# LSTM-Based EEG Classification in Motor Imagery Tasks

Ping Wang, Aimin Jiang, *Member, IEEE*, Xiaofeng Liu, Jing Shang, and Li Zhang

***Abstract*— Classification of motor imagery electroencephalograph signals is a fundamental problem in brain–computer interface (BCI) systems. We propose in this paper a classification framework based on long short-term memory (LSTM) networks. To achieve robust classification, a one dimension-aggregate approximation (1d-AX) is employed to extract effective signal representation for LSTM networks. Inspired by classical common spatial pattern, channel weighting technique is further deployed to enhance the effectiveness of the proposed classification framework. Public BCI competition data are used for the evaluation of the proposed feature extraction and classification network, whose performance is also compared with that of the state-of-the-arts approaches based on other deep networks.**

***Index Terms*— Electroencephalograph (EEG), long short-term memory (LSTM), motor imagery, one dimension-aggregate approximation (1d-AX).**

## I. Introduction

CLASSIFICATION of electroencephalography (EEG) data in motor imagery tasks of brain-computer interface (BCI) combines many research fields, including signal processing, pattern recognition, machine learning and so on. Most BCI research aims at helping people who are severely paralyzed regain control over their environment and communicate with their social environment [1]. Specifically, BCI systems enable a subject to send commands to an electronic device simply through brain activity. In motor imagery tasks of a BCI system, EEG data are collected by electrodes attached on scalp. Previous research [2] on neurophysiology has demonstrated that brain activities in response to motor imagery generally reside within three frequency bands, including alpha (8-13 Hz), beta (14-30 Hz) and gamma (30-35 Hz). When a subject imagines different kinds of movements, the characteristics of EEG are generally different. However, it is not easy to infer the category of actions which the subject is imagining simply by raw EEG signals, since they are vulnerable to various noise and interferences. Therefore, effective preprocessing and feature extraction are extremely necessary.

EEG features widely used in practice can be roughly categorized into three groups: 1) discriminative features manually selected in spatial, time and frequency domain, e.g., common spatial patterns (CSP) [3] and its variants, such as common sparse spectral spatial patterns (CSSSP) [4]; 2) statistical features, e.g., mean channel energy (MCE) [5], and entropy [6]; 3) data-driven adaptive features extracted by restricted Boltzmann machine (RBM) [7], neural network (NN) [8] and etc. It is worth mentioning that NN is an efficient framework, which can achieve feature extraction and classification simultaneously.

Currently, five kinds of classifiers are widely used in various motor imagery tasks: 1) linear classifiers, e.g., linear discriminant analysis (LDA) and support vector machine (SVM); 2) nonlinear Bayesian classifiers, e.g., Bayes quadratic and hidden Markov model (HMM); 3) nearest neighbor classifiers, e.g., $k$ nearest neighbors (KNN); 4) neural networks, e.g., multi-layer perceptron (MLP) [9] and radial basis function (RBF) neural network [10]; 5) combinations of different classification methods by boosting or voting.

Recent advances of machine learning have witnessed the success and popularity of deep learning techniques, among which artificial neural network (ANN) is one of the most important parts. Convolutional neural networks (CNN) and recurrent neural networks (RNN) are two popular forms of ANN, and find a variety of applications in practice. For instance, CNN draws extensive attention in image classification, and various CNN frameworks have been proposed, e.g., VGG-16 [11] and Residual-Net [12]. RNN is a class of ANN whose connections between computing units form a directed graph along a sequence and thus popular in time-series data analysis, such as speech recognition, machine translation, time series prediction and so on. Hidas *et al.* [13] proposed a video recommendation method based on users' short-term behavior data. LipNet proposed by Assael *et al.* [14] can achieve an accuracy of 95.2% on sentence-level of GRID corpus [15].

Although CNN and RNN have achieved great success, they are not so popular in the EEG classification of motor

Manuscript received July 14, 2018; revised September 13, 2018 and October 10, 2018; accepted October 11, 2018. Date of publication October 18, 2018; date of current version November 20, 2018. This work was supported in part by the National Natural Science Foundation of China under Grant 61471157, Grant 61772090, and Grant 61801055, in part by the Fundamental Research Funds for the Central Universities under Grant 26120182018B23014 and Grant 2018B47114, in part by the Key Development Program of Jiangsu Province of China under Grant BE2017071, Grant BE2017647, and Grant BE2018004-04, and in part by the Projects of International Cooperation and Exchanges of Changzhou of China under Grant CZ20170018. *(Corresponding author: Aimin Jiang.)*

P. Wang, A. Jiang, and J. Shang are with the College of Internet of Things Engineering, Hohai University, Changzhou Campus, Changzhou 213022, China (e-mail: tonywang@hhu.edu.cn; jiangam@hhuc.edu.cn; shangj@hhu.edu.cn).

X. Liu is with the Jiangsu Key Laboratory of Special Robots, Hohai University, Changzhou Campus, Changzhou 213022, China (e-mail: liuxf@hhuc.edu.cn).

L. Zhang is with the Department of Mathematics and Physics, Hohai University, Changzhou Campus, Changzhou 213022, China (e-mail: zhangl@hhuc.edu.cn).

Digital Object Identifier 10.1109/TNSRE.2018.2876129

imagery tasks as in the fields aforementioned. For CNN, one major reason is that, so far, it is difficult to find an effective conversion approach to express EEG signals in a form carrying all the information suitable to CNN. For RNN, training a network by gradient descent algorithm [16] is still a challenge. Another difficulty for RNN is that it is insensitive to inputs fed to a network earlier. To address the long-term dependence issue, researchers have also paid great efforts to develop long short-term memory (LSTM) networks. LSTM, one kind of RNNs, was first proposed by Hochreiter and Schmidhuber [17] and improved by Graves [18]. Nowadays, there are a number of variants of LSTM [e.g., Gated Recurrent Unit (GRU) [19], peephole connection LSTM [20], developed from different perspectives.

Although ANN is capable of simultaneous feature extraction and classification, it is still difficult to apply ANN directly on raw EEG data, since they are highly vulnerable to measurement noise and interferences. Appropriate preprocessing and feature extraction techniques are extremely important to the success of ANN in motor imagery tasks. Many EEG feature extraction methods based on neural network have been proposed recently. For example, Lu *et al.* [21] apply Deep Belief Networks (DBN) to generates auto-encoded features. Ren and Wu [22] utilize Convolutional Deep Belief Networks (CDBN) to extract meaningful features for classification. However, so far the LSTM has not drawn much attention in the research of EEG-based motor imagery tasks. In this paper, we shall develop an LSTM-based framework to extract essential features of time-varying EEG signals. To enhance the generalization performance of the proposed LSTM-based framework and reduce the possibility of over-fitting, 1d-AX along with channel weighting technique is also adopted in this paper to make EEG signal representation more concise, which ease the subsequent training of LSTM networks. Channel weighting coefficients can be automatically optimized with other network parameters of LSTM networks.

The rest of the paper is organized as follows. Section II describes the related work. Section III shows the overall architecture and details of the proposed classification strategy. Section IV presents experiment results for the evaluation of the proposed classification framework. Section V concludes the paper.

## II. Related Work

Considering the existing EEG feature extraction methods discussed in previous section, in this section we focus on the related work regarding CSP and NN.

CSP is one of the most classical methods to extract hand-made features of multi-channel EEG data. It aims to find a spatial filter $\mathbf{w}$ which maximizes the variance of spatially filtered data in one class while minimizing the variance of filtered data in another class [4], [23]. Let $\hat{\mathbf{X}} = [\hat{\mathbf{x}}_1 \ \hat{\mathbf{x}}_2 \ \ldots \ \hat{\mathbf{x}}_N] \in \mathbb{R}^{c\times N}$ represent EEG signals, where $c$ denotes the number of channels and $N$ the number of samples. By defining the time average of EEG signals

$$\mu = \frac{1}{N}\sum_{n=1}^{N}\hat{\mathbf{x}}_n, \tag{1}$$

The variance of signals filtered by $\mathbf{w} \in \mathbb{R}^c$ is computed by

$$\sigma^2(\hat{\mathbf{X}}, \mathbf{w}) = \frac{1}{N}\sum_{n=1}^{N}\left|\mathbf{w}^T\left(\hat{\mathbf{x}}_n - \mu\right)\right|^2. \tag{2}$$

For binary classification problem, the classical CSP problem is formulated as

$$\underset{\mathbf{w}}{\text{minimize}} \ E_{\hat{\mathbf{X}}\in C_i}\left[\sigma^2(\hat{\mathbf{X}}, \mathbf{w})\right] \tag{3a}$$

$$\text{subject to} \ \sum_{j=1,2} E_{\hat{\mathbf{X}}\in C_j}\left[\sigma^2(\hat{\mathbf{X}}, \mathbf{w})\right] = 1 \tag{3b}$$

where $C_i$ $(i = 1, 2)$ represents the dataset of class $i$ and $E_{\hat{\mathbf{X}}\in C_i}[\cdot]$ computes the expectation over $C_i$. In practice, the generalized eigenvalue decomposition can be used to achieve optimal solutions to (3), and discriminative CSP features are then fed to classifiers (e.g., LDA and SVM). Various CSP-based approaches have been developed to enhance the classification performance. For example, temporal filters can be jointly optimized with spatial filters [4], such that appropriate spectral channels can be appropriately chosen. Sparsity is also introduced to reduce inter-channel interference by removing irrelative or highly noisy recording channels [24].

