Diyala Journal of Engineering Sciences

Journal homepage: https://djes.info/index.php/djes

ISSN: 1999-8716 (Print); 2616-6909 (Online)

Dual Optimization of Deep CNN for Motor Imagery EEG Tasks Classification

Ali Al-Saegh[1*], Amar Daood[1] and Mohammad H. Ismail[2]

[1]Department of Computer Engineering, College of Engineering, University of Mosul, Mosul.
[2]Department of Computer and Information Engineering, College of Electronics Engineering, Ninevah University, Mosul, Iraq

ARTICLE INFO

Article history:
Received June 27, 2024
Revised September 28, 2024
Accepted October 8, 2024
Available online December 1, 2024

Keywords:
Convolutional neural network
Deep learning
Genetic algorithm
Optimization of CNN hyperparameters
Optimization of cropping parameters

ABSTRACT

Motor imagery (MI) electroencephalographic (EEG) signals have many real-world applications such as controlling a wheelchair through thought, body motor rehabilitation, person identification, and more. However, MI signals are quite complex due to their non-stationary nature and high dimensionality, meaning there is no specific rhythm from the many sources of signals. The deep convolutional neural network (CNN) overcomes these challenges by automatic feature extraction and selection. However, CNN architecture comprises many parameters (weights of network connections) and hyperparameters (related to network architecture) requiring optimization. The backpropagation algorithm optimizes the parameters, while the hyperparameters are adjusted manually through a lengthy trial-and-error process. This paper proposes a GACNN model to optimize those hyperparameters. The proposed model employs the genetic algorithm (GA) to optimize the number of convolution filters at each CNN layer, the size of filters, and the dropout probability. Additionally, the GA optimizes the cropping augmentation (used to boost the EEG training samples) parameters, namely window size and step size. The GACNN achieved promising results in MI EEG analysis compared to state-of-the-art studies. Experimental results showed a 17% increase in Cohen's kappa coefficient, indicating the model's classification accuracy. A decrease of about 25% in standard deviation (SD) is recorded, indicating that the GACNN showed no bias among the subjects who participated in the experiment.

1. Introduction

Electroencephalography (EEG) has generated significant attention in recent decades employing it for a myriad of real-world applications, including body motor rehabilitation, neuro-marketing, disease detection, person identification, neuro-entertainment, and many others. These applications predominantly rely on the interaction between humans and computer devices. Motor Imagery (MI) is a notable paradigm for recording EEG signals [1]. 'Motor' refers to any joint in the human body, while 'imagery' implies that a person only thinks (intends) to move a specific joint without physically doing so. Various tasks (joint movements) have been extensively experimented with and utilized in real-world applications including right-hand, left-hand, tongue, and feet [2-3]; in addition to other fine-body parts such as diametric grasp, lateral grasp, wrist deviation, wrist flexion, wrist extension, and finger-related tasks [4], other tasks may also exist.

EEG signals are acquired via several channels (electrodes) ranging from three to

* Corresponding author.
E-mail address: ali.alsaegh@uomosul.edu.iq
DOI: 10.24237/djes.2024.17405

more than one hundred channels. This multitude of channels produces massive features, each with various frequency components. Furthermore, EEG signals exhibit non-stationary behavior; i.e., the signals do not show a repetitive specific rhythm along with noise and artifacts stemming from the recording devices and unrelated body movements, such as the eye blink [5]. Therefore, the curse of dimensionality problem appears, making building a robust identification model challenging.

Various conventional machine learning algorithms have been developed for EEG analysis. The Common Spatial Pattern (CSP) has been widely used to discover several spatial filters to maximize the distance between each pair of available classes, such as feet and tongue [6]. A variation of CSP that addresses the loss of temporal information is the Filter-Bank Common Spatial Pattern (FBCSP), in which the number of filter banks is first determined, and then the CSP energy is estimated for each temporally filtered signal [7]. However, such conventional feature extraction methods involve human intervention to design features, which can be challenging in the case of EEG signals due to the curse of dimensionality.

Deep Learning (DL) algorithms have recently gained high attention and have been adopted in diverse research fields, including time-series signal processing, image processing [8-10], and video processing [11]. This high attention is due to the capability of DL for automatic feature extraction and selection [12] that reduces the burden on the designer and ensures the analysis of each latent feature type. This is extremely beneficial, especially for EEG signals due to their high-dimensional features.

The DL-MI EEG classification framework depends on the Event-Related Synchronization/Desynchronization (ERS/ERD) phenomena that occur during the performance of MI tasks [1]. The two cerebral phenomena are characterized by increments and decrements in signal voltage, resulting from synchronizing and desynchronizing various cerebral signals within specific frequency bands. DL detects and extracts these changes as distinct features of an MI class.

Various types of EEG recordings have been recently analyzed using DL algorithms. In [13], the authors developed a fused DL model consisting of a Convolutional Neural Network (CNN) and Long Short-Term Memory (LSTM) to differentiate high/low arousal and high/low valence emotions. In [14], the authors developed a real-time stroke identification system using DL and a metaheuristic optimizer. EEG recordings were analyzed using a hybrid classifier consisting of a CNN and a Bidirectional Gated Recurrent Unit (BiGRU), while harmony search was the optimization algorithm. Many other uses of DL in analyzing EEG signals exist [15-17].

Despite the remarkable advantages of the DL approach, it has several disadvantages. DL deals with large-scale neural networks which require a considerable amount of memory during identification to hold a large number of learnable parameters [18]. A substantial amount of training samples is required to tune the learnable parameters; otherwise, the underfitting problem may occur [19]. This can result in a trivial and non-generalizable model unable to capture adequate latent information for future predictions.

Researchers have suggested several methods to deal with large-scale deep networks. Dropout is a commonly used technique during the CNN construction, which randomly ignores several learnable parameters. This reduces the network size by a factor related to the dropout probability and helps prevent the overfitting problem. However, the dropout probability is often set arbitrarily without optimization leading to possible over- or under-estimation of the task.

On the other hand, the lack of data can be tackled using augmentation methods. Various methods exist for images and time-series signals. The augmentation process relies on geometric variation, noise addition, and cropping [20]. The augmentation process generates new samples from the available ones, thereby increasing the dataset size. Specifically, several augmentation methods exist for EEG analysis, such as cropping, CutCat, and Gaussian noise [1], [21]. Augmentation methods are applied randomly; for example, the

window and step size of the cropped part are arbitrarily set leading to possible loss of latent features.

Accordingly, it is important to design a CNN tailored to the task and generate new feasible training samples. The known Backpropagation algorithm (BP) optimizes the network's learnable parameters (weights) without considering the network's hyperparameters, such as the number of convolution filters (kernels), size of filters, and dropout probability. These network's hyperparameters affect the resulting network size and thus its architecture complexity. Therefore, it is advantageous to jointly optimize the learnable parameters and hyperparameters, eliminating the burden on the designer and achieving a tailored CNN.

The Genetic Algorithm (GA) is a powerful heuristic optimization algorithm used to search for optimal solutions within a complex error surface and a large parameter space [22], [23]. GA starts with an initial set of solutions, quantifies these possible solutions, produces a new generation of solutions, and repeats this process until reaching the optimized set of solutions.

In this paper, the GACNN model is proposed. The model employs a deep CNN composed of four convolution layers for MI EEG-related feature extraction and selection. While BP tunes the network's parameters, GACNN leverages GA to tune the network's hyperparameters and cropping parameters. GACNN aims to produce a tailored CNN for a specific task.

The problem statement of the paper can be stated in the following two main points:

- The backpropagation optimization algorithm does not optimize CNN's hyperparameters, such as the number of convolution filters (kernels), size of filters, and dropout probability.
- Previous studies have set cropping augmentation parameters arbitrarily, which impacts the information contained in the generated trials.

