Can a Single Model Deep Learning Approach Enhance Classification Accuracy of an EEG-based Brain-Computer Interface?

Sujit Roy*, Student Member, IEEE, Karl McCreadie, Girijesh Prasad, Senior *Member, IEEE*

***Abstract*— Convolutional Neural Network (CNN) has outperformed many traditional linear and polynomial classifiers in various domains. CNN and other deep learning methods have gained attention in the Brain-Computer Interface (BCI) domain also. Here, we investigate a CNN-based model with two different optimizers for reducing error of classification of brain states using EEG motor imagery data. Two different optimizers namely stochastic gradient descent (SGD) and adaptive momentum (Adam) are investigated for increasing classification accuracy using the well-known BCI competition IV 2b dataset. The study was conducted to investigate the feasibility of a single deep learning model for all subjects without compromising on information decoding rate for any of the BCI participants. Using two different models, mean cross-validation accuracy of 80.32% (±2.2) was achieved across participants, which is significantly higher (*p*<0.05) compared to a state-of-the-art deep learning approach.**

I. INTRODUCTION

Accurate pattern recognition in brain-computer interface (BCI) systems is crucial to let participants interact with their environment effectively [1][2]. Several studies related to preprocessing, feature extraction and classification algorithms have been conducted towards enhancing motor imagery (MI) task detection accuracy [3][20]. Popular methods such as common spatial pattern (CSP) [7] [15], principal component analysis (PCA) [11], bandpower [12] and independent component analysis (ICA) [13] have been used extensively for preprocessing and feature extraction. Introduction of deep neural networks and their revolutionary impact in the domains of image recognition, natural language processing (NLP) and speech signal processing have opened up new avenues for the neuroscience community dealing with BCIs. However, their application in the area of electroencephalography (EEG) and magnetoencephalography (MEG) based BCIs is very limited.

Deep learning, as per convention, does not require tailored features to perform optimally in image classification or NLP, but this situation is debatable in the case of electromagnetic brain signals. In an EEG/MEG-based BCI, the number of trials needed for data collection are high. Also, the non-stationary nature of the brain signals leads to inter-trial and inter-session inconsistencies. These signals have very low signal-to-noise ratio (SNR), due to multitude of noise resulting from different muscular or physiological artefacts (such as heart-rate, muscle activity, breathing etc.) and background brain activity. SNR in single-trial EEG and MEG measurements is typically assumed to be < 1 for evoked responses and ≈1 for oscillatory activity, which puts these data in stark contrast to those in traditional applications of deep learning which are typically related to image classification [4]. For a better classification performance, noise suppression is highly recommended. Deep learning if used efficiently can be of great help in analysing EEG and MEG data, as the dimensionality and spatio-temporal resolution are very high due to the large amount of sensors capable of sampling with millisecond accurate temporal resolution.

Deep learning architecture has been used on EEG data to review performance compared to feature-based classification using SVM or LDA. Recently, a study by Zubarev et al. [4] based on an adaptive neural network showed improvement in classification performance. The two different models, i.e. latent factor CNN and vector autoregressive CNN, were compared with linear SVM, RBF-SVM and shallow FBCSP-based CNN model. They also compared with the existing architecture of EEGNet [6] and VGG19 [14]. They concluded that incorporation of prior knowledge about the process generating MEG observations helped in reducing the model complexity substantially while maintaining high accuracy and interpretability. In [5], a short time Fourier transform-based CNN method was evaluated along with stacked auto encoders, which gave them better classification accuracy on the BCI competition IV 2b dataset. The mean classification accuracy obtained using 10-fold cross-validation was 74.8% (±2.3%) for only CNN-based architecture and after combining with stacked auto encoders the mean accuracy across subjects was 77.6% (±2.1%). However, authors did not explain the concept behind the use of stacked auto encoders and it was not clear

*SR is supported by Department for the Economy (DfE), Northern Ireland and UKIERI DST. G.P. is supported by the NIFBM Facility project (1303/101154803) and UKIERI DST Thematic Partnership project (DSTUKIERI- 2016-17-0128).

S.R., K.M. and G.P. are with the Intelligent Systems Research Centre, School of Computing & Intelligent Systems, Ulster University, Derry~Londonderry, N. Ireland, UK. Corresponding author: Sujit Roy: roy-s2@ulster.ac.uk.

where in the model encoding and decoding happened. From the overview, it looked like more of a feed forward neural network. There has been further development of EEGNet [6] which has been used for depth-wise and separable convolutional layers to construct EEG-specific model for classification.

