Deep Convolutional Neural Network for Motor Imagery EEG Classification

Prachi Dwivedi[1], Ashish Dubey[2]
[1,2]*Department of Electronics & Communication, SRCEM, BANMORE(M.P)*

Abstract **- This research is based on the Deep Convolutional Neural Networks (DCNN) approach to the Detection of Motor Imagery (MI) tasks in EEG (Electroencephalogram), the use of brain-computer interface (BCI) techniques for direct communication between the human body and outside the world, which has significant forecast applications in the field of cognitive science & medical rehabilitation. The technology of DL (Deep Learning) has produced noteworthy results in BCI systems in recent years, in particular through the use for recognition and interpretation in MI of CNNs (Convolutional Neural Networks). The proposed procedure converts input EEG signals first into images by applying images of MI tasks EEG signals to DCNN level after a time-frequency transformation. For image classification, we train DCNN models. The efficiency of the proposed method is assessed on the IV dataset of BCI competition III. Evaluation metrics like the proposed accuracy method results quantitatively. Findings reveal that the 99.93% accuracy value of the proposed system is the highest one among existing accuracy scores.**

Index Terms **- BCI (Brain-computer interface), EEG (Electroencephalogram), CNNs (Convolutional Neural Networks), DCNN Model.**

I.INTRODUCTION

By analyzing electrical signals produced by brain nervous systems, BCIs offer a new direction between a human brain and computer [1]. Motor imagery EEG is a type of commonly utilized signal in BCI systems. In most areas of movement, these MI-based mechanisms will stimulate neuronal activity if participants can visualize moving parts of the body [2]. Uncertainty these neuronal processes are decrypted, effects of de-coding for patients by extreme motor neuro-disease (like Parkinson's Disease, Polio) may be utilized to monitor external devices like wheelchairs & service robots [3]. Therefore, in the application of motor imagery BCIs the EEG pattern classification plays an important part.

Several machine learning algorithms have been suggested to accurately decode EEG motor imagery. In the first instance, several of these approaches were used to extract discriminative time-frequency functionality from EEG tests, for example, dynamic connectivity analysis [4], frequency band analysis [5], Filter Band Common Spatial Pattern (FBCSP) [6] & continuous wavelet transforms [7]. These discriminatory characteristics are then linked to vectors also more utilized to train classifiers for EEG classifications, such as support vector machines [8] or decision-making bodies [9].

Preprocessing raw EEG signals may increase the signal-to-noise ratio of EEG & precision of classification, however, they do not have to be. CNN is the biologically inspired multi-layer perceptron variants built to use limited pre-processing. CNN for instance to classify the raw EEG signals directly. CNN enhances raw MI-EEGs' capabilities for feature representation & classification based on natural language processing and speech recognition. Developed a deeper neural network layer to allow imagining or executing tasks with raw EEG signals. CNN with raw EEG signals suggested a better prediction for driver's performance cognitive condition that produced good outcomes. It can also be shown to produce a good MI-EEG classification effect by using initial signals. CNN can explicitly reach multi-dimensional data to prevent the complex artificial feature extraction procedure that may extract distinctive feature details.

Existing DCNN's effectiveness is highly dependent on greedy learning of model parameter over a no. of iterations dependent upon a correctly built network architecture with detailed labeled training data. During the whole learning process, the majority of DCNN models handle all training samples consistently.

However, we find that, after a series of training iterations, the error decrease rate is initially high but decreases. Most samples may be attributed to the well-known propagation of network parameters by adjusting to a certain no. of training iterations, while confusable samples which are difficult to learn from & represent a very limited number of training datasets have no significant probability of contributing to the learning method. Some earlier DCNNs consider this phenomenon as a signal for reducing a learning rate manually or as an early end criterion [10], which neglects the ability of the confusing samples, which have been poorly learned to this stage.

This document is structured in the following way. In Section II, related work is summarized. Section III proposes suggested watermarking algorithms. In section IV, the results of the computer simulation are proposed & addressed. Finally, Section V includes concluding remarks and recommendations for future work.

II.REVIEW OF LITERATURE

Kim, J. et al. [11] Propose a new approach for the classification of CNN-based MI using a new input form. The input EEG signal is used to remove MI functionality by continuous wavelet transform (CWT).

Amin, S. U. et al. [12] In this article, EEG motor imagery data is used to detect advantages of multilevel extraction and fusion of convolutional features from various CNN layers that are abstract input representations at multiple levels. With raw EEG data, their presented CNN model will learn strong spectral & temporal characteristics. They show that such multi-level feature fusion performs models which utilize only the last features layer. Their findings for EEG decoding & classification are higher than state of the art.

Li, Y. et al. [13] present an end-to-end EEG decoding framework that utilizes raw multi-channel EEG as inputs, to improve decoding accuracy via channel-projection mixed-scale convolutional neural network (CP-Mixed Net) supported by amplitude-perturbation data augmentation. In particular, firstly CP-Mixed Net block is intended to learn primary representations in temporal & spatial from EEG signals.

Li, D. et al. [14] suggest DFFN (Densely Feature Fusion Convolutional Neural Networks). By integrating morphological data of EEG signals, they develop 2 low complexity methods of data representation and then develop & enhance the DFFN framework for this input format.