Consequently, the primary contributions achieved in this paper are:

- **Hyperparameter optimization:** GACNN uses GA to optimize the core CNN's hyperparameters, namely the number of convolution filters (kernels), size of convolution filters (i.e., width and height), and dropout probability.
- **Augmentation optimization:** GACNN uses GA to optimize the cropping augmentation's parameters, namely window size and step size.

The rest of the paper is organized as follows: Section 2 discusses several previous works related to the paper's subjects. Section 3 provides a theoretical background of the main techniques and presents the proposed GACNN model. The experimental settings are discussed in Section 4. Section 5 presents the experimental results with a discussion, and Section 6 concludes the paper.

2. Literature review

Many studies have explored two method categories to analyze and decode EEG signals. The first category includes traditional signal processing techniques that depend on manual feature design, which can be time-consuming and error-prone. The second category is the DL techniques that alleviate the burden on the designer through automatic feature extraction and selection.

Authors of [24] adopted a deep CNN for person identification using EEG recordings. The CNN extracted the dominant spatial features from the signals, while Recurrent Neural Networks (RNNs) were used to extract the temporal features. Two models were examined: the CNN-GRU, which uses a CNN with a Gated Recurrent Unit (GRU) as the RNN, and the CNN-LSTM, which uses a CNN with an LSTM as the RNN. The authors achieved a high identification accuracy in practical applications. Transfer learning was used in [25] to train two deep CNN networks to classify MI EEG signals. Transfer learning involves training a pre-trained network by tuning only the parameters of the last layers; it is useful when there are not enough training samples, especially with large-scale networks. The authors used two established CNN architectures ResNet-50 and Inception-v3,

in conjunction with an LSTM network. Originally, these CNN architectures were trained on natural images, therefore, the authors generated spectral images using continuous wavelet transform to train the networks on the EEG data.

In [6], the goal was to detect covariance shifts in EEG distributions. Feature extraction was achieved by detecting variations in principal component analysis, after which a k-nearest neighbor classifier was trained. The authors of [26] used multivariate intrinsic mode functions to extract cross-channel information from EEG signals. A class-related covariance matrix was then constructed and the Riemannian geometry algorithm was utilized to measure the distance between the known covariance matrix and an unknown test trial. An 8-bit numerical quantization was applied in [27] to the weights and activations of EEGNet (a well-known benchmark deep CNN in EEG analysis), the new version of the network is called Q-EEGNet. An augmentation method called Cut-Concatenation (CutCat) is proposed in [1] to handle the non-stationary nature of EEG signals. In CutCat, a new trial is formed by concatenating short crops from several trials of MI EEG tasks belonging to different subjects. The generated trials were used to train a deep CNN composed of four convolutional layers.
Researchers adopted GA for various issues related to EEG signals. GA was applied in [28] to select relevant features for classifying left-hand and right-hand MI tasks. A feature vector was constructed using autoregressive parameters and discrete wavelet transform, then the GA was employed to identify the features that contributed to the highest classification accuracy while discarding less informative ones. GA was adopted in [22] to determine the optimal set of EEG channels for MI task classification. This process involved using CSP for feature extraction and the Support Vector Machine (SVM) for classification. In [23], GA was employed to reduce noise in EEG signals retrieving the optimal parameters for wavelet transform. Additionally, many studies have utilized DL in the analysis of other medical signals [29-31].

3. Materials and methods

This section provides a theoretical background on CNNs, cropping augmentation, and GA. Then, it presents the proposed GACNN model.

3.1 Convolutional neural network (CNN)

The general CNN architecture is primarily composed of feature extractors and a classifier [1]. The feature extractors are layers of convolution filters with activation functions, along with other regularization layers and operations such as pooling, batch normalization, and dropout. The first network's layer receives the training data as input, and the layer's output passes to the subsequent layers. During the training, convolution layers automatically extract latent features from the training data. While the initial layers extract basic features, more complex features are gradually captured in the subsequent layers. The extraction process is primarily accomplished by the convolution filters, being considered the core components of CNNs.

The filters themselves are essentially matrices of values with specific widths and heights. The convolution operation that occurs between the input layer (or a feature map matrix) and a filter can be mathematically represented as follows:

$$c_{ij}^{y} = \sum_{m}^{k} \sum_{l}^{k} f_{lm}^{y} \times x_{(i+l-1)(j+m-1)} \quad (1)$$

where c_{ij}^{y} is an entry in the feature map matrix generated from the y-th filter ($y = 1,2, \dots, d$) where d is the number of the applied filters of size $k \times k$, i and j are the row and column indices in the feature map matrix, f_{lm}^{y} is an entry in the filter matrix, l and m are the row and column indices in the filter matrix, and x is an entry in the input sample signal.

Given the number of input feature maps (FM_{L-1}) and the number of output feature maps (FM_{L}) at a specific convolution layer, then the number of learnable parameters at that layer can be calculated as follows [32]:

$$Nu.parm = k \times k \times FM_{L-1} \times FM_{L} \quad (2)$$

The number and size of employed filters significantly affect the feature extraction process. A larger number of filters is expected to extract a greater variety and quantity of features. However, this comes at the cost of increased memory requirements, longer training times, and higher demands on computational resources, even on edge devices during the inference process.

Dropout is a regularization technique used to mitigate the overfitting problem occurring in CNNs. Overfitting happens when a model performs well on the training data but fails to generalize to the test data. This issue may arise from the small number of training samples relative to the network size, excessive training epochs, or too many convolutional filters [33]. The dropout layer works by randomly deactivating a certain percentage of parameters based on a predefined probability.

3.2 Cropping augmentation

The massive number of deep learning learnable parameters requires extensive training samples to avoid the overfitting problem. However, obtaining large EEG datasets is challenging, especially in situations involving sensitive personal information such as certain diseases or experimental settings with subjects that induce fatigue. Augmentation methods permit the use of limited datasets to train large-scale CNNs. Creating variability is another goal of augmentation methods, which can lead to more reliable models.

Cropping is a vital augmentation method that was first suggested for 2D images and later adopted for EEG signals. This method has achieved better results compared to other methods for EEG analysis [21]. Cropping a time-series signal involves generating many short slices from the original signal by sliding a window over its samples. The method depends on the window size and step size; the smaller the values of those two variables, the larger the number of slices acquired (i.e. more training samples); and vice versa. However, as the slices become smaller, latent information may be lost. Therefore, one should consider the trade-off between the number of generated training samples and the information retained within each trial.

As EEG signals are highly variable, selecting the window and step sizes can be challenging when using the trial-and-error method. An optimization algorithm can be employed for this task.

3.3 Genetic algorithm (GA)

The GA is an optimization technique belonging to the family of evolutionary computation algorithms and has many potential applications [14], [23]. The GA procedure begins by initializing several encoded possible solutions called a population of individuals (chromosomes). These individuals compete based on a predefined fitness function tailored to a specific task, driving the survival of the fittest. Throughout the GA procedure, individuals evolve over multiple generations by creating new offspring similar to their parents representing new possible solutions. This development is achieved through three main operations: selection, crossover, and mutation.

Before these main operations, the information of a possible solution must be encoded into an individual (chromosome) as a string of values (genes). Several encoding methods exist, such as binary, integer, and real; the choice depends on the task and the types of used crossover and mutation. The selection operation uses the fitness function to rank the individuals based on their quality, meaning high-quality solutions have a greater chance of being selected and reproducing offspring. The selection operation gradually updates the possible solutions, preventing the GA from being trapped in a local optimum. Crossover produces new and diverse solutions by determining how two parents pass their encoded information (the genes) to the new offspring. Mutation produces innovative solutions to help escape local optima by changing the values of genes rarely and randomly. The replacement operation occurs after completing the three main operations. Replacement merges individuals of the current population with the newly produced ones. Afterward, the best individuals are retained to form the new generation. This procedure iterates until convergence is

achieved; either by reaching the desired fitness level or hitting the maximum number of generations.