With the advances of deep learning techniques, various CNN and RNN classification frameworks have also been proposed recently to deal with EEG classification in motor imagery tasks. In [5], Uktveris and Jusas evaluated different feature extraction methods [e.g., principal component analysis (PCA), fast Fourier transform energy map (FFTEM), and discrete cosine transform (DCT)], and found that the FFTEM behaves best in four-class classification of motor imagery tasks. FFTEM is obtained by first applying fast Fourier transform (FFT) on EEG signals of each channel and stacking spectral coefficients of all the channels so as to achieve a two-dimensional representation, which is further fed to a CNN. Its overall framework is depicted in Fig. 1. Obviously, this representation focuses on the spectral and spatial characteristics of EEG signals. To achieve essential information of EEG signals in time-frequency domain, Rus *et al.* [25] applied short-time FFT technique to obtain the power distribution in the time-frequency domain, which are fed to different machine learning classifiers, e.g., SVM, KNN and ANN.

Lu *et al.* [21] employed the restricted Boltzmann machine (RBM) to conduct classification. The architecture of this method displays in Fig. 2. Both FFT and wavelet packet decomposition (WPD) are used to pretrain stacked RBMs. Experimental results demonstrate that the RBM-based classification framework outperforms some state-of-the-arts, such as filter bank common spatial pattern (FBCSP) in combination with mutual information-based best individual feature (MIBIF) [26] and FBCSP in combination with mutual information-based rough set reduction (MIRSR) [26]. Park *et al.* [27] proposed a comprehensive strategy combining WPD, CSP and kernel extreme learning machine (ELM) [28].

## III. Proposed Framework

Fig. 3 shows the framework of the proposed algorithm, which involves five phases. Raw EEG data are first normalized

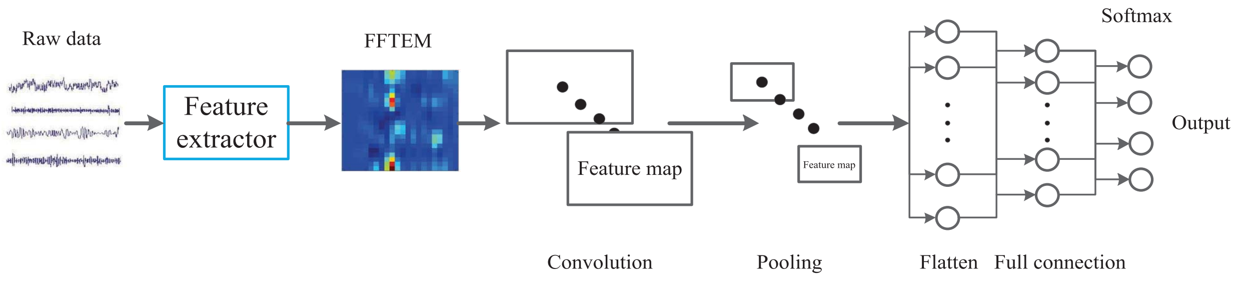


Fig. 1. Architecture of FFTEM extractor and CNN. The feature extractor combines the energy coefficients of EEG signals from all the channels into a matrix, namely FFTEM. FFTEMs are fed to CNN which contains one convolution layer, one pooling layer, one flatten layer, one full connection layer and one softmax regression layer.

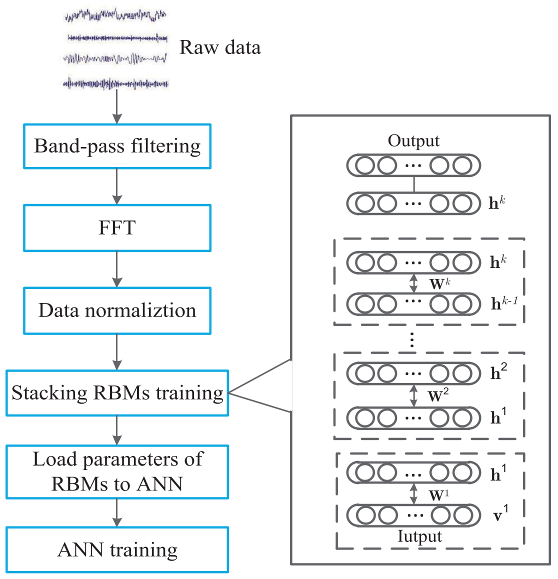


Fig. 2. Architecture of FDBN. The detail of stacking RBMs is displayed at right side, where $v^1, \ldots, v^k$ and $h^1, \ldots, h^k$ denote visual layers and hidden layers, respectively. Parameters $W^1, \ldots, W^k$ between visual layers and hidden layers are trained by contrastive divergence algorithm.

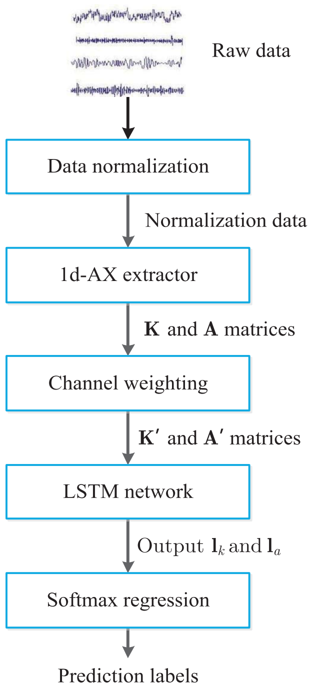


Fig. 3. Framework of proposed approach.

before extracting 1d-AX features. Secondly, the normalized data of each channel are fed to 1d-AX extractor. In the next stage, the number of channels is further reduced by channel weighting. Finally, the output is fed into an LSTM network, and the probability of each prediction label is obtained by a layer of softmax regression.

## A. Preprocessing

Bandpass filtering and normalization are two techniques most widely used in the preprocessing stage of various classification systems before feature extraction. Bandpass filtering aims to eliminate interference noise and spectral components within irrelevant bands. The frequency band between 8 and 35 Hz is recommended in motor imagery tasks. In practice, however, the exact passbands are subject-dependent and inaccurate bandpass filtering could lead to the degradation of the final classification accuracy. To avoid this problem, bandpass filtering is ignored in the proposed approach. Instead, the 1d-AX to be described in the next section is employed to achieve the effect of removing interference and smoothing normalized EEG data.

Let $\mathbf{X} \in \mathbb{R}^{c \times N}$ be an trial of observed signals and $\hat{\mathbf{X}} \in \mathbb{R}^{c \times N}$ denote preprocessed data, i.e., $\hat{\mathbf{X}} = \psi(\mathbf{X})$, where $\psi$ denotes the normalization operation, which yields normalized signals with mean and standard deviation equal to 0 and 1, respectively. Specifically, let $\mathbf{v}_i \in \mathbb{R}^N$ and $\hat{\mathbf{v}}_i \in \mathbb{R}^N$ be the $i$th row of $\mathbf{X}$ and $\hat{\mathbf{X}}$, respectively. The normalization can be described as

$$\hat{\mathbf{v}}_i = \frac{\mathbf{v}_i - \mathrm{mean}(\mathbf{v}_i)}{\mathrm{std}(\mathbf{v}_i)}, \quad i = 1, 2, \ldots, c, \tag{4}$$

where mean(·) and std(·) denote the mean and standard deviation, respectively.