Jeong, J.-H. et al. [15] In this analysis, they concentrate on classifying forearm movements using EEG signals according to advanced rotation angles. To this aim, they suggest a robust classification for HF-CNN (Hierarchical Flow CNN) model. By their experimental dataset and also with a public dataset, they test the proposed model (BNCI Horizon 2020).

Abibullaev, B., et al. [16] They first set a range of hyperparameters in this article, which is limited by our computational resources. Then we demonstrate the exact classification of sensorimotor rhythms which occur in MI tasks through a comprehensive search within this limited space.

Zhang, j., et al. [17] A method for CNN analyzing EEG signals provided by left and right-hand MI tasks is presented in this article. EEG signals are transmitted by STFT (Short-Time Fourier Transform) to time-frequency images, and then these images are supplied for classification by the network input.

Kim, J., et al. [18] In this article, they suggest a new methodology to the classification of MI depends upon CNN using a new input form. CWT is utilized for extracting the MI features with the EEG input signal.

III.PROPOSED WORK

A. PROBLEMS IDENTIFICATION

The data was not cleaned properly. The model was using all the generated images instead of only taking the proper ones. The model was too simple to handle heavy data. There was very dissimilarity in all the 5 subjects, the resulting lack of accuracy except & subject AA, AL, AV, AW, AY.

B. PROPOSED METHODOLOGY

Select the file load it into Matlab along with true indexes and true labels. From the 118 channels, we select the 49 channels that are of our use in the next work. After the selection process, we extracted only the informative data that is by applying the band-pass filter from 8Hz – 30Hz, normalization, time segmentation of 0.5s to 2.5s also done. After Matlab work, data was saved in a Matlab extension file. Load the Matlab processed file in python to form some more operations to make that readable for the upcoming work. It is saved again but now in a CSV file that is easy to work on. Then before moving the last step of

the preprocessing which is making images out of the brain signal wave, we had to split the whole large data into 250 small CSVs to work on. Now, selecting the CSVs one after another to form images, 28 images were formed per CSV data. Total 6972 images were formed per subject, for the training we don't want our model to train the error images so we did operate to only select the perfect images and rejecting the errored ones. We used DCNN (deep convolutional neural network) model for our data to be trained on. After the completion of training, accuracy calculation, the ROC curve was plotted to show our model's performance.

C. DCNN Model

The suggested classification of weld defects was a deep convolutional neural network (DCNN). The classified ability of the extracted features was compared using different approaches. Deep Learning is the Deep Neural Network (DNN) based branch of machine learning which means neural networks at a minimum of three or four layers (including the input & output layers). But for some people (particularly non-technical), whatever their depths, any neural net qualifies as deep learning. And others regard a 10-layer neural net as shallow.

CNNs is a deep learning algorithm including convolution layers that extract features maps from images by using various no. of kernels. Formerly come pooling layers that minimize these dimensions. Once again there are various categories of pooling layers, which are average pooling layers & max pooling. The network also has more layers like dropouts & dense layers.

1. Conv2V

We'll cover the Keras Conv2D class, which includes the main parameters you need to adjust for your DCNN training.

2. Maximum pooling

Maximum pooling or max pooling is a pooling process that in every patch of each feature map determines maximum or major value.

Results are pooled or sampled feature maps which high light most present feature in a patch, not average presence for average pooling. That works better for computer vision tasks like image classification in practice than average pooling has been shown.

3. Batch normalization

Batch normalization is a layer that makes it possible for each network layer to learn more independently. The output of previous layers is normalized. Activation measures the normalization input layer. The use of batch normalization learning is also effective as regularization to prevent overlapping the model. To standardize the input or outputs the layer is applied to the sequential model. It can be used between the layers of the model at many points. This is also put after the sequential model is established & after pooling layers & convolutional layers. In the following code, display how the Batch normalization layer for classifying manual numbers is described. The needed libraries and the dataset are first imported.

4. Flattening

Flattening transforms the data into a 1D (One-Dimensional) array for the next layer. The contribution of convolutional layers is flattened to generate a single long feature vector. As well as it is linked to the final classification model, known as the fully connected layer.

5. Dense

The name means that the neurons in network layers are connected (dense). Each neuron in the layer is provided with the input of all neurons of the previous layer, which means that they are densely connected. In other words, a dense layer is a fully connected layer, which means that neurons in the next layer are linked to all neurons.

6. Dropout

Dropouts are a technique of regularization used to avoid fit into the model. Drop-outs are applied to alter the number of neurons network spontaneously. The input and output connection to neurons is also off when the neurons are switched off. This is done to improve model learning. It is normally recommended that dropouts not be used after convolution layers, but mainly after the dense network layers. Only 50 percent of neurons are good to switch off. If more than 50 percent of us are turned off, the model will likely be bad & predictions are not good. Let us see how we can utilize dropouts & describe them during the construction of the DCNN model.