3.4 The proposed GACNN model

A schematic block diagram of the proposed GACNN model is shown in Figure 1. The GACNN process begins by acquiring EEG signals and generating training samples using cropping with initial window size and step size. A four-layer deep CNN is constructed with initial values for the number of filers at each layer, size of filters, and dropout rate. The GA generates these hyperparameter values as the initial population of solutions (individuals). The CNN is trained using BP and all the generated trials. Training continues until the desired classification accuracy is achieved or the maximum number of training epochs is reached. The GA undergoes fitness evaluation, selection, crossover, mutation, and replacement to produce a new generation of possible solutions. Based on the evaluated classification error and network complexity represented by the number and size of filters, the GA continues producing new generations until the desired fitness level is achieved or the maximum number of generations is attained.

The structured GACNN algorithm for optimizing the hyperparameters can be described as follows:

Algorithm: CNN optimization using BP and GA

Input:

- EEG training data: X^{train}
- Hyperparameters to optimize: Number of convolution filters per layer, size of filters, dropout probability, window size, step size.
- Maximum number of generations: G_{max}
- Population size: P_{size}
- Mutation rate: m
- Crossover rate: c

Output:

Optimized hyperparameters: θ^*

Initialization

1. Randomly initialize a population P of size P_{size}.

- Each individual $I \in P$ represents a set of hyperparameters:

$$I = \{num\_filters, filter\_size, dropout\_prob, window\_size, step\_size\}$$

2. For each individual I randomly assign values for the hyperparameters.

Training and Evaluation

3. For each generation $g = 1,2,3, \dots, G_{max}$:

- For each individual I:
  - Apply copping based on $window\_size$ and $step\_size$ from I to generate more training samples X_{crop}^{train}.
  - Build and train a CNN model $\mathcal{M}_\theta$ based on $num\_filters$, $filter\_size$, and $dropout\_prob$ from I using X_{crop}^{train}.
  - Evaluate the trained model on validation data and calculate the classification error: $\mathop{\mathbb{E}}_{x}\left[\mathcal{L}\left(x, \mathcal{M}_\theta\left(X^{\text{train}}\right)\right)\right]$.

Fitness Calculation

4. Calculate the fitness of each individual I: $fitness(I) = \frac{1}{\mathop{\mathbb{E}}_{x}\left[\mathcal{L}\left(x, \mathcal{M}_\theta\left(X^{\text{train}}\right)\right)\right]}$.

GA Operations

5. Encoding: encode each individual I using the integer representation.
6. Selection: select parent individuals based on their fitness values using the elitism strategy.
7. Crossover: with probability c perform crossover between selected parents to generate offspring.
8. Mutation: with probability m mutate the offspring by randomly changing one or more values in the offspring.
9. Replacement: replace the current population with the offspring, maintaining the best individuals.

Convergence Check

10. If the population has converged or the maximum number of generations G_{max} is reached, stop the process.

Return the Optimal Model

11. The individual I^* with the highest fitness is considered the optimal solution

- Return the optimized values of I based on:

$$\theta^* = \underset{\theta}{\operatorname{argmin}} \mathop{\mathbb{E}}_{x}\left[\mathcal{L}\left(x, \mathcal{M}_\theta\left(X^{\text{train}}\right)\right)\right].$$

The designed CNN includes four convolution layers, and the GA optimizes the relevant hyperparameters for each layer in addition to the cropping parameters. Figure 2 illustrates the gene variables of a GA's individual.

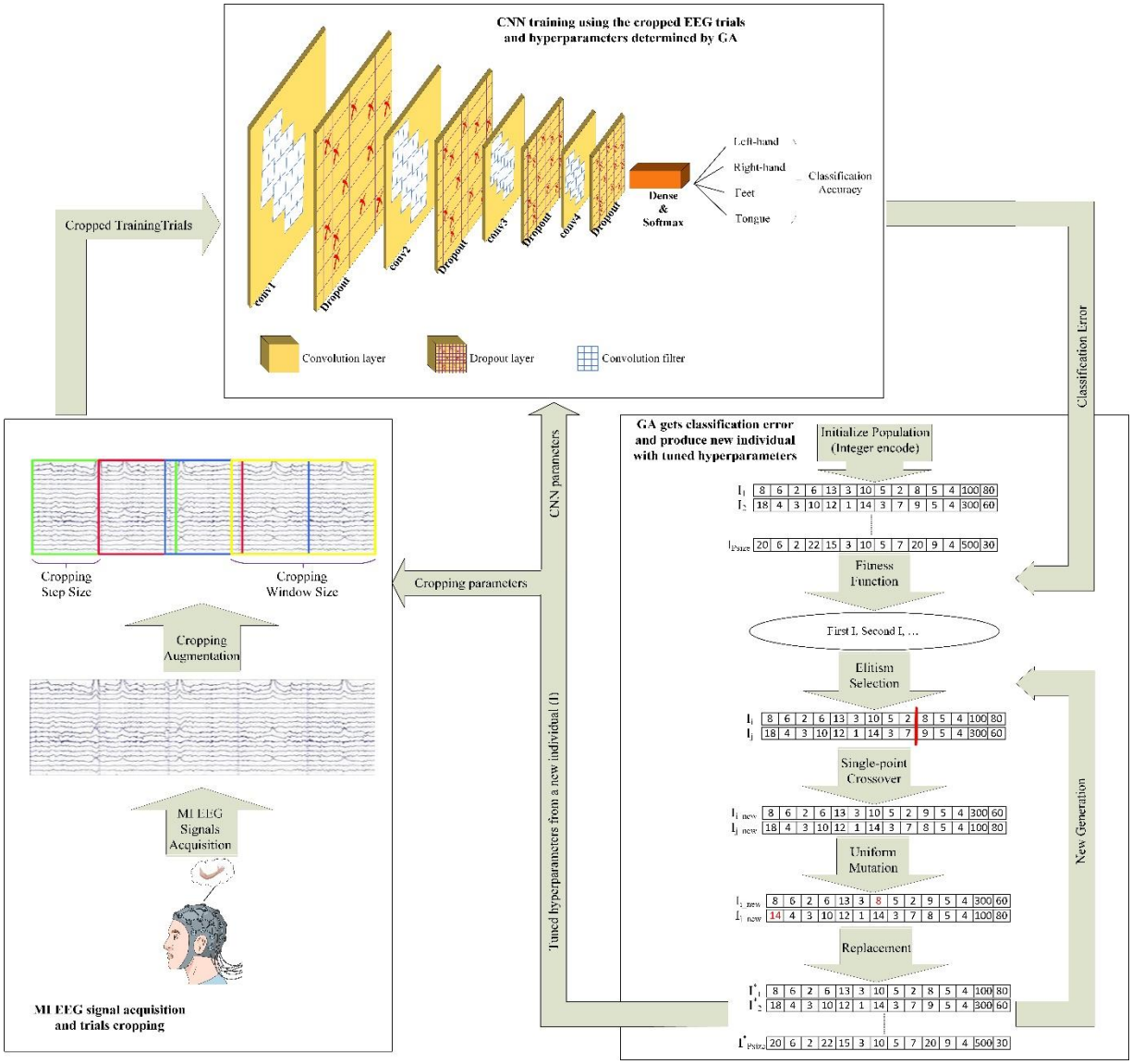

Figure 1. A schematic block diagram of the proposed GACNN model. The BP and cropped trials are used to train a deep CNN. The GA optimizes the number of convolution filters, the size of filters, and dropout probability, in addition to the cropping's window size and step size.