## B. One Dimension-Aggregate Approximation (1d-AX)

The 1d-AX is a simplified version of the 1d-SAX [29], which is considered as an improved version of symbolic

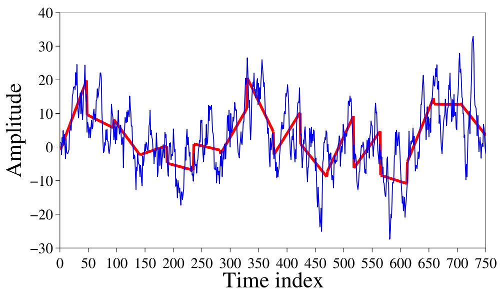


Fig. 4. Least squares estimation result of an EEG signal. The number of sample points of this EEG signal is 750, and the number of segments $m = 16$.

aggregate approximation (SAX) [30], the first symbolic representation of time series that allows for dimension reduction and indexing using a lower-bounding distance measure. The 1d-SAX segments time series into segments of equal length and computes the mean of every fragment. However, two visually different segments may have the same average value. To achieve more accurate representation, the 1d-SAX further adopts the slope of a line segment as a feature. The major steps of 1d-SAX feature extraction are given below.

1) Divide time series into segments of equal length $q$;
2) Apply linear regression analysis on each segment of normalized EEG data;
3) Quantize these regression results into a symbol from an alphabet of size $n$.

To simplify the calculation, we skip the third step in the 1d-SAX, i.e., we use the numerical values instead of symbols. Linear regression of each segment is calculated by least square estimation. Let $\hat{\mathbf{v}}_i$ be the $i$th row of $\hat{\mathbf{X}}$, that is, $\hat{\mathbf{X}} = [\hat{\mathbf{v}}_1^T \; \hat{\mathbf{v}}_2^T \; \ldots \; \hat{\mathbf{v}}_c^T]^T$. Then, $\hat{\mathbf{v}}_i$ is separated into $m$ segments of equal length $q = \lfloor N/m \rfloor$, where $\lfloor \cdot \rfloor$ means downward rounding. The $j$th segment of $\hat{\mathbf{v}}_i$ is denoted by $\hat{\mathbf{v}}_i^{(j)}$ whose elements are sampled at different time instants $\left\{ t_{i,k}^{(j)} \right\}_{k=1}^{q}$. Then, $\hat{\mathbf{v}}_i^{(j)}$ is approximated by a linear function $k_i^{(j)} \cdot t + b_i^{(j)}$, where parameters $k_i^{(j)}$ and $b_i^{(j)}$ can be computed by minimizing the least squares approximation error $\left\| k_i^{(j)} \cdot \mathbf{t}_i^{(j)} + b_i^{(j)} - \hat{\mathbf{v}}_i^{(j)} \right\|_2^2$ where $\mathbf{t}_i^{(j)} = [t_{i,1}^{(j)} \; t_{i,2}^{(j)} \; \ldots \; t_{i,q}^{(j)}]^T$. After some straightforward manipulations, we have

$$k_i^{(j)} = \frac{\sum_{k=1}^{q} \left( t_{i,k}^{(j)} - \bar{t}_i^{(j)} \right) \bar{v}_i^{(j)}}{\sum_{k=1}^{q} \left( t_{i,k}^{(j)} - \bar{t}_i^{(j)} \right)^2} \tag{5}$$

$$b_i^{(j)} = \bar{v}_i^{(j)} - k_i^{(j)} \cdot \bar{t}_i^{(j)} \tag{6}$$

where $\bar{t}_i^{(j)}$ and $\bar{v}_i^{(j)}$ represent the average values of $\mathbf{t}_i^{(j)}$ and $\hat{\mathbf{v}}_i^{(j)}$, respectively. After this step, each segment can be approximately represented by two parameters, the slope $k_i^{(j)}$ of each line segment and the mean$a_i^{(j)} = k_i^{(j)} \cdot \bar{t}_i^{(j)} + b_i^{(j)}$. Fig. 4 illustrates an example of the approximation result of the 1d-AX. The blue line represents the original EEG signals, and the red one shows the approximation result using linear regression.

Applying the above procedure to all the segments $\left\{ \hat{\mathbf{v}}_i^{(j)} \right\}_{j=1}^{m}$, we obtain

$$\mathbf{P}_i = \begin{bmatrix} k_i^{(1)} & k_i^{(2)} & \ldots & k_i^{(m)} \\ a_i^{(1)} & a_i^{(2)} & \ldots & a_i^{(m)} \end{bmatrix}, \quad i = 1, \ldots, c. \tag{7}$$

Because every trial has $c$ channels, we finally have $c$ matrices of $\mathbf{P}_i$. For the convenience of later discussion, we further construct matrices $\mathbf{K}$ and $\mathbf{A}$

$$\mathbf{K} = \begin{bmatrix} k_1^{(1)} & k_1^{(2)} & \ldots & k_1^{(m)} \\ \vdots & \vdots & \ddots & \vdots \\ k_c^{(1)} & k_c^{(2)} & \ldots & k_c^{(m)} \end{bmatrix} \tag{8}$$

$$\mathbf{A} = \begin{bmatrix} a_1^{(1)} & a_1^{(2)} & \ldots & a_1^{(m)} \\ \vdots & \vdots & \ddots & \vdots \\ a_c^{(1)} & a_c^{(2)} & \ldots & a_c^{(m)} \end{bmatrix}. \tag{9}$$

In principle, one can further incorporates more time-domain features (e.g., skewness and kurtosis) in this stage. But, as an attempt to balance the overall complexity of the system and its classification performance, the mean and the slope of data segments are employed as the most essential features for the proposed framework. Similar observations were also reported in [31], wherein Khorshidtalab *et al.* evaluated many statistical time-domain features and found that the Willison amplitude (WAMP) and slope sign change (SSC) are two promising features, if a suitable threshold value is defined for them, and the other features contribute less.

## C. Channel Weighting

Channel weighting is employed to extract useful information for the subsequent LSTM networks. In this stage, a group of spatial filters are introduced as a hidden layer, whose output is fed to LSTM networks.

Let $\mathbf{W}_k \in \mathbb{R}^{c' \times c}$ and $\mathbf{W}_a \in \mathbb{R}^{c' \times c}$ be weighting matrices. Channel weighting is conducted by

$$\mathbf{K}' = \mathbf{W}_k \cdot \mathbf{K} \tag{10}$$

$$\mathbf{A}' = \mathbf{W}_a \cdot \mathbf{A}. \tag{11}$$

Essentially speaking, to optimize weighting coefficients in $\mathbf{W}_k$ and $\mathbf{W}_a$, we have to solve the following optimization problem

$$\underset{\mathbf{W}_k, \mathbf{W}_a}{\text{minimize}} \quad J\left[ \mathcal{F}\left( \mathbf{K}', \mathbf{A}' \right) \right] \tag{12}$$

where $\mathcal{F}(\cdot)$ denotes forward propagation after channel weighting, and $J[\cdot]$ represents the cost function of whole classification system, that is to be discussed in the next section.

Because gradient descent is employed to solve (12), the gradient from $J$ to each element in $\mathbf{W}_k$ and $\mathbf{W}_a$ is necessary. Let $w_{ij}$ be the $(i, j)$th element of $\mathbf{W}_k$. The gradient of $w_{ij}$ can be computed by

$$\frac{\partial J}{\partial w_{ij}} = \sum_{n=1}^{m} \frac{\partial J}{\partial k'_{in}} \frac{\partial k'_{in}}{\partial w_{ij}} = \sum_{n=1}^{m} \frac{\partial J}{\partial k'_{in}} k_{jn} \tag{13}$$

where $k'_{in}$ and $k_{jn}$ denote, respectively, the $(i, n)$th and $(j, n)$th elements of $\mathbf{K}'$ and $\mathbf{K}$, and $\frac{\partial J}{\partial k'_{in}}$ represents the gradient of $J$

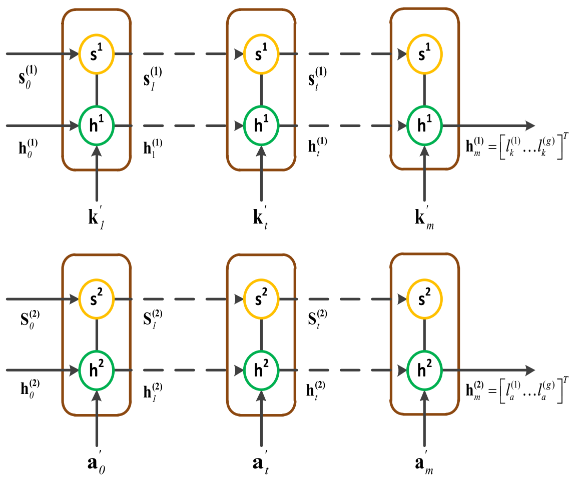


Fig. 5. Architecture of LSTM part. Both LSTM networks have $m$ cells corresponding to $m$ columns of $\mathrm{K}'$ and $\mathrm{A}'$.

