# Toward Lightweight Fusion Of AI Logic and EEG Sensors to Enable Ultra Edge-based EEG Analytics on IoT Devices

**Tahrat Tazrin**

Supervisor: Dr. Zubair Md. Fadlullah
Dr. Quazi Abidur Rahman

Department of Computer Science
Lakehead University

This dissertation is submitted for the degree of
*MSc. in Computer Science (Specialization in Artificial Intelligence)*

May 2021

# Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. This dissertation is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text and Acknowledgements. This dissertation contains fewer than 65,000 words including appendices, bibliography, footnotes, tables and equations and has fewer than 150 figures.

Tahrat Tazrin
May 2021

# Acknowledgements

I would like to express my deepest gratitude to my incredible thesis supervisors Dr. Zubair Md. Fadlullah and Dr. Quazi Abidur Rahman for their immense support and guidance throughout my master's journey. I am truly grateful that they gave me the opportunity to do my thesis under their supervision. Their intelligence and kind advice helped me learn plenty of invaluable lessons about research in computer science and have a delightful experience in my master's education. I would also like to thank Dr. Mostafa Fouda, from department of electrical and computer engineering in Idaho State University, for his insightful suggestions and guidance, which helped me expand my knowledge for research in my field. I express my utmost appreciation towards my family members for their love, encouragement, prayers and sacrifice to help me prepare for the betterment of my future. They have constantly motivated me to move forward through thick and thin and inspired me to grow as an individual. Lastly, I would like to appreciate my wonderful friends and colleagues at Lakehead University who have given me moral support throughout my journey and lent their helping hands from time to time.

# Abstract

Electroencephalogram (EEG) analysis has garnered attention in the research domain due to its ability to detect various neural activities starting from brain seizures to a person's concentration level. To make it more beneficial for the users, it is important to miniaturize the currently available clinical-grade large EEG monitors to wearables which can provide decisions at the edge. Traditionally, for performing such analysis, the raw EEG signals are collected at the edge which is then transferred to the cloud where the data is interpreted and forwarded accordingly. However, this method of transferring the user data for analysis poses a risk of security and privacy breach as well as consumes a considerable bandwidth and time which makes it inefficient in terms of scalability. In this vein, we investigated on transferring the Artificial Intelligence (AI)-logic of the analysis to the sensors, so that a localized decision can be made on the edge, without transferring the data, thus saving precious bandwidth and restoring privacy of the users. However, the main challenge in achieving such a scenario is the devices' inability to perform complex computations due to its resource constraints. Hence, we have explored various AI-based techniques throughout this thesis to find out a lightweight model which will be able to give a decent performance while consuming lower resources. We have validated our candidate models in various use-cases throughout the chapters to compute the performance of the AI models. It is believed that, this type of analysis can encourage the sensor foundries to integrate AI-logic with wearable sensors, to conduct localized EEG analysis on the sensor level, which will be more practical, cheaper, and scalable.

# Table of contents

# List of figures

# List of tables

# Chapter 1

# Introduction

## 1.1 Overview

The Electroencephalogram (EEG) has emerged as a buzzword in the Brain Computer Interface (BCI) domain due to its numerous advantages. It is a method of recording electrical activity of the brain by detecting the brain waves. The brain is very sensitive to alterations due to various activities and the consequent changes can be observed through EEG signals for e.g. the response after seeing a particular image, or speaking a particular word, or reacting to a particular event and so forth. The process of collecting the signals are non invasive while providing a high temporal resolution of the electrical activity. This is also very uncomplicated and cost effective which make it a popular approach for detecting any brain activities. The signals are commonly used to detect crucial activities such as epilepsy, seizures or similar brain disorders of a person to other abnormal brain activities such as sleep disorders, coma and brain death. Typically, the EEG signals are collected over multiple channels as shown in Fig. 1.1, since the exact location of the activated neuron in our brain is not known in most cases, and sometimes it occurs with a particular combination. It is believed that, if particular EEG signal values for a physiological risk can be determined, the severity of any damage or action can be assessed and accordingly addressed and mitigated. In a traditional setting, several small electrodes and wires are connected to different part of scalp as well as forehead and back of the neck in order to record brain's electrical activities i.e the EEG signals. The voltage fluctuations are then measured which is a result of ionic current change in the neuron of the brain. To make it more within the reach of people, there are also various commercially available EEG sensors which are able to detect the electrical activities of the brain very easily. These EEG headset/headbands are non-invasive, consumer-grade wearable devices that require no extra setups in contrast with the clinical grade EEG machines which are typically wired and comprise

of wet electrodes [1]. Some of the most popular devices among them include Neurosky, Emotiv, Muse, OpenBCI etc.

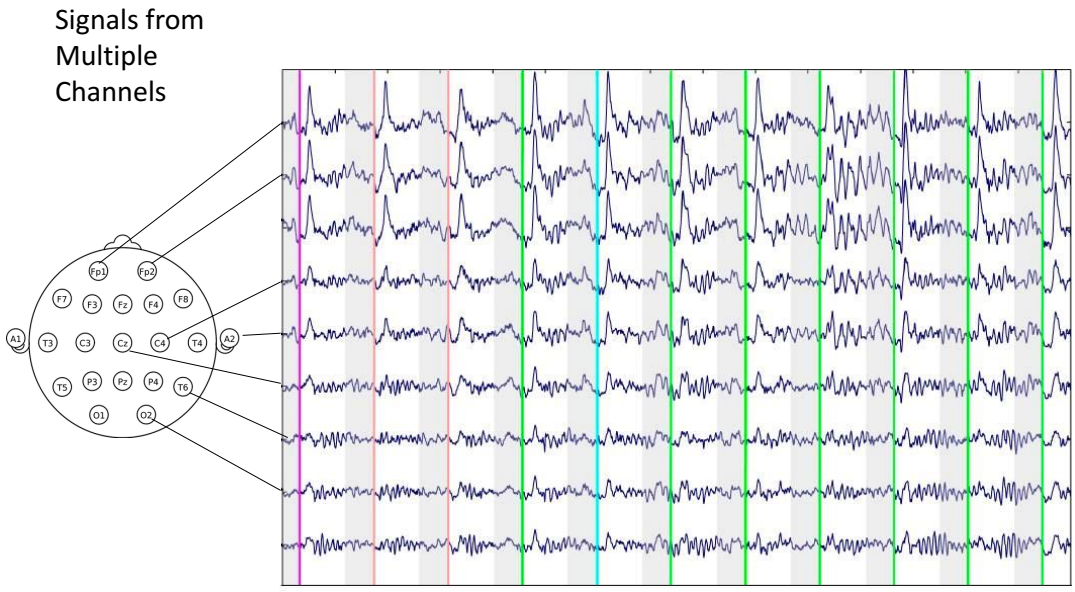



Fig. 1.1 Representation of EEG signals collected over various channels.

As stated above, the EEG signals are indicators of various crucial diseases, which is why several machine learning and deep learning techniques are implemented on the EEG signals to learn and determine particular patterns which is suggestive of a specific task. EEG analytics are therefore emerging as a hot area of research for continuous, prolonged, and non-intrusive monitoring. Various Artificial Intelligence (AI)-based techniques has been very effectively and widely used to determine various activities of a person starting from anesthesia monitoring during and post surgery, to detecting concentration level of a person. These neural activity monitoring are usually done beyond hospital settings which makes it more popular and desirable among the users. Such AI-based analysis has revolutionized various application in the field of BCI such as by introducing brain controlled wheelchairs [2] for the physically challenged people, word prediction, rehabilitation of patients undergoing stroke [3, 4], mental state detection devices [5], and so forth [6, 7]. The process of analysing the EEG signals usually comprises of a series of steps which include signal acquisition, processing, feature extraction and selection and classification. The multiple steps lead to a very accurate AI-based classification model. There are several types of classifier used for the purpose such as the heavily discussed AI-based classifiers, adaptive classifiers, matrix and tensor-based classifiers. The process of EEG data manipulation is carried out in two steps, firstly to train a machine learning model offline using a training data which

is collected prior to the training, based on a particular activity, and tuning various hyperparameters of the AI model to make it robust. Secondly an online interface that will collect the EEG signals and determine the pattern of the signal from the trained model and send to users. This is usually presented to the users using the concepts of Internet of Things (IoT).

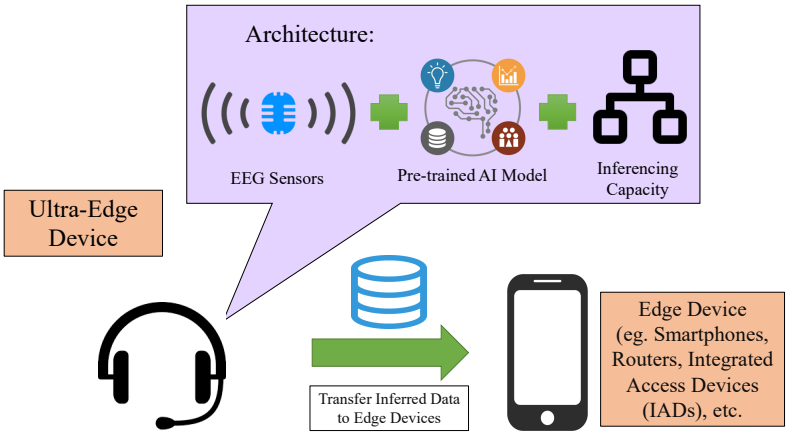



Fig. 1.2 Proposed logic-in-headbands-based edge analytics framework to facilitate localized EEG computing.

The recently emerging applications of the IoT and wearable devices are dramatically transforming the digital health landscape by offering continuous, efficient, portable as well as convenient monitoring systems [8, 9]. IoT-based health data gathering followed by cloud-based process and medical analytics have made the users more attentive toward their health. In a traditional IoT scenario, the EEG data to be analysed are collected over sensors which are connected to the internet. The collected data from the sensors are usually passed to the cloud or edge server for further processing and analysis. The entire process consumes precious bandwidth and adds communication delay while raising privacy as well as security concerns [10]. Previously, various edge/fog computing approaches were proposed to reduce data retrieval latency [11]. To facilitate a speedier and efficient data analysis method, the computation tasks need to be migrated to even further along the network edge, i.e., at the data generation point. Therefore, if a localized EEG analysis can be performed on the IoT device itself, the process of data collection and analysis can be made seamless, swift, and secure. However, this requires a lot of computational resources which poses a burden on the traditional IoT devices (i.e., ultra-edge nodes [12]). The IoT nodes are typically resource-constrained which makes it difficult to enable such analysis over the edge. Motivated by the paradigm shift of edge computing and IoT for continuous monitoring, we focus on the first step to carry out EEG edge analytics at the last frontier, which is the ultra-edge of our considered cyber-physical system for ensuring users' convenience and privacy. We propose the

concepts of a lightweight model incorporated with the commercially available EEG headbands that will have inferring capability at the edge as shown in Fig. 1.2. As it can be seen in the figure that the ultra-edge device consists of EEG sensors to collect EEG data, the selected optimal algorithm as well as the ability to make an inference. The EEG analytics will be carried out in the ultra-edge node thus avoiding any communication with remote servers and the data will then be transferred to the edge devices such as smartphones, routers, etc. To overcome challenges due to computational and energy resource constraints of IoT devices (e.g., EEG headbands/headsets), we envision a smart, lightweight model, which can be seamlessly incorporated with the consumer-grade EEG headsets to reduce delay and bandwidth consumption. By systematically investigating various machine/deep learning models, we identify and select the best model for our envisioned system.

## 1.2 Problem Description

We aim to conduct brainwave signal capturing and analysis directly on IoT and wearable devices, such as EEG-headsets/headbands. It is believed that, such analysis will be able to reduce the communication delay and bandwidth while also securing data. However, this is challenging as the conventional approaches of EEG analysis entails a series of complicated steps starting from signal processing to feature extraction to classification. Due to the resource constraints of the ultra-edge IoT devices, it becomes difficult to enable such heavy computations within the device. Therefore, it is important to limit the use of computational resources as much as possible while ensuring a precise classification model. This can be done by eliminating some of the crucial steps of EEG analysis and using a lightweight classifier, but at the same time making sure the performance of the classification model does not degrade in the process. A matrix and tensor based classifier called Riemannian geometry [13] has recently gained its popularity among the classification methods due to its precise detection of patterns in the signal using covariance matrices. Several research are being conducted in the domain to improve the algorithm due to its ample advantage in classifying the EEG signals. It uses a closed form equation which also makes it suitable for such logic-in-headbands scenario. However, it employs a spatial covariance matrix to extract the spatio-temporal features of the signal, which increases its complexity with increasing number of channels. Since our aim is to conduct a lightweight analysis for the convenience of the users, we need to ensure that the inference is done in polynomial time. In this vein, we considered another variety of classifiers which is the AI-based classification models. The AI-based models have a much better generalization capability and good performance, while having a reduced inferring time.

In addition to that, the computational delay can be further reduced by minimizing the number of signal processing steps which also adds complexity to the computation. Usually, to get a better representation of the data in both temporal and spectral domain, a signal transformation technique is applied such as Fourier Transform, Wavelet Decomposition and so forth. These methods are used to increase the performance of the model while reducing the computational load of the AI-models by reducing the dimension and providing a better representation of the data. Yet, there is a computational complexity of these transformation steps itself. With increasing number of inputs, it is noticed that the complexity of the models increase exponentially. Thus the problem becomes challenging to be solved in polynomial time which is not ideal for an ultra-edge IoT device. Therefore, we further explored in the AI domain to find out the an algorithm which will be able to classify the raw EEG signals directly, thus getting rid of some of the preprocessing steps if EEG analysis. Therefore, we systematically investigate machine/deep learning techniques to identify the candidate light-weight AI logic/inference models required for the EEG data analytics on these ultra-edge IoT devices.

