MOCNN: A Multi-scale Deep Convolutional Neural Network for ERP-based Brain-Computer Interface

MOCNN: A Multi-scale Deep Convolutional Neural Network for ERP-based Brain-Computer Interfaces

Jing Jin, *Senior Member, IEEE*, Ruitian Xu, Ian Daly, Xueqing Zhao, Xingyu Wang,
and Andrzej Cichocki, *Fellow, IEEE*

***Abstract*—Event-related potentials (ERPs) reflect neurophysiological changes of the brain in response to external events and their associated underlying complex spatiotemporal feature information is governed by ongoing oscillatory activity within the brain. Deep learning methods have been increasingly adopted for ERP-based Brain-Computer Interfaces (BCIs) due to their excellent feature representation abilities, which allow for deep analysis of oscillatory activity within the brain. Features with higher spatiotemporal frequencies usually represent detailed and localized information, while features with lower spatiotemporal frequencies usually represent global structures. Mining EEG features from multiple spatiotemporal frequencies is conducive to obtaining more discriminative information. A multi-scale feature fusion octave convolution neural network (MOCNN) is proposed in this paper. MOCNN divides the ERP signals into high-, medium- and low-frequency components corresponding to different resolutions and processes them in different branches. By adding mid- and low-frequency components, the feature information used by MOCNN can be enriched, and the required amount of calculations can be reduced. After successive feature mapping using temporal and spatial convolutions, MOCNN realizes interactive learning among different components through the exchange of feature information among branches. Classification is accomplished by feeding the fused deep spatiotemporal features from various components into a fully connected layer. The results, obtained on two public datasets and a self-collected ERP dataset, show that MOCNN can achieve state-of-the-art ERP classification performance. In this study, the generalized concept of octave convolution is introduced into the field of ERP-BCI research, which allows effective spatiotemporal features to be extracted from multi-scale networks through branch width optimization and information interaction at various scales.**

***Index Terms*—Brain-computer interfaces, event-related potentials, deep learning, multi-scale, octave convolution neural network.**

I. Introduction

BRAIN-COMPUTER Interface (BCI) systems allow individuals to directly communicate with the external environment or control external devices through brain signals [1], [2]. Common electroencephalogram (EEG) signal components that are used to control BCIs include event-related potentials (ERPs) [3], [4], steady-state visual evoked potentials (SSVEPs) [5], [6], slow cortical potentials (SCPs) [7], and the sensorimotor rhythm [8]-[12]. ERPs refer to voltage changes that are time-locked to sensory, motor, or cognitive events during ongoing EEG activity [13]. ERP-BCI systems have become one of the most significant BCI systems because of their advantages, such as relatively fast speed, applicability to most people, and ease of operation [14]. ERP-BCI systems are being applied to decode the intentions of patients with mobility difficulties and have successfully assisted them in many applications such as completing typing tasks, controlling household appliances, and driving wheelchairs [15]-[18]. These systems can enhance their user’s independent living ability, which improves their quality of life. Therefore, it is of great practical significance to study ERP-BCIs. However, it is difficult to establish a stable mathematical model of the ERP due to the non-stationarity, low Signal to Noise Ratio (SNR), and inter-person variability of EEG signals [19]-[23]. This hinders the reliable identification of ERPs.

Previous studies have proposed various traditional machine learning algorithms that can be applied to the classification of ERPs within ERP-based BCIs. For example, Rivet et al. adopted xDAWN to project raw EEG signals into an estimated signal subspace to optimize the SNR [24]. Blankertz et al. introduced the shrinkage linear discriminant analysis

This work was supported by the Grant National Natural Science Foundation of China under Grant 62176090 and STI 2030-major projects 2022ZD0208900; in part by Shanghai Municipal Science and Technology Major Project under Grant 2021SHZDZX，This research is also supported by Project of Jiangsu Province Science and Technology Plan Special Fund in 2022 (Key research and development plan industry foresight，fundamental research fund for the central universities JKH01231636 and key core technologies) under Grant BE2022064-1. (Corresponding author: Jing Jin.)

Jing Jin, Ruitian Xu, Xueqing Zhao and Xingyu Wang are with the Key Laboratory of Smart Manufacturing in Energy Chemical Process, East China University of Science and Technology, Ministry of Education, Shanghai 200237, P.R. China (e-mail: jinjingat@gmail.com; xrt_1016@163.com; xueqingzhao2021@163.com; xywang@ecust.edu.cn). Jing Jin is also with Shenzhen Research Institute of East China University of Science and Technology, Shenzhen, 518063, PR China.

Ian Daly is with the Brain-Computer Interfacing and Neural Engineering Laboratory, School of Computer Science and Electronic Engineering, University of Essex, Wivenhoe Park, Colchester, Essex CO4 3SQ, UK (e-mail: i.daly@essex.ac.uk).

Andrzej Cichocki is with the Laboratory for Advanced Brain Signal Processing, RIKEN Brain Science Institute, Japan and the Systems Research Institute of Polish Academy of Science, 01-447 Warsaw, Poland, Department of Informatics, Nicolaus Copernicus University, 87-100 Torun, Poland (e-mail: a.cichocki@riken.jp).

(SKLDA) method to achieve accurate covariance matrix estimation in the high dimensional space of ERP signals [25]. Sajda et al. designed a hierarchical discriminant component analysis (HDCA) method, a two-stage method that combines first spatial and then temporal activities [26]. Mobaien et al. formed a regularized version of xDAWN (RxDAWN) by adding constraints to the original problem in order to enhance the ERP signal [27]. Traditional methods are mainly based on linear constraints, which allows them to be trained faster and results in relatively robust decoding models. However, their performance may be limited by the efficacy of the feature extraction stage [28].

Deep learning, as a nonlinear approach [29], has shown great potential in ERP detection in recent years. Schirrmeister et al. studied deep convolutional networks with a range of different architectures, ultimately designing the DeepConvNet model with five convolutional blocks for end-to-end decoding of the raw EEG signal [30]. Lawhern et al. proposed a compact convolutional neural network, EEGNet, constructed by deep convolution and separable convolution for EEG-based BCIs [31]. Santamaria-Vazquez et al. integrated inception modules in the convolutional neural network to form the EEG-Inception model [20]. Li et al. learned phase information to improve the EEG classification performance in a rapid serial visual presentation (RSVP) task with a phase preservation neural network (PPNN) [32]. Li et al. combined self-supervised learning with supervised learning through a multi-task collaborative network (MTCN) to extract more generalized ERP features [33]. The effectiveness of these deep learning models for ERP classification has been verified in experiments.

EEG is considered to be a complex signal consisting of transient and oscillatory patterns across different time lengths that reflects brain activity [20], [34]. EEG is also a multi-scale signal, meaning sampling the signal with different granularities yields different levels of information about the signal. It is generally the case that smaller/denser sampling can reveal more details, while larger/sparser sampling can reveal the overall trend of a process [35]. The combination of details and trends makes multi-scale learning a potentially suitable tool for EEG decoding. Multi-scale CNN (MSCNN) models have been successfully applied to ERP detection [20], [36]-[38], and the experimental results show that more characteristic information can be extracted from ERP signals at multiple scales. However, research into ERP recognition based on MSCNN still faces some challenges.

First most existing MSCNN models assign the same width, namely the same number of filters, to each scale, implementing their respective feature representations, but ignore the optimal width allocation between different scales under the same resource constraints, preventing further improvement of the overall performance. Second, before feature fusion, feature mapping is always carried out independently in each scale branch without the benefit of the enhancement effect of intermediate information exchange between the features of each scale. Finally, multi-scaling is usually achieved by convolution of different kernel sizes to the same input, which can lead to information redundancy [20], [36]-[38].

To address these challenges, we develop a multi-scale feature fusion octave convolution neural network model (MOCNN) specifically for ERP classification. Octave convolution (OctConv), proposed by Chen et al. in the computer vision field [39], is used to store and process low- and high-frequency features separately to extract richer feature maps with less computational cost and more robustness. We introduce OctConv and build a multi-branch network architecture with the goal of enhancing the accuracy and efficiency of ERP-BCI systems and making them more practical for real-world applications. The major contributions of this paper can be summarized as follows:

- We propose a multi-branch convolutional neural network that captures ERP features at multi-scale resolution and optimizes the distribution of each branch width under the condition that the network width is kept constant.
- We apply the generalized concept of OctConv to ERP detection. The high-, medium- and low-frequency components of ERP signals are processed separately and interactive learning among the branches is implemented, this allows the discriminative multi-scale information to be extracted and achieving mutual complement of information between branches.
- Through extensive experimentation on two public datasets and a self-collected dataset, we obtain classification results that outperform the state-of-the-art ERP detection methods. Meanwhile, the computational cost of MOCNN is also significantly reduced compared with other deep learning methods.

This paper is organized as follows. Section II introduces the datasets used and the data processing flow of MOCNN. Section III illustrates the experimental results. Sections IV and V provide the discussion and conclusions, respectively.

II. Materials and Methods

A. Datasets Description

The three datasets used in this work are taken from Brain/Neural Computer Interaction (BNCI) Horizon 2020 database, Lausanne Federal Institute of Technology (EPFL) BCI group, and our spelling experiment, named datasets I, II, and III, respectively. The details of these datasets are described as follows.