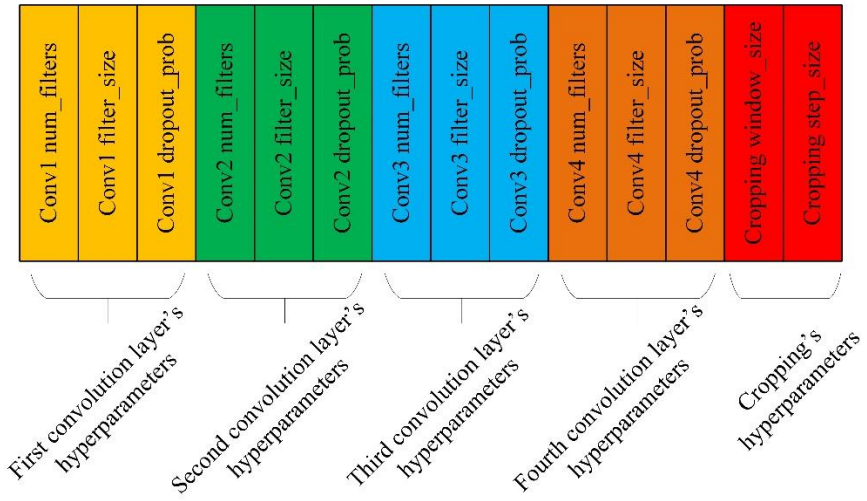

Figure 2. The gene variables of a GA's individual

4. Experimental setup

This section presents the MI EEG dataset and the evaluation metrics for quantifying the experiments. It then provides the details of experimental settings related to the training process and the GA.

4.1 Dataset

The well-known Brain-Computer Interface (BCI) Competition IV 2a (BCIC IV 2a) dataset [34] is analyzed in the experiments. The dataset comprises recordings from nine subjects, each has performed two separate sessions on two different days. The signals are captured using 22 electrodes (channels) placed on the scalp according to the 10-20 international system. Each recorded session consists of 288 trials evenly distributed among four MI tasks: right-hand, left-hand, feet, and tongue, i.e., 72 trials for each task. The trial timing scheme, illustrated in Figure 3, begins with a fixation sign displayed on a screen, followed by a cue indicating one of the tasks for the subject to perform. Afterward, a short break is provided before the next cue is displayed. The signals are sampled at a rate of 250 Hz. The dataset is freely available and can be downloaded from the link: https://www.bbci.de/competition/iv/.

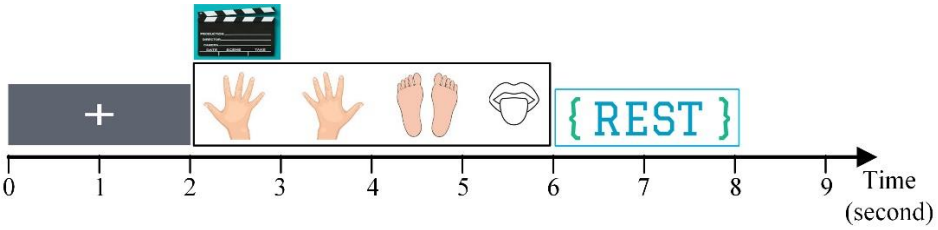

Figure 3. The timing scheme of an MI EEG trial

4.2 Training settings

The experiments are conducted on nine computers to parallelize computations. Each computer is equipped with an AMD EPYC 7282 16-core CPU, 64 GB of RAM, and the Nvidia RTX 4090 GPU. The parallelization approach involves processing the data for each subject on a separate computer. The nine computers are accessed via the online platform vast.ai. This parallelization has accelerated the training process as GA requires multiple passes through the training data to achieve the optimized parameters.

The model's code is written in Python utilizing the following programming environment and machine learning libraries: TensorFlow, Keras, NumPy, scikit-learn, SciPy, Seaborn, and Matplotlib.

The available training data is split into 70% of each MI label for training and 30% for testing. The early stopping mechanism is implemented with 40 epochs of waiting for a better model than the current one, with a maximum of 500 epochs of training in case no more improvement occurs. This technique helps in preserving the best-trained model quickly while preventing overfitting. The learning rate starts with 0.001 and adapts based on the evaluated error. The activation function is the exponential linear unit (ELU). The batch size is set to 100, i.e., the number of trials at each training iteration is 100.

4.3 Genetic algorithm settings

Each GA operation can be applied using various algorithms. Integer encoding is used to represent the gene values of individuals because all hyperparameters have integer values. The number of individuals (population size) is 30. The selection operation is applied utilizing the elitism algorithm. The single point crossover is applied and the uniform type is used for the mutation operation. These GA settings were chosen after several experimental trials.

The ranges of values for the number of filters and filter sizes are not determined for the GA, while the dropout range is 0%-100%, and the range of both window size and step size is 0-1000.

4.4 Evaluation metrics

This paper deals with a classification task with predefined classes: right-hand, left-hand, feet, and tongue. Therefore, the classification accuracy metric (Acc) is suitable for quantifying the proposed model. It can be calculated as follows:

$$Acc(\%) = \frac{TP+TN}{TP+TN+FP+FN} \times 100\% \quad (3)$$

where TP is the true positive rate, TN is the true negative rate, FP is the false positive rate, and FN is the false negative rate.

Other classification metrics are also evaluated to provide deeper insight into the performance of the proposed system. The metrics include precision, sensitivity, specificity, and F1-score. Their formulas are given as follows:

$$Precision = \frac{TP}{TP+FP} \quad (4)$$

$$Sensitivity = \frac{TP}{TP+FN} \quad (5)$$

$$Specificity = \frac{TN}{TN+FP} \quad (6)$$

$$F1-score = \frac{2\times TP}{2\times TP+FP+FN} \quad (7)$$

Cohen's kappa value (κ) measures the consistency of agreement between two raters. it is more robust than classification accuracy as it rejects the impact of true classification occurring by chance. The κ value [35] can be calculated as follows:

$$\text{kappa } (\kappa) = \frac{Acc-EA}{1-EA} \quad (8)$$

Table *1*: Comparison of GACNN results with previous studies on the BCIC IV 2a dataset

. This table includes results from previous studies for comparison purposes. The selected studies use the same analyzed dataset (BCIC IV 2a) for a fair comparison. Two of the papers ([6] and [26]) utilize conventional classification models, while the other two ([27] and [1]) use DL models.

The table presents the classification results in terms of accuracy and kappa. Almost all

where $EA = 1/n$ is the expected accuracy, and n is the number of classes, for instance EA is equal to 0.25 in a 4-class problem. The value of κ ranges from 0 indicating incorrect classifications to 1 indicating correct classification.

The Standard Deviation (SD) is calculated to compare the performance of the proposed model with other previous works. SD indicates the spread of data points about their mean value [36]. A lower SD value means that the data points are close to each other, and in a classification problem, it indicates fair handling of the different sources of observations. For $x_i(i = 1, \dots, N)$ data points, SD is calculated as follows:

$$SD = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \bar{x})^2} \quad (9)$$

where x_i is one of the N values, and $\bar{x}$ is the average of those N values.

The Confusion Matrix (CM) is also provided to present detailed results for each subject, based on the classification of the four MI tasks.

5. Results and discussion

This section discusses the achieved experimental results presenting the performance of GACNN, comparing it to several state-of-the-art studies, and highlighting the best-retrieved values of the optimized hyperparameters.

5.1 Performance of GACNN

The experimental results of the proposed GACNN model for classifying the 4-class MI EEG tasks are presented in

models exhibit poor performance for subjects 2, 5, and 6 compared to the results of other subjects (the worst three results for each model are highlighted in yellow), confirming that GACNN's results are consistent with those of previous studies. However, GACNN achieved the best outcomes for these three subjects, it outperformed the other models by attaining the highest average classification accuracy. The higher kappa value achieved by GACNN indicates better consistency in classification

results as it accounts for the effect of true classifications occurring by chance.