The architectures discussed so far have mainly been generalised for individual subjects and rarely been used for cross-subject transfer learning and the performance drops over different sessions or on different participants. It is a well-known fact that hyper parameters and optimisers play a crucial role in deep learning based classification, but there has been no study on the method of optimiser selection towards training CNN models for better accuracy, whereas hyperparameters are finalised either by grid search or by hit and trial. For effective cross-subject transfer learning, there is a need to find a common CNN architecture trainable across multiple subjects. To this end, a study has been undertaken involving two optimisation methods, i.e. SGD [9] and Adam [10], which have significant effect on CNN learning rate and classification performance.

II. Material and Methods

A. EEG Dataset

To conduct this study, we have used dataset 2b from BCI competition IV [16]. The dataset includes a motor imagery paradigm for left and right hand movement. The dataset includes 3 sessions of training data and 2 sessions of evaluation data. The imagery task is of 4 s, where participants have to imagine specific movement of left/right hand when the cue appears at t = 3 s, as shown in Figure 1.

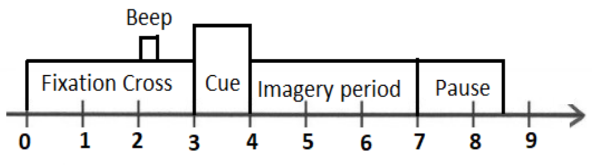

Figure 1. BCI comp IV 2b data set protocol timing

B. Deep Neural Network Designing

CNNs are deep neural networks with several convolutional-pooling layer pairs along with fully connected layer(s) at the output. CNN are good at identifying shapes in the form of images. Input image is convolved with several 2-D filters in the convolutional layer. Weights and biases in the CNN layers are learned through back-propagation algorithm to reduce the classification error. Designing an optimal deep neural network capable of automatically extracting features is still a big challenge. Deep network design is majorly based on intuition along with experience of designer and data type. There are various deep neural network architectures such as Alexnet [18], Deep Belief Networks (DBN) [19], VGG19 [14], etc., which are being used in various domains like image, speech or natural language processing. In this study, two deep network architectures based on CNN have been designed with two different optimisers. This was done with the objective of enhancing classification accuracy while ensuring faster convergence to error tolerance. Faster convergence can help in saving computational time and cost. It is observed in the study that hyperparameters along with optimisers substantially influence the classification performance of trained DNN. Details of the CNN architecture are discussed further.

B.1. Architecture-1

Input Layer: To create the input image for the CNN, a short time Fourier transform (STFT) was applied on the 2-sec-long motor imagery trial (4 s -6 s). The frequency bands considered for the input are theta band (4–8 Hz), mu band (8–12 Hz) and beta band (12–32 Hz). Data from 3 channels were available, i.e. C3, C4, and Cz and sampling frequency was 250 Hz. To create an input image from a channel we extracted theta, mu, and beta frequency bands from the STFT, i.e. broad band (4–32 Hz) EEG.

The extracted image size was 20 × 32 for the theta and mu bands (4–13 Hz) and 41 × 32 for the beta band (13–32 Hz). Using cubic interpolation, size of beta band (41 × 32) was reduced to 20 × 32. The same approach was applied for all the 3 channels. For the final input to the CNN the image was concatenated vertically, hence the resulting input size for one electrode will be 40 × 32 and similarly using all 3 channels it will be 120 × 32 (Figure 2).

Total of 30 filters were trained with size of 120 × 3 via this network. At convolutional layer, input is convolved with trainable filters and put through output function to form the output map [8].

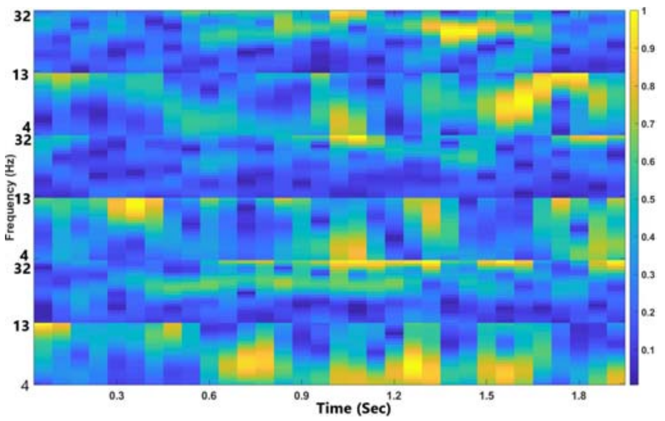

Figure 2. Input image fed to CNN using all three electrodes for right hand activity by subject number 4 including 3 frequency bands.