Model: "sequential"

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 55, 55, 96)	34944
activation (Activation)	(None, 55, 55, 96)	0
max_pooling2d (MaxPooling2D)	(None, 27, 27, 96)	0
batch normalization	(None, 27, 27, 96)	384
conv2d_1 (Conv2D)	(None, 17, 17, 256)	2973952
activation_1 (Activation)	(None, 17, 17, 256)	0
max_pooling2d_1 (MaxPooling2	(None, 8, 8, 256)	0
batch_normalization_1 (Batch	(None, 8, 8, 256)	1024
conv2d_2 (Conv2D)	(None, 6, 6, 384)	885120
activation_2 (Activation)	(None, 6, 6, 384)	0
batch_normalization_2 (Batch	(None, 6, 6, 384)	1536
conv2d_3 (Conv2D)	(None, 4, 4, 384)	1327488
activation_3 (Activation)	(None, 4, 4, 384)	0
batch_normalization_3 (Batch	(None, 4, 4, 384)	1536
conv2d_4 (Conv2D)	(None, 2, 2, 256)	884992
activation_4 (Activation)	(None, 2, 2, 256)	0
max_pooling2d_2 (MaxPooling2	(None, 1, 1, 256)	0
batch_normalization_4 (Batch	(None, 1, 1, 256)	1024
flatten (Flatten)	(None, 256)	0
dense (Dense)	(None, 4096)	1052672
activation_5 (Activation)	(None, 4096)	0
dropout (Dropout)	(None, 4096)	0
batch_normalization_5 (Batch	(None, 4096)	16384
dense_1 (Dense)	(None, 4096)	16781312
activation_6 (Activation)	(None, 4096)	0
dropout_1 (Dropout)	(None, 4096)	0
batch_normalization_6 (Batch	(None, 4096)	16384
dense_2 (Dense)	(None, 1000)	4097000
activation_7 (Activation)	(None, 1000)	0
dropout_2 (Dropout)	(None, 1000)	0
batch_normalization_7 (Batch	(None, 1000)	4000
dense_3 (Dense)	(None, 2)	2002
activation_8 (Activation)	(None, 2)	0

Total prams: 28,081,754
Trainable prams: 28,060,618
Non-trainable prams: 21,136

Fig. 1. DCNN Model sequential

PROPOSED ALGORITHM

Step 1: Select the particular 49 channels that have the main signal frequency in Matlab and also apply a bandpass filter on the selected data to remove the unwanted frequency.

Step 2: Creating a CSV file from the precise data (Matlab file) to overcome the difficulty for the next work.

Step 3: Split the CSV into 250 different CSVs to a better process. Thus, it makes 28 images per CSV and interpolating and images also.

Step 4: Saving only the proper images and rejecting the error images.

Step 5: Pass the images into the model for training applying 10-Fold Validation for improving accuracy and validation accuracy.

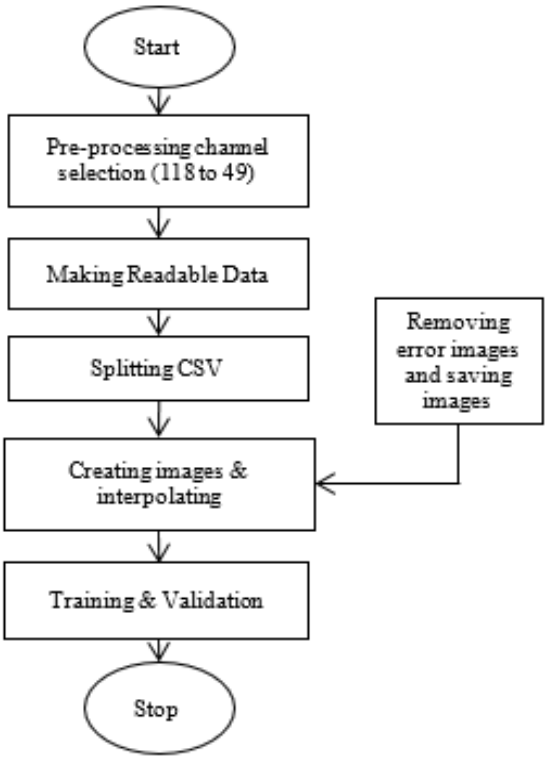

Fig.2. Flow diagram of proposed System

IV. RESULTS AND DISCUSSION

Proposed CNN (DCNN) model hardware & software platforms are Intel (R) Core (TM) i7-10700k 5.10 GHz CPU, 32 GB RAM DDR4, Python 3.8, Jupyter notebook, & TensorFlow 2.4.3 (GPU), GPU NVIDIA RTX 2060 SUPER cuDNN and CUDA ToolKit 11.2v.

A. DATASET DESCRIPTION

We have used data sets for comparison in this report to help assess the efficiency of our proposed algorithm. Dataset is public BCI competition III dataset IVa, in which EEG signals are obtained by 5 subjects (represented by al, aa, ay, aw, and av) with 118 electrode amplifiers. -ere are 2 kinds of MI tasks in the dataset, i.e., right foot MI & right-hand MI. -e time duration of every MI trial is 3.5 seconds, also the sampling rate is set to 1000 Hz. For every subject, there are 280 MI trials (140 trials for foot MI &140 trials for hand MI) in total. More information about this dataset may be originated at http://www.bbci.de/competition/iii/.

