by opening and closing the fists on the side where a target appears on the screen.
II. The participant imagines hand movements by opening and closing the fists on the side where a target appears on the screen.
III. The subject is instructed to open their fists when a target appears at the top of the screen, close them when the target appears at the bottom, and then relax, depending on the target's position on the screen.
IV. The subject is instructed to imagine opening their fists when a target appears at the top of the screen, closing them when the target appears at the bottom, and then relaxing, depending on the target's position on the screen.

Features were extracted from the raw data, and the EEG dataset's Region of Interest (ROIs) and channels are listed in TABLE 1. The 64 EEG electrodes' scalp positions were indicated by the letters "A" to "F", as shown in FIGURE 1, representing the six regions. Furthermore, we use only task (II) and (IV) combinations.

The dataset consists of 64 EEG signals, each sampled at 160 samples per second, and an annotation channel. This dataset comprises over 1500 EEG recordings, each lasting one or two minutes, and obtained from 109 volunteers involving 14 experimental runs and 4 tasks. Each annotation in this dataset includes one of three codes (T0, T1, or T2): T0 corresponds to rest, T1 corresponds to the onset of the motion (real or imagined) of the left fist (in runs 3, 4, 7, 8, 11, and 12) and both fists (in runs 5, 6, 9, 10, 13, and 14). T2 corresponds to the onset of the motion (real or imagined) of the right fist

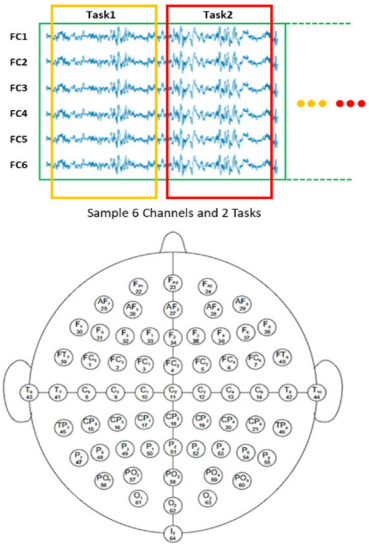

FIGURE 1. The tasks and channels of EEG regions of interest (ROI).

TABLE 1. ROI and channels of brain EEG.

ROI	Channels
A	[FC1, FC2], [FC3, FC4], [FC5, FC6]
B	[C5, C6], [C3, C4], [C1, C2]
C	[CP1, CP2], [CP3, CP4], [CP5, CP6]
D	[FC3, FC4], [C5, C6], [C3, C4], [C1, C2], [CP3, CP4]
E	[FC1, FC2], [FC3, FC4], [C3, C4], [C1, C2], [CP1, CP2], [CP3, CP4]
F	[FC1, FC2], [FC3, FC4], [FC5, FC6], [C5, C6], [C3, C4], [C1, C2], [CP1, CP2], [CP3, CP4], [CP5, CP6]

(in runs 3, 4, 7, 8, 11, and 12) and both feet (in runs 5, 6, 9, 10, 13, and 14). The sample sizes vary across different classes, as evident from the channels depicted in FIGURE 1. These recordings are categorized into Regions of Interest and are presented in TABLE 1.

The division is based on the 10-20 system rule. The international 10-20 system was created by Herbert Jasper in the mid-20th century as a standardized method for positioning EEG electrodes on the scalp. The ''10'' and ''20'' refer to the actual and nominal percentages of the distance between specific landmarks on a subject's head. The regions and channel selection follow the standard placement of EEG electrodes referring to the motor cortex and motor function areas, mainly corresponding to the brain activities related to motor imagery. Each region includes both right and left EEG channels on the scalp to ensure capturing the motor function activities. The main purpose of the channel selection for the ROIs in our paper is:

I. To reduce the computational complexity of any processing task performed on EEG signals by selecting the relevant channels and hence extracting the features of major importance.
II. To reduce the amount of overfitting that may arise due to the utilization of unnecessary channels, for the purpose of improving the performance.
III. To reduce the setup time in some applications.

In this paper, five classes are defined as L: opening and closing the left fist, R: opening and closing the right fist, LR: opening and closing both fists, F: opening and closing both feet, and B: described as a baseline (which enables the network to discriminate between EEG signals associated with intentional movements and irrelevant or noisy signals generated when participants are not actively issuing commands to the device under control). Regarding task combinations, the class label 'L' corresponds to T1 when associated with the left fist gesture in task (II). Conversely, the class label 'LR' represents T1 in the context of left-right fists in task (IV). Similarly, for T2, the class label 'R' pertains to the right fist gesture in task (II), whereas the class label 'F' signifies T2 in the context of both feet movements in task (IV). In the context of task combinations, T1 corresponds to the imagined movement of the left fist during the experimental run associated with task (II). Similarly, for T1 in task combination (IV), it signifies the imagined movement of both fists. Regarding T2, it pertains to the imagined movement of the right fist in the experimental runs related to task combination (II). Furthermore, in the context of task combination (IV), T2 represents the imagined movement of both feet. Each run adheres to a stereotyped repeated timeline consisting of 4 seconds of baseline, followed by 4 seconds of T1, another 4 seconds of baseline, and finally 4 seconds of T2.

To facilitate model training and evaluation, we partitioned the dataset based on its volume. Specifically, 80% of the data was allocated for training, while the remaining 20% was reserved for testing. Within this test data, we further allocated 10% for validation and 10% for final testing. To ensure data integrity, we prevented any sample duplication across these distinct sets. Due to the data properties and characteristics, we used SMOTE [43] which is an oversampling technique where synthetic samples are generated for the minority class. This algorithm helps overcome the overfitting problem posed by random oversampling. After applying SMOTE, the number of instances of ROIs A, B, C, D, E, and F are 110160, 110160, 110160, 183600, 220320, and 330480, respectively. Without data augmentation, the accuracy will be very low. Using SMOTE for data augmentation causes the number of minority class instances to become closer to the number of instances in other classes.

B. FEATURE EXTRACTION

Processing biological signals are often challenging due to the data's non-stationary, nonlinear, and stochastic nature. In order to achieve accurate classification, it is necessary to observe the temporal behavior of the signals rather than rely on a single point in time.

Therefore, to achieve accurate classification, observing short segments of the wave and generating a feature matrix based on the temporal segment is necessary. When using a non-temporal learning process, it is essential to perform feature extraction on the non-stationary and random EEG signal. For segmenting the data, sliding windows of 1 second

Algorithm 1 EEG Signal Feature Extraction Algorithm

Output: For every w_t, features extracted from raw data
Input: EEG raw data sequence;
Variables initialization *init*=1, w_t =0;
While getting a sequence of raw data (> 1min) **do**
if *init* **then**
  prev_lag = 0;
  post_lag = 1
end
init=0;
for each sliding window (w_t- prev_lag) to (w_t + post_lag) **do**
  Compute the mean of all w_t values $y_1 y_2 y_3 \cdots y_n$; $\bar{y}_k = \frac{1}{N}\sum_{i=1}^{N} y_{ki}$;
  Compute the asymmetry and waves peakedness represented by 3^{rd} and 4^{rd} order moments kurtosis and skewness $g_{1k} = \frac{\sum_{i=1}^{N}(y_{ki}-\bar{y}_k)^3}{N_{sk}^3}$ and $g_{2k} = \frac{\sum_{i=1}^{N}(y_{ki}-\bar{y}_k)^4}{N_{sk}^4} - 3$;
  Compute the minimum and maximum value of each signal $w_{min}^t = min(w_t)$ and $w_{max}^t = max(w_t)$;
  Compute the sample variances K×K matrix S for each signal pair $S_{kl} = \frac{1}{N-1}\sum_{i=1}^{N}(y_{ki}-\bar{y}_k)(y_{li}-\bar{y}_l)$; ∀k, ℓ ∈ [1, k];
  Compute the covariance matrix eigenvalues S, which are the λ solutions to: $\mathbf{det}(S - \lambda I_k) = \mathbf{0}$, where I_k is the K×K identity matrix, and **det**(·) is the matrix determinant;
  Compute the logarithm of each element in the upper triangular matrix of the covariance matrix S, and the matrix for S is defined via Taylor expansion $e^B = I_k + \sum_{n=1}^{\infty} \frac{S^n}{n\mathcal{W}}$, then B $\in \mathbb{C}^{K\times K}$ is a matrix logarithm of S;
  Compute the frequency's magnitude components of each signal, achieved using a Fast Fourier Transform (FFT), magFFT (w_t);
  Get the frequency values of FFT's ten most energetic elements, and for each signal, get FFT (w_t,10);
end
$w_t = w_t + 1s$;
$prev\_lag = 0.5s$; $post\_lag = 1.5s$;
Output Features Fw_t extracted within the current w_t
end

with 0.5-second overlap were introduced. Algorithm 1 outlines the hand-crafted features extracted. The samples' mean, standard deviation, maximum and minimum values were calculated to determine the differences in the two 1-second windows divided into two 0.5-second half-windows. The sample means, paired differences of the sample mean, maximum (minimum) values, and paired differences between the quarter windows were all generated by two-quarter windows of 0.25 seconds each. A Discrete Fourier Transform was applied to analyze the time windows, and all resultant values were treated as attributes. Algorithm 1 is utilized to produce a vectorized representation of the behavior of the EEG wave, resulting in the generation of numerical characteristics.

