Learning temporal information for brain-computer interface using convolutional neural networks

Sakhavi, Siavash; Guan, Cuntai; Yan, Shuicheng

2018

Sakhavi, S., Guan, C., & Yan, S. Learning Temporal Information for Brain-Computer Interface Using Convolutional Neural Networks. IEEE Transactions on Neural Networks and Learning Systems, in press.

<https://hdl.handle.net/10356/87699>

<https://doi.org/10.1109/TNNLS.2018.2789927>

© 2018 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works. The published version is available at: [<http://dx.doi.org/10.1109/TNNLS.2018.2789927>].

Downloaded on 26 Apr 2023 23:21:50 SGT

Learning Temporal Information for Brain-Computer Interface using Convolutional Neural Networks

Siavash Sakhavi, *Student Member, IEEE*, Cuntai Guan, *Fellow, IEEE*, and Shuicheng Yan, *Fellow, IEEE*

Abstract—Deep learning methods and architectures have successfully become state-of-the-art classification algorithms in computer vision and natural language processing problems. However, the application of these methods in Brain-Computer Interfaces has been limited. In this paper, we propose a classification framework for classifying motor imagery EEG data by introducing a new representation of the data and utilizing a convolutional neural network architecture. The convolutional neural network has been designed in a way to learn temporal information from the input signals and the optimum parameters of the network are found using cross-validation. By studying the convolutional weights of the trained networks, we gain insight into the temporal characteristics of EEG. Furthermore, our framework outperforms the best classification method in the literature on the BCI competition IV-2a 4-class motor imagery dataset.

Keywords—Machine Learning, Deep Learning, Convolutional Neural Network, Brain-Computer Interface, Signal Processing, Motor Imagery.

I. INTRODUCTION

DEEP LEARNING (DL), as a sub-category of machine learning, is currently considered as the state-of-the-art method in computer vision and natural language processing (NLP) applications. This phenomenon is loosely associated with the revolutionary paper by Hinton [1] which reignited the interest in neural networks- the building block of modern deep architectures. In 2012, a specific neural network architecture named the convolutional neural network (CNN) won the ImageNet competition [2] by utilizing better learning algorithms, faster computational resources and large annotated datasets. This architecture caused many computer vision researchers to regain interest in the topic of neural networks and the design of deep architectures. The concept of hierarchical convolutional data processing (used in CNN) was initially introduced by Fukushima in the neo-cognition framework [3] in 1980 and further developed by LeCun in the LeNet-5 architecture [4] in 1998. Since the 2012 success of CNN, convolutional architectures have become an important tool in vision-related applications and have rapidly been modified and improved. These modifications and improvements can be categorized into modification in network architecture design (Network in

Network [5], Inception [6], Identity Mapping [7]), modification in network activation functions (ELU [8], PReLU [9]), improvements in optimization algorithms (ADAM [10]) or improvements in network regularization (Batch normalization [11], Weight Normalization [12]).

DL and the use of CNNs are not limited to computer vision tasks and have been utilized in other fields such as speech recognition, text understanding, and more recently, Brain-Computer Interfaces (BCI), which is the main focus of this paper.

Motor imagery (MI) BCI systems base their framework on the fact that there is a change of activation in certain areas of the brain when a patient/subject imagines moving any part of their body. For example, when a person imagines moving his/her right arm, there is a desynchronization of neural activity in the the primary motor cortex in the left side of the brain. This desynchronization, which is called event-related desynchronization (ERD) in neuroscience literature, can be seen in the EEG signal as a transition from resting state energy level to a lower energy level. The spatial location, temporal onset, amount of decrease, and stability of the ERD are all subject-dependent factors, which pose a challenge in designing a single framework for detecting changes in neural activity that can be accurate and functional for a wide variety of users.

The neurological phenomenon of motor imagery-based ERD inspired computer scientists and BCI researchers to propose classification methods based on the common spatial patterns (CSP) Algorithm [13]. The CSP algorithm finds a set of linear transformations, i.e. spatial filters, that maximize the distance of the multiple classes (i.e. right hand, left hand, feet) of data recorded during a MI-EEG task. After estimating the spatial filters, the relative energy of the filtered channels is computed and considered as the representation of the data. This representation of the high-dimensional EEG data can be easily fed into a linear classifier, such as SVM, and has had a good performance [14]. However, in this representation, by reducing the signal from a series to a single value, the temporal information is destroyed and the dynamic of the signal is neglected, which may contain valuable information regarding EEG.

The Filter-Bank CSP algorithm (FBCSP) [15], [16] extends the CSP algorithm by considering the fact that not all frequency bands contain discriminative information, and therefore, by passing the signal through a filter-bank, computes the CSP energy features for each of the temporally filtered signal outputs. The features are then selected and classified. The extra step of performing CSP on each filtered input helps boost the classifier performance and shows the benefits of signal decomposition before spatial filter estimation.

Siavash Sakhavi is with the Department of Electrical and Computer Engineering, National University of Singapore (NUS) and Institute for Infocomm Research (I²R) at Agency for Science, Technology and Research (A*STAR) e-mail: (sakhavi@u.nus.edu).

Cuntai Guan is with Institute for Infocomm Research (I²R) at Agency for Science, Technology and Research (A*STAR) and Nanyang Technological University (NTU).

Shuicheng Yan is with the Department of Electrical and Computer Engineering, National University of Singapore.

There have been many other successful attempts to improve the CSP algorithm: Sparse CSP [17] adds a regularization factor on the spatial filter estimation, imposing sparsity on the weights. Stationary CSP [18], divergence-CSP [19] and Probabilistic CSP [20] each try to solve the CSP problem either by changing the objective function or by defining a more generalized computational framework around the problem. Some algorithms simultaneously try to optimize frequency and spatial filters such as [21], [22]. Although they have individually improved the CSP algorithm and increased the classification accuracy, they still contain the same caveat of the original CSP method: the negation of temporal dynamics.

The utilization of DL methods in EEG-Based BCI has been relatively scarce. The high-dimensionality (multi-channel and sampling rate) of EEG data, channel correlation and the presence of artifacts (i.e. movement) and noise pose a challenge in designing the right framework for EEG classification using deep learning. Based on the nature of the data, the framework must include a data preparation stage in which the signal dimensionality is reduced and exploited into a new representation without any significant loss in information. Based on this representation, the next stage of the framework, which is the network architecture, must be designed to extract meaningful features from the input. With the challenges in mind, nevertheless, DL methods have been successfully implemented by authors for EEG classification.

Cecotti introduced a CNN classifier for classification of a P300 (a positive peak seen in the EEG signal 300 ms after presenting a stimulus to the subject) speller task [23]. The CNN in this paper was used in both temporal and spatial manner: a convolution was first performed on the spatial EEG channels and therefore, mixing them and in the next layer, a convolution was performed in time along the temporal samples of the EEG signal. Filtered EEG signals were used as inputs. The author also used a similar architecture in another paper for steady-state visual evoked potentials (SSVEP) [24] and rapid serial visual presentation (RSVP) task [25].