B. MODEL DETAILS: DCNN

A model hyperparameter is a configuration external to the model and the value of which cannot be measured from data. They are also used for estimating model parameters in processes. The practitioner also specifies them. Sometimes they can be set by heuristics. A predictive modelling problem is also targeted. For a hyperparameter model of a particular issue, we cannot know the best value. We can use thumb rules, copy values used for other issues or check by test and error for the best value. When you set the algorithm for machine learning to address a particular problem, such as a random search, you set the model's hyperparameter or determine the parameters of the model which lead to the best predictions.

Hyper-dense layer parameters are also part of DCNN architecture.

Hyper parameter	
Parameter	Value
Padding	Valid
Optimizer	Adam
Activation function	Relu
Regularization	Dropout (40%)
Cost function	Categorical cross entropy
Batch size	128
Classes	2
Subjects	5
Kernel	11*11
Input shape	227*227*3
Filter shape	96
Pooling	Maxpooling2D, size(2*2)
Strides	(2*2)
Training images	80%
Validation images	20%

Fig.3. Hyper Parameters

1. Padding

In general, padding is utilized to add rows & columns of zero to maintain spatial scale constantly after convolution, such that accuracy could be improved by maintaining details at the border.

2. Optimizer

For training the DL model, adam is a replacement optimization algo of stochastic gradient descent.

To provide an optimization algorithm, Adam combined the best properties of RMSProp & AdaGrad algorithms to manage sparse gradients of noisy problems.

$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t$

$v_t = \beta_2 \, v_{t-1} + (1 - \beta_2) g_t^2$

3. Activation function

The rectified linear activation or ReLU function for short is a linear partly function that produces the input directly if it is positive, otherwise, zero will be produced. For certain kinds of neural networks, it is now the main activation mechanism since a model that is used is easier to learn and often performs well.

$$F(x) = \begin{cases} 0.01x \; for \; x < 0 \\ x \; for \; x \geq 0 \end{cases}$$

4. Regularization

Dropout is easily implemented by selecting randomly dropping nodes with a certain probability (for example 20%) for every weight update cycle. Dropout is being deployed in Keras as well. Dropout is used only during model training & not when evaluating the model's skills.

$E_R = \frac{1}{2}(t - \sum_{i=1}^{n} p_i \; w_i \; I_i \;)^2 + \sum_{i=1}^{n} pi(1 - pi) \, w_i^2 I_i^2$

5. Cost function Categorical cross-entropy

A cost function is an error measurement among the predicted value of your model & the actual value. E.g., we want the yi value for the data point xi to be predicted.

$\sum_{i=1}^{n} (y_i - f_\theta(x_i))^2$

6. Batch size

Instead, the batch size should be analyzed by the running time of the training because the hyperparameter of the training rate does not impact its effect on time. The batch size is restricted to the memory of your hardware, but it does not. Leslie prefers to use a batch size that matches the memory of the hardware and allows higher learning rates.

$$\text{Number of batches} = \left\lfloor \frac{\text{size of training dataset}}{\text{batch size}} \right\rfloor$$

$$= \left\lfloor \frac{60.000 \; images}{128 \; images} \right\rfloor$$

$$= \lfloor 468{,}75 \rfloor$$

$$= 469$$

7. Kernel/Filter Size

A filter is a weights matrix with which the data is integrated. The filter on convolution measures how similar an input patch is to a feature. A vertical edge or arc or any kind may be a feature. During training, the weight of the filter matrix is derived. As much local information as possible is collected by smaller filters, and larger filters reflect more global, high level & representative information.

8. Pooling-layer Parameters

Pooling layers also have the same convolution layer parameters. For all pooling solutions, Max-Pooling is commonly used.

9. Number of Channels

Several color channels for input are equal, but in subsequent steps, no. of filters for convolution operation is equal. The further sources, the more filters, the more characteristics are learned and the more chances are exaggerated and vice versa.

10. Stride

Generally, for each elemental multiplication of input weights with weights in buffer, this is no. of pixels you choose to skip when crossing input horizontally & vertically. It is used to significantly reduce the input image size.

Table 1. Result in term of Training Accuracy of each subject

Subject	AA (%)	AL (%)	AV (%)	AW (%)	AY (%)	Mean (%)
Base Algorithm	99.02	74.10	83.46	81.31	73.03	82.184
Proposed Algorithm	99.98	99.90	99.94	99.86	100.0	99.936

Training Accuracy of each subject Comparison of the precision of our process classification and 5 other competing approaches; (BCI competition III dataset IVa).

Table 2.Result in term of Training Loss of each subject

Subject	AA (%)	AL (%)	AV (%)	AW (%)	AY (%)	Mean (%)
Base Algorithm	3.270	49.73	34.48	37.23	50.55	35.052
Proposed Algorithm	0.02	0.035	0.019	0.043	0.015	0.0264

Proposed Accuracy Graph of Each Subject (1-AA, 2-AL, 3-AV, 4-AW, 5-AY)