## 1.3 Contribution

In this dissertation, we tried to address a key question which is: how can we miniaturize currently available clinical-grade large EEG monitors to wearables which can provide automated/local decisions? The solution to this question requires a major paradigm shift in computing. In this vein, we investigate how to migrate the AI-based edge computing on raw EEG data to the resource-constrained sensors/devices at the final frontier (ultra-edge) of our considered system. To enable that, we assumed that the ultra-edge EEG analysis devices will have the same configuration as the standard IoT devices. We used two use-cases to test our proposed AI models. We also explored various algorithms to find out the most suitable algorithm for integrating with the ultra-edge. We computed the inference time and memory consumption of the models to see if they are suitable to be used in the ultra-edge nodes. Our proof-of-concept model aims at encouraging the sensor foundries to integrate AI logic with wearable sensors, to conduct localized EEG analysis on the ultra-edge devices which will be more practical, cheaper, and scalable despite limited resources. In the future, the AI-based classification model can be deployed on various platforms for a plethora of tasks, which may range from smart device control to continuous and non-intrusive monitoring of post-surgical patients while rehabilitating at home, non-intrusive sleep monitoring, etc.

## 1.4 Organization of Thesis

The remainder of the dissertation proceeds as follows. Chapter 2 presents various methods in the existing literature that are commonly used for EEG analysis and how they are used in the IoT platforms. Chapter 3 describes the explored methodologies in detail and their various characteristics which made it a suitable candidate in the systematic analysis. In chapter 4, we present our first use-case to select a suitable classification model for the ultra-edge IoT analysis. Later in chapter 5, we presented another use-case to further reduce the complexity of the ultra-edge analysis model by eliminating signal transformation steps using a deep learning based hybrid approach. And finally, chapter 6 concludes the dissertation.

# Chapter 2

# Background and Related Works

EEG data analysis is an interesting domain of research as the behaviour of a human being (eg. concentration level, emotions etc.) as well as various critical diseases (eg. stroke, seizures etc.) can be deducted from it. It exhibits a high temporal resolution of a human brain, which makes it suitable for these type of critical detections and predictions. At the same time, it is also a particularly challenging field due to various properties of the signals such as their high dimension, spatial and temporal covariance as well as its non-stationary property. This is why various signal and noise processing techniques are used to make the EEG data more comprehensible for the classification models. Another important challenge with this type of data is the lack of ground truth which makes it difficult to establish a standard dataset to compare the performances like the MNIST digit dataset for image classification. Usually, the implementation of such EEG analysis in real world to enable Internet of Things (IoT) based analysis are done in a cloud or edge-based environment. In this section, various methods of EEG analysis (as depicted in Fig. 2.1) in the literature are reviewed.

## 2.1 EEG Signal Processing

### 2.1.1 Noise and artefact removal of the raw signals

Once the EEG signals are collected from a device, the signals contain various noise and artefacts such as, Electrocardiography (ECG) signals, Electrooculography (EOG) signals and so forth. These noise and artefacts can provide misleading information to the classifier which is not significant to the data analysis. Thus it is important to remove such features for a clearer interpretation of the collected brain signals. There are various techniques that are used in the literature for the stated purpose namely Common Spatial Patterns (CSP), Principal Component Analysis (PCA), Independent Component Analysis (ICA), adaptive filtering approach and so forth [14–17]. Among them, the ICA

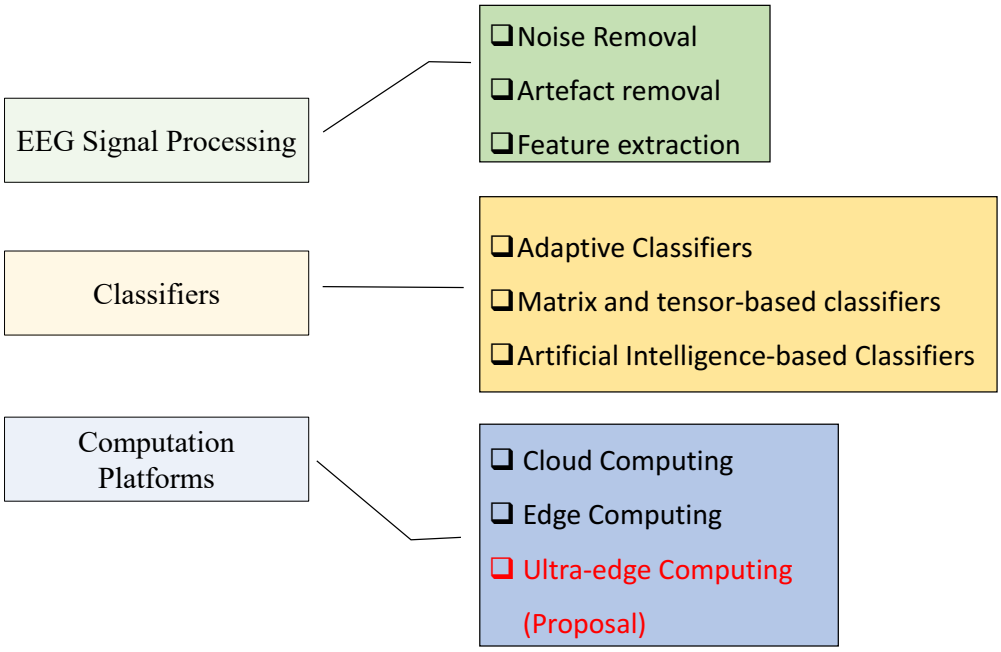



Fig. 2.1 The processing techniques of the EEG signals in the literature.

has been established as the most effective method of removing noise and artefacts due to its good performance as well as efficiency [18]. The ICA technique has the capability of separating the artefacts of the EEG signals into independent components solely based on the signals and no other features such as their channel information. It is also very effective for larger datasets. The PCA on the other hand is a widely used method for representing data as well as reduce dimensionality of the feature set. The technique divides the correlated features into principal components which are uncorrelated using the covariance matrix. The study in [19] established that the features extracted from the PCA can achieve the best performance in some cases. Additionally, the CSP technique uses the spatial filtering and converts the signals into variance matrices which can be differentiable by each labels of the dataset. However, the practicality of the method highly varies with the position of channels as stated in [20]. Finally, the adaptive filtering approach changes the features of the signals based on various properties of the signal. This approach is considered very useful for removing artefacts that have similar properties as the signals of interest unlike other methods, where usually critical signal properties are removed along with the noise/artefacts. Two of the most effective algorithm for this approach are least mean square and recursive least squares algorithm which has efficiently removed various artefacts such as ECG signals.

### 2.1.2 Feature extraction from the EEG signals

Once the noise and artefacts are removed from the EEG signals, we are simply left with the EEG signals which can be used for classifying a particular task. However, in reality, the signals are nonstationary and non-linear in nature, which is why it becomes difficult to find a pattern from the signals in time domain. In this vein, various feature extraction techniques are applied to the signals so that meaningful features can be extracted from the dataset in both frequency and time domain, without losing any information. Some common methods of feature extraction includes Fourier Transformations (FT), Wavelet Transformations (WT), Gabor Transformation (GT), Wavelet Packet Decomposition (WPD), Auto Regressive parameters (AR) and many more [21–25]. The Fourier transform method is a very popular method used to extract the frequency domain features from the time domain. The signals are first divided into a window of one-second which collides with a window of half-second. This method is also known as Discrete Fourier Transform or Power Spectral Density. The Gabor transform is a variant of FT i.e a short-time Fourier transform or a windowed FT which is able to represent the signals in time domain in addition to the frequency domain. Here a gaussian window is considered for the data and the FT is calculated for that particular window [26]. In addition to that, another very popular method of extracting features from the EEG signal is WT, which is also effective for discontinuous signals. Here, the signals are broken down into wavelets which are temporary oscillating function. These extracted features can be presented in both frequency and time domain. Like WT, the WPD can also extract features into frequency and time domain. The WPD checks the frequency of the signals and creates two separate subspaces in the time domain. At first it selects a set of features from the signal subsets and calculates the separability of the features by employing Fisher's criterion. The features with highest separability are considered in the final vector. It has the ability to extract features from the non-stationary signals with a slight expense of computation time. Finally, the AR technique can be used to extract the features of the EEG signals which are in time domain. The method uses smaller duration of data which helps to address the issue of spectral loss and also gives a good frequency resolution. It can also be used to extract features from the non-stationary EEG signals.

## 2.2 Classifiers

### 2.2.1 Adaptive classifiers

The adaptive classifiers have been a classical method of analysing EEG signals because of its ability to update its parameter weights with new data. This is very effective

for the non-stationary EEG signals as it can change its feature space as well, keeping the classifier up to date. Some of the most popular algorithms of the adaptive classifier includes Support Vector Machine (SVM), Linear Discriminant Analysis (LDA), Quadratic Discriminant Analysis (QDA), Bayesian classifier and so forth. In various event related potential tasks, a continuous adaptation was claimed to be useful while classification [27, 28]. Even though the adaptive classifier can be implemented for both supervised and unsupervised cases, the supervised classification is more popular with this type of classifier. For e.g., in [29], a Bayesian classifier was proposed to address the issue of erroneous labels of the signals in supervised cases. The proposed classifier was employed sequential Monte Carlo sampling for modelling unreliability of the defined labels. Additionally, in [30], both LDA and QDA was explored to study the adaptive classifiers with various feature types in a BCI dataset, and they concluded that the concatenation of features with the LDA approach yielded to the best performance. An adaptive probabilistic neural network was proposed in [31], which was able to deal with variation in EEG data among subjects as well as with time, where the training was performed online. Their model achieved a reasonable accuracy over various experiments in different days for several subjects. The classifier focuses on the feature distribution of the task labels in non-parametric way and changes the feature distribution with availability of new EEG data. Apart from the traditional methods, various ensemble adaptive classifiers were also explored to analyse the signals. One such instance is shown in [32], where multiple SVM classifiers were implemented to classify the EEG signals based on their majority decision. However, upon the arrival of new EEG signals, the oldest SVM model was removed and the new model was added to the ensemble model. In [33], Independent Component Analysis (ICA) or independent component correlation algorithm was applied as an adaptive classifier to extract motor-dependent features from the EEG signals. In addition to the supervised learning, various unsupervised learning techniques for the EEG signals, were also studied throughout the years. The main idea of this is to determine the class labels of the newly collected EEG signals and once the labels are determined, the adaptive classifier will be updated based on the estimated labels. In [34], an adaptive classifier was proposed which can adapt in a totally unsupervised way or use an evaluation signal along with the unsupervised method to improve the performance of the model. Also in [35], a Gaussian Mixture Model (GMM) is used to predict the labels of the unknown classes. The proposed GMM model was compared with two other existing methods and the proposed model showed a better performance in terms of adaptability. Lastly, a LDA-based unsupervised classifier was proposed in the motor imagery domain. Here, the data was considered to be balanced and only the bias of the model was adapted

with newer EEG signals instead of all the parameters, making it less computationally complex [36, 37].

### 2.2.2 Matrix and tensor-based classifiers

The most popular matrices and tensor-based classification technique that has grabbed attention in the field of BCI over the past few years is Riemmanian Geometry based classification. For classifying the signals using Riemannian geometry, the covariance matrix having property of the Symmetric Positive Definite (SPD) matrix of the EEG data is first computed. The SPD matrix are expected to have all positive eigenvalues. The covariance matrix, $\mu \epsilon \mathbb{R}^{n_c \times n_c}$ of a signal $S \epsilon \mathbb{R}^{n_c \times n_s}$ can be calculated using eq. (2.2) where $n_c$ is the number of channels, $n_s$ is the number of samples of the signals in temporal region. An advantage of calculating the covariance matrix is that the traditional preprocessing steps while classifying an EEG signal can be avoided as the covariance matrix is able to represent the spatial information as well. In order to classify the signals the Riemmanian manifold plays an important role. It is a smooth manifold that has finite amount of dimensional Euclidean tangent space over every location. The manifolds are similar to the covariance matrices. Along the Riemannian manifolds, the distance between the covariance matrices are calculated which can be employed to construct classifiers that will take the covariance matrix directly as input. The most common classification method from the Riemannian input is the minimum distance to the mean (MDM) approach [38], that employs the covariance matrix to calculate the minimum distance between two points which is considered as the class label. The classification of a newly collected EEG data to determine the class label, $C^k$, may be expressed as:

$$k = arg\ argmin_k\ \delta^2(S_j, C^k), \tag{2.1}$$

where $\delta$ denotes the Riemannian distance, and $S_j$ indicates the symmetric positive definite (SPD) matrix.