III. 1D-CNN

The conventional deep CNNs described in the previous section work exclusively with 2D data, such as images and videos. Therefore, they are often referred to as '2D CNNs'. In addition, 1D Convolutional Neural Networks (1D CNNs) have been developed as an alternative to 2D CNNs [53], [54], [55], [56], [57], [58]. The studies suggest that 1D CNNs offer benefits for specific applications and are more desirable than 2D CNNs. TABLE 2 shows a comparison between 1D-CNN and 2D-CNN.

The low computational requirements of compact 1D CNNs make them ideal for low-cost and real-time applications [53], [54], [55], [56], [57], [58], [59], [60].

Recently, compact 1D CNNs have demonstrated their superior performance in applications with limited labeled data and high signal variation, as shown in the above studies (electro-cardiograms of patients, civil, mechanical, or space structures, high power circuits, jet engines or motors, etc.). The structure of the proposed model is presented in FIGURE 2. There are two different types of layers in 1D-CNNs, as shown in FIGURE 2: 1) The CNN layers go through a process that involves 1D folding, activation functions, and subsampling (pooling), and 2) The fully connected (dense) layers are the same as those found in a typical multilayer perceptron (MLP) and are referred to as MLP layer. For a 1D CNN, the following hyper-parameters determine its configuration:

I. There are 4 CNN layer in the sample 1D CNN shown in FIGURE 2.
II. The filter size (kernel) in each CNN layer; in the sample 1D CNN in FIGURE 2, the filter size is 10 for two initial layers, and is 4 for the rest of CNN layers.
III. Each CNN layer has a subsampling factor (can be seen based on the input size).
IV. Activation functions and padding are available.

Traditional 2D CNNs consist of an input layer that receives the raw 1D signal and an output layer with neurons corresponding to the number of classes. As shown in FIGURE 2, a 1D CNN consists of three consecutive CNN layers. This figure illustrates that the 1D filter kernels in the hidden CNN layer l, are three-dimensional and have a subsampling factor of 2. The k^{th} neuron performs a sequence of convolutions, which are then summed by the activation function f and subsampled. The main difference between 1D and 2D CNN is that the kernels and feature maps are represented as 1D arrays instead of 2D matrices. The CNN layer then processes the raw 1D data and learns to extract the features that are used by the MLP layer for classification.

TABLE 3 shows the model hyperparameters that we used for our system. Categorical cross-entropy was used because our task is a multiclass classification problem. Also, the value of batch size is designated according to Random Access Memory (RAM) limitation. Other parameters are selected for

TABLE 2. 1D-CNN and 2D-CNN comparison.

	1D-CNN	2D-CNN
Input data	1D data, such as time series or sequences	2D data, such as images or matrices
Convolution	1D convolution is applied along a single spatial dimension (e.g., time in a time series)	2D convolution is applied along two spatial dimensions (e.g., width and height in an image)
Applications	Time series analysis, natural language processing, audio processing	Image recognition, object detection, computer vision tasks
Kernel/filter	1D filter/kernel, e.g., [1, -1]	2D filter/kernel, e.g., [[1, 0], [0, -1]]
Pooling	1D pooling, e.g., max-pooling or average-pooling along the single dimension	2D pooling, e.g., max-pooling or average-pooling along both dimensions
Computational complexity	Generally lower than 2D-CNN due to fewer dimensions of convolution	Generally higher than 1D-CNN due to increased dimensions of convolution
Feature extraction	Captures local patterns or features in 1D data	Captures local patterns or features in 2D data, including spatial relationships between features

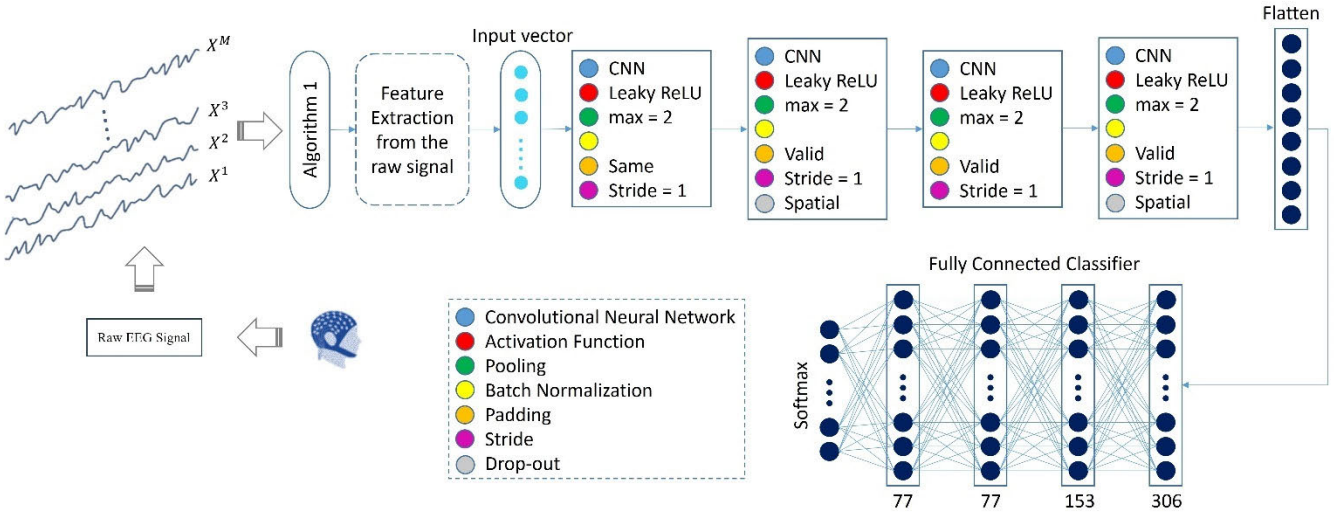

FIGURE 2. The overall structure of proposed 1D convolutional neural network model: In this model, four CNN layers undergo a process that involves 1D convolution, activation functions, and subsampling (pooling). The filter size is 10 for the first two layers and 4 for the remaining CNN layers. The fully connected (dense) layers are the same as those found in a typical multilayer perceptron (MLP) and are referred to as MLP layers. For a 1D-CNN, the hyperparameters determine its configuration.

TABLE 3. Model hyperparameter.

Parameter	Value
Cost function	Categorical cross-entropy
Regularization	0
Batch size	10
Optimizer	Adam
Learning rate	1e-4
Epoch	100

the best performance. TABLE 4 provides an overview of the network architecture in this paper.

The purpose is to combine feature extraction and classification into a single process that can be optimized for the best classification results. 1D-CNNs have this great advantage because they only need to perform a sequence of linear weighted sums of two 1D arrays, which can also lead to low computational complexity. It is possible to perform such a linear operation simultaneously during the forward and back propagation operations. Furthermore, it is an adaptive implementation since the CNN topology allows the dimension of the input layer to vary such that the sub-sampling factor of the CNN output layer can be adjusted accordingly.

One kind of neural network requires an $M \times N$ matrix as input, where M denotes the duration of the relevant time frame and N represents the count of EEG channels. 1D convolutional layer is widely used in deep learning for tasks

TABLE 4. Our deep 1D-CNN model definitions: 1 dimension convolutional neural network (1DCNN), batch normalization (BN), Spatial dropout (SD), Dropout (D), Pooling (P), Flatten (F), Fully connected (FC).

Type	Input size of Layers	number of Filters	Filter size	Activation Function	Stride=1, Padding
1DCNN-1	(640, 2)	64	10	Leaky ReLu	Same
BN1	(640, 64)				
1DCNN-2	(640, 64)	64	10	Leaky ReLu	Valid
BN2	(631,64)				
SD1	(631,64)				
1DCNN-3	(631,64)	32	4	Leaky ReLu	Valid
Max-P1	(628,32)				
1DCNN-4	(314,32)	32	4	Leaky ReLu	Valid
SD2	(311,32)				
F	(1,9952)				
FC	(1,306)			Leaky ReLu	
D	(1,306)				
FC	(1,153)			Leaky ReLu	
D	(1,153)				
FC	(1,77)			Leaky ReLu	
D	(1,77)				
FC	(1,77)			Softmax	

such as image and speech recognition, natural language processing, and many other applications where the input data has a sequential or temporal structure.