The *k*th feature map at a given layer can be represented as

$$h^{k}_{ij} = f(a) = f((w^{k} \times x)_{ij} + b_{k}) \quad (1)$$

where x is the input image, w^k is the weight matrix and b_k is the bias value for k= (1, 2,…,30). The output function f is selected as rectified linear unit (Relu) function.

At max pooling layer sampling factor of 10 was applied with zero padding. Max poling is connected to fully connected layer having two outputs, which is for left hand imagery and right hand imagery. Parameters of the CNN are learned by using Stochastic gradient descent method.

Gradient descent is a method to minimize an objective function J(θ) parametrized by model's parameter $\theta \in \mathbb{R}^d$ by updating the parameters in the opposite direction of gradient of the objective function $\nabla_\theta J(\theta)$ with respect to parameters. The learning rate is defined by the number of steps to reach local minimum.

However, at each step, gradient descent requires evaluation of n derivatives, which is expensive. A popular modification is stochastic gradient descent (SGD) [17], where at each iteration t = 1, 2, …

$$w^t = w^{(t-1)} - \eta \nabla \psi\left(w^{(t-1)}\right) \quad (2)$$

where, η is the learning rate and ψ represents the loss function. In a simpler way, learning of the model parameters can be expressed as Eq 3, where parameters perform update for each training example $x^{(i)}$ and label $y^{(i)}$.

$$\theta = \theta - \eta \cdot \nabla\theta\ J\ (\theta;\ x\ (i);\ y(i)) \quad (3)$$

The advantage of SGD is that computation time is 1/n of standard gradient descent, because every step depends upon single derivative $\nabla\psi_i(\cdot)$. Table 1 contains the parameters used to train the model using participants data by Architecture-1 method.

The Architecture-1 (Table 1), has one convolutional 2-D layer with l2 regularisation and ReLU-activation. The details of parameters are shown in Table 1. Batch normalisation was done and the model was trained for 300 epochs with the batch size of 40. For validation, 33% of the data was randomly extracted. The learning rate for the model was 0.0001 and the initial momentum was 0.9. The dropout rate was 0.6.

Table 1. Architecture-1: CNN-SGD Parameters

Layer	Filters	Size	Output	Options
Input		[120, 32, 1]		
Conv2d	30	[120, 3]	[1,30,30]	
MaxPooling2D			[1,3,30]	
Flatten			90	
Dense			2	Activation = sigmoid

B.2. Architecture -2

After implementing Architecture 1 on Comp IV 2b dataset, it was observed that the classification accuracy of participant 2 and 3 was not good. To improve performance of the specified subjects, it was decided to increase the layers. However, even on changing hyperparameters or numbers of layer, there was not much effect on classification accuracy observed. Since the participants' data were noisy compared to other participant, and SGD converges slowly over the data, it was decided to change the optimiser to Adam to converge faster using a large learning rate.

Input layer was kept exactly same as architecture-1 and 30 trainable filters were used with the size of 120×3. At convolutional layer input is convolved with trainable filter with linear activation. Batch normalization was performed keeping the learning rate as 0.01. Batch normalisation reduces the amount by what the hidden unit values shift around (covariance shift). The performance improved but still issue related to convergence was observed. Then, 2nd convolution layer was added with trainable 2-D filters of size 3×3 with a stride of [2, 2]. Total number of filters were 40. Stride controls how the filter convolve around the input volume. Again batch normalisation and Relu activation were used. To further improve the performance and convergence 3rd convolution layer was added keeping the same parameters and stride was reduced to [1,1]. Parameters of CNN are learned by adaptive moment estimation (Adam).

Adam can be explained as a combination of SGD with momentum and Root Mean Square Error Propagation (RMSprop). It is an adaptive learning rate method, where the learning rate is computed from different parameters. Adam keeps exponentially decaying average of past gradients m_t similar to momentum.

Adam uses an exponentially moving average which is computed on the current mini-batch gradient:

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t \quad (4)$$
$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t 2 \quad (5)$$

where m_t and v_t are an estimation of the mean and uncentred variance of gradient (g) and β is a new hyper parameter.