Table 1: Comparison of GACNN results with previous studies on the BCIC IV 2a dataset

Subject	[6] CSD		[26] SS-MEMDBF		[27] EEGNet		[27] Q-EEGNet		[1] CutCat		GACNN	
	Acc (%)	κ	Acc (%)	κ	Acc (%)	κ	Acc (%)	κ	Acc (%)	κ	Acc (%)	κ
1	90.28	0.870	**91.49**	**0.887**	81.10	0.748	81.00	0.747	82.64	0.769	84.00	0.787
2	58.33	0.444	60.56	0.474	52.20	0.363	53.10	0.375	60.76	0.477	**63.10**	**0.508**
3	**97.22**	**0.963**	94.16	0.922	91.30	0.884	91.20	0.883	94.79	0.931	92.70	0.903
4	67.36	0.565	76.72	0.690	59.10	0.455	58.10	0.441	**81.25**	**0.750**	77.40	0.699
5	59.03	0.454	58.52	0.447	68.60	0.581	68.40	0.579	47.22	0.296	**72.40**	**0.632**
6	65.97	0.546	68.52	0.580	52.00	0.360	50.10	0.335	67.36	0.565	**69.00**	**0.587**
7	70.83	0.611	78.57	0.714	76.80	0.691	75.20	0.669	89.58	0.861	**94.20**	**0.923**
8	90.97	0.880	**97.01**	**0.960**	80.00	0.733	81.20	0.749	80.56	0.741	86.90	0.825
9	90.28	0.870	**93.85**	**0.918**	79.30	0.724	79.70	0.729	78.13	0.708	84.40	0.792
Average	*76.70*	*0.689*	*79.93*	*0.732*	*71.16*	*0.615*	*70.89*	*0.612*	*75.81*	*0.677*	**80.46**	**0.739**

Additionally, GACNN outperformed the results of the study that first used cropping augmentation [21]. GACNN achieved increases in classification accuracy of 12% and 10% compared to the deep and shallow networks proposed in the study, respectively. The detailed results for different subjects were not provided in that paper; therefore, they are not listed in

Table *1*: Comparison of GACNN results with previous studies on the BCIC IV 2a dataset

.

The SD results for both accuracy and kappa are presented in

. GACNN achieved the lowest SD value, indicating fair data analysis concerning the different subjects. In other words, GACNN exhibits the least bias among the four MI classes related to subjects.

Figure 4 summarizes GACNN's achievements compared to results of the other studies based on the increments and decrements in accuracy, kappa, and SD. GACNN outperformed all methods in the studies across all metrics, achieving higher accuracy, higher kappa, lower SD for accuracy, and lower SD for kappa.

Table 2: The achieved SD for accuracy and kappa

Models	SD for Acc	SD for κ
CSD [6]	15.33	0.204
SS-MEMDBF [26]	14.99	0.2
EEGNet [27]	13.96	0.186
Q-EEGNet [27]	14.3	0.191
CutCat [1]	14.87	0.198
GACNN	**10.7**	**0.143**

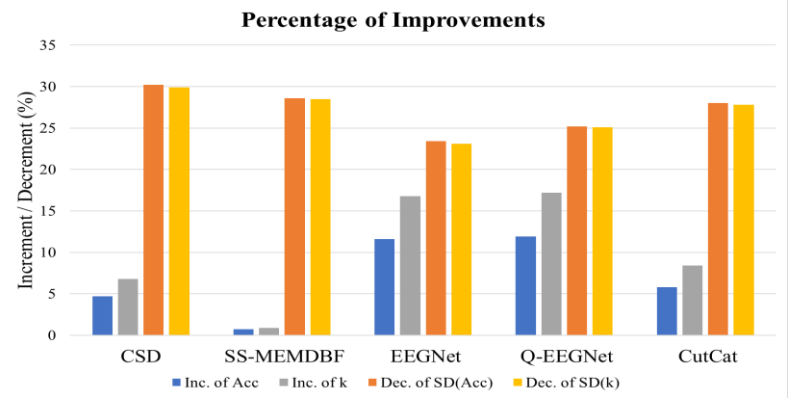

Figure 4. Summary of improvements. Compared to other studies, GACNN demonstrates higher classification accuracy (Acc), higher kappa (κ), and lower standard deviation (SD) for both accuracy and kappa

Further analysis of the achieved performance is provided in

Table *3*: Overall evaluation metrics achieved by GACNN for each MI class

. The table presents results related to precision, sensitivity, specificity, and F1-score, for each MI class. These results demonstrate the satisfactory performance of GACNN, indicating no bias toward any specific MI class.

Table 3: Overall evaluation metrics achieved by GACNN for each MI class

Evaluation metric	Left Hand	Right Hand	Feet	Tongue
Precision	0.80	0.79	0.84	0.82
Sensitivity	0.84	0.79	0.80	0.82
Specificity	0.93	0.92	0.95	0.94
F1-score	0.82	0.79	0.82	0.82

5.2 Sensitivity of GACNN

The confusion matrix (CM) provides a deeper understanding of the experimental results by presenting the classification accuracy for each MI task and subject. **Error! Reference s ource not found.** displays a CM for each subject. The diagonal cells show the percentage of correctly classified test trials, while the off-diagonal cells provide the percentage of incorrectly classified trials. Each row of the CM represents the actual or ground truth labels, while each column represents the predicted labels.

Results presented in many papers such as [1], [37-39] demonstrate variable sensitivity across the different classes of the BCIC IV 2a dataset. Specifically, the classification accuracy for right-hand and left-hand tasks tends to be higher compared to feet and tongue tasks. In contrast, the GACNN results indicate nearly equal accuracy across different subjects and MI classes. This highlights the fair sensitivity of the proposed model in handling these diverse classes. **Error! Reference source not found.** d epicts the average CM for the nine subjects.