Equation (1) denotes 1D convolutional layers, where Q represents the time window covered by the filter and N is equal to 2.

$$y_r = f(\sum_{q=1}^{Q} \sum_{n=1}^{N} w_{qn} x_{r+q,r+n} + b) \tag{1}$$

The number of feature maps in a convolutional layer is determined by the number of filters applied to the input. The size of the feature maps depends on the size of the input image, the size of the filter, the stride used in the convolution operation, and whether or not padding is used.

$$R = [\frac{M - (K-1) + 2 \times P}{S}] \tag{2}$$

Equation (2) shows the filter feature map of size R (where $R = M$ if the *stride* $= 1$) with padding results in the output y_r (Equation (1)); x represents the input range that intersects with the filter; w is the connection weight of the convolution filter; b is the bias term; and f is the activation function. This equation can help us determine the size of the filter feature map (R) following convolution.

The purpose of padding is to preserve the input-output ratio equivalent to the stride during convolution, and this is achieved by adding pixels with a value of 0 to the edges of the image. The symbol S represents the stride value, which determines the number of positions skipped by the filter during each shift while performing convolution. The first convolutional layer is followed by a batch normalization (BN) [39]. BN involves adjusting the input to the next layer to speed learning and improves generalization through regularization effects. The training and testing phases are handled differently for this technique. While training, the inputs of the entire batch are normalized by BN, which refers to the group of instances used to calculate the loss and gradient for the learning algorithm. Additionally, it sets the normalized inputs to zero. The model can effectively scale the inputs by learning to do so.

$$\mu = \frac{1}{b} \sum_{i=1}^{b} X^{(i)}$$

$$\sigma^2 = \frac{1}{b} \sum_{i=1}^{b} (X^{(i)} - \mu)^2 \tag{3}$$

Equation (3), estimates the mean and variance that depends on parameters and are derived from the stack.

In this case, $X^{(i)}$ is an instance of the batch. The normalized value of each instance refers to the transformed value of a feature or variable in the dataset such that it has a mean of zero and a standard deviation of one. Normalization is a common preprocessing step in machine learning that helps to scale the features of the dataset to a similar range, making it easier for the model to learn and reduce the impact of outliers. The calculation of $\hat{X}^i$ refers to the normalized value of each instance in the batch, which has been centered around zero, based on the total number of instances in the batch denoted by "b."

$$\hat{X}^i = \frac{X^{(i)} - \mu}{\sqrt{\sigma^2 + \xi}} \tag{4}$$

Equation (4) calculates $\hat{X}^i$ ($\xi = 10^{-5}$ avoids zero divisions).

Since normalization may not be sufficient for a particular task, *BN* employs an additional training step. By introducing the scaling and shifting parameters, BN allows the model to learn the optimal scale and shift for the normalized activations, which can improve the performance of the model. The scaling and shifting parameters are learned independently for each feature dimension of the activations, allowing the model to adapt to the specific characteristics of each feature.

$$z^i = \gamma \hat{X}^i + \beta \tag{5}$$

Equation (5) involves adjusting values through scaling and shifting using the trained parameters.

A scaling parameter is obtained by multiplying each input value by the corresponding scaling parameter, and an offset parameter is obtained by subtracting the input value from the scaling parameter (both parameters were trained by backpropagation).

An activation layer is a layer in a neural network that applies an activation function to the output of a previous layer or input, introducing non-linearity and enabling the network to learn complex patterns and relationships in the data.

$$y_j^{(l)} = f(\sum_{i=1}^{I} w_{ji}^{(l)}.x_i^{(l-1)} + b_j^{(l)}) \tag{6}$$

Equation (6) computes the activation $y_j^{(l)}$ for each unit in each of these layers. Unit j of layer l is connected to unit i of the previous layer via the weight $w_{ji}^{(l)}$, and bias $b_j^{(l)}$ which is the bias term of unit j, with f being the unit transfer function. Except for the output layer, all layers have a rectified linear unit (ReLU) transfer function f.

$$f(x) = \begin{cases} x & if\ x > 0 \\ 0.01.x & otherwise \end{cases} \tag{7}$$

Equation (7) denotes the ReLU function. Softmax is a commonly used activation function in the output layer of a neural network for multi-class classification tasks. Five categories of the MI classification task are encoded using a Softmax function.

$$\hat{y}_i = argmax(\frac{e^{y_i}}{\sum_{i=1}^{5} e^{y_i}}) \tag{8}$$

Equation (8) shows the Softmax function which is applied in the last output layer. It is a mathematical function that transforms a vector of real values into a probability distribution over the classes.

Cross-entropy loss is a commonly used loss function in machine learning and deep learning, particularly in classification tasks. It is used to measure the difference between the predicted probability distribution and the true probability distribution of the classes.

$$loss = -\sum_{i=1}^{5} (y_i^*.log\hat{y}_i) \tag{9}$$

Equation (9) denotes the categorical cross-entropy loss function [18] which optimizes the neural network's parameters. In this equation, $\hat{y}_i$ is the i^{th} output prediction and y_i^* is the corresponding target value (1 for the correct class and 0 otherwise).

The MI-EEG BCI system suggested in this paper utilizes a type of neural network known as a one-dimensional convolutional neural network (1D-CNN), which operates by sliding convolutional kernels solely over the elements of a single dimension of the input pattern. The proposed architecture of the network shown in FIGURE 2, consists of several layers, each with a specific function:

The first layer takes the input EEG data with a size of (640, 2) and applies 64 filters with a size of 10 to extract 64 feature maps. The activation function used is Leaky ReLU, and padding is set to 'same' to preserve the input shape. The normalization process may not be optimal for this task, and therefore, the batch normalization (BN) technique is used to add an additional step during training. This involves using trained parameters to further adjust and scale the input values as needed, allowing the network to learn more effectively. For this reason, the next layer is a batch normalization layer (BN1) that is applied to normalize the feature maps obtained from the previous convolutional layer. The second layer takes the normalized feature maps from the previous layer and applies 64 filters of size 10 to extract 64 new feature maps. The activation function used in this layer is also Leaky ReLU, and padding is set to 'valid' to reduce the output size. Another batch normalization layer is applied to the feature maps obtained from the second convolutional layer (BN2). After that, a spatial dropout layer (SD1) is applied to the feature maps obtained from the second batch normalization layer to prevent overfitting.

The third convolutional layer takes the output from SD1 and applies 32 filters of size 4 to extract 32 new feature maps. The activation function used is Leaky ReLU again, and padding is set to 'valid' to reduce the output size. Next, a max pooling (P) layer is applied to the feature maps obtained from the third convolutional layer to further reduce the output size. 1DCNN-4 is the fourth layer that takes the output from Max-P1 and applies 32 filters of size 4 to extract 32 new feature maps. Another spatial dropout layer (SD2) is applied to the feature maps obtained from the fourth convolutional layer to prevent overfitting. The output of the fourth convolutional layer is flattened into a 1D feature vector (F) and passed through three fully connected layers (FC) with Leaky ReLU activation functions. Another dropout layer is applied after the first fully connected layer to avoid overfitting. The output of the last fully connected layer is passed through the final fully connected layer with 77 neurons and a Softmax activation function, which produces the final output of the network as a probability distribution over the possible classes. Our consideration for the scale of each FC layers was problem complexity. More complex problems often require deeper networks with more layers. The four FC layers in our model are designed to capture a high level of abstraction in the data.

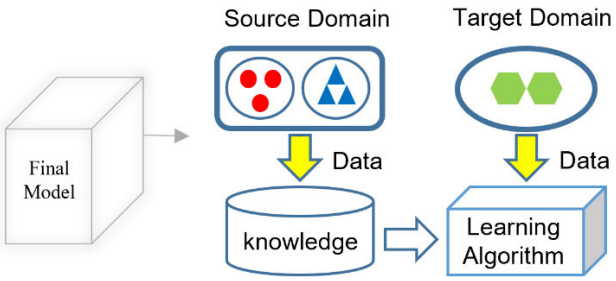

FIGURE 3. This figure illustrates transfer learning from original weights to a Motor Imagery domain using the source domain from the original weights [64].

IV. TRANSFER LEARNING

In CNN-based MI-BCI classification algorithms, there is a large amount of data, which increases the computation time and requires a large amount of training data [61], [62]. Using pre-trained weights from different subject cases can be an efficient transfer learning method. Nevertheless, the transfer of knowledge in the MI-BCI system presents a formidable challenge due to substantial differences among subjects [63]. For developing an efficient MI-BCI classifier based on transfer learning, paying particular attention to fine-tuning the model parameters is essential.

Each model must have some initial weights as part of the training process. Gradient descent is usually used to optimize the weights of neural networks. As a result, the weights tend to reach local minima, although globally optimized network weights can be achieved with some effort. Deep convolutional networks learn generic, simple features in all image processing tasks. On the other hand, higher layers learn high-level features specific to the trained dataset. Shallow tuning involves adjusting or fine-tuning only the last few fully connected layers of a deep neural network to improve the extraction of high-level features. Using a pre-trained network, features are extracted and trained on another classifier, such as a Support Vector Machine. However, this method does not significantly increase the accuracy of the model.