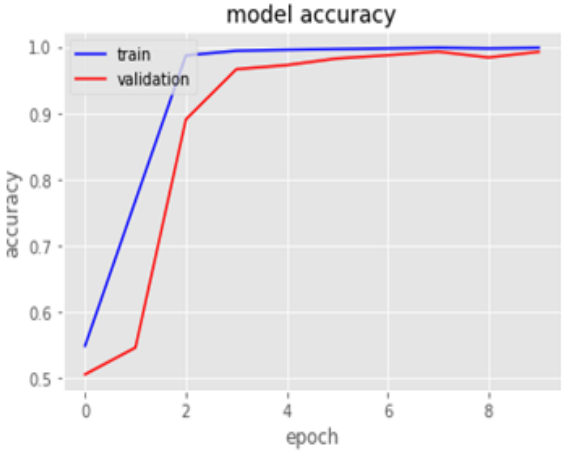

Fig.3. Model Train and Validation Accuracy of AA

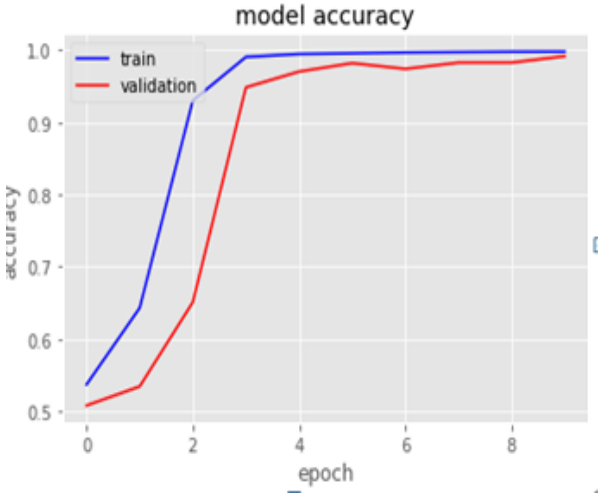

Fig. 4. Model Train and Validation Accuracy of AL

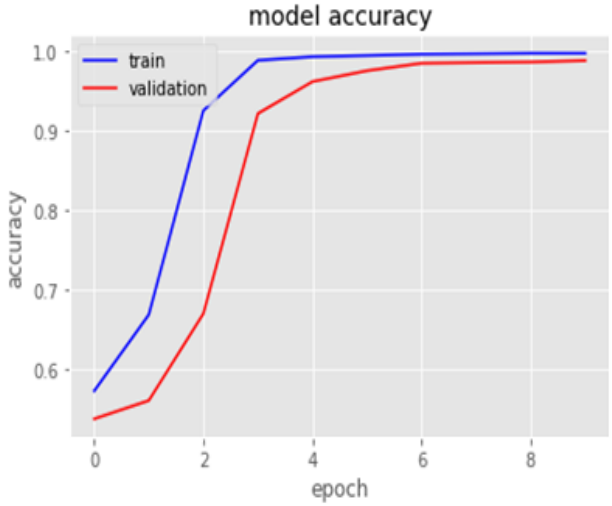

Fig. 5. Model Train and Validation Accuracy of AV

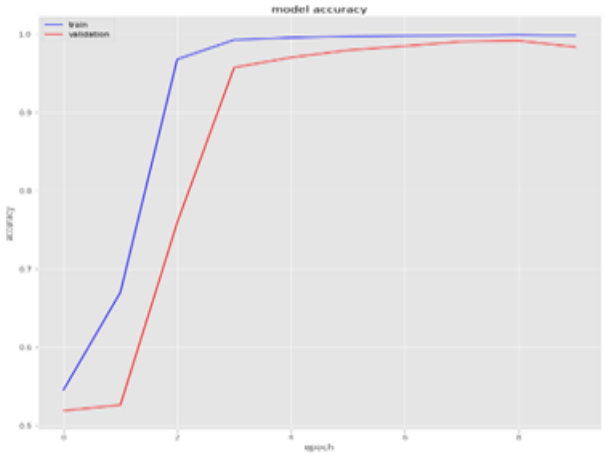

Fig. 6. Model Train and Validation Accuracy of AW

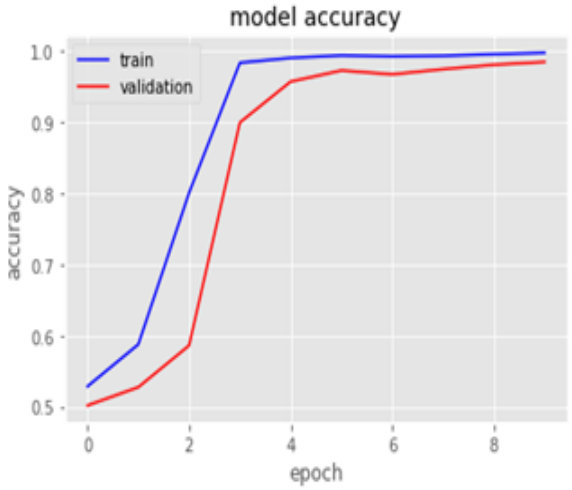

Fig. 7. Model Train and Validation Accuracy of AY Proposed Scattered Graph of Each Subject (1-AA, 2-AL, 3-AV, 4-AW, 5-AY)