In another work, Stober *et al.* used two representations (raw signal and spectral features) for classifying music imagery EEG signals using CNNs [26]. The results showed the capacity of CNNs to classify imagery-based EEG. In a follow-up paper, Stober *et al.* used a convolutional auto-encoder (CAE) to pre-train a CNN on the same dataset in a unique fashion using cross-trial encoding and similarity-constraint encoding [27]. These two techniques increase the number of samples for the network to learn from and can be used as a solution in problems with low number of data samples in future studies.

In a more recent paper, Bashivan *et al.* introduced a novel representation for EEG Signals by using an image of the the topological map of the the EEG signal's FFT on the scalp, in a specific time interval [28]. By doing so, a sequence of images is generated for the whole EEG trial and then fed into a combined CNN and Long Short-Term Memory (LSTM) for classification.

Hajinoroozi *et al.* have focused on applying DL techniques for driver cognitive performance using Deep Belief Networks (DBN) and Restricted Boltzmann Machine (RBM) on raw representations of the EEG [29], [30].

As for the application of DL for motor imagery EEG (MI-EEG), the number of papers is also very limited. An *et al.* [31] proposed using manually extracted features from the channels based on FFT and then fed them into a Deep Belief Network (DBN). Although this can be considered as a demonstration of DL, it only uses a DBN as a classifier and does not interpret the network as a feature learning algorithm.

In the paper proposed by Yang *et al.* [32], building upon the success of FBCSP, an augmented-CSP (ACSP) algorithm is proposed by using overlapping frequency bands. The log-energy features are extracted for each frequency band and arranged on a two-dimensional matrix. By training a convolutional network on the frequency-energy matrix, the network learns to discriminate the features. Furthermore, a map selection algorithm is proposed to select specific feature maps after the convolution operation. The interpretation of the weights in the network is unknown and the features selected neglect the time dynamics.

Sakhavi *et al.* [33] proposed a parallel MLP and CNN architecture from which the predictions of the networks were joined via averaging. The MLP architecture receives the log-energy features of the FBCSP algorithm and the CNN receives a temporal representation of the selected EEG channels and frequencies from the FBCSP algorithm. The temporal representation is the channel-relative instantaneous energy of the envelope of the EEG signal, extracted using the Hilbert transform. In other words, instead of compressing the signal into a single value, a new temporal representation of the signal is utilized in which the temporal dynamics is intact. The CNN network is then applied in a channel-wise fashion on each class independently and combined before the fully-connected layer, yielding a significant increase in the classification results.

In this paper, we propose a new architecture built upon the idea of our previous paper [33]. By using the FBCSP algorithm as our data preparation method, we propose an novel envelope representation for the MI-EEG. This representation of the data is important in several ways. First, it is a temporal representation of the EEG: it preserves the information relative to the dynamics of the EEG signal throughout recording and thus, is more valuable than a single value energy representation. Second, in terms of information content, it lowers the dimension without distorting the signal, as shown in section III-B. Third, by combining this envelope representation with a CNN, the kernels learned in the network give us insight into the morphology and pattern in the input that gives rise to class discrimination. This representation and the proposed architecture have achieved state-of-the-art performance in the BCI competition IV-2a dataset thereby significantly increasing the average accuracy.

II. DATA

In this study, we have focused on the 2008 BCI competition IV-2a EEG dataset [34]. This is a four-class motor imagery (Left, Right, Feet and Tongue) data recorded from 22 Ag/AgCl electrodes with a 250 Hz sampling rate in two sessions from nine subjects. Each session has 72 trials per class resulting in 288 samples per session. The timing scheme consisted of

a fixation of 2 s, cue time of 1.25 s, followed by period of a motor imagery of 4 s. Previous attempts on classification of the data shows a variety of subject performances based on the accuracy score and Cohen's kappa (Table VII).

III. METHODS

The main process of feature extraction is based on FBCSP. In section III-A, we will review the algorithm and build upon it in section III-B for extracting the temporal features from EEG. For further emphasis, we want to note that our temporal feature extraction method *modifies and borrows* the results from the original FBCSP algorithm.

A. Filter-Bank Common Spatial Patterns (FBCSP)

FBCSP was first introduced as an extension to the original Common Spatial Patterns (CSP) algorithm and gained attention by winning the 2008 BCI Competition IV-2a [16], [15]. CSP is considered as a data-driven spatial filtering algorithm. Spatial filtering algorithms have the objective of de-correlating EEG channels individually or finding a combination of the channels that have valuable information regarding the task at hand. CSP and FBCSP try to find a linear combination of channels (i.e. spatial filter) that discriminates two classes of data. The procedure for FBCSP is as follows:

- 1) The EEG signals from all recorded channels are filtered using a filter bank with nine subsequent bandpass filters, starting at 4 Hz and with a bandwidth of 4 Hz (4–8 Hz, 8–12 Hz, ...). All filters are type II Chebyshev filters.
- 2) Spatial filters for each output of the filter bank are computed using CSP. This is done by maximizing the following objective function:

$$w^* = \arg \max_w \frac{w^T \Sigma_{c_1} w}{w^T (\Sigma_{c_1} + \Sigma_{c_2}) w} \quad (1)$$

Where Σ_{c_1} and Σ_{c_2} correspond to the channel covariance matrix for classes c_1 and c_2 respectively in a specified time segment and w is the spatial filter. This objective function, also called the Rayleigh Quotient, has an analytical solution which is equivalent to solving a Generalized Eigenvalue Decomposition (GEVD) problem.

- 3) Spatial filters corresponding with the $2 \times N_W$ extreme eigenvalues (N_W largest and N_W smallest eigenvalues) are selected. Each of the extreme spatial filters are then paired with each other correspondingly (spatially-filtered channel pairs).
- 4) Energy (variance) of the spatially-filtered channels is calculated (E_C) and normalized to the total energy of the channels in a given frequency band ($\bar{E}_C = \frac{E_C}{\sum_{i=1}^{2 \times N_W} E_C}$). The logarithm of energy is computed as the final feature.
- 5) Features coming from all nine filter bands are concatenated and a mutual information-based feature selection is performed on $2 \times N_W \times 9$ spatially-filtered channels, choosing N_S filtered channels and their pairs. Depending on whether the selected features are already pairs

with each other or not, a maximum of $2 \times N_S$ features may be selected.

- 6) Because CSP is designed for a two-class problem, in the case of multiclass tasks, a one-vs-rest or one-vs-one strategy must be appointed. In FBCSP, the former is chosen and it will lead to a maximum of $classNumber \times 2 \times N_S$ features.

The values $N_W = 2$ and $N_S = 4$ were selected using cross-validation. With the competition data having four classes, the maximum number of features used for classification will be 32. It should be noted that the features can be handled in two ways: concatenation of all features into one large vector, or using features extracted by class-specific spatial features individually. In [16], the latter has been used. Because of the success of FBCSP in classification framework, we decide to build a feature extraction procedure for temporal features utilizing a modified version of the FBCSP algorithm which will be described in the next section.