We also found that the degree of fine-tuning required for different applications varies, suggesting that shallow or deep tuning may not be the best option. Adapting the layers of a pre- trained model, also known as fine-tuning, can be an effective method to attain optimal performance for a specific use case. Instead of tuning all layers or only the last layer, we tuned several layers of the model along with the last layer by trial and error, which we called semi-deep fine-tuning, which is shown in the block diagram in FIGURE 4.

Furthermore, we investigated some strategies shown in TABLE 5 to check for underfitting, overfitting, and bias-variance tradeoffs for the best performance based on the amount of data and our proposed method. In addition, we figured out that when the network has a high degree of freedom (DOF), it will converge slower in this situation. As a result, we selected a condition to have an optimal convergence. The DOF and convergence relationship in transfer learning pertains to the extent to which the pre-trained model's weights

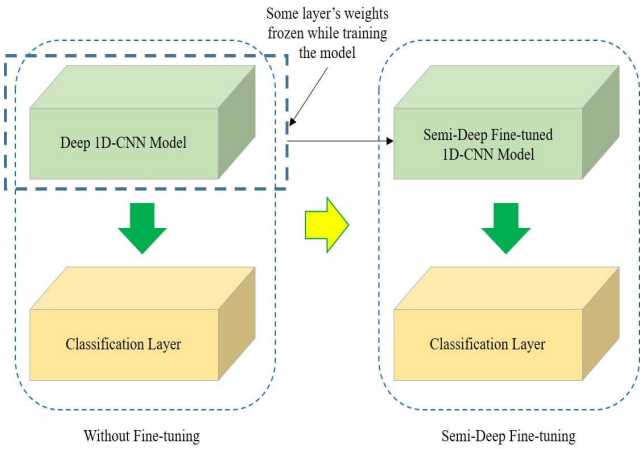

FIGURE 4. Semi-deep fine-tuning approach.

TABLE 5. Semi-deep fine-tuning strategies for overfitting (OV), underfitting (UN), bias-variance tradeoff (BVT), and degree of freedom (DF): where F means that the network convergence is fast, OK means that the network convergence is done optimally, and S means that the network convergence is slow.

Freezing layers\Status	OV	UN	BVT	DOF
Training from scratch	Yes	No	No	S
Total freezing except for the last layer	No	Yes	No	F
Semi-deep fine-tuning	No	No	Yes	OK

can be modified and the rate at which the model reaches its optimal solution—generally, the more freedom and the faster the convergence, the better the performance.

A. TUNING STRATEGY

Fine-tuning a 1D Convolutional Neural Network (1DCNN) requires careful consideration. The first step involves selecting a suitable architecture for the task at hand. The selection of the architecture should depend on the type of data that must be processed. After modeling the architecture, the next step is to define the hyperparameters, such as the learning rate, the number of layers, and the number of neurons. After that, the model should be trained on the data, and the performance should be monitored during training. Finally, the model should be fine-tuned to obtain the desired performance. Fine-tuning involves adjusting the model's hyperparameters to optimize for the specified task. This process requires careful experimentation and evaluation of the model's performance. The model's hyperparameters can be tweaked to obtain optimal performance during this process. Algorithm 2 shows the strategy for fine-tuning, as we call semi-deep fine-tuning.

B. METRICS

In a one-dimension network, we used some metrics to assess the performance. Precision is a metric that calculates the proportion of accurately identified positive instances to all the positive instances predicted by the model. To evaluate the

Algorithm 2 Semi-deep Fine-tuning Algorithm

Input: Pre-trained Deep 1D-CNN with the best features

Output: High accuracy EEG motor imagery classification
Fine-tune: 1DCNN-3, 1DCNN-4.
Trained: 1DCNN-2, 1DCNN-3, 1DCNN-4.

1: Randomly initialize the weights of 1DCNN-2, 1DCNN-3, and 1DCNN-4.

2: Initialize the rest of the weights using the pre-trained weights.

3: forepoch=1,2,3,4 do
Retrain
for *acc_fine_tuning>before_fine_tuning* **then**
print *train_loss, validation_loss, train_accuracy, validation_accuracy.*

performance of a model, recall measures the proportion of actual positives that were correctly identified, while accuracy measures the proportion of all correctly identified examples, not just the positives. Finally, the F1-score is a metric that uses the harmonic mean of recall and precision to evaluate a model's performance. It is often used as a single score to assess the model's overall performance.

$$\text{Precision} = \frac{TP}{TP + FP} \tag{10}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{11}$$

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{12}$$

$$\text{F1-score} = \frac{2 \times Recall \times Precision}{Recall + Precision} \tag{13}$$

Equations (10) and (11) compute separately precision and recall in multiclass classification for each class in the dataset, respectively. This allows evaluation of the performance of the model for each class individually. Equation (12) calculates accuracy over the entire dataset based on the true positives (TP), true negatives (TN), false positives (FP) and false negatives (FN). The *F1-score* of a specific class calculated by Equation (13) is the harmonic mean of precision and recall for all classes. Additionally, a confusion matrix can be used to visualize the model's performance, which helps to identify which classes the model classifies correctly and which classes it misclassifies.

V. RESULTS

To successfully apply DL, it is crucial to have a sufficient amount of data. Acquiring such datasets may pose difficulties. As recently suggested, a potential remedy to this issue is employing transfer learning via fine-tuning. This approach involves starting with a pre-trained model for a specific task and then fine-tuning only particular neural network layers for a related but different task. One major challenge with this approach is selecting the appropriate layers for fine-tuning, as no general rule guarantees optimal performance.

Furthermore, the fine-tuning technique can consume significant time, and selecting the appropriate layers often relies on a trial-and-error approach. In the future, we will work towards identifying the optimal layers for fine-tuning to achieve desired results more efficiently. Convergence in the fine-tuning phase is significant, especially when dealing with limited amounts of accurate data. TABLE 6 presents average metrics based on ROIs. The results we achieved are superior to other state-of-the-art methods, as depicted in TABLE 7 for comparison.

TABLE 6. Average metrics based on ROIs: Loss (L), Recall (R), Precision (P), F1-score (F1), Accuracy (Acc).

ROIs\Metrics	L	P	R	F1	Acc
A	0.10	98.47	97.23	98.08	98.33
B	0.11	98.10	96.91	97.76	98.04
C	0.09	98.63	97.18	98.09	98.53
D	0.02	100	99.72	99.45	99.81
E	0.01	99.95	99.73	99.81	99.90
F	0.03	99.85	99.52	99.17	99.70

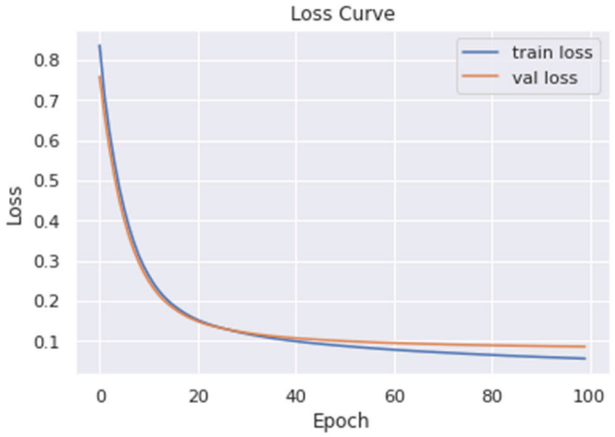

FIGURE 5. Train and validation loss of the proposed deep 1-dimension model using Physionet MI dataset. This figure shows the progression of loss during the training of the proposed model. The training process did not lead to overfitting as indicated by the curve patterns, with particular attention to those in the validation set, which plateaued.

Additionally, FIGURE 5 shows a loss curve that converges to a minimum value for train and validation loss. FIGURE 6 illustrates learning progress for both training and validation accuracy, and FIGURE 7 demonstrates a close similarity in the loss values between the training and validation steps, which supports our model assessment. Furthermore, as depicted in FIGURE 8, the accuracy curve indicates an enhancement in performance with a low cost. For further elaboration, we performed tuning for four epochs, but the curve plateaued at the third epoch, as the output remained constant after that.

In order to show the versatility and robustness of our work, we applied our pre-trained model to other datasets. After preparing the datasets for fine-tuning the pre-trained model, the vectorized inputs will be fed into the semi-deep

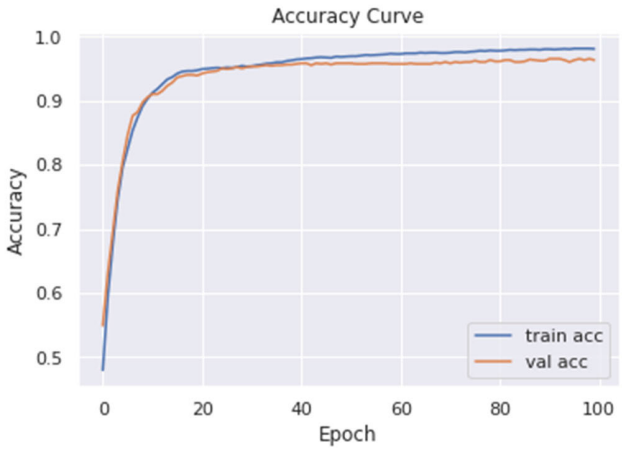

FIGURE 6. Train and validation accuracy of the proposed deep 1-dimension network before applying semi-deep fine-tuning using Physionet MI dataset. The plot exhibits improved accuracy in training and validation sets. The curves indicate a plateau pattern, as particularly noticed for the validation set, which is indicative of lack of overfitting problems during training.