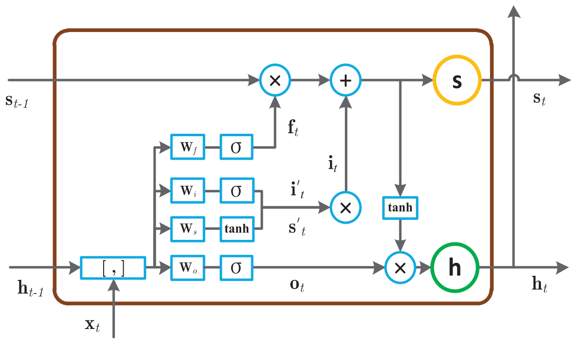


Fig. 6. Architecture of LSTM cell.

with respect to $k'_{in}$, which can be computed by the chain rule in back propagation (BP) algorithm [32]. The gradient of $J$ with respect to each element of $\mathbf{W}_a$ can be obtained using the step similar to (13).

For the convenience of latter discussion, we express $\mathbf{K}'$ and $\mathbf{A}'$ as $\mathbf{K}' = \left[\mathbf{k}'_1 \ \dots \ \mathbf{k}'_m\right]$ and $\mathbf{A}' = \left[\mathbf{a}'_1 \ \dots \ \mathbf{a}'_m\right]$, respectively, where $\mathbf{k}'_t, \mathbf{a}'_t \in \mathbb{R}^{c'}$ for $t = 1, \dots, m$.

## D. Long Short-Term Memory (LSTM) Network

After channel weighting, $\mathbf{K}'$ and $\mathbf{A}'$ are treated as time series and fed to two LSTM networks. Actually, $\mathbf{K}'$ and $\mathbf{A}'$ are processed by two LSTMs simultaneously. Fig. 5 shows the parallel processing of $\mathbf{K}'$ and $\mathbf{A}'$ in two LSTM networks. For the convenience of presentation, we use $\mathbf{s}_t$ to replace $\mathbf{s}_t^{(\mathbf{i})}$ $(i = 1, 2)$ and $\mathbf{h}_t$ to replace $\mathbf{h}_t^{(\mathbf{i})}$ $(i = 1, 2)$. The detailed architecture of an LSTM cell is given in Fig. 6, where operator [ , ] concatenates two vectors into a vector of larger size. Let $\mathbf{x}_t$ be the input vector at current time instant, i.e., $\mathbf{x}_t = \mathbf{k}'_t$ or $\mathbf{a}'_t$ for $t = 1, 2, \cdots, m$. Suppose that $\mathbf{s}_t \in \mathbb{R}^g$ and $\mathbf{h}_t \in \mathbb{R}^g$ are the cell state and the output of the hidden layer at time instant $t$, respectively. Both $\mathbf{s}_0$ and $\mathbf{h}_0$ are initialized by zeros. At time instant $t$, an LSTM cell generates $\mathbf{s}_t$ and $\mathbf{h}_t$

$$\mathbf{s}_t = \mathbf{s}_{t-1} \circ \mathbf{f}_t + \mathbf{i}_t \tag{14}$$
$$\mathbf{h}_t = \mathbf{o}_t \circ \tanh(\mathbf{s}_t) \tag{15}$$

where $\circ$ represents element-wise multiplication, $\mathbf{f}_t$, $\mathbf{i}_t$ and $\mathbf{o}_t$ denote the outputs of forget gate, input gate and output gate, respectively.

Forget gate decides how many information of $\mathbf{s}_{t-1}$ will be retained. The output of forget gate is computed by

$$\mathbf{f}_t = \sigma\left(\mathbf{W}_f \cdot \left[\mathbf{h}_{t-1}, \mathbf{x}_t\right] + \mathbf{b}_f\right) \tag{16}$$

where $\mathbf{W}_f \in \mathbb{R}^{g \times (c'+g)}$ and $\mathbf{b}_f \in \mathbb{R}^g$ are coefficients to be estimated in training stage, and $\sigma(\cdot)$ represents a sigmoid function defined by

$$\sigma(z) = \frac{1}{1 + e^{-z}}. \tag{17}$$

If the input of $\sigma(\cdot)$ is an vector, (17) is implemented on each element of its input vector. In practice, the bias of forget gate $\mathbf{b}_f$ is generally initialized by ones, which is beneficial for the cell to keep primitive information.

Input gate decides how many information of $\mathbf{x}_t$ and $\mathbf{h}_{t-1}$ will be retained in current $\mathbf{s}_t$. The output of input gate is given by

$$\mathbf{i}_t = \mathbf{i}'_t \circ \mathbf{s}'_t, \tag{18}$$

where

$$\mathbf{i}'_t = \sigma\left(\mathbf{W}_i \cdot \left[\mathbf{h}_{t-1}, \mathbf{x}_t\right] + \mathbf{b}_i\right) \tag{19}$$
$$\mathbf{s}'_t = \tanh\left(\mathbf{W}_s \cdot \left[\mathbf{h}_{t-1}, \mathbf{x}_t\right] + \mathbf{b}_s\right). \tag{20}$$

In (19) and (20), $\mathbf{W}_i, \mathbf{W}_s \in \mathbb{R}^{g \times (c'+g)}$ and $\mathbf{b}_i, \mathbf{b}_s \in \mathbb{R}^g$ are weighting matrices and biases, respectively, and $\tanh(\cdot)$ denotes a hyperbolic tangent function defined by

$$\tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}. \tag{21}$$

Output gate decides how many information of $\mathbf{s}_t$ will be retained in the output of hidden layer $\mathbf{h}_t$. Its output is given by

$$\mathbf{o}_t = \sigma\left(\mathbf{W}_o \cdot \left[\mathbf{h}_{t-1}, \mathbf{x}_t\right] + \mathbf{b}_o\right) \tag{22}$$

where $\mathbf{W}_o \in \mathbb{R}^{g \times (c'+g)}$ and $\mathbf{b}_o \in \mathbb{R}^g$.

In summary, given either $\mathbf{K}'$ or $\mathbf{A}'$, there are eight groups of parameters (i.e., $\mathbf{W}_f$ and $\mathbf{b}_f$, $\mathbf{W}_i$ and $\mathbf{b}_i$, $\mathbf{W}_s$ and $\mathbf{b}_s$, $\mathbf{W}_o$ and $\mathbf{b}_o$). The training of the LSTM network can be accomplished by back propagation through time (BPTT) algorithm [33].

## E. Softmax Regression

LSTM networks can only process the input 1d-AX features. For the purpose of EEG classification, we still need a softmax regression layer to predict the probability of each class. Let both outputs of the LSTM networks be $\mathbf{h}_m^{(1)} = \left[l_k^{(1)} \ \dots \ l_k^{(g)}\right]^T$ and $\mathbf{h}_m^{(2)} = \left[l_a^{(1)} \ \dots \ l_a^{(g)}\right]^T$, respectively. Then, they are combined to a single vector $\mathbf{l} = \left[l_k^{(1)} \ \dots \ l_k^{(g)} \ l_a^{(1)} \ \dots \ l_a^{(g)}\right]^T$, which is further fed into a softmax regression layer. Fig. 7 shows the details of input concatenation and softmax regression.

The parameters of softmax regression are denoted as $\mathbf{W}_{sf} \in \mathbb{R}^{n_c \times 2g}$, where $n_c$ is the number of categories of EEG signals to be classified. The conditional probability of

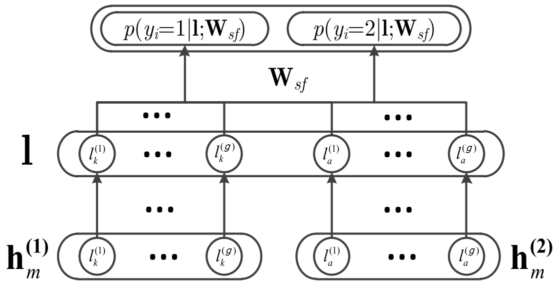


Fig. 7. Architectures of concatenation and softmax regression layer, and $y_i$ represents the prediction label of $i$th input in a mini-batch.