B. Extracting Temporal Features

Assuming we have performed the FBCSP process as described in III-A, we extract the temporal features from the following procedure:

- 1) After FBCSP, we have indices of the selected filtered channels for each frequency band and each class. We use these indices to extract the corresponding EEG signals. It should be noted that, in contrast with FBCSP, which has a variable feature dimension output, we force the selection algorithm to select $2 \times N_S$ spatially-filtered channel pairs. This ensures that the dimension of the input and therefore, the structure of the designed network is consistent between subjects. Furthermore, we use the whole period of motor imagery (0 s to 4 s) as the time segment to estimate the covariance matrices for the CSP algorithm.
- 2) The signal envelope of each signal is extracted using the Hilbert transform [35]; the Hilbert transform gives the analytic form of a signal which is complex-valued and is interpreted as the one-sided version of the original signal's frequency spectrum. Taking the amplitude of the analytic form gives an estimate of the envelope.
- 3) After extracting the envelope, we have considered three possible representations for the EEG (details regarding how to choose the representations will come in section IV-B):
 - Using the raw (or a smoothed version of) EEG envelope (R1)
 - Taking the power of the envelope, which can be interpreted as instantaneous energy (R2)
 - Dividing the envelope power by the total energy of each of the channels in each trial, similar to step 4 in the FBCSP algorithm described in III-A (R3)
- 4) Due to the spectral nature of the envelope being low-frequency, we can down-sample the signal without information loss. This comes in handy to lower the

dimension of the data, especially when the number of samples is limited. This is a natural benefit of using the FBCSP algorithm: the filter-bank intrinsically reduces the number of samples needed to represent the original data.

- 5) After extracting the envelope signals for each class, instead of taking a one-vs-rest strategy for classification, we concatenate all four classes making a single matrix of signals.

With the same values of $N_W = 2$, $N_S = 4$ and with 4 classes, the number of channels will be 32. As for the dimension of the features in time, the original sampling frequency of the data is 250 Hz and by choosing 4 s interval of data, we will have 1000 samples. Note that the envelope will have a cutoff frequency of 4 Hz, which means a sampling frequency of 8 Hz is sufficient for the signal (Nyquist rate) but we have chosen to reduce the frequency to 10 Hz, yielding 40 time points for the 4 s interval. Overall, the dimension of the data fed into the network will be 32×40 . From here on we will refer to the channel dimension as feature channels.

C. Designing the CNN Architecture

When the convolutional neural network (CNN) LeNet5 was first introduced by LeCun [4], it comprised of a sequence of convolutions and sub-sampling layers mainly constituting of max-pooling. Since then, most architectures follow the same procedure with some additions such as Dropout [36], Batch Normalization [11], Inception [6] and/or Identity Mapping [7]. These additions, based on their nature, can lead to faster training of the network, better conservation of information throughout the hierarchical process, and/or avoid the network to over-fit.

For designing the network, the nature of the input (in our case the envelope representation), should be taken into consideration. Each of the 32 feature channels may be from different frequency bands (based on the feature selection algorithm). In terms of spatial filters, each feature channel has a unique spatial filter (based on the selected eigenvalues in the CSP algorithm) which is designed for discriminating one class against the other classes. Furthermore, it is possible for the spatial filters to be correlated due to one spatial pattern being discriminative for two classes or more. With these facts in mind, we will consider different scenarios and their interpretation of utilizing convolution for our EEG representation.

Scenario 1: Convolution only across time with a common kernel shape for all feature channels. In this type of convolution, the assumption is that feature channels selected for classification, although intrinsically different and independent, share a common morphology. This morphology can be captured from each channel using a common kernel which learns the morphology leading to the discrimination of classes. The choice of such convolutional kernel will result in preserving the channels during the convolutional layers of a network (channel-wise convolution) and reduction of the temporal dimension. After convolution, the fully-connected layers will mix all the channels and temporal values and then classify

them. We will call the network architecture utilizing this convolution “Channel-wise CNN” (CW-CNN).

Scenario 2: Convolution only across channels. This operation can be interpreted as mixing the channel signals with each other. For example, if the size of the convolutional kernel for this layer is the same as the number of channels, the output of the convolution operation is a new signal which is the linear combination of all the given channels. Having a kernel size smaller than the number of channels is not ideal because it implies that a common linear combination can be shared amongst the channels but usually the EEG channels in this stage are independent and their order in the input matrix is not important. We call this scenario “Channel Mixing CNN” (CM-CNN). This type of convolutional layer is better used with CW-CNN because the FBCSP input is already a linear mixture of the original EEG channels and an extra channel mixing is redundant. Only after some processing can a new mixture of the channels make sense.

Scenario 3: Convolution across both time and channels using a two-dimensional kernel. The evident result of this type of convolution, in addition to convolution in time, is the mixing of the feature channels after they have been convoluted. This scenario produces a new time series which captures information from all the channels simultaneously. The receptive field of the kernel in the channel dimension determines which channels should be mixed with each other. For example, if all 32 channels of the input are mixed, the output will be a single channel feature which is the summation of the convolution of all the other feature channels with their own unique filter. Or, in another case, if the feature channels related to the class-specific spatial filters are mixed, the output is a channel summing the information of a class. We will call this type of architecture two-dimensional convolution scenario 2D-CNN, because of its similarity to most two-dimensional CNN architectures

Another way of performing two-dimensional convolution is by breaking up the two-dimensional convolution into two one-dimensional convolutions in two separate layers. This method of implementing the two-dimensional convolution makes the convolution in time and space independent from each other and increases the flexibility of the network but with the cost of increasing the number of parameters due to the introduction of a new computational layer. It can be viewed as adding a channel mixing to the CW-CNN network. Therefore, this architecture will be called Channel-wise Convolution with Channel Mixing (C2CM).

Figure 1 shows illustrations of convolution in time, convolution in channel and two-dimensional convolution in detail.

Amongst all the models, 2D-CNN has the smallest number of parameters followed by C2CM. The main contributor to the number of parameters trained in a CNN is the connection of the last convolutional layer to the fully-connected layers. In all CNN architectures, the last convolutional feature maps are vectorized and stacked into one large vector and fed into fully connected layers. The smaller the dimension of the feature map of the last layer, the lower the number of linear units used. In the case of both 2D-CNN and C2CM, because the feature map size is reduced due to convolution in both dimensions,

the number of parameters is significantly lower compared to CW-CNN. A lower number of parameters is desired especially when the number of training samples is relatively low to avoid over-fitting and allow for better training of the network.

We will be presenting the results for architectures C2CM and CW-CNN in section IV.

Fig. 1. The three types of convolution possible to be implemented on any feature map (from left to right): convolution in time, convolution in channel and 2D convolution. A mapping from multiple squares to one square is a linear combination of the multiple squares into a single value. The number of squares in the output of each convolution corresponds with the actual effect of convolution. Colour shows a independent time-series. The gray values in the output means the channels values are mixed.

D. Parameter Selection via Cross-Validation

For each of the scenarios in section III-C, we must choose the network parameters. Number of layers (convolutional, fully-connected), kernel size, number of hidden nodes, convolution stride, pooling method, regularization methods (Batch Normalization, Dropout) and other network related parameters are considered as hyper-parameters and can be optimized. These hyper-parameters, in addition to the different representations for the EEG brought in section III-B, must be correctly chosen based on cross-validation. Practically, it is not feasible to search through the parameter space due to time and computation limitations. Instead, we use coordinate descent as a sub-optimal method to perform cross-validation for the network parameters [37]. In coordinate descent, a set of parameters, $\Theta = [\theta_1, \theta_2, \dots, \theta_N]$, are initialized and then the objective function or score function is optimized for each θ_i ($i = 1, \dots, N$) independently, while updating the values of the initial Θ with the newly optimized parameters. After N optimizations, the Θ vector will be completely updated and a new iteration of optimization can initiate. For better results, the algorithm can be repeated for several iterations.