1) Dataset I

The dataset was from a BCI-speller based experiment using a rapid serial visual presentation (RSVP) paradigm, which was conducted by Acqualagna et al [40]. Twelve healthy participants took part in the experiment. The study was performed in accordance with the declaration of Helsinki and all participants gave written informed consent. In the RSVP paradigm, 30 symbols with different colors and different capitalizations were randomly presented, one after another, in the center of the screen. The participants were required to concentrate on the target letter and count its number of occurrences silently. The EEG was recorded at 1000 Hz using BrainAmp amplifiers and an actiCap active electrode system with 63 channels (Fp1/2, AF3/4, Fz, F1-10, FCz, FC1-6, FT7/8, T7/8, Cz, C1-6, TP7/8, CPz, CP1-6, Pz, P1-10, POz, PO3/4/7-10, Oz and O1/2). Each participant completed both offline and online phases of the experiment. Each participant was given 24 characters in the offline phase and an average of 41 characters in the online phase to spell. In this study, the EEG data from 11 participants were used because one participant's data lacked two channels (P8 and O2). We randomly selected trials focused on 20 of the characters from the offline data for use as our training data, the trials for the remaining 4 characters were used as our validation data, and the online data was used as our testing data. For more information on Dataset I, please refer to http://bnci-horizon-2020.eu/database/data-sets.

2) Dataset II

The dataset was from an experimental evaluation of a P300-based BCI system for disabled participants developed by Hoffmann et al. [41]. EEG data was recorded from 4 disabled and 4 healthy participants. In the P300 paradigm, six images were flashed in random sequences on a laptop screen, with each flash of an image lasting 100 ms and 400 ms interstimulus intervals. The participants were required to focus on specified target images. The EEG was recorded at a 2048 Hz sampling rate using a Biosemi Active Two amplifier from 32 electrodes (Fp1, AF3, F7, F3, FC1, FC5, T7, C3, CP1, CP5, P7, P3, Pz, PO3, O1, Oz, O2, PO4, P4, P8, CP6, CP2, C4, T8, FC6, FC2, F4, F8, AF4, Fp2, Fz, and Cz) placed at the standard positions of the international 10-20 system. A total of 24 runs over four recording sessions were completed by every participant, and each run is composed of an average of 22.5 epochs of six flashes. In our study, we include the EEG data from all participants. We randomly selected 10 runs as the training data, 4 as the validation data, and the remaining 10 as the testing data. For more information on Dataset II, please refer to www.epfl.ch/labs/mmspg/research/page-58317-en-html/bci-2/bci_datasets/.

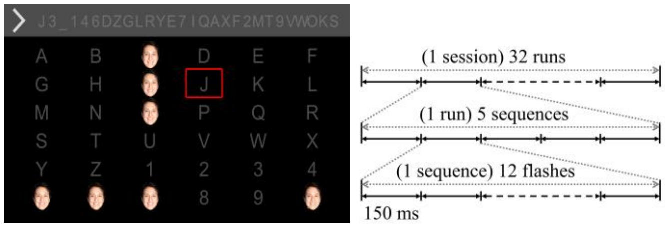
3) Dataset III

Participants: Eight participants (5 males and 3 females, aged 19-26 years) with normal or corrected-to-normal vision were recruited for this P300 spelling experiment. None of the participants reported a previous history of visual impairment, neurological disease, or injury. The experimental procedures (Document Number: ECUST-2022-054) were approved by the Local Institutional Review Board.

Paradigm: The visual stimulation interface was a 6×6 checkerboard layout (including the letters A-Z, numbers 1-9, and underscores) as shown in Fig. 1 (a), with a text box above displaying the characters to be spelled. In the paradigm, the stimuli were presented in random order via a set of binomial flashes [42], [43], and the flashing characters were overlaid with pictures of faces (faces generated by code, not real people) [44]. Each participant participated in one experimental session. The timing of one session is shown in Fig. 1 (b). Participants were asked to complete random spellings of 32 characters during a session, spelling one character as a run. The character to be spelled was prompted for 1s at the start of each run. Each run consisted of 5 sequences, and each sequence contained 12 flashes, with characters flashed in accordance with the binomial rule defined in this paradigm. The flashes were presented in such a way that faces appeared and disappeared for 75 ms each, thus a single flash event lasted for 150 ms.

EEG recording: EEG signals were recorded at a sampling rate of 1000 Hz using wireless EEG equipment with 59 scalp electrodes (Fpz, Fp1/2, AF3/4/7/8, Fz, F1-8, FCz, FC1-6, FT7/8, Cz, C1-6, T7/8, CP1-6, TP7/8, Pz, P3-8, POz, PO3-8, Oz and O1/2) arranged in a standard 10-20 montage.

In this study, 20 runs were randomly selected for use as the training data, 4 runs for use as the validation data, and the remaining 12 runs for use as the testing data for each participant.



(a) (b)

Fig. 1. Experimental paradigm. (a) The binomial face visual stimulation interface. (b) The timing of one session.

B. Data Preprocessing

The same preprocessing is performed on all three datasets. The EEG data is filtered using a Butterworth bandpass filter with frequencies ranging from 0.5 to 30 Hz, then the sampling rate is reduced to 128 Hz to reduce the data volume. The trials used for classification are extracted within the time window of 0 to 1000 ms after the onset of the reinforcement period. Each trial was then normalized by z-score to ensure uniform measurements.

C. MOCNN

We introduce the concept of OctConv into ERP detection. Specifically, we integrate an OctConv module into our multi-scale convolutional neural network model to construct our proposed MOCNN model. The framework of our MOCNN model is illustrated in Fig. 2. This MOCNN model consists of five major modules: FirstOctConv, SpatialConv, CoreOctConv, LastOctConv, and Classification. The detailed design of each module is as follows.

1) Module FirstOctConv

The main purpose of the Module FirstOctConv is to take input signals, divide them into distinct frequency components,

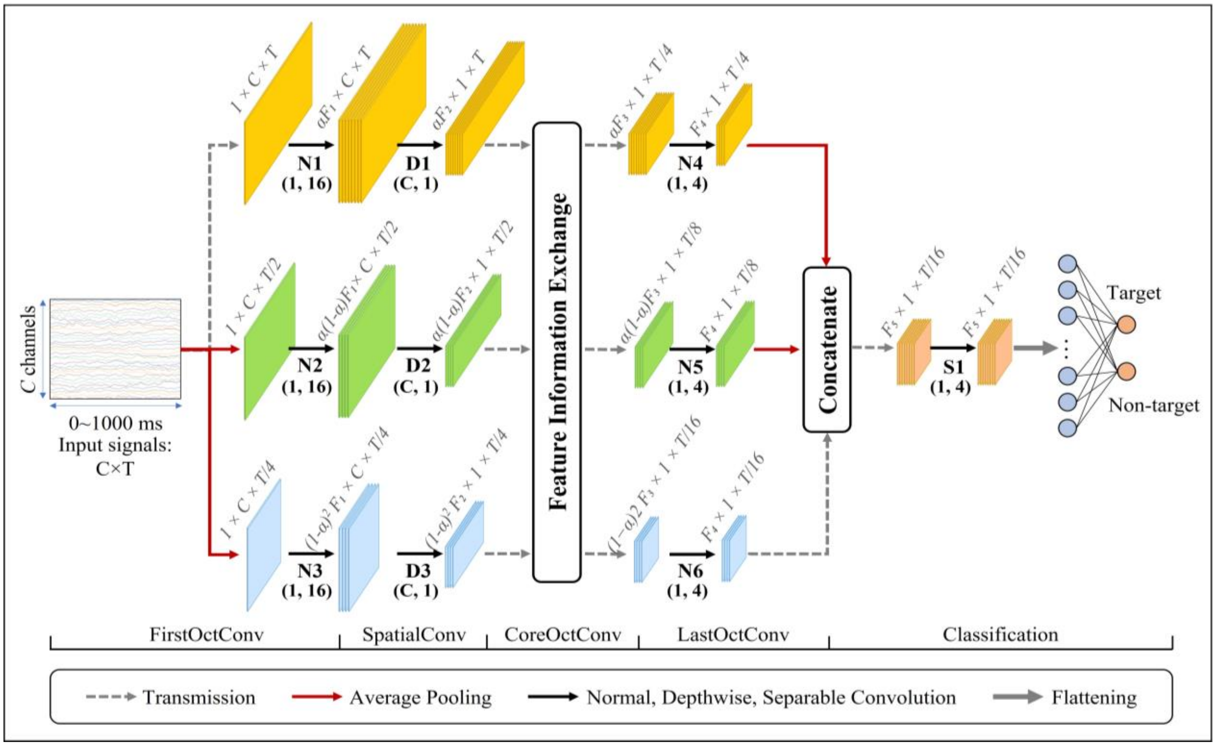


Fig. 2. The framework of MOCNN. MOCNN is divided into five modules: FirstOctConv, SpatialConv, CoreOctConv, LastOctConv, and Classification. The three types of convolution in this framework are Normal Convolution, denoted by N; Depthwise Convolution, denoted by D; and Separable Convolution, denoted by S.

and then distribute these components across the branches to extract temporal features. Information generally contains components of different frequencies. The higher frequencies are usually encoded in fine detail, while the lower frequencies are usually encoded in the global structure [39]. In this paper, we adopt a signal decomposition approach that separates the input into high-, medium-, and low-frequency components by adjusting the time resolution. The three components are encoded and decoded separately by the three parallel branches of the model. The input feature map of the parallel branches can be expressed as:

$$\begin{aligned} &X_{in} = \{X_{h_1}, X_{m_1}, X_{l_1}\} \\ &X_{h_1} = X;\ \ X_{m_1} = p(X, \vartheta_{m_1});\ \ X_{l_1} = p(X, \vartheta_{l_1}) \end{aligned} \tag{1}$$

where subscript h denotes high-frequency, m medium-frequency, and l to low-frequency components. X_{h_1} is equal to the preprocessed signal $X \in R^{C \times T}$, X_{m_1} is obtained by performing an average pooling operation $p(\cdot)$ with size $\vartheta_{m_1} = (1,2)$ on X, and X_{l_1} is obtained by $p(\cdot)$ with size $\vartheta_{l_1} = (1,4)$. Temporal convolutions (N1, N2, and N3) are then applied to the three branches. In this module, the total number of filters remains constant, and the hyperparameter α determines the allocation of filters for learning high-, medium-, and low-frequency components. Hence, the outputs of N1, N2, and N3 can be expressed as:

$$\begin{aligned} &Y_{nh_1} = g(X_{h_1}; \alpha F_1; \kappa_1) \\ &Y_{nm_1} = g(X_{m_1}; \alpha(1-\alpha)F_1; \kappa_1) \\ &Y_{nl_1} = g(X_{l_1}; (1-\alpha)^2 F_1; \kappa_1) \end{aligned} \tag{2}$$

where $g(X; \omega; \kappa)$ denotes a convolution on X with the number of filters defined by ω and the kernel size κ. A similar pattern of filter allocation also exists in the later modules. Then batch normalization (BN) is used after each convolution.