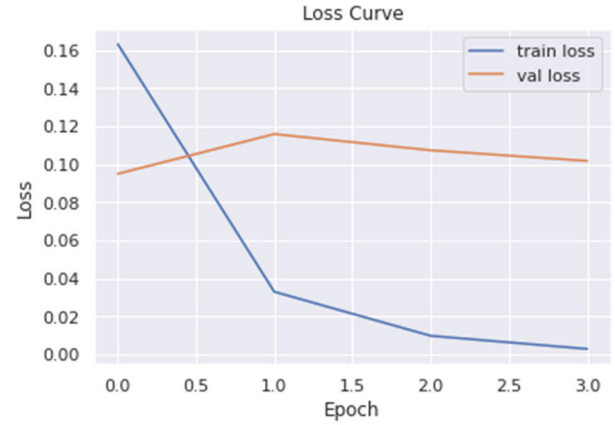

FIGURE 7. Train and validation loss after applying semi-deep fine-tuning using Physionet MI dataset. The figure shows that after just four epochs, the train and validation loss plateaued with relative values, indicating that this process did not lead to overfitting.

TABLE 7. Comparison of the outcomes with other state-of-the-art works that used the same dataset on the test set. Classes (C), Average Accuracy (Avg-Acc), Accuracy after Transfer Learning (F-Acc).

Authors\Metrics	Classes	Avg-Acc	F-Acc
Xie et. al **[16]**	4	65.73	68.51
Karácsony et. al **[65]**	4	76.37	-
Lun et. al **[42]**	4	97.28	-
Mattioli et. al **[66]**	5	99.38	99.46
Liu et. al **[67]**	4	87.19	-
Ghimire et. al **[68]**	3	69.08	-
Wu et. al **[69]**	3	86.12	-
Our performance	5	**99.90**	**100**

fine-tuning step. The results shown in TABLE 8 depend on using the original number of instances in the datasets or applying augmentation.

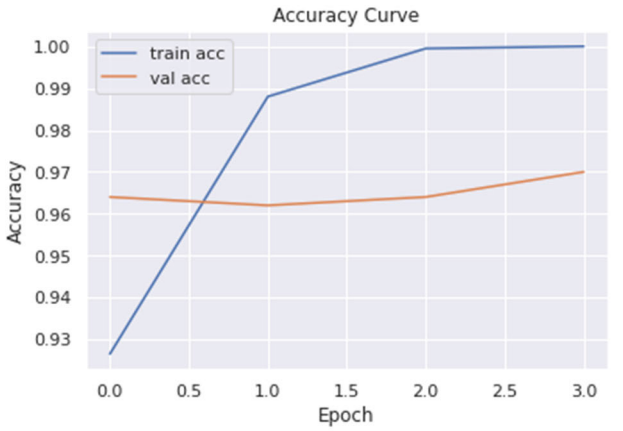

FIGURE 8. Train and validation accuracy after applying semi-deep fine-tuning using Physionet MI dataset. The plot shows a plateau pattern, which means a lack of overfitting issues. After four epochs and applying the semi-deep fine-tuning, the pre-trained model reached the highest value in accuracy, indicating the fact that this approach could achieve the best result with as low-cost as possible.

TABLE 8. Classification accuracy of the proposed model on other datasets with and without data augmentation (aug.) steps. The BCI competition sets are used.

Dataset	Classes	Accuracy after semi-deep fine-tuning (4 epochs)		No. of epochs: Accuracy (With aug.)
		No data aug.	With data aug.	
IV-2a **[70]**	4	92.4%	98.3%	16 epochs: 100%
IV-2b **[70]**	2	89.51%	96.39%	25 epochs: 99.1%
III-Iva **[71]**	2	93.17%	97.58%	18 epochs: 99.82%

Statistical analysis has been utilized to validate the adequacy of the sample size and to substantiate the hypothesis assertions regarding the enhanced classification accuracy achieved through semi-deep fine-tuning compared to 1D-CNN. The primary objective is to determine the minimum sample size required for the experiments. The statistical power of a test refers to the likelihood of rejecting the null hypothesis $\boldsymbol{H_0}$ when it is incorrect, thereby supporting the alternative hypothesis $\boldsymbol{H_1}$ when it is accurate.

Consequently, two contradictory hypotheses can be articulated as follows:

$$H_0 : \mu_A \leq \mu_B \tag{14}$$

$$H_1 : \mu_A > \mu_B \tag{15}$$

where μ_A is the classification accuracy of 1D-CNN after applying fine-tuning step, and μ_B is the classification accuracy before semi-deep fine-tuning.

The subsequent equations (16)-(21) have been utilized for determining the sample size:

$$N_2 = k.N_1 \tag{16}$$

$$\bar{q} = 1 - \bar{p} \tag{17}$$

TABLE 9. The performance of the system described in this study was compared with other state-of-the-art systems that utilized a CNN and similar dataset. Test accuracy is reported.

Type	Dataset	Number of Classes	Number of fine-tuned layers	Epochs	Accuracy
A. M. Roy **[72]**	BCI competition IV-2a	4	-	350	94.06%
Li, Ang, et al. **[73]**	BCIC IV dataset 2a and Lee2019	4	-	500	80.86%
Liu, Dezheng, et al. **[74]**	Iva and dataset IIIb of the BCI Competition III, and data set 1 of the BCI Competition IV	2	3	34	84.05% 70.32% 84.29%
Mattioli, F., C. et al. **[66]**	BCI Competition IV 2a dataset and PhysioNet Motor Movement/Imagery	4+1	2	33	98.59%
Proposed work	**PhysioNet Motor Movement/Imagery**	**5**	**3**	**4**	**100%**

$$\bar{p} = \frac{p_1 + kp_2}{1+k} \tag{18}$$

$$q_1 = 1 - p_1 \tag{19}$$

$$q_2 = 1 - p_2 \tag{20}$$

$$N_1 = \left\{ z_{1-\alpha/2} \cdot \sqrt{\bar{p}.\bar{q}.(1+\frac{1}{k})} + z_{1-\beta} \cdot \sqrt{p_1.q_1 + (\frac{p_2.q_2}{k})} \right\}^2 / \Delta^2 \tag{21}$$

where p_1, p_2 are proportion of Groups A and B, Δ is $|p_2 - p_1|$, absolute difference between two proportions, N_1 is the sample size for Group A, N_2 is the sample size for Group B, α is probability of type I error ($\alpha = 0.01$), β is probability of type II error ($\beta = 0.2$), z is the critical z value for a given α or β, and k is the ratio of sample size for Group B to Group A.

In this work, Group A corresponds to the mean classification accuracy of semi-deep fine-tuning step, and Group B corresponds to the mean classification accuracy of 1D-CNN. The conclusive result of the statistical power analysis indicates that a minimum sample size of 2444 is required.

This is while the sample size that we are working on is much more than 2444. This is more than sufficient to confirm the accuracy claims that 1D-CNN results in less classification accuracy compared to semi-deep fine-tuning.

Our paper reports achieving the highest classification accuracy with just four epochs, surpassing other state-of-the-art methods as shown in TABLE 9. This efficiency is notable as extensive training often leads to overfitting. The improvement in accuracy post-transfer learning is attributed to the pre-trained model's ability to optimize weights for better classification outcomes.

Fig. 9 presents a 2D visualization plot of the feature maps of our 1D-CNN model for the EEG classification before applying semi-deep fine-tuning. The t-Distributed Stochastic Neighbor Embedding (t-SNE) method was used to project the high-dimensional feature space into a lower 2D representation by creating probability distributions that can capture the distance relationships between the data points. The graph clearly indicates that our model extracts powerful features to accurately differentiate the five classes. After applying semi-deep fine tuning, the model achieves even better classification performance, as seen from the results reported in TABLE 9.