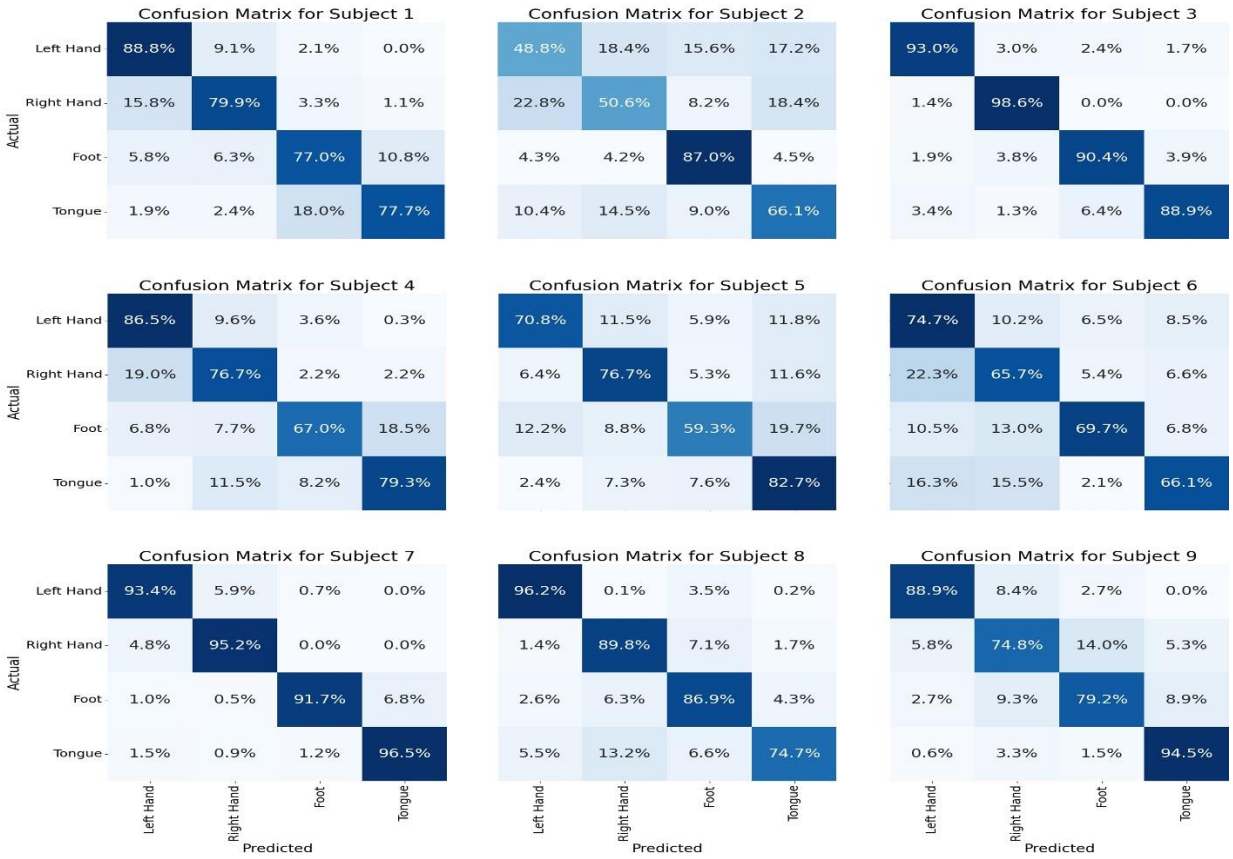

Figure 5. Subjects' confusion matrices (CMs). A CM shows the accuracy of correct classifications in the diagonal cells for each task and incorrect classifications in the off-diagonal cells

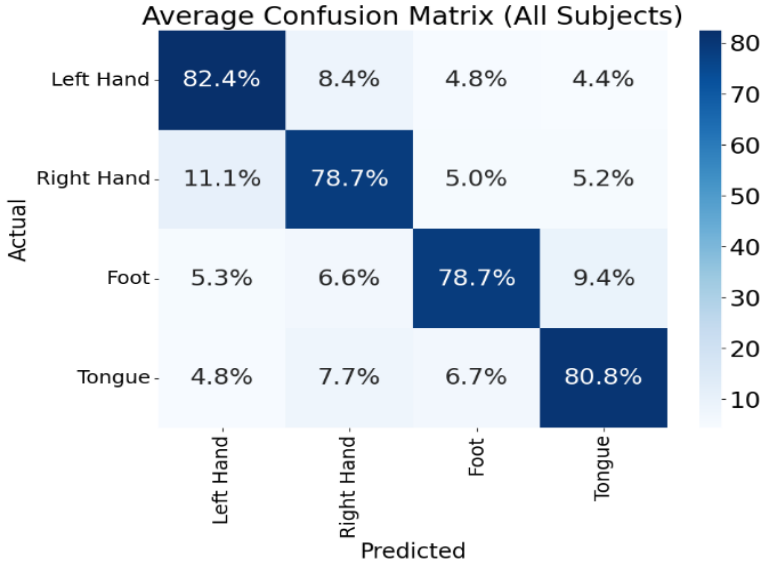

Figure 6. Averaged CM for the nine subjects. GACNN demonstrates fair sensitivity across the four MI tasks

5.3 Optimization results

The GA is employed to optimize the CNN architecture including the number of convolution filters, size of filters, and dropout probability. Table 4 outlines the values obtained for these three hyperparameters in each CNN layer.

Given that EEG signals exhibit subject-specific features [26], [40], it is essential to optimize the network architecture for each subject. Subject-specific models are applicable in various domains such as stroke therapy,

wheelchair control, and sleep disorder treatment.

Moreover, as previously mentioned, GA searched for suitable settings for cropping augmentation. Figure 6 illustrates the model performance regarding fitness versus the cropping window size and step size. The range of collected crops starts from the beginning of the MI trial (second 2) and ends 0.5 seconds after the end of the MI trial (second 6.5). The retrieved values are a window size of 1000 and a step size of 5. These values ensure proper capturing of latent features and generate more training trials. Logically, a shorter crop contains less information, while a longer crop provides a feasible amount of information. This indicates that ERS and ERD features persist for about four seconds after completing the MI task.**Error! R eference source not found.** illustrates the convergence of the optimization process achieved by the GA. The process required 113 generations to reach maximum fitness, i.e., the best hyperparameters.

It is worth mentioning that the complexity of GACNN's algorithm, arising from the repeated training process at each generation impacts only the training stage. This complexity does not affect the resulting model as the training occurs only once; thereafter, the resulting CNN model can be employed in real devices.

GACNN demonstrated superior performance compared to previous studies, achieving a 17% increase in kappa and a 25% decrease in SD relative to the Q-EEGNet model, a well-known benchmark model in EEG analysis. This confirms that the automatic selection of the CNN's hyperparameters performed by GA has influenced the final CNN model. The approach is particularly beneficial for real-world applications such as wheelchair control through thought, motor rehabilitation, stroke therapy, and sleep disorder treatment, all of which require accurate classifications.

Table 4: Details of the CNN architecture. The GA optimized the number of convolution filters, size of filters, and dropout probability for each subject

Subject 1			
	Nu. of filters	Filter size	Dropout
Conv1	39	22 x 1	0.12
Conv2	57	1 x 7	0.28
Conv3	251	1 x 12	0.37
Conv4	472	1 x 6	0.23

Subject 2			
	Nu. of filters	Filter size	Dropout
Conv1	54	22 x 1	0.17
Conv2	121	1 x 7	0.45
Conv3	98	1 x 2	0.5
Conv4	386	1 x 2	0.32

Subject 3			
	Nu. of filters	Filter size	Dropout
Conv1	30	22 x 1	0.22
Conv2	80	1 x 24	0.32
Conv3	192	1 x 8	0.3
Conv4	290	1 x 12	0.32

Subject 4			
	Nu. of filters	Filter size	Dropout
Conv1	64	22 x 2	0.23
Conv2	121	1 x 1	0.41
Conv3	242	1 x 7	0.18
Conv4	334	1 x 9	0.48

Subject 5			
	Nu. of filters	Filter size	Dropout
Conv1	47	22 x 3	0.12
Conv2	33	1 x 6	0.16
Conv3	239	1 x 19	0.35
Conv4	360	1 x 27	0.3

Subject 6			
	Nu. of filters	Filter size	Dropout
Conv1	54	22 x 2	0.38
Conv2	123	1 x 7	0.32
Conv3	126	1 x 21	0.26
Conv4	162	1 x 30	0.23

Subject 7

Subject 8

Subject 9

	Nu. of filters	Filter size	Dropout
Conv1	41	22 x 5	0.41
Conv2	57	1 x 8	0.32
Conv3	242	1 x 7	0.43
Conv4	425	1 x 28	0.11

	Nu. of filters	Filter size	Dropout
Conv1	54	22 x 2	0.23
Conv2	94	1 x 9	0.28
Conv3	242	1 x 7	0.37
Conv4	472	1 x 12	0.48

	Nu. of filters	Filter size	Dropout
Conv1	37	22 x 2	0.38
Conv2	72	1 x 7	0.46
Conv3	114	1 x 27	0.5
Conv4	492	1 x 22	0.15