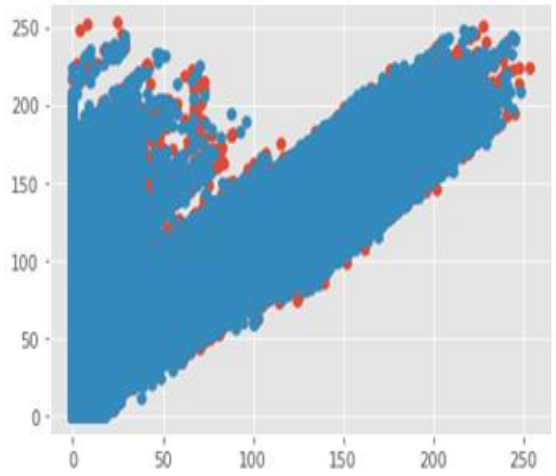

Fig. 8. Scattered graph of AA

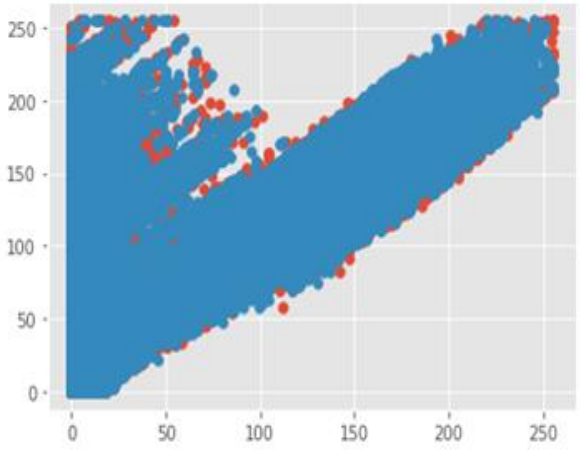

Fig. 9. Scattered graph of AL

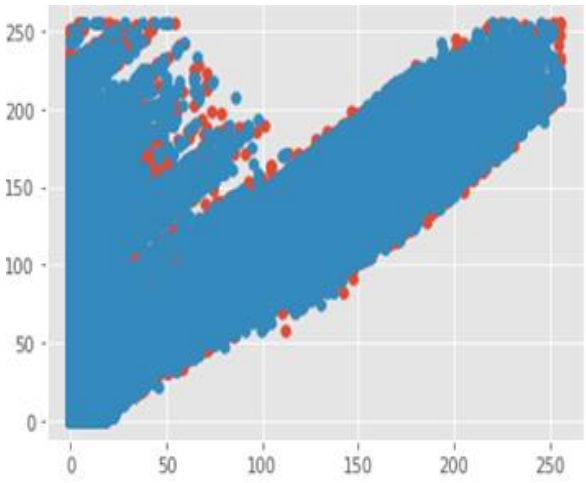

Fig. 10. Scattered graph of AV

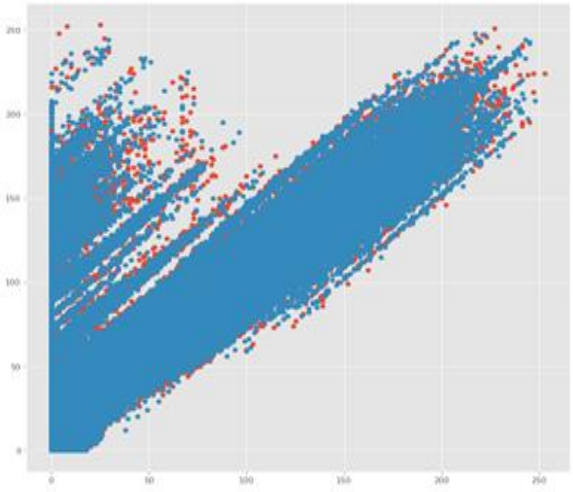

Fig. 11. Scattered graph of AW

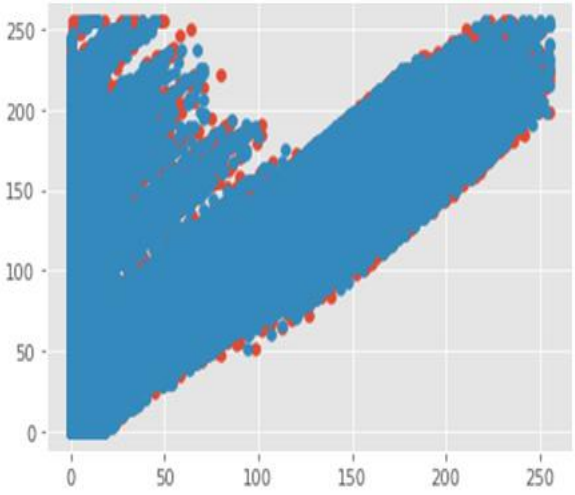

Fig.12. Scattered graph of AY Subject wise image plot

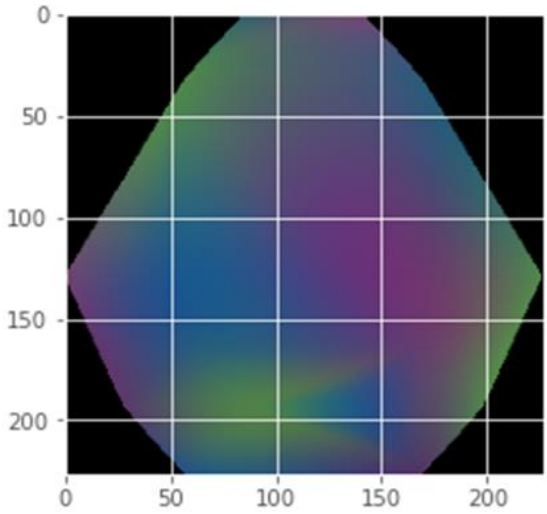

Fig.13. Subject of AA

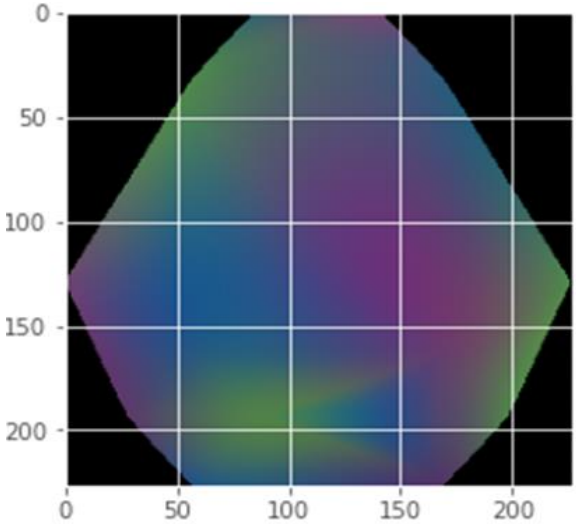

Fig.14. Subject of AL

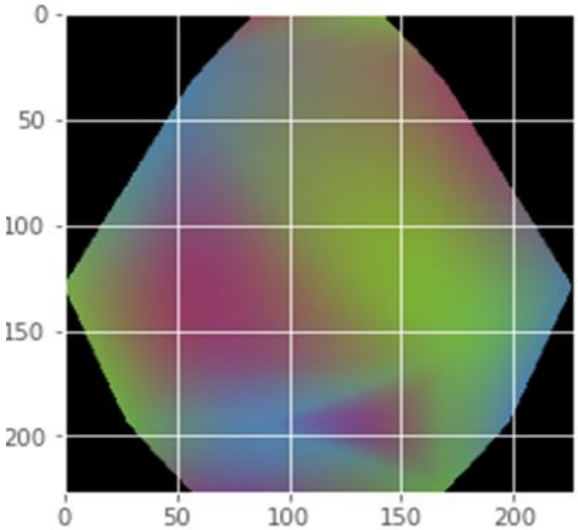

Fig.15. Subject of AV

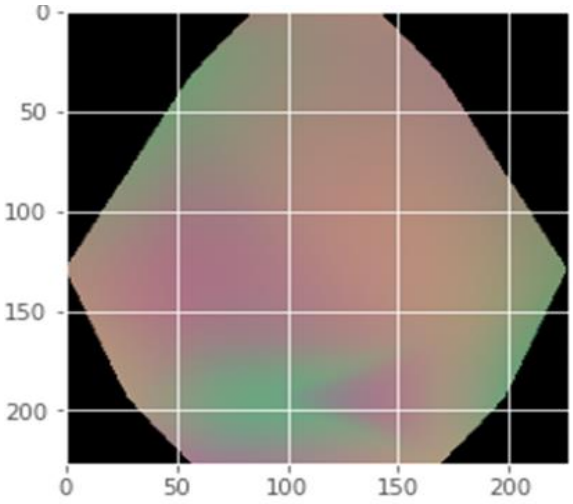

Fig.16. Subject of AW

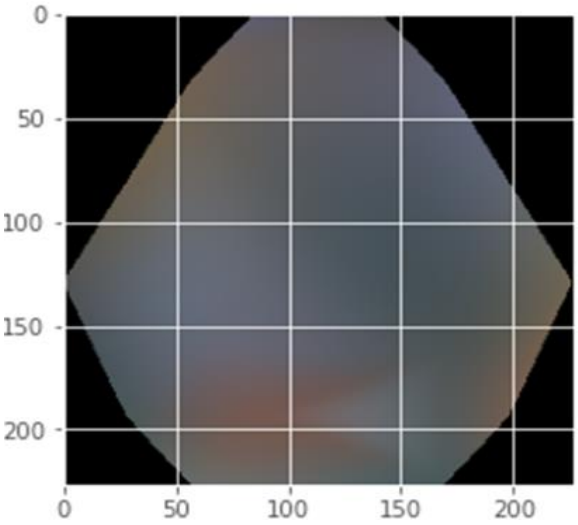

Fig.17. Subject of AY

Have represented the subject of brain-computer interface (BCI) that include AA, AL, AV, AW, AY.

V.CONCLUSION

In this context, a deep learning method to identifying focal EEG signals is proposed. DL algorithm for EEG model classification in limb motor imagery is proposed in this article. A multi-layer DCNN model for the EEG motor imaging classification is designed and the motor imagery EEG signals' spatial frequency characteristics are evaluated by the parameters of neural network convolution layers. The public BCI competition III IVa dataset was used during experiments. We have introduced a MI multi-class classification method in this article. The system has proven to be a widespread single model for this

classification task experimentally. In further applications such as image segmentation, computer vision, etc., deep CNN architectures have provided good classification accuracy. In this article, we suggested a new way of using deep CNN for the processing of EEG signals. Our test results show that the DCNN model is better able to classify than any other machine learning solution. Better precision with the lowest variance demonstrates that the CNN model in the field of EEG signal processing can be used extensively. The temporal filter is used in this architecture to collect frequency data from the raw EEG signal.