However, for some algorithms like LDA and SVM, it is not possible to give the matrix as input, which can be solved by vectorizing the matrices to feed to the model. In [39], various Riemannian geometry classifier and Riemannian geometry-based adaptive classifiers are discussed such as MDM, Fisher geodesic MDM (FgMDM), unsupervised, supervised, biased MDM, FgMDM and so forth. After thorough experimentation of the models in two datasets, they concluded that the Riemmanian geometry is able to perform better with the adaptive classifiers rather than individually. In addition to that, various optimization problem are also being analysed based on the manifold, which is able to solve novel problems [40, 41].

$$\mu = \frac{SS^T}{Trace(SS^T)} \tag{2.2}$$

The Riemannian classifiers that employs the Riemannian manifolds are proven to give good performance in various studies for both normal and irregular data [42–44]. However, the complexity increases with increasing number of channels, thus degrading the performance of the model. In some cases, the classification is performed by projecting the data into tangent space. This was also proven as a very effective method of classification in terms of accuracy in various studies such as in [42, 45]. Apart from a good performance by the classifier, one more advantage of this classifier is its simplicity. The Riemannian based models require less preprocessing steps. Also, since the classification process relies on the covariance matrix of the data, the tuning of hyperparameters are not required in the process, saving training time. The models are also more generalized as the distance between the covariance matrices does not change under any condition like matrix inversion, or any other type of data transformation. However, the approach also have some major disadvantages. For greater dimension, the distance between the covariance matrices increases thus more noise gets added to the value. Additionally, when the number of channels increases in the dataset, the computation of geometric mean and Riemannian distance also gets more complex. Therefore, for a bigger scenario the model might not be suitable. However, further studies are being performed in the field to reduce the complexity of the model, due to its ample advantages.

### 2.2.3 AI-based classifiers

The use of AI-based classifier in the field of BCI is also a very well-known approach of analysing the brain signals. Various machine learning and deep learning methods have been implemented over the years to find patterns in the EEG data. Some of the very popular methods of analysing the data includes, Random Forest, K-Nearest Neighbors (KNN), Support Vector Machine (SVM), Bayesian classifiers, Convolutional Neural Network (CNN), Recurrent Neural Network (RNN), Multilayer Perceptron (MLP) etc. The KNN algorithm has been repeatedly used in the BCI domain to perform classification of the EEG signals [46–48]. However in [49] and [50], it is apparent that KNN is unable to handle the large dimensions of data. However, if the dimension is small, it can perform well [51] despite of having a simple architecture. On the other hand, SVM can be very efficiently used to handle EEG signals having large dimensions with the help of different types of kernels. Some of the notable studies performed on EEG data employing SVM includes [52–54] and so forth. Additionally, the SVM models are usually very generalizable and are not affected by overfitting [55]. However,

the they might get very complex while classifying the data. The random forest is also very efficient in classifying EEG data due to its ensemble properties. The parallel computation property of the random forest makes it suitable for the classification of EEG signals efficiently [56]. In various studies it has been able to outperform several other algorithms [57–59]. However, one of the disadvantages of this algorithm is that with increasing number of trees, the number of hyperparameters in the model increases which makes the model more complex while training.

In addition to that, the deep learning algorithms are also used in the domain to classify EEG data due to its precise performance. The CNN is the most used deep learning model in the field of EEG according to the survey in [60]. It states that almost 40% of the EEG analysis using deep learning algorithms are analysed using CNN. The CNN is an excellent algorithm for detecting features of a data. It is most commonly used for image data, however, it has also gain its popularity in determining features from different physiological signals. Some of the EEG research using CNN algorithm out of numerous studies can be found in [61–64], which managed to achieve a significant performance. Apart from CNN, the RNN algorithms are considered extremely efficient for finding patterns in a sequence. Using a fixed time window, the RNN models maps the input over a period of time to the output, which makes it suitable for time-series datasets. In the RNN, the LSTM and GRU have risen as very effective for analysing biosignals as can be seen in the studies [65–69]. Apart from that, various hybrid architecture has also been proven to perform well from time to time in various papers in the literature. Most of this architecture contains CNN at the beginning, to extract meaningful features from the signals which are then passed to an RNN layer for the time-series analysis part. It has been able to perform well in several cases [70–73]. The autoencoders are also used in various studies to extract featured from the signals which is then fed to a neural network [74, 70]. One major disadvantage of EEG data is that its time consuming to collect the signals from different people for different activities. So there is always a shortage of data. Therefore, in some studies, data augmentation techniques such as Generative Adversarial Network (GAN), which is able to generate realistic synthetic data which can be used while training the models. It is found that the synthetic data has been able to improve the performance of the model [75–77].

As seen in this section, various types of classifiers are used to classify the EEG signals namely adaptive classifiers, matrix and tensor based classifiers and finally the AI-based classifiers. There are many strengths and weaknesses of these classification which have been summarized in table 2.1. It can be seen that, despite of the advantages of adaptive classifiers, the models update themselves whenever a new data comes, This might be computationally expensive for the ultra-edge IoT device. Thus, the other two

Table 2.1 Table representing the advantages and disadvantages of various EEG signal classification techniques.

| Classifier Type | Example | Advantages | Disadvantages |
|---|---|---|---|
| Adaptive classifiers | Quadratic Discriminant Analysis (QDA) Linear Discriminant Analysis (LDA) | Model updates based on new data | Computationally expensive |
| Matrix and tensor-based classifiers | Riemannian Geometry | -Precise performance -Lower preprocessing steps | Complexity increases with dimensions |
| Artificial Intelligence-based classifiers | K-nearest neighbors Random forest Neural networks | -Lightweight inference -Accurate performance | Extensive training is required |

types of classifiers can be considered as the two candidate choice for deploying on the ultra-edge, and hence they are scrutinized further in the later chapter.

## 2.3 IoT-based EEG analysis platforms

Traditionally, the EEG analysis is carried out in the cloud due to better computational power and resources. In [78], an auto encoder-based deep learning classifier was proposed to predict epileptic seizures. In order to store the data and perform heavy computation, the model was created in a cloud-based environment. Additionally, to reduce bandwidth and usage of computational resources, the authors used PCA to reduce the dimension of the signals. On the other hand, in [79], the authors proposed a system that collects EEG data from the commercially available device named Neurosky Mindwave that will collect and store the signals locally on an android application via bluetooth. However, while analysing the data, the signals are sent to cloud for further data analysis. The study in [80], at first discusses about the absence of an appropriate computing platform while analysing EEG signals. The authors then proposed a lightweight CNN model which will be trained on the cloud. Similarly, Blondet et al., in their paper [81], proposed a cloud-based mobile application which would collect the data with EEG sensors and pass to the mobile application. The data will then be transferred to the cloud for further computation and the results will then be transferred to the mobile application. Apart from these, some notable cloud-based EEG analysis techniques can be found in [82–85]. In [86], a hybrid cloud and edge based system is proposed where various kinds of EEG data will be stored at the cloud and a signal analysis will be performed at the edge based on the data available on cloud. The edge analysis exclusively is also growing to be a well known platform

for EEG signal analysis. For instance, in [87], the data is collected at the sensor level and sent to mobile edge cloud platform for computation. The computed results are then sent to an alert system which will notify about any possible seizures in this case. In [88], EEG data was initially collected at Mobile/Infrastructure Edge Node (MEN) which was later analyzed in cloud server for efficient transmission and fast detection of epileptic seizures. An accuracy of 98.3% was achieved by the classifier in the process. In the study [89], a platform was proposed where EEG data can be collected through consumer-grade headsets like Emotiv EPOC and Neurosky Mindwave using raspberry pi and transmitted over the internet to track neural activity in real-time.

## 2.4 Limitations of the existing literature

Table 2.2 represents the methodologies of some of the studies discussed in the previous sections of this chapter and whether they have mentioned a sensor level integration capability for their models. It can be seen that, none of the studies have mentioned sensor level computation. Most of the EEG analysis are carried out in cloud or edge platform as stated in Section 2.3. However this entails a great amount of time and bandwidth to transfer the big data. According to [88], the biosignals data collected from each patient everyday can range from 8 to 10 gb, which, in a large scale scenario, might get very inefficient. In a typical cloud based architecture, data needs to be sent to the cloud continuously and this quantity of data from only one person will consume significant bandwidth and hence introduce latency in the process. On the other hand, for edge computation, there is still a need to transmit the data to the closest edge node. The energy consumption is greatly reduced in the process, however, there are still some challenges. The more the distance of the sensors from the edge node, the more energy will be required to send the data to the node after being collected in the signals. Thus it hampers the flexibility of usage of such systems. The ultra-edge computing can thus be employed to address these challenges. If the computation is performed in the same platform where data is being collected, a significant amount of bandwidth and energy can be saved while reducing the delay of communication as well.

In order to deploy an ultra-edge IoT device, it is important to minimize the computational energy of the devices. The computation energy largely depends on the type of classification model being used in the device. Therefore, in the next section, we selected some classification models based on the advantages and disadvantages learned from the literature, to find out the best-suited classification model for the ultra-edge device.

Table 2.2 Table representing various EEG signal analysis studies in the literature and whether there was a statement of sensor level computation compatibility for the proposed approaches.

| Case study | Preprocessing techniques | Classification techniques | Mention of sensor level integration |
|---|---|---|---|
| Motor imagery brain computer interface [30] | Logarithmic band power, Adaptive autoregressive parameters | LDA, QDA | X |
| Motor Imagery, P300 BCI [38] | Covariance matrix | MDM | X |
| Epileptic seizure detection [61] | Fourier transform | CNN | X |
| Emotion Classsification [66] | Raw signals with removed noise and artefacts | GRU, LSTM | X |
| Steady state visually evoked potential [43] | Covariance matrix | MDM, MDM-Potato | X |
| Mental state [59] | Frequency features collected from Neurosky Mindwave device | Random Forest | X |
| Seizure Detection [54] | Wavelet Transform | SVM | X |
| Depression analysis [80] | Noise and Artefact removal, CSP | CNN | X |
| Anomaly in EEG signals [86] | Noise and artefact filtering | Cloud search, Edge Tracking | X |
| Epileptic seizure detection [88] | Statistical attributes of frequency domain | Stacked autoencoders | X |

# Chapter 3

# Background Theories

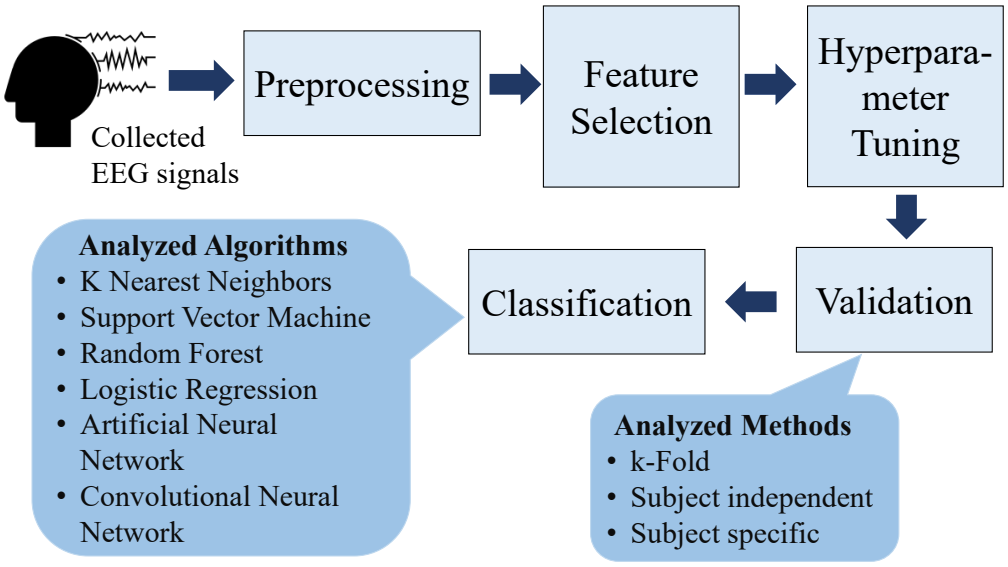



Fig. 3.1 Systematic methodology of classifying the EEG signals.

## 3.1 Preprocessing Methods

### 3.1.1 Feature Scaling

The process of feature scaling refers to the change of all feature values to a certain range so as to make the model more generalized. It is an important step in various types of scenarios such as distance-based algorithms, the algorithms which employs gradient descent and so forth. In a distance based algorithms such as KNN or SVM, the values of different features may greatly affect the way the classifier behaves. Features having higher values, will carry a greater weight than feature having smaller values and the model will hence be biased towards features having larger values. Additionally, in

methods such as logistic regression, neural networks etc., that uses gradient descent optimization technique to train the model are greatly affected by scaling. This is because the gradient descent algorithm focuses on minimizing a function's local minima by taking small steps towards the local minima. However, if the features have different ranges, the step size for each feature will be different from one another which might result in a delayed convergence. In such cases, the scaling ensures that the step size of all the features are changed at the same rate and the gradient descent moves smoothly into the direction of the minima. On the other hand, the tree-based algorithms such as decision trees, random forest does not get affected by the different ranges of each feature since the tree splits based on an individual feature and therefore does not depend on other features.

$$x_{scaled} = \frac{x - x_{min}}{(x_{max} - x_{min})}. \tag{3.1}$$

There are two commonly used scaling techniques which are normalization or min-max feature scaling method and standardization or z-score scaling method. Throughout the experimentation, the normalization technique was used for scaling the EEG signals since they do not follow a gaussian distribution. The method normalizes the values in a scale of $0 - 1$ using eq. (3.1). Thus, the standard deviation is reduced and the domination of higher values in the entire model is avoided.