$p(y = j|\mathbf{l}; \mathbf{W}_{sf})$ for each class $j = 1, \cdots, n_c$ can be calculated by

$$\begin{bmatrix} p(y = 1|\mathbf{l}; \mathbf{W}_{sf}) \\ \vdots \\ p(y = n_c|\mathbf{l}; \mathbf{W}_{sf}) \end{bmatrix} = \frac{1}{\sum_{j=1}^{n_c} e^{\mathbf{w}_j^T \cdot \mathbf{l}}} \begin{bmatrix} e^{\mathbf{w}_1^T \cdot \mathbf{l}} \\ \vdots \\ e^{\mathbf{w}_{nc}^T \cdot \mathbf{l}} \end{bmatrix} \tag{23}$$

where $y$ represents the prediction label, and $\mathbf{w}_j^T$ denotes the $j$th row of $\mathbf{W}_{sf}$. While dealing with binary classification problem, it will degenerate to the logistic regression. For training the whole system, we employ log-likelihood cost as the final cost function

$$L(\theta) = -\frac{1}{n}\left[\sum_{i=1}^{n}\sum_{j=1}^{n_c} \mathcal{I}(y_i = j) \log p\left(y_i = j|\mathbf{l}; \mathbf{W}_{sf}\right)\right] \tag{24}$$

where $\theta$ is a vector containing all the parameters to be tuned, $n$ is the number of trials of EEG signals, $y_i$ represents the prediction label of $i$th input in a mini-batch, and $\mathcal{I}(\cdot)$ is the indicator function which takes 1 if its argument is true and 0 otherwise. To avoid over-fitting in validation phase, a regularization term can also be incorporated in the cost function

$$J = L(\theta) + \lambda\rho(\theta). \tag{25}$$

Note that, in practice, not all the parameters $\theta$ are used in the regularization term $\rho(\theta)$. To reduce the probability of the over-fitting, we adopt $l_1$ norm in $\rho(\theta)$ to achieve sparse solution.

## IV. Experimental Results

To verify the performance of the proposed framework, experiments based on public BCI competition dataset are presented in this section.

### A. Dataset Description

The data used in our experiment is taken from Dataset 2A of BCI Competition IV [34], which is collected from 9 subjects. There are two sessions for each subject, and each session contains 288 trials, yielding total 5184 trials. However, some trials are removed because they contain missing values. After removing these unavailable trials, there are approximately 4800 trials. EEG signals of each trial are sampled by 22 channels (i.e., $c = 22$) with sampling frequency 250 Hz, resulting in 750 sample points (i.e., $N = 750$). This dataset involves four different motor imagery tasks, namely, the imagination of movements of left hand, right hand, both feet, and tongue. For a fair comparison, we focus on binary classification, that is, $n_c = 2$. But for a comprehensive evaluation, we conduct 6 groups (c.f., Table I) of binary classification experiments. Each group contains approximately 2400 trials.

TABLE I
Experiment Groups of Binary Classification

| Group | Category 1 | Category 2 |
|---|---|---|
| I | left hand | right hand |
| II | left hand | both feet |
| III | left hand | tongue |
| IV | right hand | both feet |
| V | right hand | tongue |
| VI | both feet | tongue |

### B. Experiment Using Hybrid EEG Data of All Subjects

In this subsection, we apply EEG data of all the subjects to train the proposed network and evaluate its performance. Training and validation trials are assigned by $5 \times 5$ folds cross-validation, yielding about 1920 trials for training and 480 for validation in each group listed in Table I. Different algorithms are trained and tested using the same set of EEG data.

To evaluate the performance of the proposed 1d-AX-based LSTM (AX-LSTM for short) network, we conduct a series of comparisons with the other approaches using both the traditional CSP method and some deep networks, including multi-channel deep belief network (MCDBN) [35], FDBN [21], FFTEM [5], wavelet form of Gramian angular field using residual network (GAFRN) [36] and wavelet form of Gramian angular field using tiled convolutional neural network (GAFTCNN) [36].

In our experiment, parameters are chosen as $m = 16$, $c' = 6$, $g = 8$, and $\lambda = 0.02$. Only channel weighting coefficients $\mathbf{W}_k$ and $\mathbf{W}_a$ are used in $l_1$ norm of the regularization term in (25) and initialized by random values following the truncated normal distribution [37]. To accelerate the convergence of training procedure and enhance the generalization capability of the proposed model, we apply batch normalization (BN) [38] on outputs $\mathbf{K}'$ and $\mathbf{A}'$ after channel weighting. In LSTM cells, the biases of forget gate $\mathbf{b}_f$ are initially set to 1 so as to remember primary information at the beginning of the training. Other tunable parameters in LSTM cells are randomly initialized by random values of uniform distribution within [0, 1].

We also employ dropout [39] to enhance the generalization capability of the proposed model. Specifically, each unit of gate outputs of LSTM cells is retained with a fixed probability 0.6 independent of other units. We apply adaptive moment estimation (Adam) algorithm [40] to minimize the cost function in (25). Learning rate $\eta$, decay coefficients of the first and second moment estimation $\beta_1$ and $\beta_2$ are initialized as $\eta = 0.001$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, respectively. Back-propagation procedure is completed by tensorflow on the platform of Python. All the parameters in our model are trained with batch size 64. Each fold in cross validation has run more than 20000 batches.

Table II lists classification accuracies of all the methods employed for comparison in this section. The best result of

TABLE II
CLASSIFICATION ACCURACIES USING EEG DATA OF ALL SUBJECTS

| | CSP | MCDBN | FDBN | FFTEM | GAFRN | GAFTCNN | AX-LSTM |
|---|---|---|---|---|---|---|---|
| I | 72.7 | 52.6 | 64.6 | 61.3 | 62.2 | 65.1 | **74.8** |
| II | 71.0 | 54.2 | 65.6 | 64.7 | 65.2 | 62.9 | **74.2** |
| III | **74.0** | 50.5 | 61.9 | 64.4 | 66.2 | 68.5 | 72.5 |
| IV | 71.3 | 59.8 | 66.8 | 63.2 | 60.3 | 64.3 | **76.3** |
| V | 71.9 | 63.2 | 70.2 | 66.3 | 68.5 | 67.5 | **74.3** |
| VI | 64.9 | 54.4 | 64.6 | 60.2 | 70.1 | 68.6 | **79.6** |

TABLE III
PAIRED $t$-TEST BETWEEN AX-LSTM AND OTHER METHODS

| AX-LSTM | CSP | MCDBN | FDBN | FFTEM | GAFRN | GAFTCNN |
|---|---|---|---|---|---|---|
| I | $1.033 \times 10^{-4}$ | $6.126 \times 10^{-28}$ | $1.140 \times 10^{-17}$ | $3.495 \times 10^{-19}$ | $3.353 \times 10^{-18}$ | $3.120 \times 10^{-18}$ |
| II | $2.374 \times 10^{-7}$ | $1.541 \times 10^{-22}$ | $1.947 \times 10^{-13}$ | $5.552 \times 10^{-16}$ | $6.976 \times 10^{-16}$ | $6.066 \times 10^{-18}$ |
| III | $9.933 \times 10^{-3}$ | $7.435 \times 10^{-23}$ | $3.508 \times 10^{-15}$ | $4.282 \times 10^{-12}$ | $5.295 \times 10^{-13}$ | $2.289 \times 10^{-8}$ |
| IV | $7.215 \times 10^{-10}$ | $5.443 \times 10^{-21}$ | $8.493 \times 10^{-17}$ | $8.785 \times 10^{-21}$ | $4.140 \times 10^{-22}$ | $1.259 \times 10^{-17}$ |
| V | $3.798 \times 10^{-5}$ | $2.178 \times 10^{-18}$ | $1.598 \times 10^{-10}$ | $7.728 \times 10^{-14}$ | $6.328 \times 10^{-12}$ | $1.832 \times 10^{-11}$ |
| VI | $9.677 \times 10^{-24}$ | $3.896 \times 10^{-26}$ | $2.994 \times 10^{-19}$ | $8.299 \times 10^{-23}$ | $7.006 \times 10^{-15}$ | $3.309 \times 10^{-18}$ |

TABLE IV
NUMBER OF MODEL PARAMETERS

| | MCDBN | FDBN | FFTEM | GAFRN | GAFTCNN | AX-LSTM |
|---|---|---|---|---|---|---|
| # of parameters | 48954 | 23302 | 60058 | 5794 | 3706 | 746 |

TABLE V
CLASSIFICATION ACCURACIES OF DWEC, FFTEM, AX-LSTM AND FDBN FOR EACH SUBJECT ON GROUP I

| Method | Sub. 1 | Sub. 2 | Sub. 3 | Sub. 4 | Sub. 5 | Sub. 6 | Sub. 7 | Sub. 8 | Sub. 9 | Avg. | Std. |
|---|---|---|---|---|---|---|---|---|---|---|---|
| DWEC | 95.40 | 64.20 | 96.80 | 67.30 | 75.90 | 65.20 | 78.10 | 96.10 | 93.20 | 81.50 | 14.20 |
| FFTEM | 68.72 | 54.97 | 69.08 | 55.07 | 72.66 | 60.61 | 70.13 | 83.49 | 83.14 | 68.41 | 9.93 |
| AX-LSTM | 75.12 | 71.38 | 72.24 | 72.92 | 82.62 | 69.64 | 88.98 | 80.28 | 75.07 | 76.47 | 5.92 |
| FDBN | 71.08 | 55.56 | 76.87 | 65.62 | 69.08 | 64.98 | 71.68 | 92.37 | 82.38 | 72.18 | 10.09 |

binary classification for each group is highlighted in boldface. It can be clearly observed that the AX-LSTM outperforms other methods except in the experiment of Group III. Note that, in general, the performance of feature extraction in motor imagery tasks are subject-dependent. But experimental results presented in this subsection demonstrate that training a uniform feature extractor/classifier using the proposed framework for different subjects is still a promising way in practice.