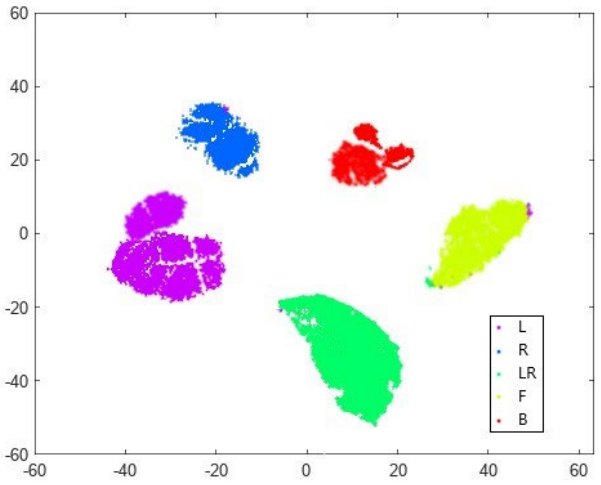

FIGURE 9. A 2D representation of our 1D-CNN features using t-SNE method.

VI. DISCUSSION

In recent years, machine learning and deep neural networks have been successfully utilized to classify various forms of data, including linguistic features, images, sounds, and natural texts. Due to a large amount of data, their classifiers are very powerful. However, in practical BCI applications, collecting high-quality and large EEG datasets is difficult due to strict requirements on the experimental environment and available subjects. During model training, a small sample size can lead to overfitting, negatively affecting decision accuracy. When a subject performs a specific EEG task, a significant amount of data is required to calibrate a BCI system. The calibration process will inevitably be time-consuming for a new user. For example, calibration times are extended when the subject performs an SSVEP spelling task. Nevertheless, collecting calibration data is time-consuming and labor-intensive, resulting in a less efficient system.

Due to its end-to-end structure and competitive performance, DL has demonstrated its effectiveness in processing EEG data [32]. A limitation in practical operation is the computational power and the small amount of data. This problem can be addressed with a hybrid structure that combines transfer learning (TL) and DL. For example, fine-tuning pre-trained networks is an effective method.

Future research should focus on developing such a hybrid structure using DL technology. Freezing particular layers and fine-tuning specific layers have been shown to enhance the performance of DNNs [69]. Using a parallel multiscale filter bank CNN, it is possible to fine-tune a model with 10, 20, 50, or 100 samples in the target domain. Based on the results of this experiment, it was found that this method would improve classification accuracy [75].

Decoding methods for two classes of MI tasks have relatively matured in the available literature. Further research is required for decoding studies related to four MI task classes, as the number of MI task classes is more significant. Additionally, obtaining large EEG datasets can be challenging. Deep learning-based decoding models may encounter issues such as overfitting and reduced resilience when the training data is restricted. Increasing the accuracy in the task imaging engine, especially when the number of classes is more than 4, is challenging and requires much work.

TABLE 8 and TABLE 9 display the comparison between our system's transfer learning performance with and without augmentation on different datasets and the performance of our system compared with the state-of-the-art works with a similar dataset and fewer epochs. Despite the inclusion of the challenging baseline class and other MI-related classes, our system achieved superior performance compared to the other systems. Moreover, TABLE 9 presents our system's performance in the transfer learning task.

The proposed model enhances 1D-CNN accuracy by incorporating serialized features from Algorithm 1, emphasizing the significance of inputting quality features into CNNs (as shown by the visualization graph of FIGURE 9 using t-SNE). Annotators use signal characteristics to label EEG data, which, when mathematically processed, improves model performance by filtering out noise. Our paper proposes using serialized feature learning and transfer learning to address challenges like noisy and limited data. Training models with structured data and fine-tuning pre-trained models with new EEG datasets can expedite research and yield better models. While our paper doesn't focus on real-world implementation, we suggest future research on online testing pipelines for signal classification.

VII. CONCLUSION

Our approach for EEG signal classification, based on transfer learning, has been proposed to achieve high accuracy. Our system utilizes a feature-extracted deep 1D-CNN, which can be fine-tuned to provide a dependable model. We have utilized a semi-deep fine-tuning method to improve network performance without requiring training from scratch. Our proposed system achieved an accuracy level of 100%, surpassing the current state-of-the-art model accuracy of 99.46%. Other works have not explored our approach of using a feature-extracted deep 1D-CNN. We initially trained our models by dividing the data into train and validation subsets and manually selecting specific hyperparameters, such as the activation function and optimization algorithm, using gradient descent. During the semi-deep fine-tuning step, some layers were tuned, and the remaining layer weights were initialized using the pre-trained weights. Future studies should investigate the implications of distinct sets of hyperparameters, train the CNN for individual patients for accurate classification results, and implement real-time classification. In upcoming endeavors, it may be beneficial to consider a synchronized feature selection approach to properly align network inputs. We suggest introducing the process of choosing attributes as a problem of combinatorial optimization in more complex experiments. Additionally, exploring the potential for exchanging knowledge between different biological signal domains, including ECG, is a promising research avenue for the future.

REFERENCES

[1] N. A. Alzahab, L. Apollonio, A. Di Iorio, M. Alshalak, S. Iarlori, F. Ferracuti, A. Monteriù, and C. Porcaro, "Hybrid deep learning (hDL)-based brain–computer interface (BCI) systems: A systematic review," *Brain Sci.*, vol. 11, no. 1, p. 75, Jan. 2021.

[2] S. Ikegami, K. Takano, N. Saeki, and K. Kansaku, "Operation of a P300-based brain–computer interface by individuals with cervical spinal cord injury," *Clin. Neurophysiol.*, vol. 122, no. 5, pp. 991–996, May 2011.

[3] E. Buch, C. Weber, L. G. Cohen, C. Braun, M. A. Dimyan, T. Ard, J. Mellinger, A. Caria, S. Soekadar, A. Fourkas, and N. Birbaumer, "Think to move: A neuromagnetic brain–computer interface (BCI) system for chronic stroke," *Stroke*, vol. 39, no. 3, pp. 910–917, Mar. 2008.

[4] A. Vallabhaneni, T. Wang, and B. He, "Brain–computer interface," in *Neural Engineering*. Boston, MA, USA: Springer, 2005, pp. 85–121.

[5] B. Z. Allison, E. W. Wolpaw, and J. R. Wolpaw, "Brain–computer interface systems: Progress and prospects," *Exp. Rev. Med. Devices*, vol. 4, no. 4, pp. 463–474, Jul. 2007.

[6] S. C. Chen, A. R. See, Y. J. Chen, C. H. Yeng, and C. K. Liang, "The use of a brain computer interface remote control to navigate a recreational device," *Math. Problems Eng.*, vol. 2013, no. 1, pp. 1–8, 2013.

[7] Z. Chen, J. Liao, J. Chen, C. Zhou, F. Chai, Y. Wu, and P. Hansen, "Paint with your mind: Designing EEG-based interactive installation for traditional Chinese artworks," in *Proc. 15th Int. Conf. Tangible, Embedded, Embodied Interact.*, Feb. 2021, pp. 1–6.

[8] R. Ramchurn, S. Martindale, M. L. Wilson, and S. Benford, "From director's cut to user's cut: To watch a brain-controlled film is to edit it," in *Proc. CHI Conf. Hum. Factors Comput. Syst.*, May 2019, pp. 1–14.

[9] E. Curran, "Learning to control brain activity: A review of the production and control of EEG components for driving brain–computer interface (BCI) systems," *Brain Cognition*, vol. 51, no. 3, pp. 326–336, Apr. 2003.

[10] F. Ferracuti, V. Casadei, I. Marcantoni, S. Iarlori, L. Burattini, A. Monteriù, and C. Porcaro, "A functional source separation algorithm to enhance error-related potentials monitoring in noninvasive brain–computer interface," *Comput. Methods Programs Biomed.*, vol. 191, Jul. 2020, Art. no. 105419.

[11] F. Ferracuti, A. Freddi, S. Iarlori, S. Longhi, A. Monteriú, and C. Porcaro, "Augmenting robot intelligence via EEG signals to avoid trajectory planning mistakes of a smart wheelchair," *J. Ambient Intell. Humanized Comput.*, vol. 14, no. 1, pp. 223–235, Jan. 2023.

[12] F.-B. Vialatte, M. Maurice, J. Dauwels, and A. Cichocki, "Steady-state visually evoked potentials: Focus on essential paradigms and future perspectives," *Prog. Neurobiol.*, vol. 90, no. 4, pp. 418–438, Apr. 2010.

[13] A. Combaz and M. M. Van Hulle, "Simultaneous detection of P300 and steady-state visually evoked potentials for hybrid brain–computer interface," *PLoS ONE*, vol. 10, no. 3, Mar. 2015, Art. no. e0121481.

[14] S. Salelkar and S. Ray, "Interaction between steady-state visually evoked potentials at nearby flicker frequencies," *Sci. Rep.*, vol. 10, no. 1, pp. 1–16, Mar. 2020.

[15] H. Altaheri, G. Muhammad, and M. Alsulaiman, "Physics-informed attention temporal convolutional network for EEG-based motor imagery classification," *IEEE Trans. Ind. Informat.*, vol. 19, no. 2, pp. 2249–2258, Feb. 2023.

[16] J. Xie, J. Zhang, J. Sun, Z. Ma, L. Qin, G. Li, H. Zhou, and Y. Zhan, "A transformer-based approach combining deep learning network and spatial–temporal information for raw EEG classification," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 30, pp. 2126–2136, 2022.