### 3.1.2 Dimensionality Reduction

As reviewed in the literature, various dimensionality reduction techniques are used for classifying the EEG signals to reduce the computational complexity and training time. Among them, the Principal Component Analysis (PCA) is a very well-known approach. If $\chi \epsilon \mathbb{R}^N$ is a data space, the PCA computes a mapping $\gamma = f(\chi) : \mathbb{R}^X \rightarrow \mathbb{R}^M$ where $M < X$ so that maximum information of $\chi$ can be preserved in $\gamma$. It is done by capturing the highest variance of dataset with minimum principal components where the first principal component always holds the highest variance of data, and the second principal components have the second highest variance and so forth. The PCA first creates a covariance matrix, $\sigma$, which is diagonal from the input EEG signals [90]. Then the covariance matrix and identity matrix are made equal by normalizing the dimensions using a transformation technique [91]. At first, the eigenvalue decomposition of $\sigma$ is calculated using $\sigma = \chi\chi^T$. Based on the eigenvalues, $\lambda$ sorted in descending order, the eigenvectors are then selected. The value of the dimension of the PCA features, $M$, can be determined by eq. (3.2), i.e the minimum $M$ principal components whose cumulative eigenvalues is more than 90% of the total eigenvalues. Thus the principal components

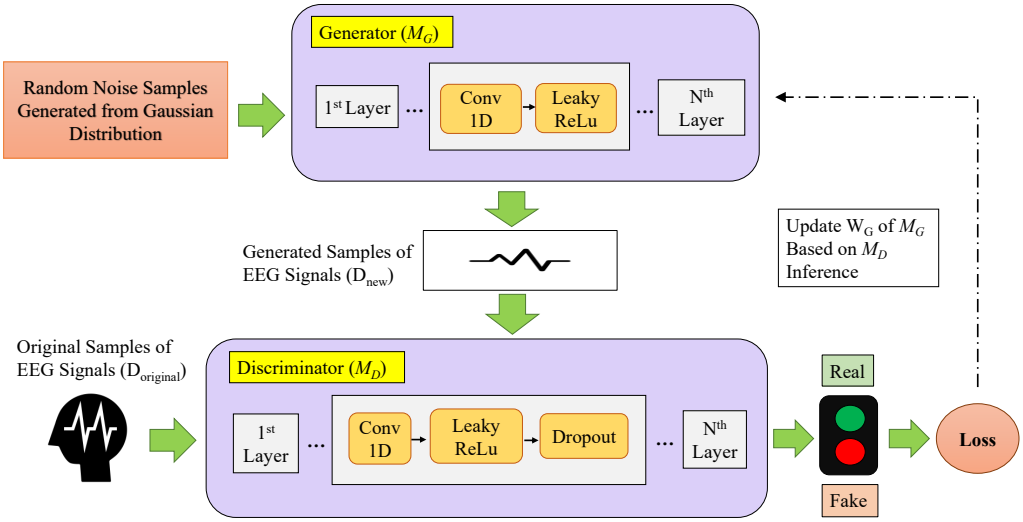



Fig. 3.2 Proposed deep convolutional general adversarial network model.

are computed accordingly.

$$\frac{\sum_{i=1}^{M} \lambda_i}{\sum_{i=1}^{N} \lambda_i} 100\% \geq 90\% \tag{3.2}$$

### 3.1.3 Data augmentation techniques

Data augmentation is a very important preprocessing step while dealing with EEG signals. Since the EEG signals need time to be collected for different experiments, there are many publicly available EEG datasets which are not big enough or the classes of the data are imbalanced to construct a robust model. The generative adversarial network, introduced in [92], has emerged as one of the most popular data augmentation technique due to its capability of generating synthetic EEG traces accurately. In a basic GAN, the models employ the multilayer perceptron to generate synthetic data which will be indistinguishable from the original data, however where are various types of other GAN used for the purpose of data augmentation such as Deep Convolutional GAN (DCGAN) [93], Wasserstein GAN (WGAN) [76] etc. We have used DCGAN for constructing synthetic EEG signals because it interprets the original EEG signals ($s$) more precisely by employing deep convolution layers. Fig. 3.2 depicts our customized DCGAN architecture for augmenting EEG signals. The model consists of two main components, namely a generator ($M_G$) and a discriminator($M_D$). $M_G$ takes as input random noise/latent vectors ($\eta$) produced by a Gaussian distribution, and generates signals $s'$ that replicates $s$. The objective of $M_G$ is to generate such signals, which are difficult for $M_D$ to differentiate; i.e., by enhancing $M_G$ as shown in eq. (3.3). To accomplish this, the generator is constructed with $n_G$ hidden layers where each $i^{th}$ layer

consists of a convolution layer followed by an activation function $\alpha$:

$$min_{M_G} V(M_G) = \mathbb{E}_{\eta \sim p(\eta)}[\log(1 - M_D(M_G(\eta)))]. \tag{3.3}$$

On the other hand, $M_D$ is trained to distinguish between $s$ and $s'$ by stating if the generated signal is real or fake. The objective is, therefore, to be precise enough to infer the maximum likelihood of the observed data as stated in eq. (3.4),

$$max_{M_D} V(M_D) = \mathbb{E}_{s \sim p(s)}[\log M_D(s)] + \mathbb{E}_{\eta \sim p(\eta)}[\log(1 - M_D(M_G(\eta)))]. \tag{3.4}$$

The results are used to update the weights of $M_G$ that are denoted by $W_G$. Then, $M_G$ is trained again based on these new weights. Thus, the discriminator is constructed with $n_D$ hidden layers, where each $i^{th}$ layer is composed of a convolution layer followed by $\alpha$ and a dropout layer. The dropout layer introduces randomness to prevent the discriminator from becoming stuck in local minima.

Thus, the overall objective function of the entire adversarial network is expressed as follows. Note that the model is trained for $I$ iterations.

$$min_{M_G} max_{M_D} V(M_D, M_G) = \mathbb{E}_{s \sim p(s)}[\log M_D(s)] + \mathbb{E}_{\eta \sim p(\eta)}[\log(1 - M_D(M_G(\eta)))]. \tag{3.5}$$

## 3.2 AI-based Classification Algorithms

Various AI-based algorithms that has been used in the literature were explored to find an optimal model for carrying out EEG analysis at the ultra-edge. In this section, the algorithms used for classifying the EEG signals are discussed.

**K Nearest Neighbour** (KNN) is a lazy learning method that considers the k nearest points around a data point and make a classification decision for the data point accordingly instead of computing for the whole dataset. The algorithm calculates the distance of the new data point with all the available datapoints in a dataset and choose the $k$ datapoints closest to it. In our case, the Euclidean distance, $E_d$ was considered as the distance metric which uses eq.(3.6), where $x_1$ and $x_2$ are two datapoints whose distance are being calculated and n is the number of datapoints. The number of neighbours, $k$, is considered as a hyperparameter for the model which is altered to find an optimal model. The KNN algorithm is a very simple algorithm and performs well in various cases. Also, the number of hyperparameters are reasonable which makes it easier to implement. However, since it calculated the distance of a datapoint with all the existing instances of a dataset, with increasing number of instances, the time consumption of the model increases.

$$E_d(x_1, x_2) = \sqrt{\sum_{i=1}^{n} (x_{2i} - x_{1i})^2} \tag{3.6}$$

The **Logistic Regression** is a classification method that is known to classify a problem with great precision using statistical methods. It does so by calculating the probability, $P$, of class labels after choosing a threshold between the two class labels from the training dataset. The decision boundary can be linear or non-linear. Often, the polynomial order is altered to change the complexity of the decision boundary. The $P$ is correlated to the explanatory variables of the dataset as shown in eq. (3.7) where $\lambda_i$ is the coefficient of the datapoint $x_i$. The logistic regression is also a very fast and simple algorithm, which performs well when the dataset has greater number of instances. The relation between the features can also be known since the model gives either a positive or negative value. The model can also be updated based on the newly collected data using gradient descent approach. However, in short and wide datasets having higher dimension data and lower number of instances, the model might tend to overfitting. Additionally, the logistic regression model struggles to solve non-linear problems since its decision surface is linear. However it is a big drawback considering most of the real-world data are usually non-linear.

$$logit(P) = ln(\frac{P}{1-P}) = \lambda_0 + \sum_{i=0}^{n} \lambda_i x_i \tag{3.7}$$

**Support Vector Machine** (SVM) is a very effective classification method that utilizes various kernels to transform high dimension data in order to create an optimal decision boundary that separates the class values. The decision boundary also known as hyperplane, is chosen in such a way so that it maximizes the distance of the datapoints from the plane for the classes with the help of a hinge loss (eq. (3.8)). The points closest to the hyperplane are called support vectors and the boundary mostly depends on them. Thus, a change in the support vectors can cause the decision boundary to change. This algorithm has the capability to deal with multi dimensional feature set. Since higher dimensions are also considered while developing a hyperplane, SVM is said to separate the class values even more accurately than other algorithms. An idea of feature dimensionality of the dataset can be found employing various kernels like linear, sigmoid and radial basis function (rbf) in the model. However, in some cases, the data cannot be perfectly separated if an optimal boundary is built. In that case, a penalty factor known as regularization parameter can be introduced that is a trade-off between an optimal boundary and more accurate classification. It is an efficient algorithm which consumes comparatively less memory and usually works really well for data points can be separated completely based on their classes. Nonetheless, when the classes are not entirely distinguishable, i.e, there is noise in the dataset, the performance of the SVM degrades.

$$\delta(x, y, f(x)) = (1 - yf(x))_+ \tag{3.8}$$

**Random Forest**, on the other hand, is an ensemble classifier that employs multiple decision trees to classify a dataset. In this algorithm, the class labels are determined based on multiple decision trees. An individual tree is created based on bootstrap sampling of the instances. For each decision tree, the importance of each feature is calculated, based on their entropy, gini impurity or information gain value. The entropy measures the impurity of a dataset using eq. (3.9) where $d$ is the dataset, $c$ is the number of classes. The lower the entropy, the more pure the dataset is. From this entropy value, the information gain from each feature, $f$ can then be determined using eq. (3.10) where $d_i$ is the subset of dataset having the $i^{th}$ feature. Finally the gini impurity of a dataset $d$ is another metric for evaluating the purity of the dataset which can be calculated using eq. (3.11). The feature importance then determines the feature which best classifies the dataset among all. Based on the feature importance, the tree is split, dividing the data into multiple branches. The process is repeated until the data is divided into their respective class labels completely, or the required number of features are explored. This method does not require any scaling method as only one feature is dealt at a time. Additionally, the overfitting is also reduced since multiple decision trees are used. However, since multiple decision trees are used, the computational time and complexity is comparatively higher.

$$Entropy(d) = -\sum_{i=1}^{c} p_i log_2 p_i \tag{3.9}$$

$$Entropy(d, f) = Entropy(d) - \sum_{i \epsilon f} \frac{|d_i|}{|d|} Entropy(d_i) \tag{3.10}$$

$$Gini\ Impurity(d) = 1 - \sum_{i=1}^{c} {d_i}^2 \tag{3.11}$$

The **Artificial Neural Network** (ANN) is the most basic form of neural network. It mainly consist of three layers containing neurons which are input, hidden and output layer. The EEG signals are passed through the input layer which are passed forward to the fully connected hidden layers. The inputs are assigned a random weight at first, which are then updated with the help of forward and back propagation techniques. After the inputs are passed to the output layer, the activation of the output layers checks for specific criteria i.e whether the output is equal to the label. If it does not meet the criteria, an error is calculated using a cost function and the data are backpropagated through the same network. Therefore, it is important to minimize the cost function of a network, which is usually done using the gradient descent technique. Here, the

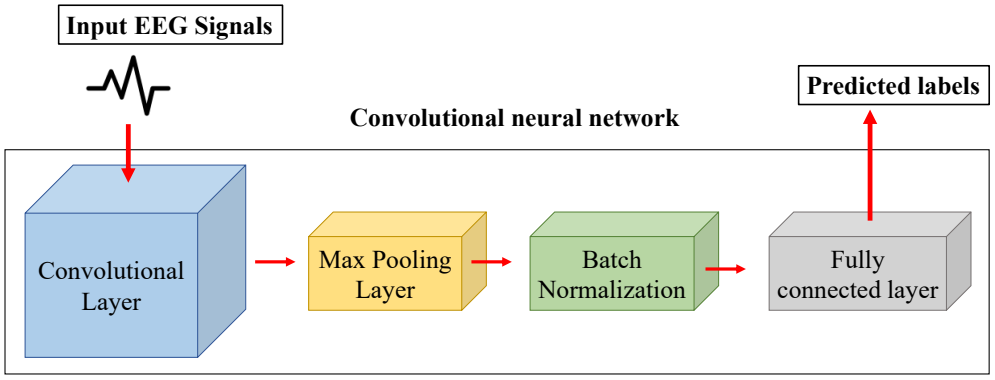



Fig. 3.3 The architecture of a typical convolutional neural network.

technique searches for the local minima of the function while taking small steps towards the direction of the gradient. At each step, the weights are updated by multiplying with a parameter called learning rate. The learning rate plays a very important role while training the model. The ANN is able to determine various complex and non-linear relations and usually have a great generalization power. However, the network requires heavy computational resources and sometimes tend to overfitting. The overfitting can be avoided by taking smaller number of neurons or adding a penalty factor [94].

The **Convolutional Neural Network** (CNN) are deep learning models that mainly consist of convolutional layers, max pooling layers. The convolutional layers are very efficient for detecting important features of a data. It employs numerous convolutional kernels which are integer matrices that are slid across the input data to perform computations over a small window. In [95], it was stated that the convolutional layers are able to eliminate the effect of noise in the signals, which makes it more suitable for EEG data analysis. Each convolutional kernel creates its own feature map. The size of feature maps usually consume a great amount of memory, thus, a max pooling layer is applied to reduce the size of the feature maps without losing any information. The max pooling layer considers a small window and within that window size in the feature maps, the maximum value is chosen. For a better approximation of the features, a non-linear activation function is applied in the network. The 'Rectified Linear Unit (ReLU)' is most commonly used for the purpose. It goes through the feature maps, and change any negative number there is to 0, thus getting rid of all the negative numbers. In many cases, to reduce the overfitting of the model by regularization, and also to help make stable predictions a batch normalization layer is used with the convolutional and max-pooling layers. The typical architecture of a CNN layer is depicted in Fig. 3.3. These layers are repeated many times to extract diverse, more meaningful information from the input data. Thus, the input signals are directly fed to the CNN model and the model is able to capture hierarchical patterns in the signals [96]. However, one

big disadvantage of the model is, it typically requires large number of data which is difficult to find in the EEG domain. The data augmentation techniques such as generative adversarial network and many more can be explored to address the issue. Also, the training usually requires longer and the model have a higher number of hyperparameters such as the activation function, the number of kernels, the number of layers and so forth.