For the current paper, we have chosen two values to be selected via cross-validation: the size of the kernel and the number of convolutional nodes. Table I shows the values chosen for each of the two values plus the other parameters' initialization values. Multiple values in curly brackets show the

TABLE II. SAMPLE ARCHITECTURE

Layer Type	Patch size / Stride	Input Size	Hidden Unit	Parameters
Convolution	$1 \times 7 / 1 \times 3$	32×40	32	256
ReLU	-	32×12	-	0
Convolution	$1 \times 3 / 1 \times 3$	32×12	32	3104
ReLU	-	32×4	-	0
Linear	-	4096	512	2,097,664
ReLU	-	512	-	0
Linear	-	512	4	2052
LogSoftMax	-	4	4	0

number of layers used during cross validation. For convolution kernels, the values in the bracket are as follows: kernel width, kernel height, stride in width and stride in height. For example, $\{\{4, 1, 3, 1\}, \{3, 1, 2, 1\}\}$ indicates that the convolution has two layers with the first layer having a kernel size of 1×4 with a stride of 3 and the second layer having a kernel size of 1×3 with a stride of 2. We have performed a 10-fold cross-validation only once in order to select the parameters. The convolutional layer parameters (*convParams*) are first selected using cross-validation and then the selected values are used for cross-validation for selecting the number of convolutional nodes (*hidNodes*). For C2CM structure which has an additional computational layer, every value is the same as the CW-CNN structure. The channel mixing layer is positioned after the channel-wise convolutions and has the same number of hidden nodes as the previous convolutional layers.

The kernel size and stride combination for the first and second layer are chosen based on the input size and in a way that the output of the convolutional neural network is a integer value. It should be noted that in conventional CNN architecture used in computer vision, the kernel size is high for the first layer but is reduced in the subsequent layers. In order to find the rule of the kernel, we gave different values for the first layer from a size 4 corresponding to a interval of 400 ms to a size of 20 which corresponds with a interval of 2 s. Furthermore, In contrast with conventional architectures, we have chosen to not use any sub-sampling method and instead, rely solely on changing the stride.

The training of the networks was carried out with the following configurations:

- ADAM [10] is used as the optimization method. The parameters are set to default values mentioned in the original paper [10].
- Negative log-likelihood has been considered as the optimization criterion.
- In all layers, we have inserted a batch normalization [11] layer before the activation layer and a dropout layer after the activation with a probability of 50%.

As an example, considering the input size calculated in section III-B, Table II shows the input size at each layer for a

TABLE I. CROSS VALIDATION PARAMETERS

Cross Validation Parameter	Initial value for other parameters	Cross Validation Values
Convolutional kernel sizes (convParams)	MLP Nodes = none, Conv. Nodes = {32, 32}	$\{\{4, 1, 3, 1\}, \{3, 1, 2, 1\}\}, \{\{7, 1, 3, 1\}, \{3, 1, 3, 1\}\}, \{\{8, 1, 2, 1\}, \{5, 1, 3, 1\}\}, \{\{10, 1, 3, 1\}, \{3, 1, 2, 1\}\}, \{\{10, 1, 2, 1\}, \{4, 1, 2, 1\}\}, \{\{16, 1, 3, 1\}, \{3, 1, 2, 1\}\}, \{\{20, 1, 2, 1\}, \{5, 1, 2, 1\}\}$
Number of convolutional nodes (hidNodes)	Chosen values for convParams, MLP Nodes = none	$\{8, 8\}, \{16, 16\}, \{32, 32\}, \{64, 64\}, \{128, 128\}, \{256, 256\}$

Fig. 2. Visualization of Sample Architecture CW-CNN (above) and C2CM (below). After the convolution layer, the feature maps are flattened into a single vector and fed to the fully-connected network. It can be seen that the more the convolutional layers, the lower the dimension of the last convolution operation and therefore, less parameters needed to connect to the fully-connected layers.

sample two-layer CW-CNN architecture, while figure 2 shows a sample visualization of both CW-CNN and C2CM networks.

IV. RESULTS

A. Baseline Method

In our study, we have used both Cohen's kappa and accuracy to evaluate our method. The FBCSP feature extraction algorithm in combination with a linear C -SVM classifier has been used as the baseline. Features are extracted from the 0.5 s to 2.5 s after the cue for both train and test datasets. Kappa values from the reported in the original FBCSP paper are used. We have also included results from a paper from Bashashati [38] which has used Bayesian optimization to find the best parameters for FBCSP and are the best results we could find on the BCI competition IV-2a dataset in terms of accuracy. In terms of kappa, we have used the values in [39] which has shown to be highest value of kappa amongst many methods

(accuracy was not reported in the paper). The baseline results can be seen in table VII.

In order to verify the significance in increase/decrease of the accuracies, a one-sided Wilcoxin signed-rank test [40] is used. This test is appropriate in conditions where the number of paired samples to compare is relatively small and non-Gaussian.

B. EEG representation & Architecture Comparison

In order to select one of the three EEG representations described in section III-B, we use a simple architecture with a set of chosen parameters and perform a 10-fold cross-validation over each of the representations. The architecture is similar to the architecture in table II, having two convolutional layers (32 nodes each) without the fully-connected layer. The average cross-validation values are obtained by repeating the measurements for 10 networks (10 networks $\times$ 10 folds). Our assumption is that with a common architecture, the better

TABLE III. CROSS-VALIDATION ACCURACY RESULTS FOR FEATURE REPRESENTATION GIVEN THE C2CNN ARCHITECTURE IN TABLE II.

Feature	Classifier	Subject 1	Subject 2	Subject 3	Subject 4	Subject 5	Subject 6	Subject 7	Subject 8	Subject 9	Average
Envelope (R1)	CNN	85.23	69.73	90.15	65.57	77.42	52.41	93.68	90.04	84.75	
Instantaneous Energy (R2)		83.38	69.42	87.38	65.27	75.43	47.71	92.38	87.45	82.40	76.76
Relative Instantaneous Energy (R3)		85.04	65.67	87.07	64.28	73.72	48.59	91.97	87.80	83.55	76.41

TABLE IV. CROSS-VALIDATION ACCURACY RESULTS FOR CLASSIFIERS GIVEN THE R1 FEATURE.