FUTURE WORK

We will study a transfer learning technique in future studies in the following directions for more analysis. First, human behavior is an open-minded and dynamic issue that cannot be considered a simple problem with multiple classifications. The most famous problem, however, is to create a classification for the less classified dataset to incorporate comprehensive unknown data. Secondly, it is still a subject for future studies to explore and compare other active and quick clustering or classification algorithms with the one proposed in the present report. Third, effective behavior perception helps one to perceive the condition of the individual and create human-centered applications. The development of an effective living framework to consider the behavior of users would also facilitate home-based healthcare.

REFERENCES

[1] L. He, D. Hu, M. Wan, Y. Wen, K. M. von Deneen, and M. Zhou, "Common Bayesian network for classification of EEG-based multiclass motor imagery BCI," IEEE Trans. Syst., Man, Cybern., Syst., vol. 46, no. 6, pp. 843–854, Jun. 2016.

[2] Y. Zhang, C. S. Nam, G. Zhou, J. Jin, X. Wang, and A. Cichocki, "Temporally constrained sparse group spatial patterns for motor imagery BCI," IEEE Trans. Cybern.

[3] K. K. Ang and C. Guan, "EEG-based strategies to detect motor imagery for control and rehabilitation," IEEE Trans. Neural Syst. Rehabil. Eng., vol. 25, no. 4, pp. 392–401, Apr. 2017.

[4] Y. Wang, K. C. Veluvolu, and M. Lee, "Time-frequency analysis of band-limited EEG with BMFLC and Kalman filter for BCI applications," J. Neuroengineering Rehabil., vol. 10, no. 1, pp. 1–16, 2013.

[5] Y. Li, M.-Y. Lei, W. Cui, Y. Guo, and H.-L. Wei, "A parametric time frequency-conditional granger causality method using ultra-regularized orthogonal least squares and multiwavelets for dynamic connectivity analysis in EEGs," IEEE Trans. Biomed. Eng., to be published.

[6] Y. Li, W. G. Cui, M. L. Luo, K. Li, and L. Wang, "High-resolution time-frequency representation of EEG data using multi-scale wavelets," Int. J. Syst. Sci., vol. 48, no. 12, pp. 2658–2668, May 2017.

[7] W. Wu, Z. Chen, X. Gao, Y. Li, E. N. Brown, and S. Gao, "Probabilistic common spatial patterns for multichannel EEG analysis," IEEE Trans. Pattern Anal. Mach. Intell., vol. 37, no. 3, pp. 639–653, Mar. 2015.

[8] L. Wang et al., "Automatic epileptic seizure detection in EEG signals using multi-domain feature extraction and nonlinear analysis," Entropy, vol. 19, no. 6, p. 222, 2017.

[9] S. Min, B. Lee, and S. Yoon, "Deep learning in bioinformatics," Brief Bioinform., vol. 18, no. 5, pp. 851–869, 2016.

[10] L. Prechelt. Early stopping-but when? In Neural Networks: Tricks of the trade, 55-69, 1998.

[11] Kim, J., Park, Y., & Chung, W. (2020). Transform-based feature construction utilizing magnitude and phase for convolutional neural network in EEG signal classification. 2020 8th International Winter Conference on Brain-Computer Interface (BCI). doi:10.1109/bci48061.2020.9061635

[12] Amin, S. U., Alsulaiman, M., Muhammad, G., Bencherif, M. A., & Hossain, M. S. (2019). Multilevel Weighted Feature Fusion Using Convolutional Neural Networks for EEG Motor Imagery Classification. IEEE Access, 1–1. doi:10.1109/access.2019.2895688

[13] Li, Y., Zhang, X.-R., Zhang, B., Lei, M.-Y., Cui, W.-G., & Guo, Y.-Z. (2019). A Channel-Projection Mixed-Scale Convolutional Neural Network for Motor Imagery EEG Decoding. IEEE Transactions on Neural Systems and

Rehabilitation Engineering, 1–1. doi:10. 1109 /tnsre.2019.2915621

[14] Li, D., Wang, J., Xu, J., & Fang, X. (2019). Densely Feature Fusion Based on Convolutional Neural Networks for Motor Imagery EEG Classification. IEEE Access, 7, 132720–132730. doi:10.1109/access.2019.2941867

[15] Jeong, J.-H., Lee, B.-H., Lee, D.-H., Yun, Y.-D., & Lee, S.-W. (2020). EEG Classification of Forearm Movement Imagery Using a Hierarchical Flow Convolutional Neural Network. IEEE Access, 8, 66941–66950. doi:10.1109/ access. 2020.2983182

[16] Abibullaev, B., Dolzhikova, I., & Zollanvari, A. (2020). A Brute-force CNN Model Selection for Accurate Classification of Sensorimotor Rhythms in BCIs. IEEE Access, 1–1. doi:10.1109 /access. 2020.2997681

[17] Zhang, J., Yan, C., & Gong, X. (2017). Deep convolutional neural network for decoding motor imagery-based brain computer interface. 2017 IEEE International Conference on Signal Processing.