To verify whether the performance difference between the proposed AX-LSTM method and other methods is statistically significant, paired $t$-test is conducted between the results of the AX-LSTM and other methods. The obtained $p$-values are given in Table III. It can be seen that all the obtained $p$-values are less than 0.01, which implies that the performance improvement of the AX-LSTM over other methods is statistically significant.

Due to the difference of network frameworks, different networks require different numbers of parameters, which are also listed in Table IV. Compared to other networks, AX-LSTM requires less tunable parameters, which highly reduces the risk of over-fitting.

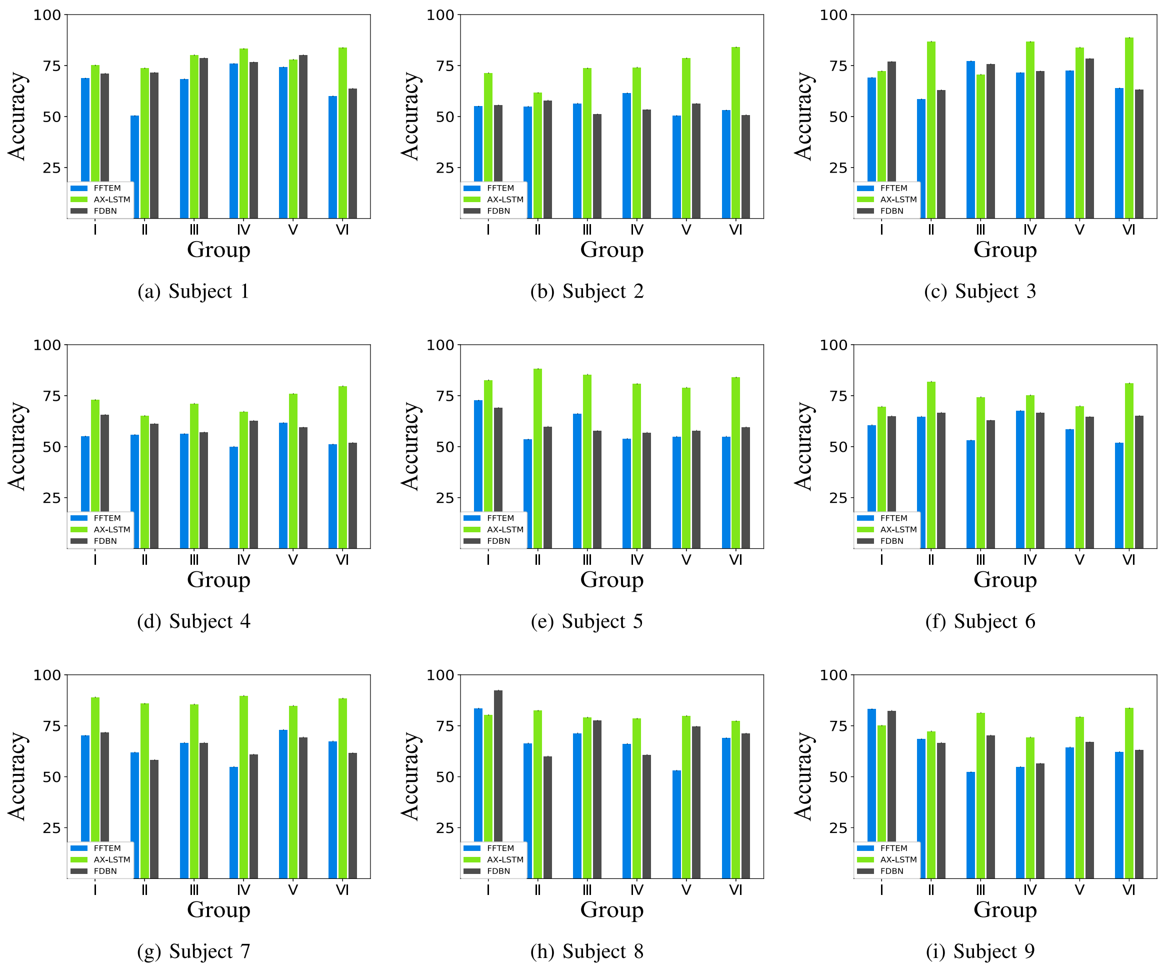
## C. Experiments Using EEG Data of Each Subject

In this subsection, we evaluate the performance of the proposed AX-LSTM using EEG data of each subject. Two of the most competitive deep-network-based methods (i.e., FFTEM and FDBN) in Subsection IV-B are also employed for comparison. We use the first session to train different feature extractors and classifiers, and the second one to validate their performance. In this experiment, parameters of the AX-LSTM network are set equal to $m = 16$, $c' = 6$, $g = 12$, and $\lambda = 0.02$. Since each subject has the limited amount of training data, as an attempt to avoid over-fitting, only average values of each segment (i.e., $\mathbf{A}$) in the 1d-AX are used as inputs of channel weighting. Consequently, only $\mathbf{W}_a$ is used in $l_1$-norm regularization of (25).

Fig. 8 shows classification accuracy of each subject. In most of experiments presented in this subsection, the proposed method achieves better performance than the other two deep-network-based approaches, which demonstrates the capability of the LSTM network on analyzing time-series EEG signals. To gain more insights, classification accuracies of each subject on Group I are listed in Table V. Compared to the other two deep-network-based methods, the proposed AX-LSTM attains more than 4% improvement on the mean accuracy and always the minimum standard deviation of accuracies on different subjects. For a comprehensive comparison, another state-of-the-art method called dynamically weighted ensemble classification (DWEC) [41] is also incorporated in Table V. It can be observed that the DWEC shows the impressive performance on some subjects, while the proposed AX-LSTM has much lesser standard deviation of classification accuracies, demonstrating its consistent performance on different subjects.

## D. Experiments of Prediction Using EEG Data of Different Subjects

As stated before, EEG data are subject-dependent, which implies that it is generally hard to predict motor imagery labels of one subject using EEG data recorded from different subjects. However, in practice, collecting EEG data is still a time-consuming and tedious process. If a network can be



(a) Subject 1 (b) Subject 2 (c) Subject 3 (d) Subject 4 (e) Subject 5 (f) Subject 6 (g) Subject 7 (h) Subject 8 (i) Subject 9

Fig. 8. Prediction accuracies using EEG data of each subject. (a) Subject 1. (b) Subject 2. (c) Subject 3. (d) Subject 4. (e) Subject 5. (f) Subject 6. (g) Subject 7. (h) Subject 8. (i) Subject 9.

employed to predict motor imagery labels of EEG data of a new subject without any further data collection and training, it will greatly facilitate practical application. To evaluate the generalization property of the AX-LSTM, in this subsection we use EEG data of eight subjects to train the proposed AX-LSTM network, which is then implemented to predict motor imagery labels of EEG data of the remaining subjects. All the parameters of the AX-LSTM are set to the same values as in Subsection IV-B. Prediction accuracies of each subject are given in Table VI. It can be observed that the proposed AX-LSTM can achieve prediction accuracies above 60% in experiments of all the subjects and better performance (i.e., prediction accuracies above 70%) in experiments of subjects 3, 5, 7 and 8.