Table 2. Architecture-2: CNN-Adam Parameters

Layer	Filters	Size	Activation	Options
Input		[120,32,1]		
Conv2d	30	[120,3]	Linear	stride = [1,1]
BatchNorm				epsilon= e-5 momentum = 0.01
Activation			Relu	
Conv2d	40	[3,3]	Linear	stride = [2,2]
BatchNorm				epsilon= e-5
Activation			Relu	
Conv2d	40	[3,3]	Linear	stride = [1,1]
BatchNorm				epsilon= e-5 momentum = 0.01
Activation			Relu	
Average pooling2D		[2,2]		
Fully connected				Output size 2

The update rule for Adam is

$$\theta_{t+1} = \theta_t - \frac{n}{\sqrt{\hat{v}_t + \varepsilon}} \widehat{m}_t \quad (6)$$

where θ is the model parameter, $\theta \in \mathbb{R}^d$.

The proposed default values were 0.9 for β1, 0.999 for β2, and 10^{-8} for ϵ [10]. It was shown empirically that Adam works well in practice and compares favourably to other adaptive learning-method algorithms.

The Architecture-2 (Table 2), which is the Adam-based CNN, has 3 convolutional 2-D layers with l2 regularisation. The input remains the same and details of the parameter are shown in Table 2. Initial learning rate was 0.01 and batch size was 50 and the model was trained for 15 epochs.

III. Results

Python with MNE and MATLAB 2018b were used for creating scripts for building learning models and evaluating their performance. The system had Windows 10 with an i7 8th gen processor. Nvidia RTX2080 Ti was used as GPU.

For building models, first 3 sessions from the BCI competition IV 2b dataset were used and ten-fold cross validation was used for evaluation.

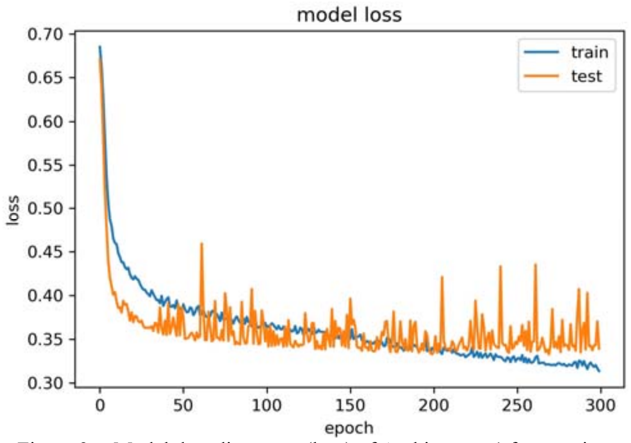

Figure 3. Model decoding error (loss) of Architecture 1 for partcipant 4 using 33% as validation set.

Figure 3 displays the model decoding error for the participant 4 using architecture 1. To analyse number of epochs and learning rate for all the participants, data of participant 4, session 1,2 and 3 were trained, keeping 33% of the cumulative data as validation set. The plot clearly shows that the model does not overfit or underfit as the test errors are converging. For this specific subject a clear stagnation of validation loss can be seen at nearly 175-200 epochs. However, the plot clearly indicates fluctuation which may be due to the low amount of data to train and validate. The participants' data was almost linearly separable, so we can see early convergence. For other participant the number of epochs was not exactly same rather was in range of 250-300. Keeping batch size of 40 and training up to 300 epochs, model was not overfitting or there was no issue of overlearning, thus it was decided to train the model up to 300 epochs keeping it same for multiple subjects. However, if independent model is being trained for a subject then the learning can be stopped at 200 epochs, keeping other parameters same.

Figure 4 shows the ten-fold classification accuracy for left hand vs right hand imagery task using architecture 1 and architecture 2. It can be observed that for some participants i.e. S01, S02, S03, S07 and S09 architecture 2 has performed better than architecture 1. But for S04, S05, S06 and S08 architecture 1 has performed better than architecture 2. Hence, it would not be possible to generalize one architecture for all participants without compromising on classification performance.