2) Module SpatialConv

The SpatialConv module serves the primary purpose of extracting spatial features. It encompasses layers D1, D2, and D3, along with their corresponding BN layers, activation layers, and dropout layers. D1, D2, and D3 are depth wise convolution layers with a depth of 1, which compress the channel dimension from C to 1, thus reducing the spatial data dimension. The outputs of D1, D2, and D3 can be expressed as:

$$\begin{aligned} &Y_{dh_2} = d(X_{h_2}; \alpha F_2; \kappa_2; \alpha F_1) \\ &Y_{dm_2} = d(X_{m_2}; \alpha(1\text{-}\alpha)F_2; \kappa_2; \alpha(1\text{-}\alpha)F_1) \\ &Y_{dl_2} = d(X_{l_2}; (1\text{-}\alpha)^2 F_2; \kappa_2; (1\text{-}\alpha)^2 F_1) \end{aligned} \tag{3}$$

where $d(X, \omega, \kappa, \tau)$ denotes a depth wise convolution on X with the number of filters denoted by ω, kernel size κ, and group size τ, and $X_2 = \{X_{h_2}, X_{m_2}, X_{l_2}\}$ denotes the input of the SpatialConv module. Depth wise convolution performs convolution independently on each filter dimension, which can effectively reduce the number of parameters [45]. Then, a BN layer, an ELU activation function, and a dropout layer are applied after each deep convolution layer to accelerate training and reduce the risk of overfitting.

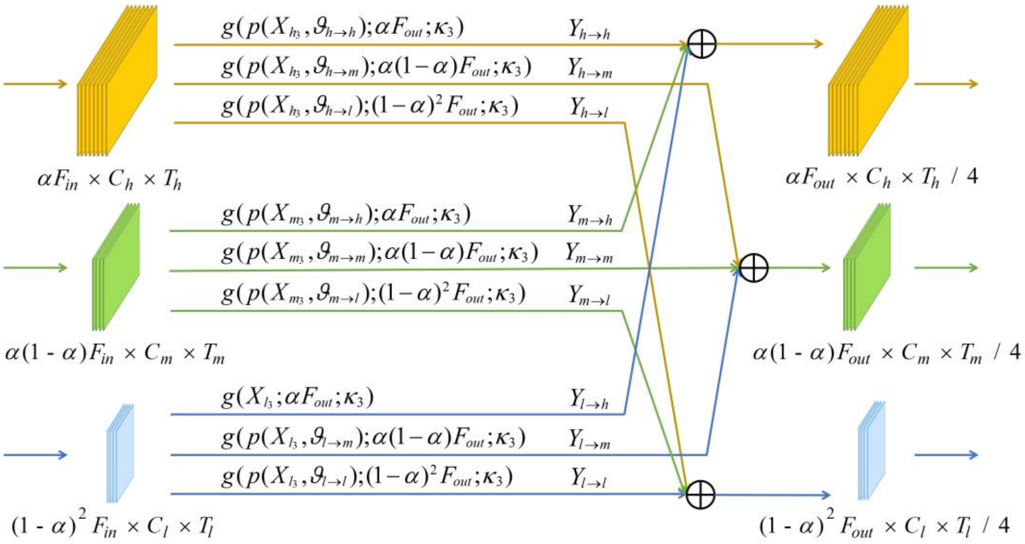

$g(p(X_{h_3},\vartheta_{h\to h});\alpha F_{out};\kappa_3)$
$Y_{h\to h}$
$g(p(X_{h_3},\vartheta_{h\to m});\alpha(1-\alpha)F_{out};\kappa_3)$
$Y_{h\to m}$
$g(p(X_{h_3},\vartheta_{h\to l});(1-\alpha)^2F_{out};\kappa_3)$
$Y_{h\to l}$
$\alpha F_{in}\times C_h\times T_h$
$\alpha F_{out}\times C_h\times T_h\ /\ 4$
$g(p(X_{m_3},\vartheta_{m\to h});\alpha F_{out};\kappa_3)$
$Y_{m\to h}$
$g(p(X_{m_3},\vartheta_{m\to m});\alpha(1-\alpha)F_{out};\kappa_3)$
$Y_{m\to m}$
$g(p(X_{m_3},\vartheta_{m\to l});(1-\alpha)^2F_{out};\kappa_3)$
$Y_{m\to l}$
$\alpha(1\text{ - }\alpha)F_{in}\times C_m\times T_m$
$\alpha(1\text{ - }\alpha)F_{out}\times C_m\times T_m\ /\ 4$
$g(X_{l_3};\alpha F_{out};\kappa_3)$
$Y_{l\to h}$
$g(p(X_{l_3},\vartheta_{l\to m});\alpha(1-\alpha)F_{out};\kappa_3)$
$Y_{l\to m}$
$g(p(X_{l_3},\vartheta_{l\to l});(1-\alpha)^2F_{out};\kappa_3)$
$Y_{l\to l}$
$(1\text{ - }\alpha)^2F_{in}\times C_l\times T_l$
$(1\text{ - }\alpha)^2F_{out}\times C_l\times T_l\ /\ 4$

Fig.3. Main design of the CoreOctConv module. Yellow rectangles indicate high-frequency features, green ones indicate medium-frequency features, and blue ones indicate low-frequency features.

3) Module CoreOctConv

The goal of the CoreOctConv module is to effectively deal with high-, medium- and low-frequency features and achieve efficient inter-frequency communication. As seen in Fig. 3, the input and output of this module are defined as $X_3=\{X_{h_3}, X_{m_3}, X_{l_3}\}$, $Y_3=\{Y_{h_3}, Y_{m_3}, Y_{l_3}\}$, where $X_{h_3}\in R^{\alpha F_{in}\times C_h\times T_h}$, $X_{m_3}\in R^{\alpha(1\text{-}\alpha)F_{in}\times C_m\times T_m}$, $X_{l_3}\in R^{(1\text{-}\alpha)^2F_{in}\times C_l\times T_l}$, $Y_{h_3}\in R^{\alpha F_{out}\times C_h\times T_h/4}$, $Y_{m_3}\in R^{\alpha(1\text{-}\alpha)F_{out}\times C_m\times T_m/4}$, $Y_{l_3}\in R^{(1\text{-}\alpha)^2F_{out}\times C_l\times T_l/4}$, with F_{in} and F_{out} representing the number of input and output feature maps, the hyperparameter α controls the ratio of filter number to learn high-, medium-, and low-frequency features. The terms Y_{h_3}, Y_{m_3} and Y_{l_3} are given by:

$$\begin{aligned}
Y_{h_3} &= Y_{h\to h}+Y_{m\to h}+Y_{l\to h}\\
Y_{m_3} &= Y_{h\to m}+Y_{m\to m}+Y_{l\to m}\\
Y_{l_3} &= Y_{h\to l}+Y_{m\to l}+Y_{l\to l}
\end{aligned}\tag{4}$$

where $Y_{a\to b}$ indicates the convolution update from feature a to b. Therefore, $Y_{h\to h}$, $Y_{m\to m}$, $Y_{l\to l}$ represent intra-frequency updates, $Y_{h\to m}$, $Y_{h\to l}$, $Y_{m\to h}$, $Y_{m\to l}$, $Y_{l\to h}$, $Y_{l\to m}$ represent inter-frequency updates. Specifically,

$$\begin{aligned}
Y_{h\to h} &= g(p(X_{h_3},\vartheta_{h\to h});\alpha F_{out};\kappa_3)\\
Y_{m\to m} &= g(p(X_{m_3},\vartheta_{m\to m});\alpha(1-\alpha)F_{out};\kappa_3)\\
Y_{l\to l} &= g(p(X_{l_3},\vartheta_{l\to l});(1-\alpha)^2F_{out};\kappa_3)\\
Y_{h\to m} &= g(p(X_{h_3},\vartheta_{h\to m});\alpha(1-\alpha)F_{out};\kappa_3)\\
Y_{h\to l} &= g(p(X_{h_3},\vartheta_{h\to l});(1-\alpha)^2F_{out};\kappa_3)\\
Y_{m\to h} &= g(p(X_{m_3},\vartheta_{m\to h});\alpha F_{out};\kappa_3)\\
Y_{m\to l} &= g(p(X_{m_3},\vartheta_{m\to l});(1-\alpha)^2F_{out};\kappa_3)\\
Y_{l\to h} &= g(X_{l_3};\alpha F_{out};\kappa_3)\\
Y_{l\to m} &= g(p(X_{l_3},\vartheta_{l\to m});\alpha(1-\alpha)F_{out};\kappa_3)
\end{aligned}\tag{5}$$

where $p(\cdot)$ and $g(\cdot)$ still represent pooling and convolution operations described earlier. As high-, medium-, and low-frequency features differ in the filter and time dimensions and cannot be directly combined, pooling with varying sizes ($\vartheta_{h\to h}=\vartheta_{m\to m}=\vartheta_{l\to l}=(1,4)$; $\vartheta_{h\to m}=\vartheta_{m\to l}=(1,8)$; $\vartheta_{h\to l}=(1,16)$; $\vartheta_{m\to h}=\vartheta_{l\to m}=(1,2)$) is employed to adapt the temporal dimension during inter-frequency updates, while convolution is used to adjust the filter dimension during intra-frequency updates. After the information exchange, the corresponding features of each component need to be transmitted into their respective BN layer, ELU layer, and dropout layer.