## E. Effects of Channel Weighting on Classification Accuracy

Channel weighting technique can reduce the number of effective channels and consequently the size of inputs to LSTM network. To evaluate its influence on final classification accuracy, another set of experiments are conducted on Group VI in this subsection. Except $c'$, the same set of parameters as in Subsection IV-B are used to construct our network. The value of $c'$ varies from 2 to 22, and the variation of classification accuracy with respect to $c'$ is depicted in

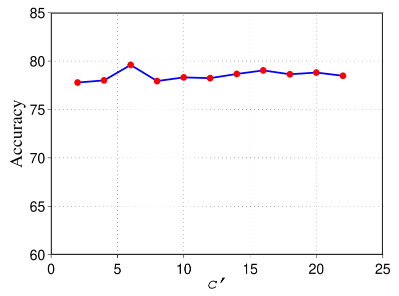


Fig. 9. Classification accuracies obtained by using different $c'$.

Fig. 9. It can be observed that the variation of $c'$ yields a slight fluctuation on classification accuracy. But the range of the fluctuation is within 2%, which implies that the AX-LSTM is robust to $c'$. Since $c' = 6$ leads to the best performance of the network, we adopt this value in the previous experiments.

## F. Effects of Segment Size on Classification Accuracy

To evaluate the sensitivity of the AX-LSTM to the change of segment size in the 1d-AX, different $m$ are used in the experiment of this subsection. Specifically, the value of $m$ varies from 4 to 36, while the other parameters are kept unchanged. Fig. 10 shows the performance of the AX-LSTM using different $m$. Generally, for a small $m$ (that is, a large segment size), one can achieve consistent classification results

TABLE VI

PREDICTION ACCURACIES USING EEG DATA OF DIFFERENT SUBJECTS

| | Sub. 1 | Sub. 2 | Sub. 3 | Sub. 4 | Sub. 5 | Sub. 6 | Sub. 7 | Sub. 8 | Sub. 9 | Avg. |
|---|---|---|---|---|---|---|---|---|---|---|
| I | 55.9 | 63.2 | 64.5 | 66.0 | 79.9 | 63.5 | 84.3 | 78.9 | 71.0 | 69.7 |
| II | 68.6 | 62.4 | 77.4 | 62.6 | 83.6 | 73.6 | 80.4 | 55.8 | 59.9 | 69.4 |
| III | 67.6 | 62.6 | 70.1 | 71.7 | 76.2 | 59.2 | 76.0 | 71.3 | 68.3 | 69.3 |
| IV | 65.6 | 69.5 | 75.7 | 62.9 | 76.4 | 67.6 | 80.2 | 68.2 | 59.0 | 69.5 |
| V | 63.9 | 65.8 | 69.5 | 70.1 | 73.8 | 65.3 | 69.1 | 72.4 | 76.3 | 69.6 |
| VI | 68.0 | 70.0 | 85.1 | 76.6 | 82.7 | 80.2 | 86.9 | 73.8 | 72.1 | 77.3 |
| Avg. | 64.9 | 65.6 | 73.8 | 68.4 | 78.8 | 68.2 | 79.5 | 70.0 | 67.8 | |

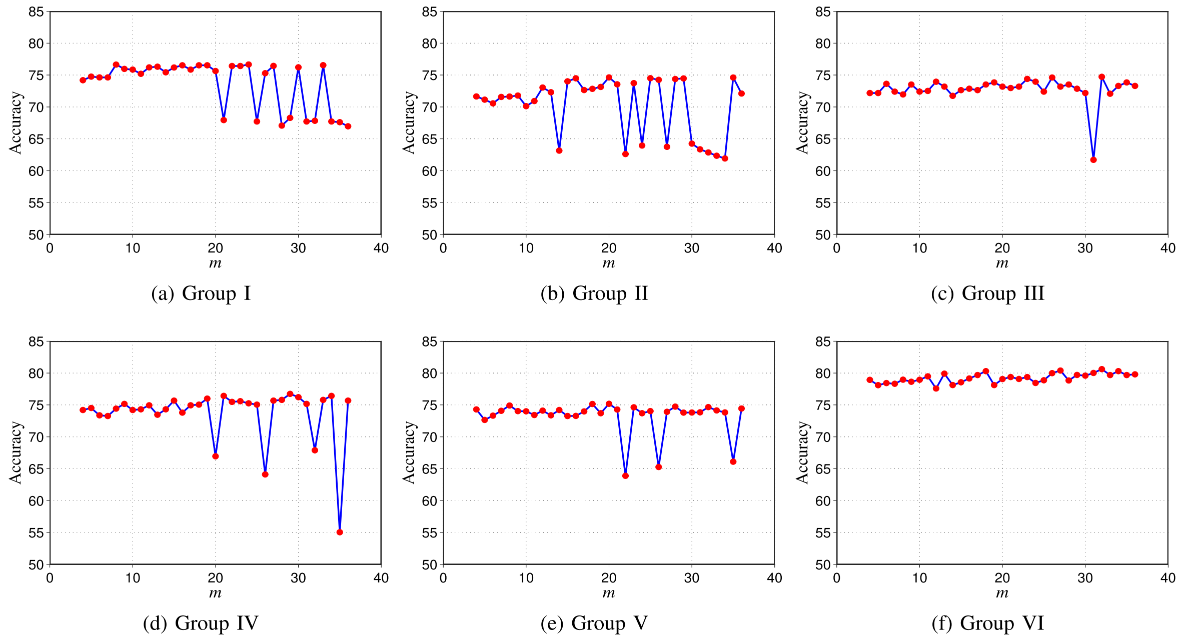


Fig. 10. Classification accuracies obtained by using different $m$. (a) Group I. (b) Group II. (c) Group III. (d) Group IV. (e) Group V. (f) Group VI.

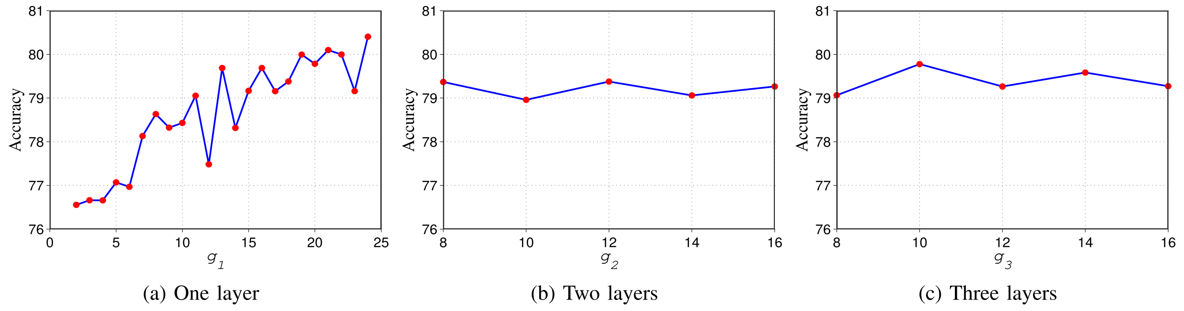


Fig. 11. Classification accuracies using different structures of LSTM network. (a) One layer. (b) Two layers. (c) Three layers.

in each binary classification experiment. But, with the increase of $m$ (or the decrease of segment size), the representation of the 1d-AX is more sensitive to interference noise. This is reflected by the performance fluctuation of the AX-LSTM in some binary classification experiments.

## G. Experiments Using Different Structures of LSTM Network

Multiple layers can be employed in the proposed LSTM network in practice. To evaluate the effects of network architecture on final classification accuracy, different numbers of LSTM layers and units in LSTM cells are used in our experiments of this subsection. Experimental results presented in this subsection are based on EEG data of Group VI. We first fix the number of LSTM layers and then change the number of output units. For clarity, we use $g_i$ to denote the size of $\mathbf{h}_t$ at the $i$th layer. Fig. 11a depicts the variation of classification accuracy of the AX-LSTM with respect to $g_1$ when using only one layer. Here, the value of $g_1$ varies from 2 to 24. With the increase of $g_1$, classification accuracy of the AX-LSTM becomes better gradually. When using two layers, we first set $g_1$ equal to 8 and, then, the value of $g_2$ varies from 8 to 16. Fig.11b shows the variation of classification accuracy with respect to $g_2$. It can be noticed that increasing the length of hidden units in the second layer hardly leads to the performance improvement of the AX-LSTM. Finally, when using three layers, we first set $g_1$ and $g_2$ both equal to 8. Then, the length of hidden units $g_3$ in the third LSTM layer takes a value varying from 8 to 16. Fig. 11c depicts the variation of classification accuracy with respect to $g_3$. Similar to the situation using two LSTM layers, the performance cannot be further improved even deploying more LSTM layers and cells.