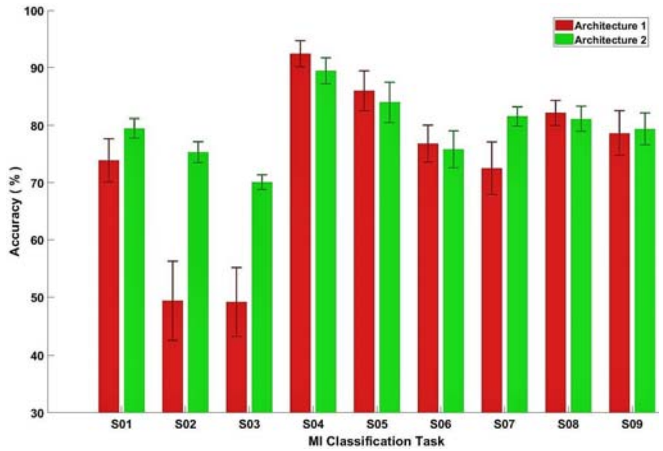

Figure 4. Ten fold Classification accuracy of CNN with SGD as optimiser and adam as optimiser for 9 participants.

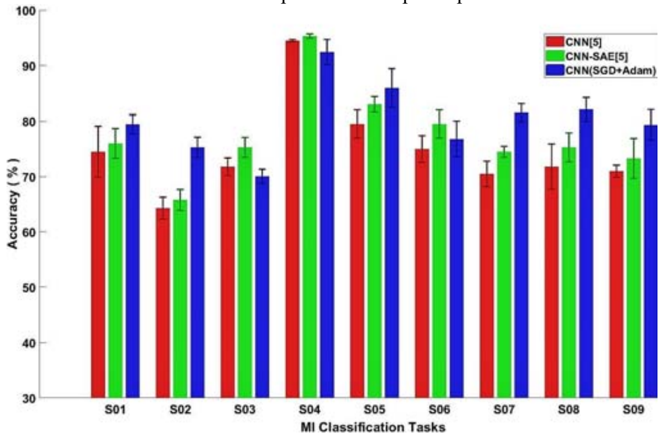

Figure 5. Ten-fold cross-validation classification accuracy and std. deviation results for CNN, CNN-SAE methods published by Tabar et.al. [5] and CNN models based on the Architecture-1 and -2.

The results obtained have been compared with results published by Tabar et al. [5] wherein a different model was created for every participant. Figure 5 shows the classification accuracy of left hand vs. right hand per participant for the BCI comp. IV 2b dataset obtained using 10-fold cross validation. The results of CNN Architecture-1 and -2 are compared with CNN and CNN-SAE architecture accuracies published by Taber et al. [5]. A clear increase in classification accuracy in all participants except s03, s04 and s06 can be seen. However, for s06 architecture-1 provided better result than independent CNN model accuracy published by [5]. Mean classification accuracy across subjects obtained using Architecture-1 and 2 is 80.32% (±2.25), which is 5.56% significantly higher ($p<0.05$) than the CNN model and 2.77% higher than CNN-SAE model proposed by Tabar et al. [5].

IV. Conclusion and Discussion

The paper has discussed two different CNN architectures, found to provide best classification accuracies on the BCI competition IV 2b data-sets from a specific group of subjects.

It has been observed, when a participant's data are noisy, SGD performs worse, which might be due to a low learning rate (10^{-04}) and too many training epochs; while Adam is able to converge quickly in just 15 epochs with less learning rate (10^{-02}) than SGD.

Especially in the case of time-series data, there is no justification to assume that one model is going to fit every subject without the need for adaptation to subject-specific temporal changes. For electromagnetic brain signals, such as EEG and MEG which provide information related to temporal changes at millisecond level, creating a single model is not going to provide enough information for classification.

However, we can divide the participants into categories by single session test on the basis of their BCI performance. If the participant is poorly performing with BCI then Architecture-2 can be used, otherwise Architecture-1 may be better suited. It is not being claimed that this is the best possible accuracy a model can provide; further tuning of parameters may provide with better accuracy. It can be thus concluded from the study that for implementing a single deep learning architecture to train a cross-subject single model, we may have to compromise on accuracy. In order to achieve that model, we will require adaptive optimization by the model which can work with variable time-series data.

Development of single generalised classification model and transfer learning across subjects are major issues for BCI systems to work smoothly. Further study will be carried out by training the model to combine different participants' data. Empirically, if different participants' data can be combined along with the sensor information the input map will become 3D i.e. [frequency, time, location]. Then model would be able to learn the features across subjects from different channels, considering it to be the time series information from a single subject. Also, the tuning of the network will be based on cumulative performance of participants. Then the variation due to new participant's features can be accounted for.

Code for replicating the experiment will be made available at the following link: https://github.com/thesujitroy/Deep-neural-network-Transfer-learning-EEG-MEG-

REFERENCES

[1] Hochberg, L.R., Serruya, M.D., Friehs, G.M., Mukand, J.A., Saleh, M., Caplan, A.H., Branner, A., Chen, D., Penn, R.D. and Donoghue, J.P., 2006. Neuronal ensemble control of prosthetic devices by a human with tetraplegia. Nature, 442(7099), p.164.