Feature	Classifier	Parameters	Subject 1	Subject 2	Subject 3	Subject 4	Subject 5	Subject 6	Subject 7	Subject 8	Subject 9	Average
Envelope (R1)	CNN	19748	85.23	69.73	90.15	65.57	77.42	52.41	93.68	90.04	84.75	78.78
	MLP	42180	85.20	68.16	91.04	60.86	73.68	52.34	90.20	88.69	85.83	77.33
	SVM	5120	82.36	69.52	90.84	62.09	74.32	52.82	91.91	86.57	83.89	77.15

TABLE V. ACCURACIES USING THE SELECTED PARAMETERS FROM CROSS VALIDATION FOR CW-CNN

	Subject 1	Subject 2	Subject 3	Subject 4	Subject 5	Subject 6	Subject 7	Subject 8	Subject 9	Average
	86.11	60.76	86.81	67.36	62.50	45.14	90.63	81.25	77.08	73.07
Test Results (50 Ensemble)										
Selected Kernels Size	{{20,1,2,1}, {5,1,2,1}}	{{7,1,3,1}, {3,1,3,1}}	{{20,1,2,1}, {5,1,2,1}}	{{20,1,2,1}, {5,1,2,1}}	{{10,1,2,1}, {4,1,2,1}}	{{20,1,2,1}, {5,1,2,1}}	{{20,1,2,1}, {5,1,2,1}}	{{7,1,3,1}, {3,1,3,1}}	{{20,1,2,1}, {5,1,2,1}}	
Selected Hidden Nodes	{64,64}	{32,32}	{32,32}	{32,32}	{32,32}	{8,8}	{8,8}	{32,32}	{32,32}	

TABLE VI. ACCURACIES USING THE SELECTED PARAMETERS FROM CROSS-VALIDATION FOR C2CM

	Subject 1	Subject 2	Subject 3	Subject 4	Subject 5	Subject 6	Subject 7	Subject 8	Subject 9	Average
	87.50	65.28	90.28	66.67	62.50	45.49	89.58	83.33	79.51	74.46
Test Results (50 Ensemble)										
Selected Kernels Size	{{10,1,2,1}, {4,1,2,1}, {1,32,1,1}}	{{4,1,3,1}, {3,1,2,1}, {1,32,1,1}}	{{10,1,2,1}, {4,1,2,1}, {1,32,1,1}}	{{20,1,2,1}, {5,1,2,1}, {1,32,1,1}}	{{20,1,2,1}, {5,1,2,1}, {1,32,1,1}}	{{20,1,2,1}, {5,1,2,1}, {1,32,1,1}}	{{4,1,3,1}, {3,1,2,1}, {1,32,1,1}}	{{4,1,3,1}, {3,1,2,1}, {1,32,1,1}}	{{10,1,3,1}, {3,1,2,1}, {1,32,1,1}}	
Selected Hidden Nodes	{256,256}	{32,32}	{256,256}	{256,256}	{128,128}	{32,32}	{256,256}	{8,8}	{256,256}	

representation would have a higher cross-validation average. The results can be seen in table III. The table shows that the CNN architecture is selecting the R1 representation for all subjects and the cross-validation average accuracy is higher than the other two representations.

For classifier comparison, we perform cross-validation of the given representation on two additional classifiers: linear SVM and MLP (two hidden layers with 32 hidden nodes, 10 networks). The cross-validation results can be seen in table IV. The table shows that, on average, most subjects have a higher cross-validation accuracy for the CNN architecture, resulting in the CNN to be the selected architecture.

C. Architecture Parameter Selection

With the chosen representation, we conduct cross-validation over the network parameters based on the values in table I. Parameters are selected based on the averaging over the results of each fold, more multiple network initializations. Here, a 10-fold cross-validation is performed on 10 network initializations. The average accuracy of these 100 networks is used for classification. The selected parameters for architectures CW-CNN and C2CM can be seen in tables V and VI respectively. The test accuracy reported in both these tables is obtained by averaging and ensemble of 50 model initializations trained on the training data.

The tables show interesting results. In the case of the CW-CNN architecture, most of the subjects select a larger kernel size with smaller number of hidden nodes whereas for the C2CM architecture most subjects are selecting lower kernel sizes but with higher hidden nodes, with the channel mixing layer being the only difference between the two architectures. This means that the channel mixing leads on to making the

network wider and thereby increasing the number of features in the output of the convolution. In contrast, without the channel mixing, the network gives more emphasis on the receptive field of the networks rather than widening the network.

Table VII shows the results for multiple classification methods including our proposed method using C2CM. The values in paranthesis are Cohen's kappa. As shown in the table, our method has a superior performance in both kappa and accuracy. The first "FBCSP" is using the SVM classifier on the FBCSP features which we have calculated. The second "FBCSP" is based on the FBCSP algorithm results in [38] and "BO" is the Bayesian optimization method proposed in the same paper. "SVM" is the use of SVM on the envelope representation features proposed in III-B. Based on the Wilcoxon signed-rank test, the increase in the mean accuracy for C2CM is significantly higher with a bound of $p < 0.05$ relative to BO, SVM and CW-CNN. In terms of average kappa, are method is not significantly higher than the method in paper [39] based on the Wilcoxon test due to the ranking of the differences between the kappa values of subjects 4 and 6. But overall, there is increase in kappa for the other subjects. For other methods, the increase in mean kappa is significant with a bound of $p < 0.05$.

V. ANALYSIS

To further understand the learned network parameters and whether the results obtained by the trained network is by random or the network is actually learning important patterns in the data, we have conducted some experiments.

A. Importance of the Kernel Morphology

As shown in the results of section IV-C, each subject has chosen a specific kernel length and certain number of hidden

TABLE VII. TABLE OF ACCURACY AND KAPPA FOR BASELINE METHODS AND OUR METHOD. VALUES IN THE PARENTHESIS ARE KAPPA VALUES AND THE NON-PARENTHESIS ARE ACCURACY.

	FBCSP [16]	FBCSP [38]	BO [38]	TSSM+SVM [39]	SVM	CW-CNN	C2CM
Subject 1	(0.676)	76.00	82.12	(0.77)	82.29 (0.764)	86.11 (0.815)	87.50 (0.833)
Subject 2	(0.417)	56.5	44.86	(0.33)	60.42 (0.472)	60.76 (0.477)	65.28 (0.537)
Subject 3	(0.745)	81.25	86.60	(0.77)	82.99 (0.773)	86.81 (0.824)	90.28 (0.870)
Subject 4	(0.481)	61	66.28	(0.51)	72.57 (0.634)	67.36 (0.565)	66.67 (0.556)
Subject 5	(0.398)	55	48.72	(0.35)	60.07 (0.468)	62.50 (0.500)	62.50 (0.500)
Subject 6	(0.273)	45.25	53.30	(0.36)	44.10 (0.255)	45.14 (0.269)	45.49 (0.273)
Subject 7	(0.773)	82.75	72.64	(0.71)	86.11 (0.815)	90.63 (0.875)	89.58 (0.861)
Subject 8	(0.755)	81.25	82.33	(0.72)	77.08 (0.694)	81.25 (0.750)	83.33 (0.778)
Subject 9	(0.606)	70.75	76.35	(0.83)	75.00 (0.667)	77.08 (0.694)	79.51 (0.727)
Average	(0.569)	67.75	68.13	(0.593)	71.18 (0.616)	73.07 (0.641)	74.46 (0.659)

nodes. It should be noted that the value for the size found for each subject is based on cross-validation and ensembles: this means that this kernel size is dependent on the data seen during training and also the initializations of the network for each model. Therefore, it is difficult to determine whether this size is meaningful from a neuroscientific aspect or not, especially when the underlying differences between people with different performances is a topic of research [41]. What we are seeking to find out in this section is whether the kernels' learned shape is truly important or the shape is just random and doesn't contribute to the classification.