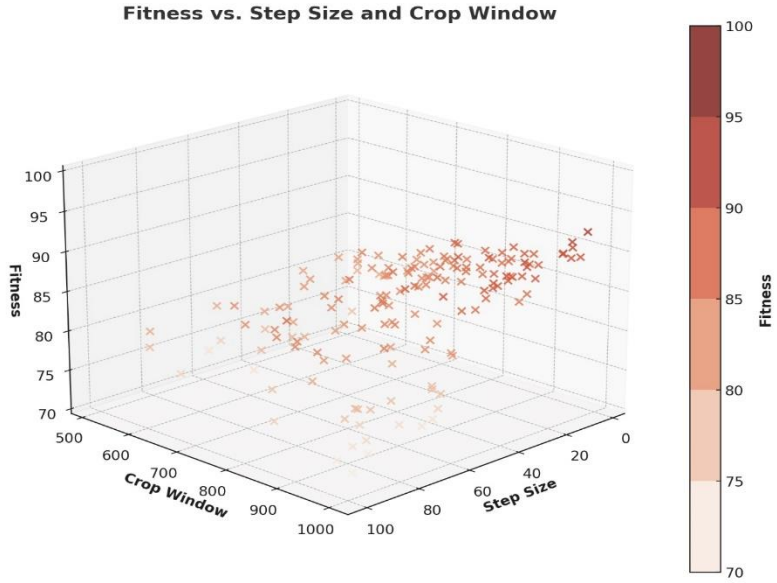

Figure 6. Optimization of cropping parameters. The individuals' fitness is evaluated across generations for various values of cropping window size and step size

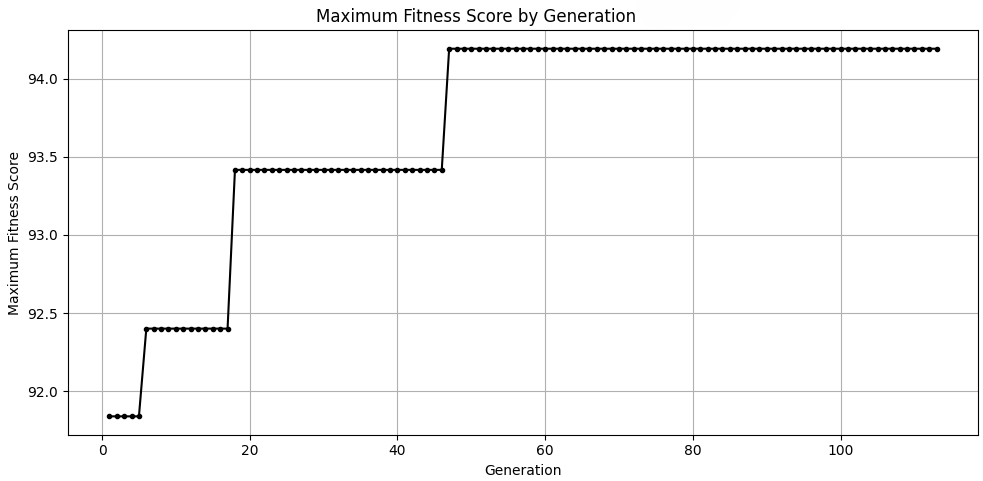

Figure 8. The convergence of GA over generations of individuals

6. Conclusion

The GACNN model is proposed in this study to address the overestimation problem in designing a deep network. GACNN employs the GA to supervise the construction of the CNN layers by optimizing the number of filters, size of filters, and dropout probability at each layer. Moreover, the model optimizes the window size and step size of cropping augmentation to generate adequate and appropriate training samples.

Practical results demonstrate the superiority of GACNN compared to state-of-the-art studies.

It achieved high classification accuracy and fair sensitivity across different MI classes.

Consequently, GACNN offers several advantages, including eliminating the burden on the programmer, preventing the trial-and-error method of selecting crucial hyperparameters related to the network and cropping augmentation, and producing a problem-fit deep network.

The only disadvantage of GACNN is its requirement for a significant amount of time to complete the training process as GA makes a small step towards optimal hyperparameter values after an entire generation. However, this issue is not a major concern, as the training process occurs only once; thereafter, the model is ready for deployment.

Additionally, the proposed model applies to other types of signals, particularly in the context of big data analysis. For future work, the following directions are suggested:

- The experiments in this paper utilized offline training. The model may be examined in online training scenarios, like weather forecasting or stock market analysis.
- The GACNN model may be extended to other types of signals beyond EEG, such as developing models for video classification or video captioning, as video analysis deals with massive datasets.

References

[1] A. Al-Saegh, S. A. Dawwd, and J. M. Abdul-Jabbar, "CutCat: An augmentation method for EEG classification," *Neural Networks*, vol. 141, pp. 433–443, Sep. 2021, doi: 10.1016/j.neunet.2021.05.032.

[2] S. R. Sreeja, Himanshu, and D. Samanta, "Distance-based weighted sparse representation to classify motor imagery EEG signals for BCI applications," *Multimed Tools Appl*, vol. 79, pp. 13775–1379, 2020, doi: 10.1007/s11042-019-08602-0.

[3] Z. T. Al-Qaysi, A. Al-Saegh, A. Hussein, and M. A. Ahmed, "Wavelet-based Hybrid Learning Framework for Motor Imagery Classification," *Iraqi Journal for Electrical and Electronic Engineering*, vol. 19, no. 1, pp. 47–56, 2023, doi: 10.37917/ijeee.19.1.6.

[4] R. Alazrai, M. Abuhijleh, H. Alwanni, and M. I. Daoud, "A Deep Learning Framework for Decoding Motor Imagery Tasks of the Same Hand Using EEG Signals," *IEEE Access*, vol. 7, pp. 109612–109627, 2019, doi: 10.1109/access.2019.2934018.

[5] A. Al-Saegh, "Comparison of Complex-Valued Independent Component Analysis Algorithms for EEG Data," *Iraqi Journal for Electrical and Electronic Engineering*, vol. 15, no. 1, pp. 1–12, 2019, doi: 10.37917/ijeee.15.1.1.

[6] H. Raza, H. Cecotti, Y. Li, and G. Prasad, "Adaptive learning with covariate shift-detection for motor imagery-based brain–computer interface," *Soft comput*, vol. 20, no. 8, pp. 3085–3096, 2016, doi: 10.1007/s00500-015-1937-5.

[7] H. Wu *et al.*, "A Parallel Multiscale Filter Bank Convolutional Neural Networks for Motor Imagery EEG Classification," *Front Neurosci*, vol. 13, no. November, pp. 1–9, 2019, doi: 10.3389/fnins.2019.01275.

[8] A. Daood, A. Al-saegh, and A. F. Mahmood, "Handwriting Detection and Recognition of Arabic Numbers and Characters Using Deep Learning Methods," *Journal of Engineering Science and Technology*, vol. 18, no. 3, pp. 1581–1598, 2023.

[9] M. Ghanim, A. Mohammed, and A. Sali, "Arabic/English Handwritten Digits Recognition using MLPs, CNN, RF, and CNN-RF," *Al-Rafidain Engineering Journal (AREJ)*, vol. 28, no. 2, pp. 252–260, 2023, doi: 10.33899/rengj.2023.138592.1236.

[10] R. Jamal Kolaib and J. Waleed, "Crime Activity Detection in Surveillance Videos Based on Developed Deep Learning Approach," *Diyala Journal of Engineering Sciences*, vol. 17, no. 3, pp. 98–114, Sep. 2024, doi: 10.24237/djes.2024.17307.

[11] A. J. Yousif and M. H. Al-Jammas, "Real-time Arabic Video Captioning Using CNN and Transformer Networks Based on Parallel Implementation," *Diyala Journal of Engineering Sciences*, vol. 17, no. 1, pp. 84–93, 2024, doi: 10.24237/djes.xxxx.13301.

[12] N. Y. Abdullah and ad Ahmed Al Kazzaz, "Evaluation of Physiotherapy Exercise by Motion Capturing Based on Artificial Intelligence: A Review," *Al-Rafidain Engineering Journal (AREJ)*, vol. 28, no. 2, pp. 237–251, 2023, [Online]. Available: http://creativecommons.org/licenses/by/4.0/

[13] M. Ramzan and S. Dawn, "Fused CNN-LSTM deep learning emotion recognition model using electroencephalography signals," *International*

Journal of Neuroscience, vol. 133, no. 6, pp. 587–597, 2023, doi: 10.1080/00207454.2021.1941947.