### 3.2.1 Cross validation techniques of the EEG signals

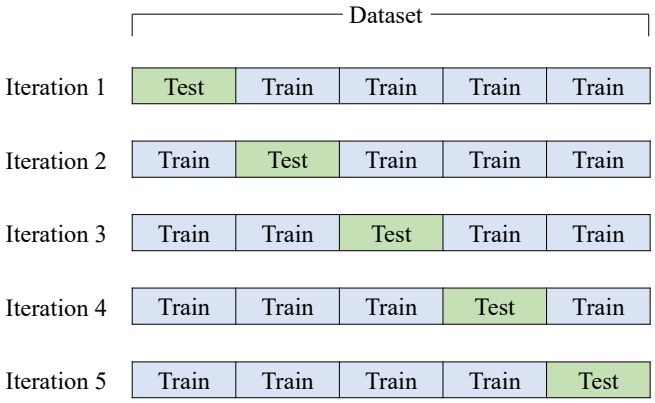
The three most commonly used cross validation techniques in the domain of EEG analysis are k-Fold cross validation, subject independent cross validation and subject specific cross validation. The cross validation is important as it helps to avoid overfitting of the model by considering all the instance of a data in both train and test set atleast once. The k-fold cross validation is a popular method of cross validation that evaluates an individual EEG signal associated with an activity. Here, the data is split into $k$ folds to evaluate its performance. In each iteration, one of the fold among the k folds is considered as the test set and the rest of the $k-1$ folds are considered as train sets as shown in Fig. 3.4(a). This method is repeated for all $k$ folds i.e until all the data are considered in both train and test splits. The average of the performance of each iteration are then usually considered as the final performance of the model. The subject independent cross validation technique on the other hand, focuses more on the subject to subject EEG variation. This means that if the EEG data are collected from $n$ subjects, at first, $n-1$ subjects data will be considered in the training set and the remaining subject will be considered in the test set as depicted in Fig 3.4(b). This process will be repeated until all the subjects appear as both train and test sets. This is an effective approach as it will be able to analyse signals of new subject's data based on the existing model that might or might not be present in the training model. Finally, the subject specific cross validation is another popular approach that validates the data present among an individual subject as shown in Fig. 3.4(c). This technique is suitable for datasets where the same experimentation is repeated for an individual subject. Therefore, if there are $e$ experiments done for one particular subject, then $e-1$ experiments will be considered as the train set of that particular subject and the remaining experiment data will be considered as the test set. The performance of all the subjects are then averaged to get the overall performance of the model.

### 3.2.2 Performance Evaluation

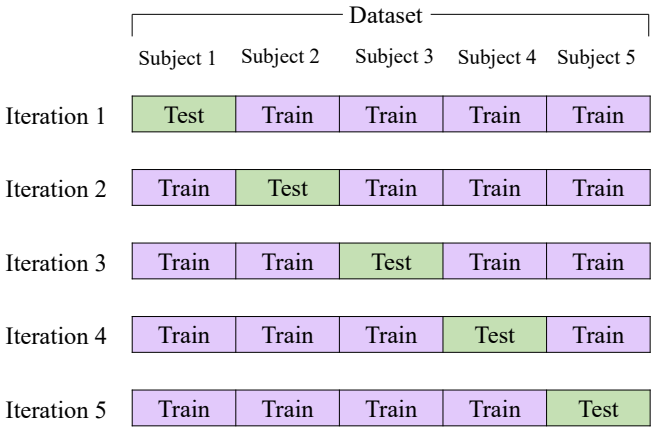
The performance of the candidate AI models trained with the EEG datasets of our EEG analysis on the ultra-edge was evaluated in terms of accuracy using eq. (3.12):

$$accuracy(y, \bar{y}) = \frac{1}{n_{test}} \sum_{i=0}^{n_{test}-1} 1(\bar{y}_i = y_i), \tag{3.12}$$

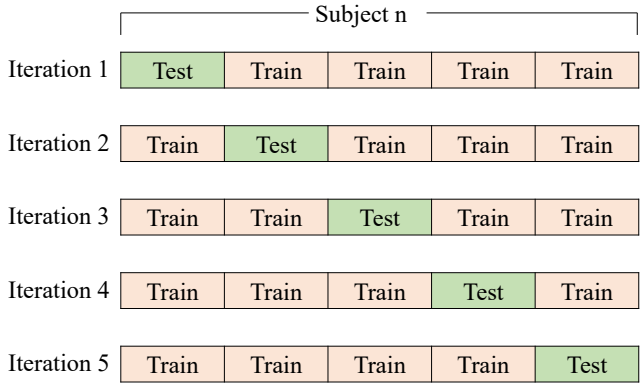
where $y$ denotes the true class of the dataset, $\bar{y}$ represents the predicted class, and $n_{test}$ indicates the number of test data.




(a) K-Fold cross validation.




(b) Subject independent cross validation




(c) Subject specific cross validation

Fig. 3.4 Figures demonstrating various validation approaches used to analyse EEG signals

# Chapter 4

# Migrating EEG Analytics to Ultra-Edge IoT Devices with Logic-in-Headbands

## 4.1 Introduction

Recently, the rapid proliferation of the Internet of Things (IoT) has revolutionized the healthcare industry with the adoption of a plethora of technologies ranging from wearable devices to the early warning or monitoring systems. Most existing IoT systems collect biomedical and environmental data, and then employ existing wireless communication systems to transmit the data to a central server or the cloud. The obtained data are processed, analyzed, and various analytics decisions are generated by the centralized/cloud environment. Thus, this traditional analytics ecosystem requires the raw and often sensitive patient data to be transferred over the Internet or dedicated networks that consumes precious network resources (e.g., bandwidth) and contributes to a significant network delay while raising various security and privacy concerns [97, 98]. Edge computing recently emerged as a viable technique to address these concerns where user-smartphones and network nodes (e.g., base stations and access points) with some computational resources can tackle the computing tasks. For instance, with the popularity of EEG analytics for continuous monitoring and assessing various neural activities beyond a hospital setting, it is critical to localize and automate the process of EEG data analytics. In this chapter, we investigate how to migrate the edge computing operations to the final frontier of the considered system, i.e., the IoT sensor level, that we refer to as the ultra-edge. The ultra-edge computing was introduced in one of our earlier research work in [12] with an entirely different use case for smart, localized ECG noise reduction. Motivated by our success in that work, in our conceptualized ultra-edge EEG computing, we aim to conduct brainwave signal capturing and analysis directly on-board IoT devices and wearable devices, such as EEG-headsets/headbands. While this ultra-edge computing concept is appealing, it requires high computational and

energy resources. On the other hand, the IoT nodes are typically resource-constrained. In this work, we address this challenge and conceptualize a Logic in Headbands-based Edge Analytics (LiHEA) architecture. Thus, we perform a systematic investigation to move the traditional cloud analytics operations from the traditional resource-heavy cloud environment towards the resource-constrained ultra-edge nodes.

Our envisioned LiHEA framework requires seamless incorporation of a lightweight AI (artificial intelligence) inference model to minimize the computational and energy burden on the EEG headband acting as the resource-constrained IoT node. The predictions, indicating anomalies or aberrations, can be transferred to edge devices and then to the hospital-server for prompt intervention. Conventional approaches, such as Riemannian Geometry-based classifiers, are inefficient for LiHEA-incorporation because of their high timing and space complexity. Therefore, in this chapter, we systematically investigate various machine/deep learning techniques (e.g., random forest, K-nearest neighbor (KNN), support vector machine (SVM), logistic regression, artificial neural network (ANN) , and convolutional neural network (CNN)) to identify the candidate lightweight AI inference models required for the LiHEA framework.

To train these AI models, we consider the use-case of EEG data analytics for detecting the confusion of an individual since this scenario may largely impact mental health assessment and concentration levels, particularly while attending online educational courses that proliferated during the novel coronavirus (COVID-19) pandemic. Indeed, confusion is regarded as a salient factor in various other scenarios such as medical settings over short-term (e.g., EEG monitoring of a patient during a surgical procedure) and long-term (e.g., post-operative EEG monitoring along with other modalities) duration. A dataset containing EEG records of the frontal lobe using a single channel headset is used to evaluate the performance of our considered AI models. Our investigation reveals that the unavailability of a large EEG dataset could be a performance bottleneck for EEG computing using deep learning techniques. To remedy this issue, we employ a deep convolutional general adversarial network (DCGAN)-based data augmentation technique to generate synthetic EEG traces to train the deep learning models to conduct comparative analytics with the machine learning counterpart with high accuracy, i.e., random forest. Even though we find a significant performance boost of deep learning algorithms with our customized DCGAN, it still cannot outperform the random forest model in terms of accuracy. We then investigate the timing complexity of the entire process for the candidate models (i.e., random forest, CNN, and DCGAN-boosted CNN) in terms of accuracy, time, and memory for deployment on IoT nodes such as Jetson Nano and raspberry pi 3/4 to simulate the inference model incorporation on the EEG headsets.

The contributions of this chapter are summarized as follows.

1. We propose the LiHEA framework to migrate EEG computing from the traditional cloud to the ultra-edge (i.e., the final frontier of the considered cyber-physical system).

2. We perform a systematic investigation of various machine/deep learning algorithms and demonstrate that the random forest emerges as the most viable model for LiHEA-incorporation in terms of accuracy as well as memory/energy overhead minimization for the EEG headset.

3. We consider a use-case for online course participation to track the confusion states of the users using the LiHEA framework. This is to assess the concentration/distraction levels of the participants, which appeared as a key concern during remote course delivery during the COVID-19 pandemic, and our proposal can be regarded as an initial step to practically address this issue. We achieved a state-of-the-art performance with the use-case compared to earlier studies in the literature, that employed the particular use-case.

4. We consider synthetic data generation to boost the performance of deep learning-based models for further comparative analytics of EEG edge computing.

## 4.2 Related Work

Brain signal sensing, collection, and analysis have been an active area of research. However, most of these research works were conducted in a clinical/centralized setting, and continuous home-monitoring and edge computing of EEG data for prompt decision making and intervention is yet to be taken into consideration. Therefore, in this section, we survey the relevant research work from two aspects, EEG computing using machine/deep learning-based classification techniques and IoT-based EEG monitoring systems.

### 4.2.1 EEG Classification Techniques

Previously, a variety of studies were conducted relevant to our considered use-case, i.e., confusion detection from EEG signals. A study was conducted to determine if EEG data were associated with confusion and whether it could outperform human observers [99]. The study concluded that confusion is positively correlated with the EEG signals. In [99], Wang *et al.* employed the Gaussian Naïve Bayes technique to perform binary classification and achieved an accuracy of 57(%). A feature selection was also performed where they concluded the Theta signal to be a significant feature for the procedure. In another study [100], Ni *et al.* compared the performances of support

vector machine (SVM), k-nearest neighbor (KNN), convolutional neural network (CNN), deep belief network, and bi-directional long short term memory (Bi-LSTM) for detecting confusion states from an EEG dataset. Among these models, Bi-LSTM exhibited the best classification accuracy of 73.6(%). Their research identified gamma wave to be the feature, which contributes the most towards confusion. Next, in [101], Tahmassebi *et al.* applied a genetic programming classifier to predict confusion and achieved an accuracy of 89%. On the other hand, in [102], a pictorial representation was employed to collect and then classify raw EEG data (without pre-processing) using CNN to ascertain confused or non-confused states, yielding an accuracy of 71.36%. Wang et. al, in another study [103], removed various confounding factors from the collected EEG data to predict confusion, lung adenocarcinoma, and the segmentation on the right ventricle of the heart. To detect confusion after removing confounding factors, an accuracy of 75% was achieved with a variation of BiLSTM. Next, Subashi *et al.* employed SVM to derive a relationship between epileptic seizures and EEG signals with 100% accuracy [104]. Furthermore, in [105], a Riemannian geometry-based classifier was introduced to classify EEG data using both filtered and unfiltered algorithm, indicating the approach to be convenient for small, noisy, or multi-class datasets.

In addition to purely clinical settings, EEG data were successfully utilized in other domains. For instance, Yeo *et al.* detected the drowsiness of car drivers using EEG data with an accuracy of 99.3% [52]. Deep learning emerged as a popular technique for enhanced EEG data analysis in various domains [60]. In a study conducted in [106], a method called SleepEEGNet was introduced that automatically annotates sleep-stages using a CNN model by extracting various features and information. The method yielded an accuracy of 84.26%. In another interesting research in [107], a combined auto-encoder and CNN structure was utilized on motor imagery data classification. On the other hand, generative adversarial network (GAN) was applied to augment EEG data to train a more robust model in various domains [75, 76]. The latter studies implemented Wasserstein GAN, and reported a substantial performance improvement. In [108] Zhang *et al.* applied a conditional deep convolutional generative adversarial network (cDCGAN) to generate synthetic EEG signals on a motor imagery dataset, and the accuracy for CNN-based classification of motor imagery was reported to improve from 83% to 86%.