## V. Conclusions

In this paper, we present a novel framework to classify EEG data in motor imagery tasks. The proposed framework involves several stages and components, i.e., preprocessing, feature extraction using the 1d-AX, channel weighting, LSTM network, and softmax regression. Different to most of existing methods using CSP techniques and deep networks, LSTM networks are appropriately deployed to extract essential features of EEG signals, such that time-varying characteristics of EEG signals can be exploited to enhance the classification performance. Furthermore, the 1d-AX along with channel weighting technique makes EEG signal representation more concise, which ease the subsequent training of LSTM networks. Experimental results based on public BCI competition dataset demonstrate the superior classification performance of the proposed AX-LSTM. For online classification, the whole network can be first trained offline by EEG data collected from different subjects. It is then fed by new EEG segments and outputs the probability of the prediction label. The scale of parameters in the proposed method is far more smaller than other deep networks, which means faster real-time processing and a small risk of over-fitting.

## References

[1] B. Graimann, B. Allison, and G. Pfurtscheller, *Brain–Computer Interfaces: A Gentle Introduction*. Dordrecht, The Netherland: Springer, 2010.

[2] S. Sanei and J. A. Chambers, *EEG Signal Processing*. New York, NY, USA: Wiley, 2007.

[3] J. Müller-Gerking, G. Pfurtscheller, and H. Flyvbjerg, "Designing optimal spatial filters for single-trial EEG classification in a movement task," *Clin. Neurophysiol.*, vol. 110, no. 5, pp. 787–798, 1999.

[4] M. Arvaneh, C. Guan, K. K. Ang, and C. Quek, "Multi-frequency band common spatial pattern with sparse optimization in brain-computer interface," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, Mar. 2012, pp. 2541–2544.

[5] T. Uktveris and V. Jusas, "Application of convolutional neural networks to four-class motor imagery classification problem," *Inf. Technol. Control*, vol. 46, no. 2, pp. 260–273, 2017.

[6] J. Hu, D. Xiao, and Z. Mu, "Application of energy entropy in motor imagery EEG classification," *Int. J. Digit. Content Technol. Appl.*, vol. 3, no. 2, pp. 83–90, 2009.

[7] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[8] S. Haykin, *Neural Networks: A Comprehensive Foundation*. Englewood Cliffs, NJ, USA: Prentice-Hall, 1999.

[9] C. M. Bishop, *Neural Networks for Pattern Recognition*. Oxford, U.K.: Oxford Univ. Press, 1995.

[10] M. Hamedi, S.-H. Salleh, A. M. Noor, and I. Mohammad-Rezazadeh, "Neural network-based three-class motor imagery classification using time-domain features for BCI applications," in *Proc. IEEE REGION 10 Symp.*, Apr. 2014, pp. 204–207.

[11] K. Simonyan and A. Zisserman. (2014). "Very deep convolutional networks for large-scale image recognition." [Online]. Available: https://arxiv.org/abs/1409.1556

[12] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 770–778.

[13] B. Hidasi, A. Karatzoglou, L. Baltrunas, and D. Tikk. (2015). "Session-based recommendations with recurrent neural networks." [Online]. Available: https://arxiv.org/abs/1511.06939

[14] Y. M. Assael, B. Shillingford, S. Whiteson, and N. de Freitas. (2016). "LipNet: End-to-end sentence-level lipreading." [Online]. Available: https://arxiv.org/abs/1611.01599

[15] M. Cooke, J. Barker, S. Cunningham, and X. Shao, "An audio-visual corpus for speech perception and automatic speech recognition," *J. Acoust. Soc. Amer.*, vol. 120, no. 5, pp. 2421–2424, 2006.

[16] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE Trans. Neural Netw.*, vol. 5, no. 2, pp. 157–166, Mar. 1994.

[17] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[18] A. Graves, "Long short-term memory," in *Supervised Sequence Labelling with Recurrent Neural Networks*. Springer, 2012, pp. 37–45.

[19] K. Cho *et al.* (2014). "Learning phrase representations using RNN encoder-decoder for statistical machine translation." [Online]. Available: https://arxiv.org/abs/1406.1078

[20] F. A. Gers and J. Schmidhuber, "Recurrent nets that time and count," in *Proc. IEEE Int. Joint Conf. Neural Netw.*, Jul. 2000, pp. 189–194.

[21] N. Lu, T. Li, X. Ren, and H. Miao, "A deep learning scheme for motor imagery classification based on restricted Boltzmann machines," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 25, no. 6, pp. 566–576, Jun. 2017.

[22] Y. Ren and Y. Wu, "Convolutional deep belief networks for feature extraction of EEG signal," in *Proc. IEEE Int. Joint Conf. Neural Netw.*, Jul. 2014, pp. 2850–2853.

[23] Y. Wang, S. Gao, and X. Gao, "Common spatial pattern method for channel selelction in motor imagery based brain-computer interface," in *Proc. Eng. Med. Biol. Soc.*, Jan. 2006, pp. 5392–5395.

[24] F. Goksu, N. F. Ince, and A. H. Tewfik, "Sparse common spatial patterns in brain computer interface applications," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, May 2011, pp. 533–536.

[25] I. D. Rus, P. Marc, M. Dinsoreanu, R. Potolea, and R. C. Muresan, "Classification of EEG signals in an object recognition task," in *Proc. IEEE Intell. Comput. Commun. Process.*, Sep. 2017, pp. 391–395.

[26] K. K. Ang, Z. Y. Chin, C. Wang, C. Guan, and H. Zhang, "Filter bank common spatial pattern algorithm on BCI competition IV datasets 2a and 2b," *Frontiers Neurosci.*, vol. 6, no. 1, p. 39, 2012.

[27] H.-J. Park, J. Kim, B. Min, and B. Lee, "Motor imagery EEG classification with optimal subset of wavelet based common spatial pattern and kernel extreme learning machine," in *Proc. IEEE Eng. Med. Biol. Soc. (EMBC)*, Jul. 2017, pp. 2863–2866.

[28] G.-B. Huang, H. Zhou, X. Ding, and R. Zhang, "Extreme learning machine for regression and multiclass classification," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 42, no. 2, pp. 513–529, Apr. 2012.

[29] S. Malinowski, T. Guyet, R. Quiniou, and R. Tavenard, "1d-SAX: A novel symbolic representation for time series," in *Proc. Int. Symp. Intell. Data Anal.* U.K.: Springer, 2013, pp. 273–284.

[30] J. Lin, E. Keogh, S. Lonardi, and B. Chiu, "A symbolic representation of time series, with implications for streaming algorithms," in *Proc. ACM SIGMOD Workshop Res. Issues Data Mining Knowl. Discovery*, 2003, pp. 2–11.

[31] A. Khorshidtalab, M. Salami, and M. Hamedi, "Robust classification of motor imagery EEG signals using statistical time–domain features," *Physiol. Meas.*, vol. 34, no. 11, pp. 1563–1579, 2013.

[32] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[33] P. J. Werbos, "Backpropagation through time: What it does and how to do it," *Proc. IEEE*, vol. 78, no. 10, pp. 1550–1560, Oct. 1990.

[34] R. Leeb, C. Brunner, G. Müller-Putz, A. Schlögl, and G. Pfurtscheller, "BCI competition 2008–Graz data set A," Graz Univ. Technol., Graz, Austria, Tech. Rep., 2008, vol. 16.

[35] A. M. Al-kaysi, A. Al-Ani, and T. W. Boonstra, "A multichannel deep belief network for the classification of EEG data," in *Proc. Int. Conf. Neural Inf. Process.*, 2015, pp. 38–45.

[36] Z. Wang and T. Oates, "Encoding time series as images for visual inspection and classification using tiled convolutional neural networks," in *Proc. 29th AAAI Conf. Artif. Intell. Workshops*, vol. 1, 2015, pp. 1–7.

[37] D. Mishkin and J. Matas. (2015). "All you need is a good init." [Online]. Available: https://arxiv.org/abs/1511.06422

[38] S. Ioffe and C. Szegedy. (2015). "Batch normalization: accelerating deep network training by reducing internal covariate shift." [Online]. Available: https://arxiv.org/abs/1502.03167

[39] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, 2014.

[40] D. P. Kingma and J. Ba. (2014). "Adam: A method for stochastic optimization." [Online]. Available: https://arxiv.org/abs/1412.6980

[41] S. R. Liyanage, C. Guan, H. Zhang, K. K. Ang, J.-X. Xu, and T. H. Lee, "Dynamically weighted classification with clustering to tackle non-stationarity in brain computer interfacing," in *Proc. IEEE Int. Joint Conf. Neural Netw.*, Jun. 2012, pp. 1–6.