4) Module LastOctConv

The LastOctConv module serves as a fusion module with the primary responsibility of reintegrating the separately processed high-, middle-, and low-frequency components. The mode of concatenation in the filter dimension is chosen to fuse the three components. Before the cascade, convolution layers N4, N5, and N6 are adopted to further extract features. The outputs of N4, N5, and N6 can be expressed as:

$$\begin{aligned}
Y_{nh_4} &= g(X_{h_4};F_4;\kappa_4)\\
Y_{nm_4} &= g(X_{m_4};F_4;\kappa_4)\\
Y_{nl_4} &= g(X_{l_4};F_4;\kappa_4)
\end{aligned}\tag{6}$$

which are then processed by the respective BN layers, ELU layers, and dropout layer. The time dimension of the high-frequency component is average pooled with size (1,4), the medium-frequency component is average pooled with size (1,2), and the low-frequency component remains unchanged. This ensures the time dimension of all three components becomes consistent. Following this, the output of the three branches is concatenated together to produce the fused feature set.

5) Module Classification

The main function of the Classification module is to make use of the fused features to generate the classification output. In this module, a separable convolution S1 is used to further adjust the information within the fused features. A separable convolution consists of a depth wise convolution and a pointwise convolution [45]. Separable convolution has the same advantages as a depth wise convolution. In addition, pointwise convolution enables separable convolution to achieve feature fusion of different filters [31]. The high-, medium- and low-frequency features are combined in the filter dimension to form the fusion feature, and the combination of features at different time scales is optimized through S1. The output of S1 can be expressed as:

$$Y_{sh_5} = s(X_5; F_5; \kappa_5; F_5) \tag{7}$$

where $s(X,\omega,\kappa,\tau)$ denotes a separable convolution on X with the number of filters denoted by ω, kernel size κ, and group size τ. The term X_5 is used to denote the fusion feature. Before the flattening operation, the BN, ELU activation, and dropout operations are applied. Binary classification is achieved using a fully connected network, and the softmax activation function is used to obtain the probability that the input sample belongs to the target or non-target class. The output is the label with the greater probability:

$$MOCNN\left(X\right) = \begin{cases} 1, & if\ P_{\text{Target}} > P_{\text{Non-target}} \\ 0, & otherwise \end{cases} \tag{8}$$

where X is the input sample, P_{Target} represents the probability that X is the target sample, $P_{\text{Non-target}}$ represents the probability that X is the non-target sample.

D. Comparison Algorithms

1) xDAWN [24], [46]: xDawn is an unsupervised method that enhances P300 evoked potentials by projecting an originally recorded EEG signal onto an estimated evoked subspace. Bayesian linear discriminant analysis (BLDA) is used for classification after xDawn spatial filtering.

2) HDCA [26]: HDCA learns spatial weights via Fisher linear discriminants (FLD) and then temporal weights via logistic regression, enabling classification. It is easy to implement and computationally efficient.

3) EEGNet [31]: EEGNet is a compact convolutional neural network designed for Brain-Computer Interfaces based on EEG, which demonstrates excellent performance across multiple paradigms.

4) EEG-inception [20]: EEG-inception is the first model that integrates the Inception module for ERP detection. It integrates effectively with other structures in a lightweight architecture to improve the accuracy and calibration time of the auxiliary ERP-based Brain-Computer Interface.

5) PPNN [32]: PPNN utilizes a stack of dilated temporal convolution layers to extract temporal features and preserve phase information. Then the channel correlation is obtained through a spatial convolution layer. Classification is achieved with a fully connected layer.

E. Implementation Details

Our proposed MOCNN model and the deep learning models which we compare it to are all constructed using PyTorch [47]. We employ the Adam [48] optimizer with an initial learning rate of 0.001 during the training process. The mini-batch size is set to 512, and the models are trained for 500 epochs. To avoid overfitting and reduce the training time, we employ early stopping [49] to stop training when the loss of the validation data does not improve for 20 consecutive epochs. The weighted cross-entropy loss function is applied to address data imbalance. All models in this work use the same training settings. For each dataset, we run all the methods 5 times and average the results to decrease the effect of randomness.

In MOCNN, except for the one following S1, all dropout layers are implemented using spatial dropout [50] with a rate of 0.5. Spatial dropout randomly zeros out some regions rather than some elements to avoid the destruction of the spatial correlation between neurons [51]. The structural parameters are $F_1 = 16$, $F_2 = 128$, $F_3 = 32$, $F_4 = 2$, $F_5 = 6$ and the convolution parameters are $\kappa_1 = (1,16)$, $\kappa_2 = (C,1)$, $\kappa_3 = (1,8)$, $\kappa_4 = (1,4)$, $\kappa_5 = (1,4)$. The hyperparameter α is set as 0.75. Bias units are omitted in all convolution layers.

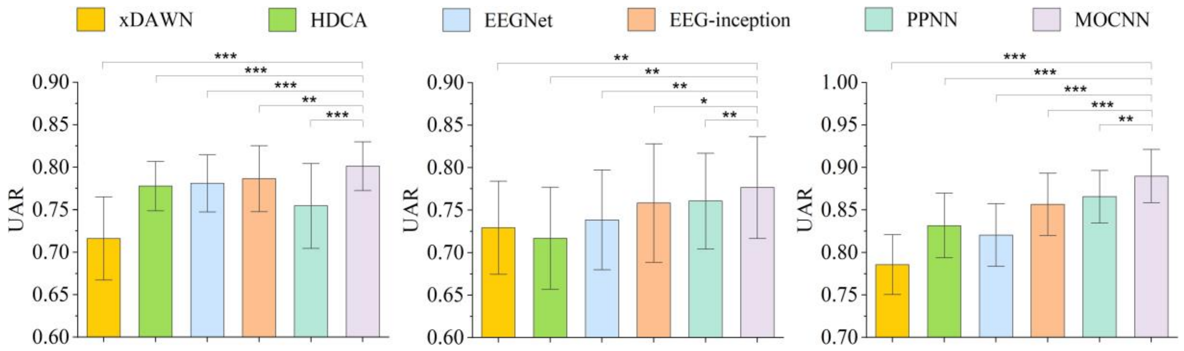
The hardware information of the computer used in the experiment is as follows: 11th Gen Intel(R) Core(TM) i7-11800H @ 2.30GHz, NVIDIA GeForce RTX 3060 Laptop GPU.

III. Results

A. Performance Evaluation of ERP Detection

ERP detection in BCI systems is an imbalanced data classification problem. Unweighted average recall (UAR) [35] is the average accuracy over all classes, which is a widely used metric for class imbalance problems. Therefore, we adopt UAR as one of the criteria to evaluate the classification performance for all methods in this study. We compare our proposed MOCNN model with several traditional algorithms (xDAWN and HDCA) and deep learning algorithms (EEGNet, EEG-inception, and PPNN). Additionally, we employ statistical performance measures based on paired t-tests [52] to compare the significance of our proposed method with previously reported methods.

Fig. 4 depicts the single trial UAR ERP detection results from Dataset I, II, and III. It can be observed that, overall, our proposed MOCNN model outperforms the other comparison methods on all three datasets. For Dataset I, MOCNN obtains the highest average UAR across the 11 participants, with UAR improvements of 11.9%, 3.0%, 2.6%, 1.9%, and 6.2% respectively. Our proposed method has an extremely significant ($p < 0.001$) improvement compared with xDAWN, HDCA, EEGNet, PPNN, and a significant ($p < 0.01$)



(a) Dataset I (b) Dataset II (c) Dataset III

Fig. 4. The ERP detection results for the three datasets. Error bars represent standard deviations. The single asterisk * denotes p<0.05; ** denotes p<0.01; and *** denotes p<0.001.

improvement compared with EEG-inception. For Dataset II, our MOCNN model obtains the highest average UAR across the 8 participants, with UAR improvements of 6.5%, 8.3%, 5.1%, 2.4%, and 2.1% respectively. Significant differences exist between xDAWN, HDCA, EEGNet, PPNN, and MOCNN, and the difference in results obtained by MOCNN and EEG-inception are significant (p < 0.05). For Dataset III, our MOCNN model obtains the highest average UAR across the 8 participants, with UAR improvements of 13.2%, 7.0%, 8.4%, 3.9%, and 2.8% separately. There are extremely significant differences between MOCNN, xDAWN, HDCA, EEGNet, and EEG-inception, and significant differences between MOCNN and PPNN.