### 4.2.2 IoT-based EEG Monitoring

Various wearables, monitoring, and detection systems have been proposed in earlier studies to make early warning systems more efficient and simple, particularly for continuous monitoring at user-homes. In [88], a feature selection and classification was performed on EEG data collected by the mobile/infrastructure edge node (MEN)

for efficient transmission and fast detection of epileptic seizures. This cloud-based classifier achieved an accuracy of 98.3%. The research work in [109] also utilized cloud computing to identify important features from voice signals in order to detect Parkinson's disease, results of which are then sent to registered doctors. Vidyaratne *et al.* utilized a fast wavelet decomposition method to detect seizures using EEG signals in real-time [110]. Furthermore, various methods were designed in other domains of healthcare. For instance, in [111], facial expressions were recorded to measure the heart rate to extract physiological signals of an individual, thereby acting as a contact-less monitoring system. Various wearables such as Ring sensors for pulse oximetry [112], attachable Bio-Patch [113], and chest-worn ECG monitor [114] have become popular among patients for monitoring their vitals. Furthermore, wearables, such as smart clothes embedded with textile-based sensors, to monitor autonomic nervous system responses were introduced by [115]. However, these wearables also required the collected data to be transmitted to the cloud for EEG data analytics.

## 4.3 Problem Description

Since EEG signals exhibit a high temporal resolution, various techniques are used to analyze EEG data to detect various important activities and irregularities of a subject (i.e., a user). The EEG classification methods include Riemannian geometry-based classification, adaptive classifiers, and machine/deep learning-based algorithms. However, the conventional Riemannian geometry-based approach and its variants require solving a computationally demanding Riemannian optimization problem. This includes the use of stochastic gradient approaches adapted to manifolds [116]. Additionally, these methods require EEG data to be pre-processed, followed by feature extraction, to classify the data. In other words, these techniques are unable to classify raw signals and require rigorous pre-processing. The classification of a newly collected EEG data to determine the class label using a Minimum Distance to the Mean (MDM) method, $C^k$, may be expressed as:

$$k = arg\ argmin_k\ \delta^2(S_j, C^k), \tag{4.1}$$

where $\delta$ denotes the Riemannian distance, and $S_j$ indicates the symmetric positive definite (SPD) matrix.

This traditional approach of conducting EEG analytics employs a closed-form equation (eq. (4.1)) to determine the class label, making it a theoretically ideal classifier for performing edge analytics to construct LiHEA. However, to classify the signals, SPD matrix needs to be calculated, which also models covariance matrices. For event-related desynchronization (ERD) tasks, the spatial covariance matrix ($C_s$) is largely utilized

because the spatio-frequential information can be extracted from this matrix. This spatial covariance matrix can be computed using eq. (4.2), where $Z$ is a multichannel segment of a signal. $Z \in \mathbb{R}^{n \times s}$.

$$C_s = \frac{1}{s} Z Z^T \tag{4.2}$$

Therefore, with an increase in the number of channels, the complexity of the equation also increases. This results in the need for much higher computational resources, e.g., computational power, memory, and energy. Therefore, making an inference in our considered LiHEA framework using the conventional Riemannian algorithm can be complicated and inefficient. Thus, it may be difficult to analyze the EEG signals in polynomial time. Additionally, for continuous EEG monitoring and analytics, cloud-based implementation of this technique requires continuous Internet connectivity for the cloud-based implementation yielding high network overheads and significant latency in transferring the data. Thus, executing such computations on edge devices, such as EEG headbands, can be challenging since they are resource-constrained. To facilitate localized analytics in polynomial time, various AI inference models could be leveraged due to their much better generalization ability in contrast with the Riemannian geometry-based counterpart. Hence, the problem considered in our work is to systematically compare various AI inference models and determine the most viable one for LiHEA to carry out ultra-edge EEG analytics.

## 4.4 Proposed Methodology for Logic in Head-bands Based Edge Analytics (LiHEA)

In this section, we provide several approaches to construct our envisioned LiHEA framework, which aims to classify EEG data precisely and efficiently at the ultra-edge sensing layer. Several machine/deep learning techniques are considered by varying various parameters to determine the most suitable AI inference model for EEG signal analysis for incorporating with the LiHEA framework. The notations, used in this section and the remainder of the chapter, are listed in Table 4.1.

Our systematic approach of implementing LiHEA is shown in the steps of Algorithm 1. Here, $D$ contains the various datasets that were used in the experiment, which can be considered to be collected from the EEG headsets acting as IoT devices of various users. Initially, for each dataset $d$, the data need to be pre-processed, and an AI model, $M_N$, is applied to classify the EEG data to make a relevant prediction, e.g., confusion state(s). KNN, SVM, random forest, logistic regression, ANN, and CNN are considered as candidate techniques for implementing $M_N$. For each model $m_n$, various

cross-validation methods, $V_N$s are used to validate the model. Therefore, for each validation method $v_n$, the models are classified and the model, $m_{max}$, that achieved the highest accuracy $A_{max}$ is saved. Subsequently, a feature selection algorithm is applied for the model that achieves the maximum accuracy for the processed $D$ datasets. A grid search approach is then used to tune the hyper-parameters of the best model with the best dataset to ensure an optimized model. The optimal accuracy $A_{op}$ is then returned by the algorithm. In the following sections, the details of our customized candidate AI inference models are presented.

### 4.4.1 Pre-processing, Classification and Validation Approaches

The frequency of each EEG signal of the power spectrum lies in a diverse range. Therefore, to process the EEG data to feed into machine/deep learning algorithms, the min-max feature scaling method is applied in the entire feature set. The method normalizes the values in a scale of 0 – 1 using eq. (4.3). Thus, the standard deviation is reduced and the domination of higher values in the entire model is avoided. The scaling operation can be represented as follows,

$$x_{scaled} = \frac{x - x_{min}}{(x_{max} - x_{min})}. \tag{4.3}$$

Among the considered candidate models, KNN, SVM, logistic regression, ANN, and CNN, are applied in the scaled datasets. On the other hand, the random forest model does not require any scaling since it focuses more on the partitioning due to its decision tree-based properties.

The aforementioned models are applied to predict confusion from the EEG data to find out which model performs optimally. The random forest model adopts an ensemble classification method that predicts the classes based on the results of multiple decision trees. In this chapter, the random forest model is customized using $\log_2 n$ number of features and the number of trees ($N_t$) are altered among a set of values ranging from $N_{t0}$ to $N_{tn}$. Next, KNN, adopting a lazy learning method, is considered that determines class labels based on the most frequent value among $k_n$ neighbors. To implement the KNN as a candidate model for LiHEA, various number of neighbors ($k_n$) are considered to determine which value of $k_n$ yields the best performance. Additionally, logistic regression is considered as a candidate model, which predicts classes by calculating the probability of class labels after choosing a threshold between the two class labels from the training dataset. On the other hand, the SVM-based candidate model utilizes various kernels to transform high dimension EEG data to create an optimal decision boundary separating the class values. Therefore, SVM exhibits the capability to deal with a multi-dimensional feature set. Therefore, to get an idea of feature dimensionality

of the dataset that is used, various kernels like linear, sigmoid, and radial basis function (RBF) kernels are applied; and the value of a penalty factor (C) is also varied by a factor 10 from $C_0$ to $C_n$, i.e., $i^{th}$ C ($C_i$) = $C_{i-1} * 10$.

In addition to the aforementioned traditional machine learning models, two deep learning approaches (convolutional neural network (CNN) and a deep neural network (ANN)) are considered as candidate models for LiHEA for the lightweight EEG classification. For CNN, $c$ number of 1-dimensional convolution layers, along with $a_c$ hidden layers, are used in order to extract the underlying features from the EEG data for various filter sizes using $\delta$ as the activation function. Each of these layers is followed by a max-pooling and batch normalization layer. On the other hand, for the ANN-based model, $a$ number of hidden layers are used to classify the EEG confusion states. In both CNN and ANN models, the output layer consists of two neurons for two-class labels for simplicity, indicating confused or non-confused states.

The algorithms are validated using three techniques, namely, k-fold cross-validation, subject independent validation, and subject-specific validation. In the k-fold cross-validation technique, accuracy is calculated for each set of training and testing values, and the average of those accuracies are taken as the final accuracy. A few values for the random seed are tried, which states the starting number of random number generator for the folds, and the results are all produced with random seed = $r$ Setting a random seed ensures that the same folds are generated during every execution. On the other hand, the subject independent validation method evaluates the change of EEG signals of a user/subject in terms of other users. For instance, $(n-1)$ subjects' ($S$) data are considered for training while one subject's data is used as the testing set. Alternatively, the subject-specific validation method states how the EEG signal changes among a person. In this case, $(m-1)$ data of a single subject/user, $E_S$, is used to train the classifier while one data is held out as the testing set. This is repeated for each user, and the average accuracy is considered as result.

### 4.4.2 Feature Selection and Model Optimization

A feature selection procedure is performed to accomplish two objectives. First, we aim to know the most significant features for detecting confusion. If the most important features can be found, more detailed information with respect to those features can be collected in the future. Additionally, better knowledge can be derived regarding the features that contributes the most toward the confusion of a user. Secondly, it is carried out to improve the performance of a given classifier. There are some unimportant features that weigh down the classifier's performance. Also, too many features in a dataset might lead to model overfitting. Therefore, to avoid these issues, some of the insignificant features are removed to enhance the model performance.

For selecting the features, the random forest feature selection method is used. In this method, the important features are selected based on decrease in gini impurity on each node. The node, where the gini impurity decreases the most, is considered as the most important feature. For both feature importance determination and feature set selection, this method is adopted. k-fold cross-validation is applied to validate the feature selection. The feature selection is applied to the training folds, and then the random forest model is constructed again based on the new set of features. The model is tested on the test fold to assess the performance of feature selection. The common set of features in all the folds is considered as the final feature set; i.e., $F_{imp} = F_{list}[0] \cap F_{list}[1] \cap ...F_{list}[n-1]$, where $F_{list}$ denotes the list of selected features of all folds. For detecting the most important features, the features are ranked based on their gini impurity values. The feature with the highest impurity is considered the most important feature.

After the feature selection is performed, the model optimization is performed by executing a grid search. Thus, we find the optimal number of parameters for enhancing the model performance. The grid search is applied in the random forest algorithm altering various parameters, such as the number of trees, the maximum number of features, the maximum depth, the minimum sample split of the trees, and the necessity of bootstrap sampling.

## 4.5 Dataset Preparation

We use the confused student EEG brainwave data from a public repository [117] to train the candidate AI models described in the earlier section. For the dataset collection, twenty online educational videos were acquired where ten videos were pre-labeled as confusing (e.g., quantum mechanics, stem cell research videos, and so forth) while the remainder of the videos consisted of comparatively easy content (e.g., basic algebra, geometry, and so on). 10 students, $S$, were shown 10 videos, $E_S$, each, 5 from each category; and the students rated their difficulty of understanding or confusion on a scale of 1 to 7. Their EEG signals were also recorded from the frontal lobe for one minute over a 0.5-second interval using a single-channel wireless EEG headset called MindSet. Considered features include (*i*) subject id and video id (ranging from 0 to 9 for 10 students and 10 videos), (*ii*) the attention and meditation of each student sampled over 1Hz collected directly from the device to check the calmness and level of focus of the student, (*iii*) the raw EEG signals collected in volts from the device, and (*iv*) the delta, theta, alpha 1, alpha 2, beta 1, beta 2, gamma 1, and gamma 2 frequency bands in the power spectrum extracted from the raw EEG data. Each of the frequency waves was considered to be responsible for a particular set of actions. The sampling frequency for

the features extracted from the MindSet was 2 Hz. The subject ID and video ID were not considered while training the model in order to avoid biasness in the model. Therefore, from the dataset, a total of 12 features were used. For each video of one student, there were approximately 120 values. Thus, the total number of rows in the dataset was approximately 12800 for 100 data points (10 videos of 10 students). Additionally, since each student tends to describe her/his confusion in a different manner, the students' ratings of 1 to 7 were converted to binary ratings where any value greater than the median is confusing (1) while less or equal to the median is considered not confusing (0). Note that the confusion rating of each of the data points was the same throughout the rows for a single datapoint. The student rating is the class that will be predicted through the candidate AI models for LiHEA. Therefore, our considered problem is reduced to a binary classification where the state of confusion of a student/user is detected from the electrical activity of the brain. This dataset was selected as it is important to highlight the use of such ultra-edge systems outside hospital settings, specially in a domain which is crucial in the time of pandemic. This type of use case is imperative for detecting confusion of students in an online class to provide valuable feedback to the teachers or in a post surgical monitoring scenario where it is important to keep track of the surgical patients.

The dataset was rearranged in four different ways for training the AI models. In the first dataset (D1), each time segment of EEG data of a person was considered as a set of features. The minimum number of EEG data available for a person was 112. Therefore, 112 EEG recordings were considered for each datapoint. As each feature set contained 12 features, the total number of features in D1 was 112 X 12. Thus, for D1, the total number of features was 1344 for 100 datapoints. In the second dataset (D2), the mean of each datapoint was considered to represent the overall EEG records. Here, the number of features was 12 for the 100 datapoints. The third dataset (D3) contained the mean and standard deviation of each datapoint in order to obtain a greater variety of data. Since the standard deviation values of both pre-defined and user-defined labels were 0, those two columns were omitted from the dataset. Therefore, the number of features was 23; 12 mean features and 11 standard deviations of the features for the 100 datapoints. The fourth and final dataset (D4) included the minimum and maximum values of the EEG records in addition to the standard deviation and mean. So, the number of features considered in this dataset was 48 for the 100 datapoints. Since there was only one confusion rating for each data point, that was considered as the class value for all the datasets.