[14] A. Sawan, M. Awad, R. Qasrawi, and M. Sowan, "Hybrid deep learning and metaheuristic model-based stroke diagnosis system using electroencephalogram (EEG)," *Biomed Signal Process Control*, vol. 87, p. 105454, 2024, doi: https://doi.org/10.1016/j.bspc.2023.105454.

[15] F. Hassan and S. F. Hussain, "Review of EEG Signals Classification Using Machine Learning and Deep-Learning Techniques," in *Advances in Non-Invasive Biomedical Signal Sensing and Processing with Machine Learning*, S. M. Qaisar, H. Nisar, and A. Subasi, Eds., Cham: Springer International Publishing, 2023, pp. 159–183. doi: 10.1007/978-3-031-23239-8_7.

[16] K. M. Hossain, M. A. Islam, S. Hossain, A. Nijholt, and M. A. R. Ahad, "Status of deep learning for EEG-based brain–computer interface applications," *Front Comput Neurosci*, vol. 16, no. i, 2023, doi: 10.3389/fncom.2022.1006763.

[17] A. Craik, Y. He, and J. L. Contreras-Vidal, "Deep learning for electroencephalogram (EEG) classification tasks: A review," *J Neural Eng*, vol. 16, no. 3, 2019, doi: 10.1088/1741-2552/ab0ab5.

[18] H. Wu, P. Judd, X. Zhang, M. Isaev, and P. Micikevicius, "Integer quantization for deep learning inference: Principles and empirical evaluation," *ArXiv*, pp. 1–20, 2020.

[19] J. Gao, P. Li, Z. Chen, and J. Zhang, "A Survey on Deep Learning for Multimodal Data Fusion," *Neural Comput*, vol. 32, pp. 1–36, 2020, doi: 10.1162/NECO.

[20] C. He, J. Liu, Y. Zhu, and W. Du, "Data Augmentation for Deep Neural Networks Model in EEG Classification Task: A Review," *Front Hum Neurosci*, vol. 15, no. December, 2021, doi: 10.3389/fnhum.2021.765525.

[21] R. T. Schirrmeister *et al.*, "Deep learning with convolutional neural networks for EEG decoding and visualization," *Hum Brain Mapp*, vol. 38, no. 11, pp. 5391–5420, 2017, doi: 10.1002/hbm.23730.

[22] J. S. Kirar and R. K. Agrawal, "A combination of spectral graph theory and quantum genetic algorithm to find relevant set of electrodes for motor imagery classification," *Appl Soft Comput*, vol. 97, no. xxxx, p. 105519, 2020, doi: 10.1016/j.asoc.2019.105519.

[23] Z. A. A. Alyasseri, A. T. Khader, M. A. Al-Betar, A. K. Abasi, and S. N. Makhadmeh, "EEG Signals Denoising Using Optimal Wavelet Transform Hybridized with Efficient Metaheuristic Methods," *IEEE Access*, vol. 8, pp. 10584–10605, 2020, doi: 10.1109/ACCESS.2019.2962658.

[24] T. Wilaiprasitporn, A. Ditthapron, K. Matchaparn, T. Tongbuasirilai, N. Banluesombatkul, and E. Chuangsuwanich, "Affective EEG-Based Person Identification Using the Deep Learning Approach," *IEEE Trans Cogn Dev Syst*, vol. 12, no. 3, pp. 486–496, Sep. 2020, doi: 10.1109/TCDS.2019.2924648.

[25] Z. hademi, F. Ebrahimi, and H. M. Kordy, "A transfer learning-based CNN and LSTM hybrid deep learning model to classify motor imagery EEG signals," *Comput Biol Med*, vol. 143, p. 105288, 2022, doi: 10.1016/j.compbiomed.2022.105288.

[26] P. Gaur, R. B. Pachori, H. Wang, and G. Prasad, "A multi-class EEG-based BCI classification using multivariate empirical mode decomposition based filtering and Riemannian geometry," *Expert Syst Appl*, vol. 95, pp. 201–211, 2018, doi: 10.1016/j.eswa.2017.11.007.

[27] T. Schneider, X. Wang, M. Hersche, L. Cavigelli, and L. Benini, "Q-EEGNet: An energy-efficient 8-bit quantized parallel EEGNet implementation for edge motor-imagery brain-machine interfaces," in *2020 IEEE International Conference on Smart Computing (SMARTCOMP)*, Bologna, Italy: IEEE, 2020, pp. 284--289. doi: 10.1109/smartcomp50058.2020.00065.

[28] C. Yaacoub, G. Mhanna, and S. Rihana, "A genetic-based feature selection approach in the identification of left/right hand motor imagery for a brain-computer interface," *Brain Sci*, vol. 7, no. 1, 2017, doi: 10.3390/brainsci7010012.

[29] Q. Mastoi *et al.*, "Novel DERMA fusion technique for ECG heartbeat classification," *Life*, vol. 12, no. 6, p. 842, 2022, doi: 10.3390/life12060842.

[30] A. A. Mutlag *et al.*, "Multi-agent systems in fog--cloud computing for critical healthcare task management model (CHTM) used for ECG monitoring," *Sensors*, vol. 21, no. 20, p. 6923, 2021, doi: 10.3390/s21206923.

[31] A. H. Ameen, M. A. Mohammed, and A. N. Rashid, "Enhancing Security in IoMT: A Blockchain-Based Cybersecurity Framework for Machine Learning-Driven ECG Signal Classification," *Fusion: Practice and Applications*, vol. 14, no. 1, pp. 221–251, 2024, doi: 10.54216/FPA.140117.

[32] M. Z. Alom *et al.*, "A state-of-the-art survey on deep learning theory and architectures," *Electronics (Switzerland)*, vol. 8, no. 3, pp. 1–67, 2019, doi: 10.3390/electronics8030292.

[33] A. Al-Saegh, S. A. Dawwd, and J. M. Abdul-Jabbar, "Deep learning for motor imagery EEG-based classification: A review," Jan. 01, 2021, *Elsevier Ltd*. doi: 10.1016/j.bspc.2020.102172.

[34] M. Tangermann *et al.*, "Review of the BCI competition IV," 2012. doi: 10.3389/fnins.2012.00055.

[35] K. Zhang *et al.*, "Data augmentation for motor imagery signal classification based on a hybrid neural network," *Sensors (Switzerland)*, vol. 20, no. 16, pp. 1–20, 2020, doi: 10.3390/s20164485.

[36] G. Varone *et al.*, "A comprehensive machine-learning-based software pipeline to classify EEG signals: A case study on PNES vs. control subjects," *Sensors (Switzerland)*, vol. 20, no. 4, Feb. 2020, doi: 10.3390/s20041235.

[37] S. U. Amin, M. Alsulaiman, G. Muhammad, M. A. Mekhtiche, and M. Shamim Hossain, "Deep Learning for EEG motor imagery classification based on multi-layer CNNs feature fusion," *Future Generation Computer Systems*, vol. 101, pp. 542–554, 2019, doi: 10.1016/j.future.2019.06.027.

[38] X. Zhang, M. Lei, and Y. Li, "An amplitudes-perturbation data augmentation method in convolutional neural networks for EEG decoding," in *2018 5th International Conference on Information, Cybernetics, and Computational Social Systems (ICCSS)*, IEEE, 2018, pp. 231–235.

[39] B. Du, Y. Liu, and G. Tian, "Improving Motor Imagery EEG Classification by CNN with Data Augmentation," in *2020 IEEE 19th International Conference on Cognitive Informatics & Cognitive Computing (ICCI*CC)*, Beijing, China: IEEE, 2020, pp. 111–118. doi: 10.1109/iccicc50026.2020.9450227.

[40] T. Z. et Al., "Subject-specific EEG channel selection using non-negative matrix factorization for lower-limb motor imagery recognition," *J Neural Eng*, p. https://doi.org/10.1088/1361-6463/aad7de, 2020.