B. Performance Evaluation of Command Recognition

Given that the ultimate goal of BCI systems is to enable users to communicate, command recognition accuracy is an important criterion to evaluate the performance of our proposed method. Tables I, II, and III summarize the average command recognition accuracies for different methods as the number of repetitions increases for Dataset I, II, and III. The bold numbers indicate the maximum values of the corresponding columns in the tables. The number of repetitions k denotes that the classification probabilities of the EEG signals in the previous k rounds are used to obtain the recognition result by superposition averaging. It can be seen that with the increase in the number of repetition times, the command recognition accuracy achieved with each method increases. Our MOCNN model achieves higher command recognition accuracy than the other methods over all repetitions, especially when only 1 repetitions is used. For Dataset I, the command recognition accuracy of the MOCNN method exceeds the suboptimal result by more than 5% using the first repetition of data. For Dataset II and III, the command recognition accuracy achieved with the MOCNN model using the first repetition of data is more than 60%.

C. Computation Complexity Analysis

In this work, the concept of multiple-accumulated operations (MACs) is adopted to represent the computational

TABLE I
AVERAGE COMMAND RECOGNITION ACCURACY (%) OF DATASET I

Repetition Method	1	2	3	4	5	6	7	8	9	10
xDAWN	36.1	51.1	59.1	67.0	71.5	75.8	76.4	79.3	81.5	82.6
HDCA	39.4	62.0	72.9	80.5	85.3	87.7	89.4	91.2	92.5	92.7
EEGNet	37.0	60.0	70.7	79.4	84.5	87.8	89.0	90.3	92.0	92.4
EEG-inception	40.8	62.5	74.1	80.8	85.1	88.9	89.9	91.3	92.4	93.1
PPNN	34.5	52.9	63.0	69.1	74.6	78.3	80.5	81.9	84.1	85.5
MOCNN	**46.4**	**65.6**	**77.1**	**84.0**	**87.4**	**89.0**	**90.3**	**91.9**	**93.1**	**93.3**

TABLE II
AVERAGE COMMAND RECOGNITION ACCURACY (%) OF DATASET II

Repetition Method	1	2	3	4	5	6	7	8	9	10
xDAWN	54.5	67.2	76.0	83.0	87.8	90.5	90.3	93.3	95.0	95.0
HDCA	48.0	63.5	73.2	77.8	81.5	81.3	84.3	87.5	90.5	90.7
EEGNet	52.3	71.5	82.5	86.5	89.2	89.5	91.8	93.5	96.5	96.7
EEG-inception	56.8	74.8	83.5	87.3	89.7	90.5	92.8	93.5	97.0	96.5
PPNN	58.8	77.0	85.8	89.0	90.5	93.0	93.7	96.5	97.5	97.5
MOCNN	**62.0**	**79.0**	**87.5**	**91.0**	**92.5**	**94.3**	**95.5**	**97.0**	**98.5**	**98.0**

TABLE III
AVERAGE COMMAND RECOGNITION ACCURACY (%) OF DATASET III

Repetition / Method	1	2	3	4	5
xDAWN	39.4	58.9	71.9	73.9	81.3
HDCA	52.3	77.1	82.7	87.1	89.8
EEGNet	43.9	68.1	78.8	82.9	87.5
EEG-inception	58.3	79.8	85.4	88.3	92.1
PPNN	56.1	77.7	88.7	91.9	94.4
MOCNN	**64.0**	**84.8**	**90.6**	**93.1**	**95.8**

TABLE IV
COMPUTATION COMPLEXITY OF DEEP LEARNING MODELS FOR THE THREE DATASETS (G: GIGA-10^9)

Dataset	Model	MACs	Reduction Rate
Dataset I	EEGNet	2.19 G	16.4%
	EEG-inception	4.19 G	56.3%
	PPNN	3.97 G	53.9%
	MOCNN	1.83 G	-
Dataset II	EEGNet	1.12 G	-3.6%
	EEG-inception	2.23 G	48.0%
	PPNN	2.02 G	42.6%
	MOCNN	1.16 G	-
Dataset III	EEGNet	2.05 G	15.1%
	EEG-inception	3.94 G	55.8%
	PPNN	3.72 G	53.2%
	MOCNN	1.74 G	-

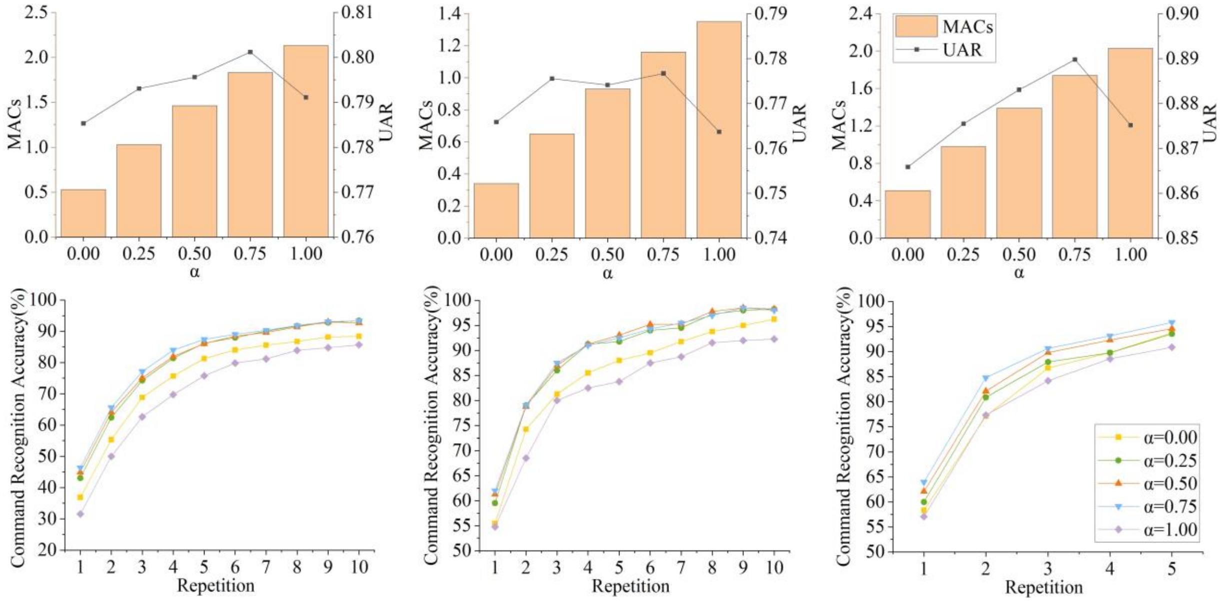
complexity [53] of deep learning models. One MAC consists of a multiplication operation and an addition operation. The dimensions of input data vary across our three datasets, resulting in a variation in the computational complexity of the models across the datasets. Table IV provides a comparison of the computational complexity between the three deep learning models and our proposed MOCNN model on the three datasets. The number of input samples during the computation is fixed, as is the mini-batch size. Comparing the MACs of the EEGNet, EEG-Inception, and PPNN models shows the reduction rates in MACs for MOCNN are 16.4%, 56.3%, 53.9% on Dataset I, -3.6%, 48.0%, 42.6% on Dataset II, and 15.1%, 55.8%, 53.2% on Dataset III. These results demonstrate that the computational complexity of our MOCNN model is lower compared to the EEG-inception and PPNN models across all datasets, and compared to EEGNet across Dataset I and III.

IV. DISCUSSION

A. The Influence of Hyperparameter α on MOCNN

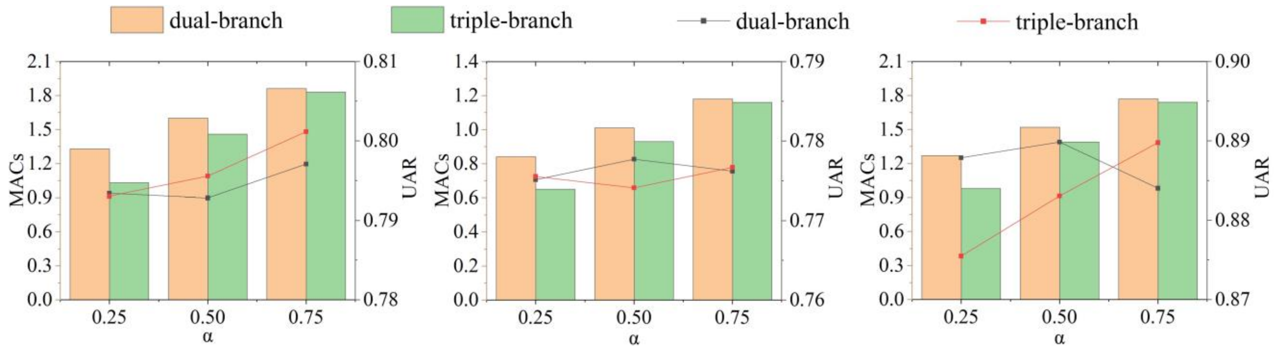
To explore the influence of different α values on the MOCNN model, we change the α value and conduct experiments on Dataset I, II, and III. Fig. 5 shows the classification performance and computational complexity of the MOCNN model under different α values ($\alpha \in \{0, 0.25, 0.50, 0.75, 1\}$). In particular, when α is equal to 0 or 1, MOCNN is a single-branch network using only low- or high-frequency information. The five comparison models are denoted as $T_\alpha \in \{T_0, T_{0.25}, T_{0.50}, T_{0.75}, T_1\}$, corresponding to the five α values.