## 4.6 Performance Evaluation

The results of the systematic methodology, that led to finding the most viable AI inference model for incorporating with the LiHEA framework, are described in detail in this section.

### 4.6.1 Classification and Validation

This section represents the performance of KNN, logistic regression, SVM, random forest, ANN, and CNN models in the datasets for classifying confusion which were validated using the k-fold cross-validation method considering $k = 5$.

#### K-Nearest Neighbor (KNN)

The values of $k_n$ were altered between 3 to 9 for all four datasets (i.e., D1, D2, D3, and D4), and the results of the KNN model are depicted in Table 4.2. KNN on D2 and D4 achieved the highest accuracy of 66-67% with $k_n = 3$ *and* 5. However, in both cases, the performance dropped when the values of $k_n$ were increased. KNN applied to the other two datasets (D1 and D3), on the other hand, demonstrated poor performance for predicting confusion.

#### Logistic Regression

Fig. 4.1 demonstrates the results of the logistic regression model. It is noticed that the highest accuracy of 65% was achieved with logistic regression with D3. The performances of the logistic regression trained with the datasets D2 and D4 were slightly lower than that with D3.

#### Support Vector Machine (SVM)

Fig. 4.2 demonstrates the performance of SVM while the hyper-parameters, such as the kernels and penalty factor (C), were altered. Fig. 4.2(a) depicts the results of SVM when different kernels, namely linear, sigmoid, and rbf, were applied with C=0.1. In all cases, linear kernels performed significantly better than rbf and sigmoid. This result signifies that the features are linearly separated. Subsequently, the value of $C_0$ and $C_n$ was considered to be 0.01 and 100, respectively, for the linear kernel. The results are reported in Fig. 4.2(b). The performance of SVM with datasets D2, D3, and D4 improved with increasing values of C while the accuracy of D1 gradually decreased. SVM achieved its highest accuracy of 66% with D2 while C was 0.1 and the adopted kernel was linear.

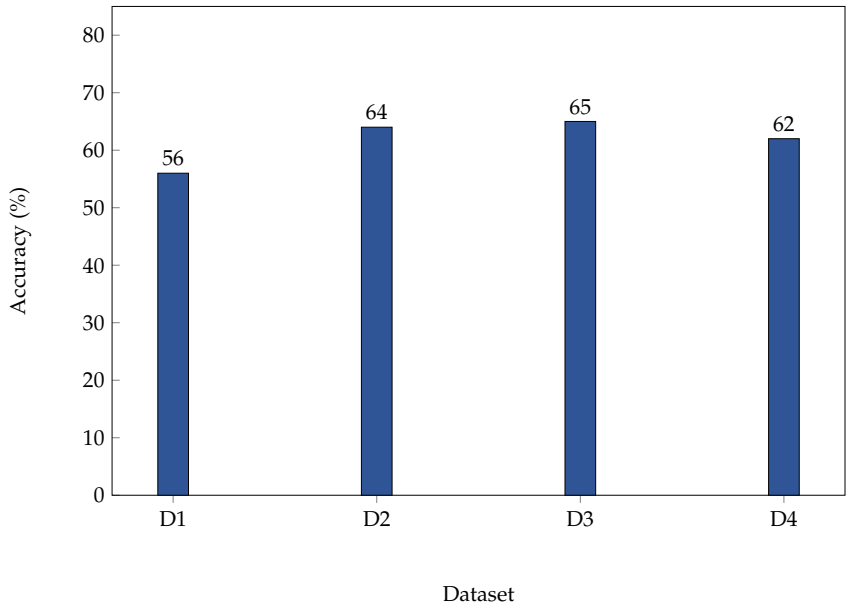


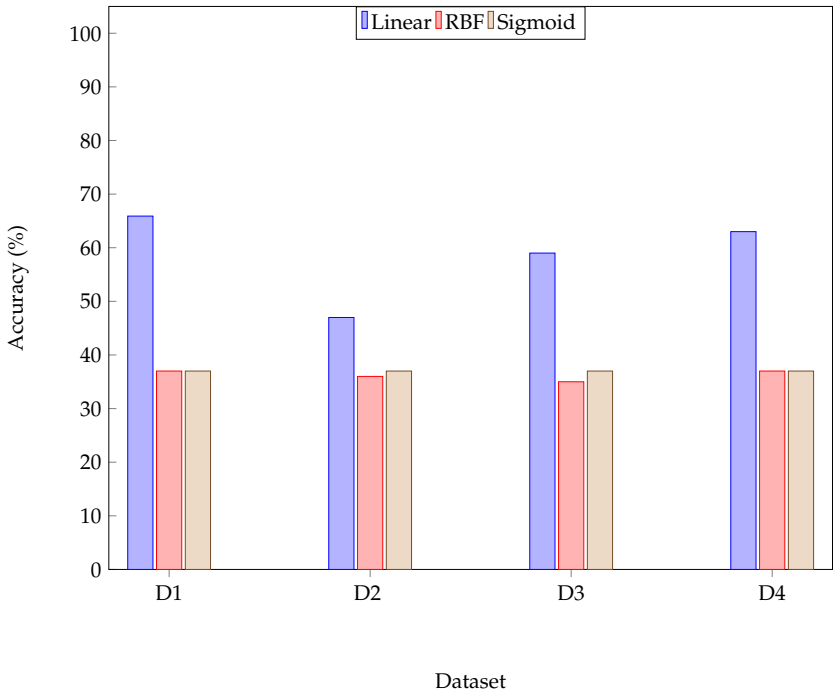

Fig. 4.1 Performance of logistic regression.

**Random Forest**

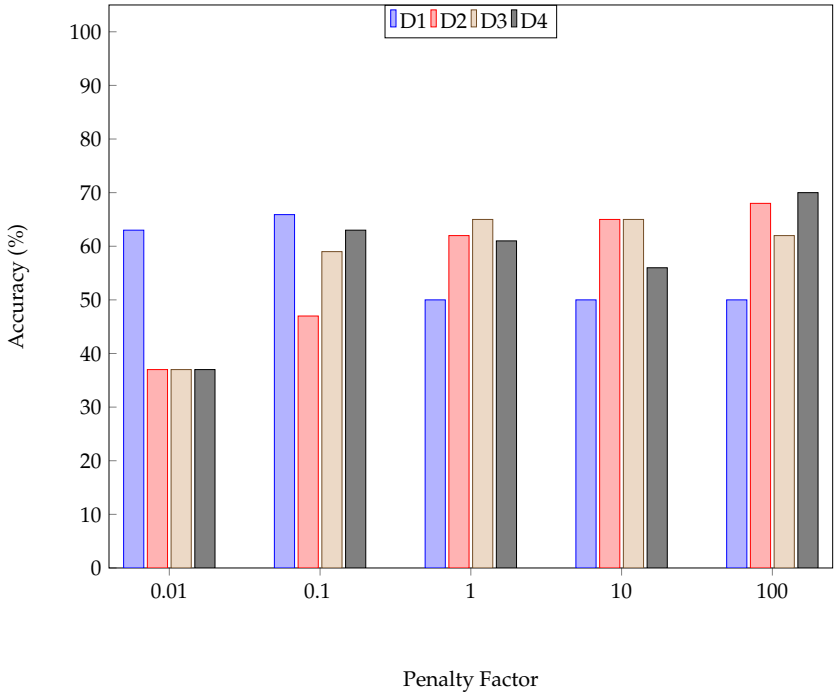
The maximum number of features for a tree in a random forest was initially set to $\log_2 n$ and $\sqrt{n}$ to find the performance of the model. The results of both models were the same in most cases with a slight difference in some of the cases. Therefore Fig. 4.3 represents the performance of random forest for $N_t$ ranging from $50 - 1000$ when the maximum number of features was $\log_2 n$. It is evident that dataset 2 has performed best for random forest with the highest accuracy of 80%. The performance was quite consistent with respect to $N_t$ with the highest being achieved at $N_t = 500$. For dataset 3, the performance was 77% when $N_t$ was 100 which came close to the highest accuracy. The overall performance of random forest was consistent for dataset 4 with accuracy varying from 67% to 68%. The performance of dataset 1 was much lower than the other datasets with the highest accuracy of 62%. Most of these datasets performed better when $N_t$ was considered to be 100.

**Deep Learning-based models**

The performances of ANN and CNN for confusion classification from the EEG datasets are demonstrated in Fig. 4.4. For ANN, the number of hidden layers $a$ was considered to be 3. Additionally for CNN, 3 one-dimensional convolution layers, $c$, were used followed by three hidden layers, $a_c$. ANN performed relatively better than CNN with the highest accuracy of 54.9% for the dataset D2. Rectified linear unit (ReLU) was used as activation function $\delta$. This result indicates that the CNN model did not have enough data to extract profound features from the EEG dataset, due to its rather small size.




(a) SVM with various kernel.




(b) SVM with various penalty factor.

Fig. 4.2 Results of SVM.

### Overall Performance: Comparative Analytics

For the overall performance evaluation, the highest accuracy of each algorithm was considered for the four datasets. The values are illustrated in Table 4.3. It can be noticed

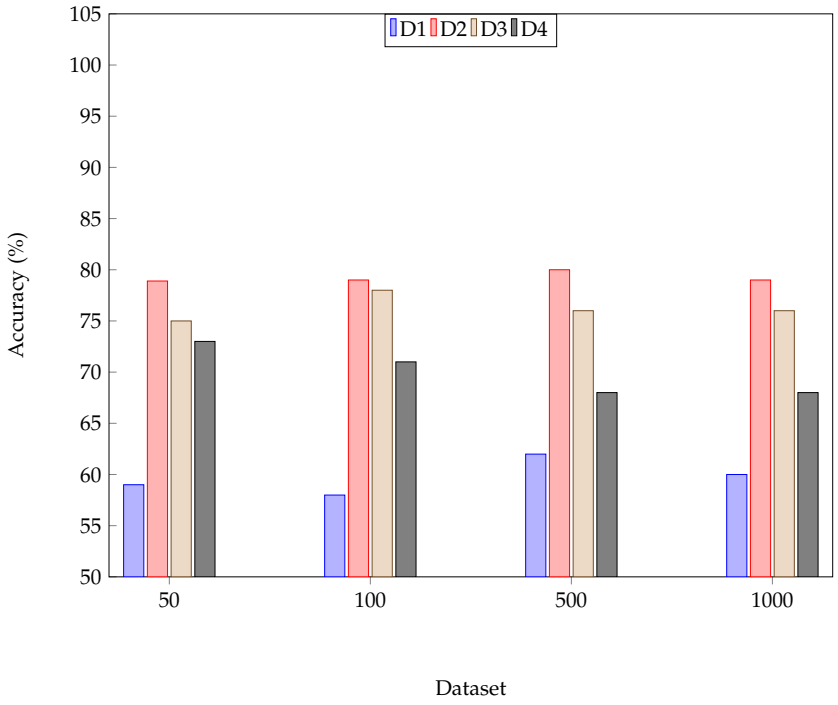



Fig. 4.3 Accuracy of random forest with various number of trees.

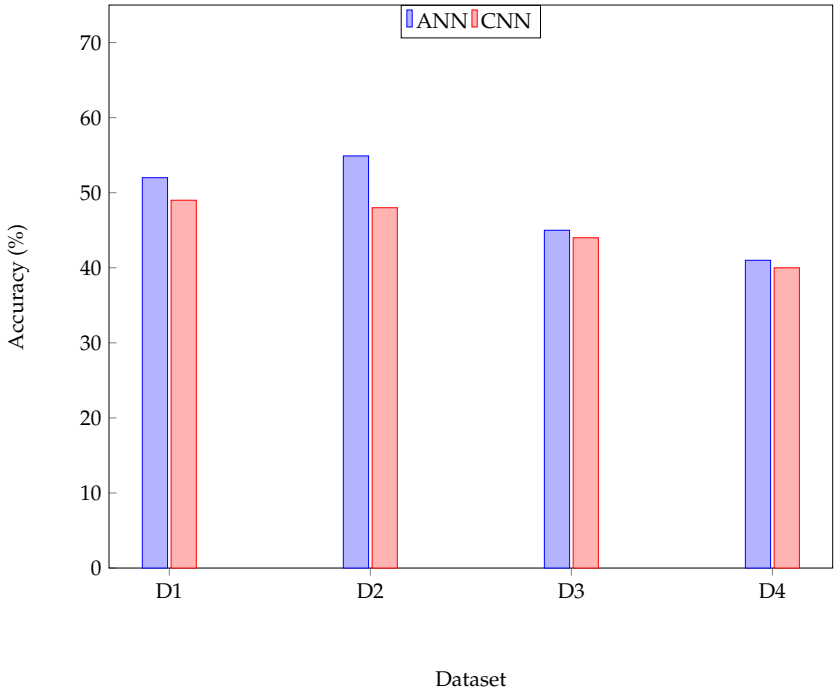



Fig. 4.4 Performance of ANN and CNN.

that the performance of random forest for datasets D2 and D3 is significantly higher than the other values in the table. Indeed, the other algorithms also exhibited comparatively better performances for these two datasets. This result signifies that datasets D2 and D3, containing the mean and standard deviation of the features, have been able to

represent the EEG data more accurately than the other arrangements/representations of data. Thus, among the four AI inference models used, the best result was achieved with random forest which was 80%. Logistic regression and SVM also performed comparatively better with results ranging from 56% to 70%. However, KNN and the two deep learning-based models (ANN and CNN) performed below par. The performance of these models with the dataset D1 was also lower than those with other datasets. One of the reasons might be due to high correlation among features as all the EEG data features were considered in this dataset. This led to significant drop in performance across all the candidate AI models for this particular dataset.