[17] T. Mwata-Velu, J. G. Avina-Cervantes, J. Ruiz-Pinales, T. A. Garcia-Calva, E.-A. González-Barbosa, J. B. Hurtado-Ramos, and J.-J. González-Barbosa, "Improving motor imagery EEG classification based on channel selection using a deep learning architecture," *Mathematics*, vol. 10, no. 13, p. 2302, Jul. 2022.

[18] X. Zhang, L. Yao, X. Wang, J. Monaghan, D. McAlpine, and Y. Zhang, "A survey on deep learning-based non-invasive brain signals: Recent advances and new frontiers," *J. Neural Eng.*, vol. 18, no. 3, Jun. 2021, Art. no. 031002.

[19] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.

[20] B. Sun, X. Zhao, H. Zhang, R. Bai, and T. Li, "EEG motor imagery classification with sparse spectrotemporal decomposition and deep learning," *IEEE Trans. Autom. Sci. Eng.*, vol. 18, no. 2, pp. 541–551, Apr. 2021.

[21] X. Chen, R. S. Gupta, and L. Gupta, "Exploiting the cone of influence for improving the performance of wavelet transform-based models for ERP/EEG classification," *Brain Sci.*, vol. 13, no. 1, p. 21, Dec. 2022.

[22] M. V. Valueva, N. N. Nagornov, P. A. Lyakhov, G. V. Valuev, and N. I. Chervyakov, "Application of the residue number system to reduce hardware costs of the convolutional neural network implementation," *Math. Comput. Simul.*, vol. 177, pp. 232–243, Nov. 2020.

[23] Y. Park and W. Chung, "BCI classification using locally generated CSP features," in *Proc. 6th Int. Conf. Brain-Comput. Interface (BCI)*, Jan. 2018, pp. 1–4.

[24] K. Keng Ang, Z. Yang Chin, H. Zhang, and C. Guan, "Filter bank common spatial pattern (FBCSP) in brain–computer interface," in *Proc. IEEE Int. Joint Conf. Neural Netw., IEEE World Congr. Comput. Intell.*, Jun. 2008, pp. 2390–2397.

[25] M. Arvaneh, C. Guan, K. K. Ang, and C. Quek, "Optimizing the channel selection and classification accuracy in EEG-based BCI," *IEEE Trans. Biomed. Eng.*, vol. 58, no. 6, pp. 1865–1873, Jun. 2011.

[26] Y. Park and W. Chung, "Novel BCI classification method using cross-channel-region CSP features," in *Proc. 6th Int. Conf. Brain-Comput. Interface (BCI)*, Jan. 2018, pp. 1–4.

[27] Y. Park and W. Chung, "Optimal channel selection using covariance matrix and cross-combining region in EEG-based BCI," in *Proc. 7th Int. Winter Conf. Brain-Comput. Interface (BCI)*, Feb. 2019, pp. 1–4.

[28] S. A. Raurale, G. B. Boylan, S. R. Mathieson, W. P. Marnane, G. Lightbody, and J. M. O'Toole, "Grading hypoxic-ischemic encephalopathy in neonatal EEG with convolutional neural networks and quadratic time–frequency distributions," *J. Neural Eng.*, vol. 18, no. 4, Aug. 2021, Art. no. 046007.

[29] H. Abbasi, A. J. Gunn, C. P. Unsworth, and L. Bennet, "Advanced deep learning spectroscopy of scalogram infused CNN classifiers for robust identification of post-hypoxic epileptiform EEG spikes," *Adv. Intell. Syst.*, vol. 3, no. 2, 2021, Art. no. 2000198.

[30] R. T. Schirrmeister, J. T. Springenberg, L. D. J. Fiederer, M. Glasstetter, K. Eggensperger, M. Tangermann, F. Hutter, W. Burgard, and T. Ball, "Deep learning with convolutional neural networks for EEG decoding and visualization," *Human Brain Mapping*, vol. 38, no. 11, pp. 5391–5420, Nov. 2017.

[31] L. Alzubaidi, J. Zhang, A. J. Humaidi, A. Al-Dujaili, Y. Duan, O. Al-Shamma, J. Santamaría, M. A. Fadhel, M. Al-Amidie, and L. Farhan, "Review of deep learning: Concepts, CNN architectures, challenges, applications, future directions," *J. Big Data*, vol. 8, no. 1, pp. 1–74, Mar. 2021.

[32] A. Craik, Y. He, and J. L. Contreras-Vidal, "Deep learning for electroencephalogram (EEG) classification tasks: A review," *J. Neural Eng.*, vol. 16, no. 3, Jun. 2019, Art. no. 031001.

[33] A. S. Abdulbaqi, M. T. Younis, Y. T. Younus, and A. J. Obaid, "A hybrid technique for EEG signals evaluation and classification as a step towards to neurological and cerebral disorders diagnosis," *Int. J. Nonlinear Anal. Appl.*, vol. 13, no. 1, pp. 773–781, 2022.

[34] Z. Wang, Y. Wang, C. Hu, Z. Yin, and Y. Song, "Transformers for EEG-based emotion recognition: A hierarchical spatial information learning model," *IEEE Sensors J.*, vol. 22, no. 5, pp. 4359–4368, Mar. 2022.

[35] D. Kim, J. Lee, Y. Woo, J. Jeong, C. Kim, and D.-K. Kim, "Deep learning application to clinical decision support system in sleep stage classification," *J. Personalized Med.*, vol. 12, no. 2, p. 136, 2022.

[36] S. Bagchi and D. R. Bathula, "EEG-ConvTransformer for single-trial EEG-based visual stimulus classification," *Pattern Recognit.*, vol. 129, Sep. 2022, Art. no. 108757.

[37] Y.-E. Lee and S.-H. Lee, "EEG-transformer: Self-attention from transformer architecture for decoding EEG of imagined speech," in *Proc. 10th Int. Winter Conf. Brain-Comput. Interface (BCI)*, Feb. 2022, pp. 1–4.

[38] W. Qu, Z. Wang, H. Hong, Z. Chi, D. D. Feng, R. Grunstein, and C. Gordon, "A residual based attention model for EEG based sleep staging," *IEEE J. Biomed. Health Informat.*, vol. 24, no. 10, pp. 2833–2843, Oct. 2020.

[39] K. Aboalayon, M. Faezipour, W. Almuhammadi, and S. Moslehpour, "Sleep stage classification using EEG signal analysis: A comprehensive survey and new investigation," *Entropy*, vol. 18, no. 9, p. 272, Aug. 2016.

[40] H. Dose, J. S. Møller, H. K. Iversen, and S. Puthusserypady, "An end-to-end deep learning approach to MI-EEG signal classification for BCIs," *Expert Syst. Appl.*, vol. 114, pp. 532–542, Dec. 2018.

[41] X. Tang, W. Li, X. Li, W. Ma, and X. Dang, "Motor imagery EEG recognition based on conditional optimization empirical mode decomposition and multi-scale convolutional neural network," *Expert Syst. Appl.*, vol. 149, Jul. 2020, Art. no. 113285.

[42] X. Lun, Z. Yu, T. Chen, F. Wang, and Y. Hou, "A simplified CNN classification method for MI-EEG via the electrode pairs signals," *Frontiers Hum. Neurosci.*, vol. 14, p. 338, Sep. 2020.

[43] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic minority over-sampling technique," *J. Artif. Intell. Res.*, vol. 16, pp. 321–357, Jun. 2002.

[44] I. Hossain, A. Khosravi, I. Hettiarachchi, and S. Nahavandi, "Multiclass informative instance transfer learning framework for motor imagery-based brain–computer interface," *Comput. Intell. Neurosci.*, vol. 2018, pp. 1–12, Jan. 2018.

[45] I. Hossain, A. Khosravi, and S. Nahavandhi, "Active transfer learning and selective instance transfer with active learning for motor imagery based BCI," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2016, pp. 4048–4055.

[46] H. Fauzi, M. I. Shapiai, and U. Khairuddin, "Transfer learning of BCI using CUR algorithm," *J. Signal Process. Syst.*, vol. 92, no. 1, pp. 109–121, Jan. 2020.

[47] P. L. C. Rodrigues, C. Jutten, and M. Congedo, "Riemannian Procrustes analysis: Transfer learning for brain–computer interfaces," *IEEE Trans. Biomed. Eng.*, vol. 66, no. 8, pp. 2390–2401, Aug. 2019.

[48] F. Li, Y. Xia, F. Wang, D. Zhang, X. Li, and F. He, "Transfer learning algorithm of P300-EEG signal based on XDAWN spatial filter and Riemannian geometry classifier," *Appl. Sci.*, vol. 10, no. 5, p. 1804, Mar. 2020.