Consistent with the evaluation criteria used in the previous section, we use MACs to assess the computational complexity of the model, while we employ UAR and command recognition accuracy to evaluate the classification performance. The results show that MACs of T_α gradually decrease with the decrease in α over the three datasets. Compared to the MACs of T_1 taken as the reference value, the MACs of $T_{0.75}$, $T_{0.50}$, $T_{0.25}$, and T_0 decreased by approximately



(a) Dataset I (b) Dataset II (c) Dataset III

Fig. 5. Classification performance and computational complexity of MOCNN under different α values on the three datasets. In the first row of subfigures, the axis on the left corresponds to the bar chart, representing the MACs of the models, and the axis on the right corresponds to the line chart, representing the average UAR of the models.



(a) Dataset I (b) Dataset II (c) Dataset III

Fig. 6. ERP detection performance and computational complexity of the MOCNN model under different numbers of branches. The axis on the left corresponds to the bar chart, representing the MACs of the models, and the axis on the right corresponds to the line chart, representing the UAR of the models.

TABLE V
AVERAGE COMMAND RECOGNITION ACCURACY (%) OF MOCNN UNDER DIFFERENT NUMBERS OF BRANCHES ON DATASET I

Repetition Model	1	2	3	4	5	6	7	8	9	10
$D_{0.25}$	45.3	**67.6**	**77.8**	83.0	86.6	88.5	89.9	91.5	**93.2**	93.1
$\mathrm{T}_{0.25}$	43.1	62.4	74.3	81.4	86.2	88.0	90.0	91.7	92.8	**93.4**
$D_{0.50}$	44.0	64.0	75.6	82.4	84.9	87.7	89.2	90.6	92.7	92.7
$\mathrm{T}_{0.50}$	44.9	64.1	75.1	81.9	86.1	88.4	89.6	91.4	93.0	92.7
$D_{0.75}$	44.5	65.3	76.3	81.9	85.6	87.7	89.4	90.7	92.4	92.3
$\mathrm{T}_{0.75}$	**46.4**	65.6	77.1	**84.0**	**87.4**	**89.0**	**90.3**	**91.9**	93.1	93.3

TABLE VI
AVERAGE COMMAND RECOGNITION ACCURACY (%) OF MOCNN UNDER DIFFERENT NUMBERS OF BRANCHES ON DATASET II

Repetition Model	1	2	3	4	5	6	7	8	9	10
$D_{0.25}$	59.5	**80.5**	85.8	91.2	92.5	95.0	**96.0**	97.2	**99.0**	98.5
$\mathrm{T}_{0.25}$	59.5	79.0	86.0	91.2	91.8	94.0	94.5	97.3	98.0	98.3
$D_{0.50}$	**62.3**	78.7	86.7	90.8	92.5	94.2	95.5	**98.0**	98.3	98.3
$\mathrm{T}_{0.50}$	61.3	78.8	87.0	91.2	**93.0**	**95.2**	95.3	97.8	98.5	98.3
$D_{0.75}$	58.0	79.0	84.5	**93.3**	91.7	94.2	95.5	**98.0**	98.8	**98.8**
$\mathrm{T}_{0.75}$	62.0	79.0	**87.5**	91.0	92.5	94.3	95.5	97.0	98.5	98.0

14.1%, 31.5%, 51.6%, and 75.1%, respectively.

As α increases, the UAR trend on the three datasets is similar, with UAR first increasing to a vertex and then decreasing. The optimal point, where UAR is maximized, is found at $\alpha = 0.75$. In terms of command recognition accuracy, $T_{0.75}$ consistently achieves the best results when the number of repetitions is 1, 2, and 3 on all datasets. On Dataset I, $T_{0.75}$ obtains optimal command recognition accuracies in the first 9 repetitions. In the last iteration, the command recognition accuracy of $T_{0.75}$ is slightly inferior to that of $T_{0.25}$. On Dataset II, the gap between the command recognition accuracies of $T_{0.50}$ and $T_{0.75}$ is very small. $T_{0.50}$ is the more advantageous one in the last 7 repetitions, but $T_{0.75}$ also allows us to obtain acceptable results. On Dataset III, the command recognition accuracy of $T_{0.75}$ has an advantage in all repetitions.

It can be seen that when $\alpha = 0.75$, the MOCNN model performs better overall. At this point, 12/16 of the filters in the MOCNN model always learn the high-frequency component, 3/16 of the filters learn the middle-frequency component, and 1/16 of the filters learn the low-frequency component. The feature information of each component is fully learned and complementary when fused. We set α to 0.75 in the previous section based on the above findings.

TABLE VII
AVERAGE COMMAND RECOGNITION ACCURACY (%) OF MOCNN UNDER DIFFERENT NUMBERS OF BRANCHES ON DATASET III

Repetition Model	1	2	3	4	5
$D_{0.25}$	64.0	84.2	89.4	91.9	94.8
$\mathrm{T}_{0.25}$	60.0	80.8	87.9	89.8	93.5
$D_{0.50}$	**64.8**	**85.4**	**91.7**	**93.6**	**96.0**
$\mathrm{T}_{0.50}$	62.1	82.1	89.8	92.3	94.6
$D_{0.75}$	63.7	83.1	89.6	92.1	95.2
$\mathrm{T}_{0.75}$	64.0	84.8	90.6	93.1	95.8

B. The Influence of the Number of Branches

To investigate the impact of the number of branches on our MOCNN model, we change the number of branches and conduct experiments on Dataset I, II, and III. As mentioned earlier the MOCNN model is a single-branch network when α is equal to 0 or 1. As shown in Fig. 5, the triple-branch MOCNN model significantly outperforms the single-branch MOCNN model.

TABLE VIII
ABLATION STUDIES OF MOCNN ON DATASET I, II AND III

Dataset	Model	UAR	Command Recognition Accuracy of Each Repetition									
			1	2	3	4	5	6	7	8	9	10
I	MOCNN-1	0.795	44.8	63.3	74.1	81.4	84.8	87.2	88.1	90.3	91.6	91.9
	MOCNN-2	0.798	45.1	65.5	**77.4**	83.0	87.0	88.2	89.3	90.8	92.5	92.7
	MOCNN	**0.801**	**46.4**	**65.6**	77.1	**84.0**	**87.4**	**89.0**	**90.3**	**91.9**	**93.1**	**93.3**
II	MOCNN-1	**0.777**	57.2	**79.5**	85.5	90.2	91.7	**94.3**	94.3	96.7	97.8	97.8
	MOCNN-2	0.771	**62.5**	78.2	**87.5**	90.8	91.0	91.5	95.0	96.7	97.5	97.5
	MOCNN	**0.777**	62.0	79.0	**87.5**	**91.0**	**92.5**	**94.3**	**95.5**	**97.0**	**98.5**	**98.0**
III	MOCNN-1	0.880	63.8	82.9	90.4	92.5	94.4	-	-	-	-	-
	MOCNN-2	**0.891**	63.9	**85.0**	**91.5**	**93.1**	95.0	-	-	-	-	-
	MOCNN	0.890	**64.0**	84.8	90.6	**93.1**	**95.8**	-	-	-	-	-

We then compare the performance between the dual-branch and triple-branch MOCNN models. The dual-branch MOCNN is implemented by removing the branches that learn the low-frequency components in the triple-branch MOCNN model, in which case the proportion of filters assigned to the high-frequency component is α and the proportion of filters assigned to the medium-frequency component is 1-α. The dual-branch and triple-branch models are further divided into three cases according to the different values of $\alpha \in \{0.25,\ 0.50,\ 0.75\}$, separately. The six comparison models are referred to as $D_\alpha \in \{D_{0.25},\ D_{0.50},\ D_{0.75}\}$ and $T_\alpha \in \{T_{0.25},\ T_{0.50},\ T_{0.75}\}$. Fig. 6 illustrates the ERP detection performance and computational complexity of the MOCNN model under different numbers of branches. It can be observed that the MACs of T_α is always less than D_α, and the closer α is to 0, the less T_α is then D_α. From the trend of the variation of UAR of the two structures with α, it is difficult to summarize a common law between them. However, it can be determined that the UAR of T_α is always optimal at $\alpha = 0.75$. Moreover, the UAR of $D_{0.50}$ is superior to $T_{0.75}$ on Dataset II and III.

In Tables V-VII, the average command recognition accuracy of the MOCNN model under different numbers of branches on the three datasets is shown. On Dataset I, the command recognition accuracy of $T_{0.75}$ is optimal in most repetitions. On Dataset II, the advantages of each model are not obvious, and the relatively reliable models are $D_{0.25}$ and $D_{0.75}$. On Dataset III, $D_{0.50}$ achieved optimality in all repetitions.

Overall, the performances of the dual-branch MOCNN and the triple-branch MOCNN models differ over the different datasets. $T_{0.75}$ and $D_{0.50}$ are the two models with the best comprehensive performance. As shown in Table IV, the computational complexity of $T_{0.75}$ is already relatively small compared to other deep learning models. Therefore, only the UAR and the command recognition accuracy are considered when further comparing $T_{0.75}$ and $D_{0.50}$. Considering that the average UAR of $T_{0.75}$ on the three datasets can always rank in the top two among the six models and the average command recognition of $T_{0.75}$ is generally superior to $D_{0.50}$ in most repetitions of Dataset I and II, we can conclude that model $T_{0.75}$ is the most suitable decoding model for our datasets.

C. Ablation Studies

Two additional ablation experiments are conducted to analyze the importance of optimizing the distribution of branch width and the information interaction between branches. The results of the ablation studies on Datasets I, II, and III are shown in Table VIII. To compare the effect of branch width optimization on the performance of MOCNN, we constructed MOCNN-1 by evenly distributing branch widths under the condition of keeping the network width roughly equal to MOCNN. To compare the effect of inter-branch information exchange on the performance of our MOCNN model, we construct a new model, MOCNN-2, by removing inter-frequency updates in the module CoreOctConv. Table VIII shows the average UAR and the average command recognition accuracy of each repetition of Dataset I, II, and III. The bold numbers indicate the maximum values of the corresponding columns of each dataset in the table. It can be seen that MOCNN is always significantly better than MOCNN-1 and is significantly better than MOCNN-2 on datasets I and II. In the first 4 repetitions on dataset III, the results of MOCNN and MOCNN-2 are close, but in the fifth repetition, MOCNN shows a significant improvement over MOCNN-2. The ablation studies indicate the necessity of optimizing the distribution of branch width and the information interaction between branches.