[2] Prasad, G., Herman, P., Coyle, D., McDonough, S. and Crosbie, J., 2010. Applying a brain-computer interface to support motor imagery practice in people with stroke for upper limb recovery: a feasibility study. Journal of neuroengineering and rehabilitation, 7(1), p.60.

[3] Lotte, F., Bougrain, L., Cichocki, A., Clerc, M., Congedo, M., Rakotomamonjy, A. and Yger, F., 2018. A review of classification algorithms for EEG-based brain–computer interfaces: a 10 year update. Journal of neural engineering, 15(3), p.031005.

[4] Zubarev, I., Zetter, R., Halme, H.L. and Parkkonen, L., 2018. Robust and highly adaptable brain-computer interface with convolutional net architecture based on a generative model of neuromagnetic measurements. arXiv preprint arXiv:1805.10981.

[5] Tabar, Y.R. and Halici, U., 2016. A novel deep learning approach for classification of EEG motor imagery signals. Journal of neural engineering, 14(1), p.016003.

[6] Lawhern, V.J., Solon, A.J., Waytowich, N.R., Gordon, S.M., Hung, C.P. and Lance, B.J., 2018. EEGNet: a compact convolutional neural network for EEG-based brain–computer interfaces. Journal of neural engineering, 15(5), p.056013.

[7] A. Chowdhury, H. Raza, Y. K. Meena, A. Dutta, and G. Prasad, "Online covariate shift detection-based adaptive braincomputer interface to trigger hand exoskeleton feedback for neuro-rehabilitation," IEEE Transactions on Cognitive and Developmental Systems, vol. 10, no. 4, pp. 1070–1080, Dec 2018.

[8] Le Cun B B, Denker J S, Henderson D, Howard R E, Hubbard W and Jackel L D 1990 Handwritten digit recognition with a back-propagation network Advances in Neural Information Processing Systems (San Francisco, CA: Morgan Kaufmann Publishers Inc.)

[9] Zhang, T., 2004, July. Solving large scale linear prediction problems using stochastic gradient descent algorithms. In Proceedings of the twenty-first international conference on Machine learning (p. 116). ACM.

[10] Kingma, D.P. and Ba, J., 2014. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.

[11] Jolliffe I 2002 Principal Component Analysis (New York: Wiley) (doi: 10.1002/9781118445112.stat06472).

[12] Pfurtscheller, G., Neuper, C., Schlogl, A. and Lugger, K., 1998. Separability of EEG signals recorded during right and left motor imagery using adaptive autoregressive parameters. IEEE transactions on Rehabilitation Engineering, 6(3), pp.316-325.

[13] Comon P 1994 Independent component analysis, a new concept? Signal Process. 36 287–314.

[14] Simonyan, K. and Zisserman, A., 2014. Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.

[15] A. Chowdhury, Y. K. Meena, H. Raza, B. Bhushan, A. K. Uttam, N. Pandey, A. A. Hashmi, A. Bajpai, A. Dutta, and G. Prasad, "Active physical practice followed by mental practice using bci-driven hand exoskeleton: A pilot trial for clinical effectiveness and usability," IEEE Journal of Biomedical and Health Informatics, vol. 22, no. 6, pp. 1786–1795, Nov 2018.

[16] Schlögl A 2003 Outcome of the BCI-Competition 2003 on the Graz Data Set (Berlin: Graz University of Technology.

[17] Johnson, R. and Zhang, T., 2013. Accelerating stochastic gradient descent using predictive variance reduction. In Advances in neural information processing systems (pp. 315-323).

[18] Iandola, F.N., Han, S., Moskewicz, M.W., Ashraf, K., Dally, W.J. and Keutzer, K., 2016. SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and< 0.5 MB model size. arXiv preprint arXiv:1602.07360.

[19] Lee, H., Grosse, R., Ranganath, R. and Ng, A.Y., 2009, June. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In Proceedings of the 26th annual international conference on machine learning (pp. 609-616). ACM.

[20] Roy, S., Rathee, D., McCreadie, K. and Prasad, G., 2019, March. Channel Selection Improves MEG-based Brain-Computer Interface. In 2019 9th International IEEE/EMBS Conference on Neural Engineering (NER) (pp. 295-298). IEEE.