For this, we modify the convolutional kernels such that each kernel is replaced by the mean of that kernel. The selection of kernel mean is mainly because we do not want to abrupt the scales of the network by changing the values of the kernel to very high values. After changing the values, we perform statistical analysis on the accuracy values with the following hypothesis: *The mean accuracy obtained by the modified network lower than the original accuracy.* It is expected that the classification results will drop but our hypothesis emphasizes whether this drop is significant or not. This hypothesis is one-sided and therefore, when using the Wilcoxin signed-rank test we use the one-sided test values for measuring the significance.

TABLE VIII. MODIFICATION OF KERNELS. L1 & L2 REFER TO THE KERNELS OF LAYER 1 AND 2. EACH COLUMN SHOWS THE TEST ACCURACY WHEN THE KERNEL OF THOSE LAYERS(S) IS MODIFIED TO BE THE MEAN VALUE OF THE KERNEL.

Subject	Trained CNN	Mod. (L1)	Mod. (L2)	Mod. (L1+L2)
Subject 1	87.50	82.99	82.29	82.64
Subject 2	65.28	60.42	61.11	57.99
Subject 3	90.28	84.38	85.42	83.33
Subject 4	66.67	58.33	61.81	58.33
Subject 5	62.50	54.17	58.33	53.47
Subject 6	45.49	48.26	43.40	48.26
Subject 7	89.58	90.63	90.28	92.01
Subject 8	83.33	84.03	80.21	82.29
Subject 9	79.51	61.11	60.76	59.38
Average	74.46	69.37	69.29	68.63

While performing this analysis, only one set of values are changed so the comparison is meaningful. This means, for example, if the kernel values of the first layer is changed to the mean value, the rest of the network values are. Table VIII shows the accuracy values for each individual subject after the

network modification from an ensemble of 50 networks. The last column of this table is the modification for both layers simultaneously. The W-score calculated between the original network and change of the first layer parameters is 6 and based on the hypothesis the maximum value at which the hypothesis is significant with a p-value of 0.05 is 8, showing that the first layer modified network is performing worse than the original network. Modification of both layer simultaneously shows an even larger decrease relative to the original network, showing that the the morphology captured by the kernels is important.

B. Qualitative Analysis of Kernel Shapes

After training the network, we want to gain understanding about what the network has learned and if possible, obtain a visualization. In image processing, methods such as deconvolution [42], back-propagation-based visualization [43], [44], and layer-wise relevance propagation (LRP) [45] have been used for understanding the network with the latter recently being used for EEG analysis as well [46].

Here we have chosen back-propagation methods proposed in Yosinski's paper [43]. In this group of methods, an initial image is fed to the network and based on the desired sensitivity of any node in the network, the initial image is recursively changed in order for a desired level of sensitivity to be achieved at the specific node. This recursive algorithm can be accompanied by a smoothing function to remove high frequency noise during the recursion.

For example, in our case, we have given the average of all training examples, regardless of class, as a initial value to the network and want to output label vector correspond to class *C*. We then change the input based on the update rule iteratively. The result is an input that when fed to the network, results in class *C*. In other words, this perceived input is what the network recognizes as class *C* and if any other input is similar to this perception of the network, it is also classified as class *C*. We have used a Gaussian kernel as our smoothing function.

Figure 3 shows a sample of the average signals for class 1 of subject 9 (blue solid) and the perceived signals for class 1 for a number channels that have high significant correlation. This reconstruction is made based on the network that has the highest classification accuracy on the test data. By visually inspecting the graphs, it can be seen that in most cases, the perceived input signal follows the average signal. It should

Fig. 3. Sample signal visualization for network perceived input for significantly correlated channels from Subject 9 and class 1. The blue (solid) signal is the average signal and the red (dash) signal is the perceived signal from the network. Only signals with significant high positive correlation ($p < 0.05$, $r > 0.5$) have been shown. The correlation and p-value have been written in the image. The x -axis represents time and y -axis is amplitude. The y -axis has been adjusted for each subplot for better visualization.

Fig. 4. Correlation between perceived signal and average signal in class 1 for subject 9. The vertical dashed lines show the channels belonging to the one-versus-rest CSP channels for each class. Only high positive correlations ($r > 0.5$) and significant correlations ($p < 0.05$) have been shown.

be noted that the algorithm converges and the perceived input correlates with each of the classes.

After obtaining the perceived inputs for all classes, the question arises regarding whether this perception is meaningful and whether it corresponds with the data. Therefore, in addition to visual inspection, we quantify the correlation by calculating the Pearson correlation between each of the channels of the

perceived input and the mean of the data for each class. Using the p-value, we determine whether this correlation is significant or not.

Figure 4 shows the correlation coefficient for subject 9 for positive values and also those that have a p-value of lower than 0.05. The vertical lines correspond to each of the one-versus-rest classes in the CSP algorithm. It can be seen in the figure that, for this particular subject, there are high correlation between most of the classes in some particular channels. This means that not all channels are important in contributing to the classification output. Figure 5 shows the signals sorted based on the p-value for the original average channel (left) and perceived input (right) based on the p-value arranged vertically from class 1 to class 4, validating that there is similarity between the original data and the perceived input.

VI. DISCUSSION & FUTURE WORK

Deep neural network architectures and deep learning algorithms have opened a door to many applications which were not possible before due to their compositional structure [47], expressivity [48], and their ability to be parallelized for speeding up the learning process. The results in this paper have identified two important facts regarding the application of deep learning in EEG. First, the representation of the signal when fed into the deep learning framework is important. Previous methods which were solely based on energy values neglected valuable temporal information. When considering the new representation, the way the information processed must be different as well because of the time element introduced by the temporal information. In other words, new representations need new processing methods. This need for new processing

Fig. 5. Sample visualization for average class signals and network perceived input for all channels sorted based on p-value for subject 9. *Left column*: average input for each class. *Middle column*: perceived input for each class. *Right column*: correlation p-value between average and perceived classes.

and classification tools results in the second important fact: deep learning methods and building blocks can be used in the context of EEG signal classification and can yield superior results relative to other methods such as SVM and MLP. These results are not because of random matrix multiplications but rather, the network is learning something from the input EEG data; Proving that the architecture is meaningful for EEG. Visualizing the architecture also verifies that the network has learned important relationships from the data and is able to construct a perceived input which is similar to the original data.

One of the main caveats of the current framework is that the data preparation algorithm or in other words the extraction of the temporal feature is independent of the classification stage. The FBCSP algorithm has been designed for the classification of energy features and the extraction of the spatial filter weights is based on the channel covariance matrix, also based on signal energy. Weights extracted based on temporal information, rather than signal energy, can be a better solution when using temporal features.

Furthermore, the training of the networks is time consuming. In BCI systems, it is desired to reduce the calibration time, i.e. the time that is necessary to record sufficient data and train a model. In the models that we have trained for this paper,

training time can vary from 30s to 150s per network for the C2CM model and 9s to 12s per network for the C2CN. Which will become high when we consider training an ensemble and even higher when we need to perform cross-validation. It is important to note that using C2CNN is much faster relative to C2CM because it has one less layer but a much larger number of parameters due to the fully connected layers.

One challenge of EEG signal classification was not answered in this paper: how can a small number of samples for one class be used to train high capacity networks without having an over-fitting issue. Although methods such as dropout and batch normalization both help avoid over-fitting problems, the small number of samples pushes us into using ensemble methods. It is possible to pre-train the network in a smarter way which could lead to a better training of the networks and better understanding of the input data after analyzing the trained networks. Auto-encoders can potentially be used for pre-training but the strategy of training these types of networks is yet to be explored for the context of MI-EEG data.