V. CONCLUSION

This study proposes a novel deep convolution neural network called Multi-scale Feature Fusion Octave Convolution Neural Network (MOCNN) for EEG classification in ERP-BCIs. MOCNN can effectively process the high-, medium-, and low-frequency components of the ERP signals separately, thus extracting more complementary feature information. Experimental evaluations conducted on two publicly available datasets and a self-collected ERP dataset demonstrate the exceptional performance of MOCNN, significantly outperforming previous xDAWN, HDCA, EEGNet, EEG-inception, and PPNN models. Moreover, we also discuss the important parameter and structure choices for MOCNN. The results show that setting $\alpha = 0.75$ and utilizing three branches enable the complete learning of essential components, leading to improved use of complementary features. Meanwhile, the ablation studies also indicate the suitability of our MOCNN design. This study introduces generalized octave convolutions in ERP classification tasks with satisfactory results.

REFERENCES

[1] J. R. Wolpaw, N. Birbaumer, W. J. Heetderks, D. J. Mcfarland, and P. H. Peckham, "Brain-computer interface technology: a review of the first international meeting," *IEEE Transactions on Rehabilitation Engineering,* vol. 8, no. 2, pp. 164-173, 2000.

[2] J. -H. Jeong, J. -H. Cho, B. -H. Lee and S. -W. Lee, "Real-time deep neurolinguistic learning enhances noninvasive neural language decoding for brain–machine interaction," *IEEE Transactions on Cybernetics,* vol. 53, no. 12, pp. 7469-7482, 2023.

[3] Á. Fernández-Rodríguez, A. Darves-Bornoz, F. Velasco-Álvarez, and R. Ron-Angevin, "Effect of stimulus size in a visual ERP-based BCI under RSVP," *Sensors,* vol. 22, no. 23, p. 9505, 2022.

[4] J. Jin, Z. Chen, R. Xu, Y. Miao, X. Wang, and T.-P. Jung, "Developing a novel tactile P300 brain-computer interface with a cheeks-stim paradigm," *IEEE Transactions on Biomedical Engineering,* vol. 67, no. 9, pp. 2585-2593, 2020.

[5] J. Jin, Z. Wang, R. Xu, C. Liu, X. Wang, and A. Cichocki, "Robust similarity measurement based on a novel time filter for SSVEPs detection," *IEEE Transactions on Neural Networks and Learning Systems,* vol. 34, no. 8, pp. 4096-4105, 2023.

[6] B. Xiong, B. Wan, J. Huang, F. Li, X. Li and P. Yang, "Cross-Stimulus Transfer Method Using Common Impulse Response for Fast Calibration of SSVEP-Based BCIs," *IEEE Transactions on Instrumentation and Measurement*, vol. 73, pp. 1-14, 2024.

[7] M. M. Makary, H. M. Bu-Omer, R. S. Soliman, K. Park, and Y. M. Kadah, "Spectral subtraction denoising preprocessing block to improve slow cortical potential based brain–computer interface," *Journal of Medical and Biological Engineering,* vol. 38, pp. 87-98, 2018.

[8] J. Jin, R. Xiao, I. Daly, Y. Miao, X. Wang, and A. Cichocki, "Internal feature selection method of CSP based on L1-norm and Dempster–Shafer theory," *IEEE Transactions on Neural Networks and Learning Systems,* vol. 32, no. 11, pp. 4814-4825, 2021.

[9] J. Fumanal-Idocin, Y.-K. Wang, C.-T. Lin, J. Fernández, J. A. Sanz, and H. Bustince, "Motor-imagery-based brain–computer interface using signal derivation and aggregation functions," *IEEE Transactions on Cybernetics,* vol. 52, no. 8, pp. 7944-7955, 2022.

[10] Q. Zheng, Y. Wang, and P. A. Heng, "Multitask feature learning meets robust tensor decomposition for EEG classification," *IEEE Transactions on Cybernetics,* vol. 51, no. 4, pp. 2242-2252, 2021.

[11] J.-H. Cho, J.-H. Jeong, and S.-W. Lee, "Neurograsp: Real-time eeg classification of high-level motor imagery tasks using a dual-stage deep learning framework," *IEEE Transactions on Cybernetics,* vol. 52, no. 12, pp. 13279-13292, 2022.

[12] F. Qi, W. Wu, Z. Yu, Z. Gu, Z. Wen, T. Yu, and Y. Li, "Spatiotemporal-filtering-based channel selection for single-trial EEG classification," *IEEE Transactions on Cybernetics,* vol. 51, no. 2, pp. 558-567, 2021.

[13] V. De Pascalis, "On the psychophysiology of extraversion," *On the Psychobiology of Personality,* vol. 46, no. 5, pp. 295-327, 2004.

[14] R. Abiri, S. Borhani, E. W. Sellers, Y. Jiang, and X. Zhao, "A comprehensive review of EEG-based brain–computer interface paradigms," *Journal of Neural Engineering,* vol. 16, no. 1, p. 011001, 2019.

[15] O. E. Korkmaz, O. Aydemir, E. A. Oral, and I. Y. Ozbek, "A novel probabilistic and 3D column P300 stimulus presentation paradigm for EEG-based spelling systems," *Neural Computing and Applications,* vol. 35, no. 16, pp.11901-11915, 2023.

[16] X. Zhang, J. Jin, S. Li, X. Wang, and A. Cichocki, "Evaluation of color modulation in visual P300-speller using new stimulus patterns," *Cognitive Neurodynamics,* vol. 15, pp. 873-886, 2021.

[17] M. Kim, M.-K. Kim, M. Hwang, H.-Y. Kim, J. Cho, and S.-P. Kim, "Online home appliance control using EEG-Based brain–computer interfaces," *Electronics,* vol. 8, no. 10, p. 1101, 2019.

[18] T. Kaufmann, A. Herweg, and A. Kübler, "Toward brain-computer interface based wheelchair control utilizing tactually-evoked event-related potentials," *Journal of Neuroengineering and Rehabilitation,* vol. 11, p. 7, 2014.

[19] D. Zhang, L. Yao, K. Chen, S. Wang, X. Chang, and Y. Liu, "Making sense of spatio-temporal preserving representations for EEG-based human intention recognition," *IEEE Transactions on Cybernetics,* vol. 50, no. 7, pp. 3033-3044, 2019.

[20] E. Santamaria-Vazquez, V. Martinez-Cagigal, F. Vaquerizo-Villar, and R. Hornero, "EEG-inception: A novel deep convolutional neural network for assistive ERP-based brain-computer interfaces," *IEEE Transactions on Neural Systems and Rehabilitation Engineering,* vol. 28, no. 12, pp. 2773-2782, 2020.

[21] B. Wang, C. M. Wong, Z. Kang, F. Liu, C. Shui, F. Wan, and C. L. P. Chen, "Common spatial pattern reformulated for regularizations in brain–computer interfaces," *IEEE Transactions on Cybernetics,* vol. 51, no. 10, pp. 5008-5020, 2021.

[22] J. Li, S. Qiu, Y.-Y. Shen, C.-L. Liu, and H. He, "Multisource transfer learning for cross-subject EEG emotion recognition," *IEEE Transactions on Cybernetics,* vol. 50, no. 7, pp. 3281-3293, 2019.

[23] C. Dai, J. Wu, D. Pi, S. I. Becker, L. Cui, Q. Zhang, and B. Johnson, "Brain EEG time-series clustering using maximum-weight clique," *IEEE Transactions on Cybernetics,* vol. 52, no. 1, pp. 357-371, 2022.

[24] B. Rivet, A. Souloumiac, V. Attina, and G. Gibert, "xDAWN algorithm to enhance evoked potentials: application to brain–computer interface," *IEEE Transactions on Biomedical Engineering,* vol. 56, no. 8, pp. 2035-2043, 2009.

[25] B. Blankertz, S. Lemm, M. Treder, S. Haufe, and K.-R. Müller, "Single-trial analysis and classification of ERP components—a tutorial," *NeuroImage,* vol. 56, no. 2, pp. 814-825, 2011.

[26] P. Sajda, E. Pohlmeyer, J. Wang, L. C. Parra, C. Christoforou, J. Dmochowski, B. Hanna, C. Bahlmann, M. K. Singh, and S.-F. Chang, "In a blink of an eye and a switch of a transistor: cortically coupled computer vision," *Proceedings of the IEEE,* vol. 98, no. 3, pp. 462-478, 2010.

[27] A. Mobaien, R. Boostani, and S. Sanei, "Improving the performance of P300-based BCIs by mitigating the effects of stimuli-related evoked potentials through regularized spatial filtering," *Journal of Neural Engineering,* vol. 21, no. 1, p. 016023, 2024.

[28] B. Zang, Y. Lin, Z. Liu, and X. Gao, "A deep learning method for single-trial EEG classification in RSVP task based on spatiotemporal features of ERPs," *Journal of Neural Engineering,* vol. 18, no. 4, p. 0460c8, 2021.