[49] P. Gaur, K. McCreadie, R. B. Pachori, H. Wang, and G. Prasad, "Tangent space features-based transfer learning classification model for two-class motor imagery brain–computer interface," *Int. J. Neural Syst.*, vol. 29, no. 10, Dec. 2019, Art. no. 1950025.

[50] A. Salami, M. B. Khodabakhshi, and M. H. Moradi, "Fuzzy transfer learning approach for analysing imagery BCI tasks," in *Proc. Artif. Intell. Signal Process. Conf. (AISP)*, Oct. 2017, pp. 300–305.

[51] A. M. Azab, L. Mihaylova, K. K. Ang, and M. Arvaneh, "Weighted transfer learning for improving motor imagery-based brain–computer interface," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 27, no. 7, pp. 1352–1359, Jul. 2019.

[52] G. Schalk, D. J. McFarland, T. Hinterberger, N. Birbaumer, and J. R. Wolpaw, "BCI2000: A general-purpose brain–computer interface (BCI) system," *IEEE Trans. Biomed. Eng.*, vol. 51, no. 6, pp. 1034–1043, Jun. 2004.

[53] S. Kiranyaz, T. Ince, and M. Gabbouj, "Personalized monitoring and advance warning system for cardiac arrhythmias," *Sci. Rep.*, vol. 7, no. 1, pp. 1–8, Aug. 2017.

[54] O. Avci, O. Abdeljaber, S. Kiranyaz, M. Hussein, and D. J. Inman, "Wireless and real-time structural damage detection: A novel decentralized method for wireless sensor networks," *J. Sound Vib.*, vol. 424, pp. 158–172, Jun. 2018.

[55] O. Avci, O. Abdeljaber, S. Kiranyaz, and D. Inman, "Structural damage detection in real time: Implementation of 1D convolutional neural networks for SHM applications," in *Structural Health Monitoring & Damage Detection*, vol. 7. Cham, Switzerland: Springer, 2017, pp. 49–54.

[56] O. Abdeljaber, O. Avci, M. S. Kiranyaz, B. Boashash, H. Sodano, and D. J. Inman, "1-D CNNs for structural damage detection: Verification on a structural health monitoring benchmark data," *Neurocomputing*, vol. 275, pp. 1308–1317, Jan. 2018.

[57] T. Tamura and K. I. Ohshima, "Mapping of sea ice production in the Arctic coastal polynyas," *J. Geophys. Res., Oceans*, vol. 116, no. C7, pp. 1–20, Jul. 2011.
[58] S. Kiranyaz, A. Gastli, L. Ben-Brahim, N. Al-Emadi, and M. Gabbouj, "Real-time fault detection and identification for MMC using 1-D convolutional neural networks," *IEEE Trans. Ind. Electron.*, vol. 66, no. 11, pp. 8760–8771, Nov. 2019.
[59] O. Abdeljaber, S. Sassi, O. Avci, S. Kiranyaz, A. A. Ibrahim, and M. Gabbouj, "Fault detection and severity identification of ball bearings by online condition monitoring," *IEEE Trans. Ind. Electron.*, vol. 66, no. 10, pp. 8136–8147, Oct. 2019.
[60] L. Eren, T. Ince, and S. Kiranyaz, "A generic intelligent bearing fault diagnosis system using compact adaptive 1D CNN classifier," *J. Signal Process. Syst.*, vol. 91, no. 2, pp. 179–189, Feb. 2019.
[61] S. Sakhavi and C. Guan, "Convolutional neural network-based transfer learning and knowledge distillation using multi-subject data in motor imagery BCI," in *Proc. 8th Int. IEEE/EMBS Conf. Neural Eng. (NER)*, May 2017, pp. 588–591.
[62] O.-Y. Kwon, M.-H. Lee, C. Guan, and S.-W. Lee, "Subject-independent brain–computer interfaces based on deep convolutional neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 10, pp. 3839–3852, Oct. 2020.
[63] M. Wronkiewicz, E. Larson, and A. K. C. Lee, "Leveraging anatomical information to improve transfer learning in brain–computer interfaces," *J. Neural Eng.*, vol. 12, no. 4, Aug. 2015, Art. no. 046027.
[64] M. Taghizadeh and K. Mohammadi, "The fast and accurate approach to detection and segmentation of melanoma skin cancer using fine-tuned YOLOv3 and SegNet based on deep transfer learning," 2022, *arXiv:2210.05167*.
[65] T. Karácsony, J. P. Hansen, H. K. Iversen, and S. Puthusserypady, "Brain computer interface for neuro-rehabilitation with deep learning classification and virtual reality feedback," in *Proc. 10th Augmented Hum. Int. Conf.*, Mar. 2019, pp. 1–8.
[66] F. Mattioli, C. Porcaro, and G. Baldassarre, "A 1D CNN for high accuracy classification and transfer learning in motor imagery EEG-based brain–computer interface," *J. Neural Eng.*, vol. 18, no. 6, Dec. 2021, Art. no. 066053.
[67] X. Liu, S. Xiong, X. Wang, T. Liang, H. Wang, and X. Liu, "A compact multi-branch 1D convolutional neural network for EEG-based motor imagery classification," *Biomed. Signal Process. Control*, vol. 81, Mar. 2023, Art. no. 104456.
[68] A. Ghimire and K. Sekeroglu, "Classification of EEG motor imagery tasks utilizing 2D temporal patterns with deep learning," in *Proc. 2nd Int. Conf. Image Process. Vis. Eng.*, 2022, pp. 182–188.
[69] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?" in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 27, 2014, pp. 1–9.
[70] C. Brunner, R. Leeb, G. Müller-Putz, A. Schlögl, and G. Pfurtscheller, "BCI competition 2008-Graz data set A," *Inst. Knowl. Discovery, Lab. Brain-Comput. Interfaces, Graz Univ. Technol.*, vol. 16, pp. 1–6, Jan. 2008.
[71] G. Dornhege, B. Blankertz, G. Curio, and K.-R. Müller, "Boosting bit rates in noninvasive EEG single-trial classifications by feature combination and multiclass paradigms," *IEEE Trans. Biomed. Eng.*, vol. 51, no. 6, pp. 993–1002, Jun. 2004.
[72] A. M. Roy, "Adaptive transfer learning-based multiscale feature fused deep convolutional neural network for EEG MI multiclassification in brain–computer interface," *Eng. Appl. Artif. Intell.*, vol. 116, Nov. 2022, Art. no. 105347.
[73] A. Li, Z. Wang, X. Zhao, T. Xu, T. Zhou, and H. Hu, "MDTL: A novel and model-agnostic transfer learning strategy for cross-subject motor imagery BCI," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 31, pp. 1743–1753, 2023.
[74] D. Liu, J. Zhang, H. Wu, S. Liu, and J. Long, "Multi-source transfer learning for EEG classification based on domain adversarial neural network," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 31, pp. 218–228, 2023.
[75] H. Wu, Y. Niu, F. Li, Y. Li, B. Fu, G. Shi, and M. Dong, "A parallel multiscale filter bank convolutional neural networks for motor imagery EEG classification," *Frontiers Neurosci.*, vol. 13, p. 1275, Nov. 2019.

MOHAMAD TAGHIZADEH received the M.Sc. degree in electrical engineering from Iran University of Science and Technology, Tehran, Iran. Garnered through a deep scientific passion from an early age, his research interests exist largely within the field of healthcare technology, digital/biomedical embedded hardware/software co-designs, biomedical signal/image processing, computer vision, artificial intelligence, machine learning deep learning, transfer learning, and the IoT.

FATEMEH VAEZ received the M.Sc. degree in telecommunications engineering in Iran. Her M.Sc. thesis focused on network coding and signal processing. Her research interests include artificial intelligence, signal processing, and computer vision.

MIAD FAEZIPOUR (Senior Member, IEEE) received the B.Sc. degree in electrical engineering from the University of Tehran, Iran, and the M.Sc. and Ph.D. degrees in electrical engineering from The University of Texas at Dallas. She is currently an Associate Professor of electrical and computer engineering technology with the School of Engineering Technology, Purdue University. She is also a Full Member of the Regenstrief Center for Healthcare Engineering (RCHE), Purdue University. She is also the Founder and the Director of the Digital/Biomedical Embedded Systems and Technology (D-BEST) Research Laboratory. She has been a Postdoctoral Research Associate with The University of Texas at Dallas collaborating with the Center for Integrated Circuits and Systems (CICS) and Quality of Life Technology (QoLT) Research Laboratory, after graduation. Prior to joining Purdue University, she was a Faculty Member with the Computer Science and Engineering and Biomedical Engineering Programs, University of Bridgeport, CT, USA, for ten years. Her research interests include healthcare technology with embedded intelligence, digital/biomedical embedded hardware/software co-designs, biomedical signal/image processing, computer vision, healthcare/biomedical informatics, artificial intelligence, machine/deep learning, and AI-based biodata augmentation. She is a Senior Member of EMBS and IEEE Women in Engineering.