Learning a generalized network from a large number of subjects is one of the goals that should be sought when using deep learning for BCI. By defining a good representation for all subjects, a deep neural network will be able to learn from all subjects and use the information for new subjects without needing to be re-trained (zero-shot learning) or by training on a small sample from the new data (transfer learning).

Overall, the application of deep learning methods in EEG processing is promising and the method proposed above can be generalized to medical applications and/or different types of EEG recordings in the future.

ACKNOWLEDGMENTS

The authors would like to thank all members of the BCI lab at A*STAR and the Learning and vision (LV) lab at NUS for their discussions and inspiration. Furthermore, the maintenance and development of Torch on Github made the implementations of the algorithms much easier and faster.

REFERENCES

- [1] G. Hinton and R. Salakhutdinov, "Reducing the Dimensionality of Data with Neural Networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.
- [2] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," in *Advances in Neural Information Processing Systems 25*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2012, pp. 1097–1105.
- [3] K. Fukushima, "Neocognitron: A self-organizing neural network for a mechanism of pattern recognition unaffected by shift in position," *Biological Cybernetics*, vol. 36, no. 4, pp. 193–202, 1980.
- [4] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-Based Learning Applied to Document Recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.
- [5] M. Lin, Q. Chen, and S. Yan, "Network In Network," *arXiv preprint arXiv:1312.4400*, 2013.
- [6] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going Deeper with Convolutions," Google, Tech. Rep. arXiv:1409.4842 [cs.CV], 2014.
- [7] K. He, X. Zhang, S. Ren, and J. Sun, "Identity Mappings in Deep Residual Networks," mar 2016. [Online]. Available: <http://arxiv.org/abs/1603.05027>

- [8] D.-A. Clevert, T. Unterthiner, and S. Hochreiter, "Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs)," nov 2015. [Online]. Available: <http://arxiv.org/abs/1511.07289>
- [9] K. He, X. Zhang, S. Ren, and J. Sun, "Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification," feb 2015. [Online]. Available: <http://arxiv.org/abs/1502.01852>
- [10] D. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," dec 2014. [Online]. Available: <http://arxiv.org/abs/1412.6980>
- [11] S. Ioffe and C. Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift," in *Proceedings of The 32nd International Conference on Machine Learning*, 2015, pp. 448–456.
- [12] T. Salimans and D. P. Kingma, "Weight Normalization: A Simple Reparameterization to Accelerate Training of Deep Neural Networks," feb 2016. [Online]. Available: <http://arxiv.org/abs/1602.07868>
- [13] H. Ramoser, J. Muller-Gerking, and G. Pfurtscheller, "Optimal spatial filtering of single trial EEG during imagined hand movement," *Rehab. Eng., IEEE Trans. on*, vol. 8, no. 4, pp. 441–446, 2000.
- [14] F. Lotte and C. Guan, "Regularizing common spatial patterns to improve BCI designs: unified theory and new algorithms," *Biomedical Engineering, IEEE Trans. on*, vol. 58, no. 2, pp. 355–362, 2011.
- [15] K. K. Kai Keng Ang, Z. Y. Zhang Yang Chin, H. Haihong Zhang, and C. Cuntai Guan, "Filter Bank Common Spatial Pattern (FBCSP) in Brain-Computer Interface," in *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*. IEEE, jun 2008, pp. 2390–2397.
- [16] 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," *Front. in neurosci.*, vol. 6, 2012.
- [17] M. Arvaneh, C. Guan, K. K. Ang, and C. Quek, "Optimizing the channel selection and classification accuracy in EEG-based BCI," *Biomedical Engineering, IEEE Trans. on*, vol. 58, no. 6, pp. 1865–73, Jun. 2011.
- [18] W. Samek, C. Vidaurre, K.-R. Müller, and M. Kawanabe, "Stationary common spatial patterns for brain-computer interfacing," *Journal of Neural Engineering*, vol. 9, no. 2, p. 26013, 2012.
- [19] W. Samek, M. Kawanabe, and K.-R. Müller, "Divergence-based framework for common spatial patterns algorithms," *Biomedical Engineering, IEEE Reviews in*, vol. 7, pp. 50–72, 2014.
- [20] W. Wu, Z. Chen, X. Gao, Y. Li, E. Brown, and S. Gao, "Probabilistic common spatial patterns for multichannel eeg analysis," *Pattern Analysis and Machine Intelligence, IEEE Trans. on*, vol. 37, no. 3, pp. 639–653, March 2015.
- [21] W. Wu, X. Gao, B. Hong, and S. Gao, "Classifying single-trial EEG during motor imagery by iterative spatio-spectral patterns learning (ISSPL)," *Biomedical Engineering, IEEE Trans. on*, vol. 55, no. 6, pp. 1733–1743, jun 2008.
- [22] F. Qi, Y. Li, and W. Wu, "RSTFC: A novel algorithm for spatio-temporal filtering and classification of single-trial EEG," *Neural Networks and Learning Systems, IEEE Tran. on*, vol. 26, no. 12, pp. 3070–3082, dec 2015.
- [23] H. Cecotti and A. Gräser, "Convolutional neural networks for P300 detection with application to brain-computer interfaces," *Pattern Analysis and Machine Intelligence, IEEE Trans. on*, vol. 33, no. 3, pp. 433–45, mar 2011.
- [24] H. Cecotti, "A time–frequency convolutional neural network for the offline classification of steady-state visual evoked potential responses," *Pattern Recognition Letters*, vol. 32, no. 8, pp. 1145–1153, 2011.
- [25] H. Cecotti, M. P. Eckstein, and B. Giesbrecht, "Single-trial classification of event-related potentials in rapid serial visual presentation tasks using supervised spatial filtering," *Neural Networks and Learning Systems, IEEE Transactions on*, vol. 25, no. 11, pp. 2030–2042, 2014.
- [26] S. Stober, D. J. Cameron, and J. A. Grahn, "Using Convolutional Neural Networks to Recognize Rhythm Stimuli from Electroencephalography Recordings," in *Advances in Neural Information Processing Systems*, 2014, pp. 1449–1457.
- [27] S. Stober, A. Sternin, A. M. Owen, and J. A. Grahn, "Deep Feature Learning for EEG Recordings," *Arxiv*, nov 2015. [Online]. Available: <http://arxiv.org/abs/1511.04306>
- [28] P. Bashivan, I. Rish, M. Yeasin, and N. Codella, "Learning Representations from EEG with Deep Recurrent-Convolutional Neural Networks," pp. 1–15, nov 2015. [Online]. Available: <http://arxiv.org/abs/1511.06448>
- [29] M. Hajinoroozi, Z. Mao, and Y. Huang, "Prediction of driver's drowsy and alert states from EEG signals with deep learning," in *2015 IEEE 6th International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP)*. IEEE, dec 2015, pp. 493–496.
- [30] M. Hajinoroozi, Z. Mao, T.-P. Jung, C.-T. Lin, and Y. Huang, "EEG-based prediction of driver's cognitive performance by deep convolutional neural network," *Signal Processing: Image Communication*, may 2016.
- [31] X. An, D. Kuang, X. Guo, Y. Zhao, and L. He, "A Deep Learning Method for Classification of EEG Data Based on Motor Imagery," in *Intelligent Computing in Bioinformatics*. Springer, 2014, pp. 203–210.
- [32] Huijuan Yang, S. Sakhavi, Kai Keng Ang, and Cuntai Guan, "On the use of convolutional neural networks and augmented CSP features for multi-class motor imagery of EEG signals classification," in *2015 37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*. IEEE, aug 2015, pp. 2620–2623.
- [33] S. Sakhavi, C. Guan, and S. Yan, "Parallel convolutional-linear neural network for motor imagery classification," in *2015 23rd European Signal Processing Conference (EUSIPCO)*. IEEE, aug 2015, pp. 2736–2740.
- [34] M. Tangermann, K.-R. Müller, A. Aertsen, N. Birbaumer, C. Braun, C. Brunner, R. Leeb, C. Mehring, K. J. Miller, G. R. Müller-Putz, G. Nolte, G. Pfurtscheller, H. Preissl, G. Schalk, A. Schlögl, C. Vidaurre, S. Waldert, and B. Blankertz, "Review of the BCI Competition IV," *Frontiers in Neuroscience*, vol. 6, p. 55, 2012.
- [35] M. Le Van Quyen, J. Foucher, J.-P. Lachaux, E. Rodriguez, A. Lutz, J. Martinerie, and F. J. Varela, "Comparison of Hilbert transform and wavelet methods for the analysis of neuronal synchrony," *Journal of Neuroscience Methods*, vol. 111, no. 2, pp. 83–98, sep 2001.
- [36] 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, pp. 1929–1958, 2014.
- [37] S. J. Wright, "Coordinate descent algorithms," *Mathematical Programming*, vol. 151, no. 1, pp. 3–34, mar 2015.
- [38] H. Bashashati, R. K. Ward, and A. Bashashati, "User-customized brain computer interfaces using Bayesian optimization," *Journal of neural engineering*, vol. 13, no. 2, p. 026001, jan 2016.
- [39] X. Xie, Z. L. Yu, H. Lu, Z. Gu, and Y. Li, "Motor Imagery Classification based on Bilinear Sub-Manifold Learning of Symmetric Positive-Definite Matrices," *Neural Systems and Rehabilitation Engineering, IEEE Trans. on*, vol. PP, no. 99, p. 1, 2016.
- [40] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *The Journal of Machine Learning Research*, 2006.
- [41] C. Jeunet, E. Jahanpour, and F. Lotte, "Why standard brain-computer interface (BCI) training protocols should be changed: an experimental study," *Journal of Neural Engineering*, vol. 13, no. 3, p. 036024, jun 2016.
- [42] M. D. Zeiler and R. Fergus, "Visualizing and Understanding Convolutional Networks," NYU, Tech. Rep. arXiv:1311.2901 [cs.CV], 2013.
- [43] J. Yosinski, J. Clune, A. Nguyen, T. Fuchs, and H. Lipson, "Understanding Neural Networks Through Deep Visualization," *International Conference on Machine Learning - Deep Learning Workshop 2015*, p. 12, jun 2015. [Online]. Available: <http://arxiv.org/abs/1506.06579>
- [44] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep Inside Convolutional Networks: Visualising Image Classification Models