[29] S. Pancholi, A. Giri, A. Jain, L. Kumar, and S. Roy, "Source aware deep learning framework for hand kinematic reconstruction using EEG signal," *IEEE Transactions on Cybernetics,* vol. 53, no. 7, pp. 4094-4106, 2023.

[30] R. T. Schirrmeister, J. T. Springenberg, L. D. J. Fiederer, M. Classtetter, 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, 2017.

[31] V. J. Lawhern, A. J. Solon, N. R. Waytowich, S. M. Gordon, C. P. Hung, and B. J. Lance, "EEGNet: a compact convolutional neural network for EEG-based brain–computer interfaces," *Journal of Neural Engineering,* vol. 15, no. 5, p. 056013, 2018.

[32] F. Li, C. Wang, Y. Li, H. Wu, B. Fu, Y. Ji, Y. Niu, and G. Shi, "Phase preservation neural network for electroencephalography classification in rapid serial visual presentation task," *IEEE Transactions on Biomedical Engineering,* vol. 69, no. 6, pp. 1931-1942, 2022.

[33] H. Li, J. Tang, W. Li, W. Dai, Y. Liu and Z. Zhou, "Multi-Task Collaborative Network: Bridge the Supervised and Self-Supervised Learning for EEG Classification in RSVP Tasks," *IEEE Transactions on Neural Systems and Rehabilitation Engineering,* vol. 32, pp. 638-651, 2024.

[34] C. Yi, R. Yao, L. Song, L. Jiang, Y. Si, P. Li, F. Li, D. Yao, Y. Zhang, and P. Xu, "A novel method for constructing EEG large-scale cortical dynamical functional network connectivity (dFNC): WTCS," *IEEE Transactions on Cybernetics,* vol. 52, no. 12, pp. 12869-12881, 2022.

[35] F. Li, H. Li, Y. Li, H. Wu, B. Fu, Y. Ji, C. Wang, and G. Shi, "Decoupling representation learning for imbalanced electroencephalography classification in rapid serial visual presentation task," *Journal of Neural Engineering,* vol. 19, no. 3, p. 036011, 2022.

[36] D. Yipeng and L. Jian, "IENet: a robust convolutional neural network for EEG based brain-computer interfaces," *Journal of Neural Engineering,* vol. 19, no. 3, p. 036031, 2022.

[37] H. Wang, Z. Pei, L. Xu, T. Xu, and J. Li, "Performance enhancement of P300 detection by multi-scale-CNN," *IEEE Transactions on Instrumentation and Measurement,* vol. 70, no. 1, pp. 1-12, 2021.

[38] D. Borra, S. Fantozzi, and E. Magosso, "A lightweight multi-scale convolutional neural network for P300 decoding: analysis of training strategies and uncovering of network decision," *Frontiers in Human Neuroscience*, vol. 15, p. 655840, 2021.

[39] Y. Chen, H. Fan, B. Xu, Z. Yan, Y. Kalantidis, M. Rohrbach, Y. Shuicheng, and J. Feng, "Drop an octave: Reducing spatial redundancy in convolutional neural networks with octave convolution," *2019 IEEE/CVF International Conference on Computer Vision (ICCV),* 2019, pp. 3435-3444.

[40] L. Acqualagna and B. Blankertz, "Gaze-independent BCI-spelling using rapid serial visual presentation (RSVP)," *Clinical Neurophysiology,* vol. 124, no. 5, pp. 901-908, 2013.

[41] U. Hoffmann, J.-M. Vesin, T. Ebrahimi, and K. Diserens, "An efficient P300-based brain–computer interface for disabled subjects," *Journal of Neuroscience Methods,* vol. 167, no. 1, pp. 115-125, 2008.

[42] J. Jin, P. Horki, C. Brunner, X. Wang, C. Neuper, and G. Pfurtscheller, "A new P300 stimulus presentation pattern for EEG-based spelling systems," *Biomedizinische Technik/Biomedical Engineering,* vol. 55, no. 4, pp. 203-210, 2010.

[43] J. Jin, B. Z. Allison, E. W. Sellers, C. Brunner, P. Horki, X. Wang, and C. Neuper, "Optimized stimulus presentation patterns for an event-related potential EEG-based brain-computer interface," *Medical & Biological Engineering & Computing,* vol. 49, pp. 181-191, 2011.

[44] J. Jin, B. Z. Allison, T. Kaufmann, A. Kübler, Y. Zhang, X. Wang, and A. Cichocki, "The changing face of P300 BCIs: a comparison of stimulus changes in a P300 BCI involving faces, emotion, and movement," *Plos One,* vol. 7, no. 11, p. e49688, 2012.

[45] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR),* 2017, pp. 1251-1258.

[46] X. Lei, P. Yang, and D. Yao, "An empirical Bayesian framework for brain–computer interfaces," *IEEE Transactions on Neural Systems and Rehabilitation Engineering,* vol. 17, no. 6, pp. 521-529, 2009.

[47] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B.Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," *Advances in Neural Information Processing Systems,* vol. 32, 2019.

[48] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *3rd International Conference on Learning Representations (ICLR),* 2015, pp. 1-15.

[49] L. Prechelt, "Early stopping-but when?," *Neural Networks: Tricks of the Trade*: Springer, 2002, pp. 55-69.

[50] J. Tompson, R. Goroshin, A. Jain, Y. Lecun, and C. Bregler, "Efficient object localization using convolutional networks," *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR),* 2015, pp. 648-656.

[51] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research,* vol. 15, no. 1, pp. 1929-1958, 2014.

[52] S. Li, J. Jin, I. Daly, C. Liu, and A. Cichocki, "Feature selection method based on Menger curvature and LDA theory for a P300 brain–computer interface," *Journal of Neural Engineering,* vol. 18, no. 6, p. 066050, 2022.

[53] L. Ang, W. Zhenyu, Z. Xi, X. Tianheng, Z. Ting, and H. Honglin, "MDTL: a novel and model-agnostic transfer learning strategy for cross-subject motor imagery BCI," *IEEE Transactions on Neural Systems and Rehabilitation Engineering,* vol. 31, pp. 1743-1753, 2023.

Jing Jin (Senior Member, IEEE) received the Ph.D. degree in control theory and control engineering from the East China University of Science and Technology, Shanghai, China, in 2010. His Ph.D. advisors were Prof. Gert Pfurtscheller at Graz University of Technology from 2008 to 2010 and Prof. Xingyu Wang at East China University of Science and Technology from 2006 to 2008.

He is currently a Professor at East China University of Science and Technology (ECUST) and Associate Dean of School of Mathematics at ECUST.

Prof. Jin currently serves as Associate Editor of Journal of Neuroscience Methods and Frontiers in Neurorobotics, Action Editor of Neural Networks, Editor of Journal of Neural Engineering. His research interests include brain-computer interface, signal processing and pattern recognition.

Ruitian Xu received the B.E. degree in automation from the East China University of Science and Technology (ECUST), Shanghai, China, in 2021.

She is currently pursuing her doctoral degree from East China University of Science and Technology, Shanghai, China. Her research interest includes brain-computer interface, machine learning, and signal processing.

Ian Daly received the M.Eng. degree in computer science in 2006 and the Ph.D. degree in cybernetics in 2011 from the University of Reading, Reading, U.K. Between May 2011 and 2013 he was a Postdoctoral Researcher in the Laboratory of Brain-Computer Interfaces, Graz University of Technology, Graz, Austria.

He is currently a Senior Lecturer, School of Computer Science and Electronic Engineering (CSEE), University of Essex, U.K. His research interests focus on BCIs, nonlinear dynamics, machine learning, signal processing, and connectivity analysis in the EEG and fMRI (functional magnetic resonance imaging). He is also interested in the neurophysiological correlates of motor control and stimuli perception and how they differ between healthy participants and individuals with neurological and physiological impairments.

Xueqing Zhao received the B.E. degree in electrical engineering and automation from the East China University of Science and Technology (ECUST), Shanghai, China, in 2019.

She is currently pursuing her doctoral degree at the ECUST, Shanghai. Her research interests include brain-computer interface, machine learning, and signal processing.

Xingyu Wang was born in Sichuan, China, in 1944. He received the B.S. degree in mathematics from Fudan University, Shanghai, China, in 1967, and the M.S. in control theory from East China Normal University, Shanghai, China, in 1982, and Ph.D. degrees in industrial automation from East China University of Science and Technology, Shanghai, China, in 1984.

He is currently a Professor at the School of Information Science and Engineering, East China University of Science and Technology, Shanghai, China. His research interests include control theory, control techniques, the application to biomedical system, and brain control.

Andrzej Cichocki (Fellow, IEEE) received the M.Sc. (Hons.), Ph.D., and Dr.Sc. (Habilitation) degrees in electrical engineering from the Warsaw University of Technology, Warszawa, Poland, in 1972, 1975, and 1982, respectively.

He spent several years at the University Erlangen, Erlangen, Germany, as an Alexander-von-Humboldt Research Fellow and a Guest Professor. From 1995 to 2017, he was a Senior Team Leader and the Head of the Laboratory for Advanced Brain Signal Processing, RIKEN Brain Science Institute, Wako, Japan. He has authored more than 500 technical journal articles and five monographs in English (two of them translated to Chinese). Currently, his research focus on multiway blind source separation, tensor decomposition, tensor networks for big data mining, and brain-computer interface. His joint publications currently report over 44,000 citations according to Google scholar, with an h-index of 97.

Dr. Cichocki served as Associate Editor of, IEEE Trans. on Signals Processing, IEEE Trans. on Neural Networks and Learning Systems, IEEE Trans on Cybernetics, Journal of Neuroscience Methods and he is a founding Editor in Chief for Journal Computational Intelligence and Neuroscience.