and Saliency Maps,” *ICLR*, pp. 1–, dec 2014. [Online]. Available: <http://arxiv.org/abs/1312.6034>

- [45] S. Bach, A. Binder, G. Montavon, F. Klauschen, K.-R. Müller, and W. Samek, “On Pixel-Wise Explanations for Non-Linear Classifier Decisions by Layer-Wise Relevance Propagation,” *PLOS ONE*, vol. 10, no. 7, p. e0130140, jul 2015.
- [46] I. Sturm, S. Bach, W. Samek, and K.-R. Müller, “Interpretable Deep Neural Networks for Single-Trial EEG Classification,” apr 2016. [Online]. Available: <http://arxiv.org/abs/1604.08201>
- [47] H. Mhaskar, Q. Liao, and T. Poggio, “Learning Functions: When Is Deep Better Than Shallow,” mar 2016. [Online]. Available: <http://arxiv.org/abs/1603.00988>
- [48] M. Raghu, B. Poole, J. Kleinberg, S. Ganguli, and J. Sohl-Dickstein, “On the Expressive Power of Deep Neural Networks,” jun 2016. [Online]. Available: <http://arxiv.org/abs/1606.05336>

Siavash Sakhavi received his PhD degree from a joint program between National University of Singapore (NUS) and the Agency for Science, Technology and Research (A*STAR) in 2018 and Master’s degree from Sharif University of Technology in 2012. During his PhD studies, he was working with the researchers at the Institute for Infocomm Research and the Neural & Biomedical Technology Department on Designing and Utilizing Deep Learning Methods for EEG Signal Processing and Brain Computer Interfaces. His research areas are

Biosignal Processing, Biomedical Image Processing, Machine Learning and Deep Learning. His interests are applying new methodologies in machine learning to real-world problems.

Shuicheng Yan is the Vice President and Chief Scientist of Qihoo 360 Technology Co. Ltd., as well as Head of 360 Artificial Intelligence Institute. He is also a tenured Associate Professor at National University of Singapore, and IEEE Fellow, IAPR Fellow and ACM Distinguished Scientist. His research areas include computer vision, machine learning and multimedia analysis, and he has authored/co-authored about 500 high quality technical papers, with Google Scholar citation over 25,000 times and H-index 70.

He is TR Highly Cited Researcher of 2014, 2015 and 2016. Besides, his team received 7 times winner or honorable-mention prizes in 5 years over PASCAL VOC and ILSVRC competitions which are core competitions in the field of computer vision, along with more than 10 times best (student) paper awards and especially a Grand Slam in ACM MM, the top conference in the field of multimedia, including Best Paper Award, Best Student Paper Award and Best Demo Award!

Cuntai Guan (S91M92SM03-F18) received his PhD degree from Southeast University in 1993. He is a Professor in the School of Computer Science and Engineering, Nanyang Technological University, Singapore, since 2016. He is the Co-Director of the Rehabilitation Research Institute of Singapore. He serves as Chief Scientist in Neeuro Pte Ltd. He is a Member of the Singapore National Medical Research Council Open Fund IRG Review Panel. He was the founding head of the Neural & Biomedical Technology Department and a Principal Scientist at

the Institute for Infocomm Research, A*STAR of Singapore. He was the Program Leader of Neuro-Technology at A*STAR. He had served as Chairman of IEEE Engineering in Medicine and Biology Chapter as well as President of the Pattern Recognition and Machine Intelligence Association (PREMIA). His research interests are in the fields of Brain-Computer Interfaces, Neural Engineering, Machine Learning, Data Analytics, and Neuro-technologies. He is an Associate Editor for IEEE Transactions on Biomedical Engineering, IEEE Access, Frontiers in Neuroscience, Brain Computer Interfaces, and Guest-Editor of for the IEEE Computational Intelligence Magazine. He received the Annual BCI Research Award, the IES Prestigious Engineering Achievement Award, Achiever of the Year (Research) Award, Finalist of President Technology Award, and National Infocomm Award. He is a Fellow of IEEE.