#### Validation Methods Comparison

Since the candidate models for LiHEA performed relatively better with the dataset D2 in contrast with the other datasets, D2 was leveraged to carry out a comparison between the various validation methods. Table 4.4 lists the performances of the three validation methods, i.e., 5-fold cross-validation (CV), subject independent validation (SIV), and subject-specific validation (SSV), for all the models implemented with dataset 2. For the 5-fold CV, $r = 10$ was used. For SIV, EEG data of 9 students were used for training the models, and the left out student's EEG data was used for testing. In SSV, for each student, 9 videos were used as training while one was employed for testing, and this was considered for all students. It is noticed that the overall performances of the 5-fold CV and SIV were slightly higher than that of SSV, although random forest shows a promising performance using the latter. SIV is a convenient way of validation since a new user's data is tested based on a model built with the existing user's EEG data. However, since the 5-fold CV demonstrated a more robust result; the results were generated using this validation technique.

### 4.6.2 Feature Selection

Random forest feature extraction was applied followed by a grid search to tune the hyper-parameters of the random forest model to improve the performance of the models. The results are stated in the following sections.

#### Determining important features

Table 4.5 lists the ranking of five most important features for the four datasets, and their gini impurity values. For datasets D1, D2 and D4, the most important feature was found to be the delta wave; and for dataset D3, it was the mean of the Theta wave. Delta activity is known to be stimulated while performing mental calculation, semantic tasks, and the Sternberg paradigm as stated in [118]. Attention was chosen as the second

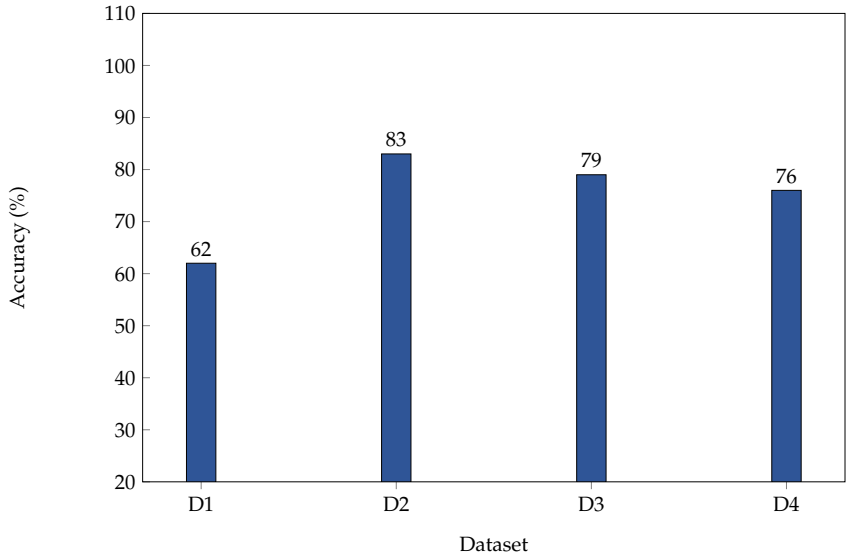



Fig. 4.5 Performance of random forest model after feature selection.

most important feature in datasets D2 and D3. This is, indeed, an important feature since the more attentive a person is, the less confused they will be. The Theta activity also appeared in the top 3 features in all the datasets. The theta frequency is known to contribute towards the learning capability, memory, and intuition [119]. Therefore, it can be established as an important feature for detecting confusion. It can also be noticed that the most important features are the mean values of datasets D2, D3, and D4. In other words, the mean values provide a better interpretation of the dataset than the other measures. Therefore, it can be concluded that delta, attention, and theta frequencies are the three most important features according to majority of the datasets.

**Enhancing model performance**

Since the random forest outperformed the other candidate AI inference models for LiHEA, we now aim to investigate the possibility of further performance improvement of the random forest model. In this vein, a set of important features are selected using the random forest feature selection method. The results of the highest accuracy gained from each dataset for 100 trees are represented in Fig. 4.5. The performances of the model with all datasets significantly improved after using the best features to construct model. The highest accuracy was achieved with the dataset D2 (83%) followed by D3 (79%).

**Model Optimization with Grid Search**

A rigorous hyper-parameter tuning was performed for the dataset D2 in order to optimize the performance of the random forest model using the selected features because of the model's promising performance. The features selected for constructing the model were attention, delta, theta, beta1 and alpha1. The accuracy of the model improved to 90% using the set of parameters illustrated in Table 4.6.

### 4.6.3 Evaluating lightweight property of the models

In this section, we validate the deployment of the AI-based algorithms on ultra-edge IoT devices such as EEG headbands. To ensure that the algorithms can be successfully integrated with the sensors, we need to evaluate various properties of the models such as their memory consumption i.e whether it is able to finish the task within the limited memory and if the model is able to make a quick and efficient inference using the limited resources. These properties can be used to evaluate the lightweight condition of the inference model and can be considered inversely proportional. In this vein we conduct a numeric analysis to assess the lightweight property of our top three candidate AI inference models (i.e., random forest, ANN, and CNN) for LiHEA. The analytical results in terms of processing time and memory consumption are demonstrated in Figs. 4.7 and 4.6, respectively. The analysis was performed to measure the overhead of the entire process starting from data collection to preprocessing followed by analysis. Here, since we used an already preprocessed dataset, we computed the preprocessing overhead separately by generating a random signal and applying fast fourier transform to account for the computation overhead in the process. In addition to that, we also calculated the overhead of the model to make an inference on the preprocessed data. Note that the inference time and the percentage of memory consumption with respect to the overall capacity of the machine of the aforementioned three algorithms were measured in a base device having an Intel Core i7, 3.00 GHz central processing unit (CPU), 16GB random access memory (RAM), powered by NVIDIA RTX 2060 graphics processing unit (GPU). Subsequently, the approximate time and memory required for other IoT devices, such as NVIDIA Jetson Nano (Quad-core ARM Cortex-A57 @ 1.43GHz and 4GB RAM), Raspberry Pi 4 (Quad-core Cortex-A72 @ 1.5GHz and and 2GB RAM), and Raspberry Pi 3 (Quad-core Cortex-A53 @ 1.4GHz and 1GB RAM), were numerically analyzed with respect to the base device. It can be viewed that, the memory consumption of CNN and ANN in Raspberry Pi 3 exceeds the allocated memory. This implies that the random forest emerges as the ideal model to be integrated with raspberry pi 3 in terms of time and memory. On the other hand, for Raspberry Pi 4 and Jetson Nano, the time and memory consumption of random forest was much lower compared to the other two

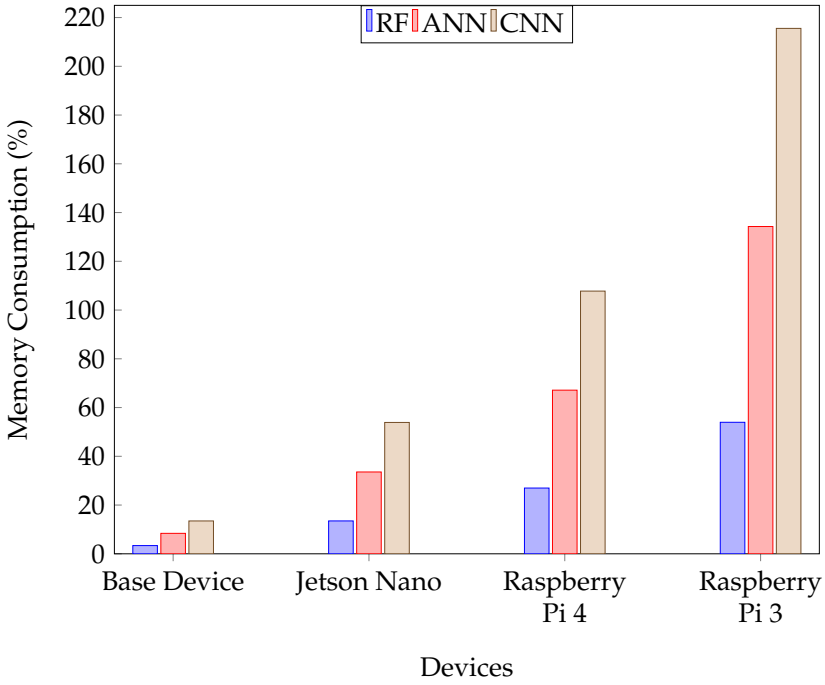



Fig. 4.6 Percentage of memory consumption of selected algorithms with respect to various devices.

algorithms. Additionally, among the two deep learning algorithms, CNN consumed a slightly more time and memory as it inspects and analyzes the features in a much more complicated manner using the convolution layers that requires substantially more computational resources.

In summary, for constructing our proposed LiHEA model, random forest can be regarded as the ideal model in terms of performance, execution time, and memory consumption for IoT devices like Raspberry Pi 3, Raspberry Pi 4. NVIDIA Jetson Nano, and so forth, that can be used for smart EEG headband implementation.

### 4.6.4 Complexity analysis of the top candidate AI inference model for LiHEA

In the remainder of the section, we present the time complexity analysis of our top candidate AI inference model (i.e., random forest) for LiHEA in both training and running phases. In the conducted analysis, we assume that the operations, such as initialization, addition, concatenation, are basic operations with constant time, i.e., contributing to the time complexity of $O(1)$.

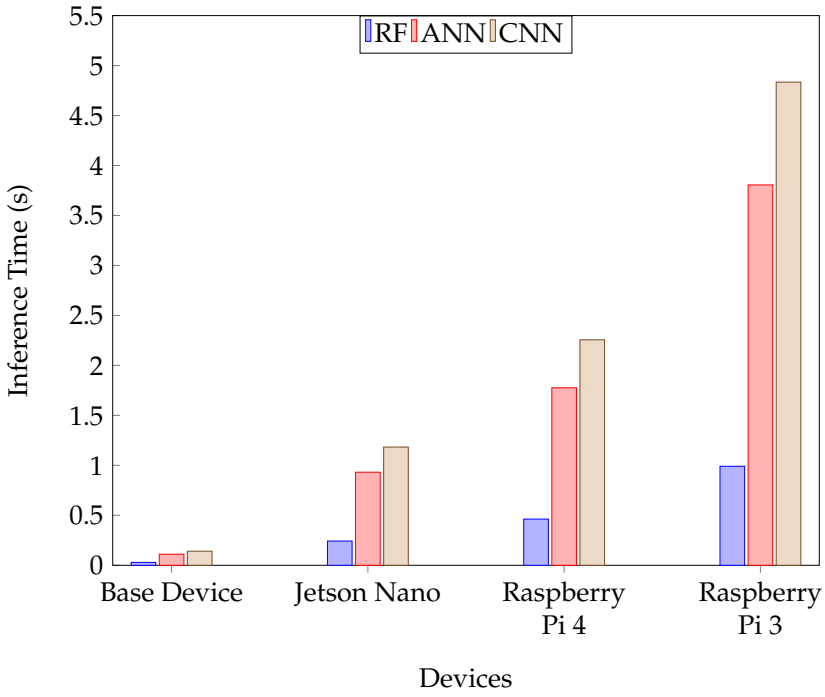



Fig. 4.7 Inference time of different algorithms in various devices.

**Training Phase**

Let the total time required to train the AI inference model for LiHEA be denoted by $T(L_{TR})$. We divide $T(L_{TR})$ into two phases, namely the data pre-processing $T(DP)$ and classification $T(CLF)$. Therefore, the total time required to train the model can be represented as:

$$T(L_{TR}) = T(DP) + T(CLF). \tag{4.4}$$

For data pre-processing, at first we considered a signal transformation step that was carried out to convert the signals in their frequency domain. The complexity of signal transformation step can be represented as $O(x_t Log(x_t)$, where $x_t$ denotes the total number of training data. Additionally, if $y_t$ is the total number of features, the time complexity of the min-max feature scaling to pre-process the data can be stated as:

$$T(DP) = O(x_t * y_t). \tag{4.5}$$

For classification, various candidate models were implemented whereby each model consist of its own time complexity for training. In this segment, the time complexity of random forest $T(CLF_{RF})$ is presented since it achieved the highest accuracy. In the random forest model, if $N_t$ denotes the number of trees, then $T(CLF_{RF})$ can be written as:

$$T(CLF_{RF}) = O(x_t^2 * y_t * N_t). \tag{4.6}$$

Therefore, the total training time required by the LiHEA model using the random forest model may be represented as:

$$T(L_{TR}) = O(x_t * y_t) + O(x_t^2 * y_t * N_t). \tag{4.7}$$

**Inference Phase**

On the other hand, during the running or inference phase, the trained model is expected to make an inference (i.e., prediction) for a newly collected set of EEG data $x_r$ in order to determine the class label of the data in terms of the confusion state of the user. Therefore, the time complexity $T(R)$ to make an inference on $x_r$ data can be expressed as:

$$T(R) = O(x_r) \tag{4.8}$$

Hence, it may be concluded that, our customized random forest-based inference model for the envisioned LiHEA framework requires linear time to make a prediction. Therefore, it addresses the lightweight requirement of such a model required in LiHEA while achieving a much higher prediction accuracy in contrast with the other candidate models as described earlier.

Table 4.1 Table representing the notations of the chapter and their respective definition.

| Notation | Definition |
|---|---|
| $A_{max}$ | Maximum accuracy |
| $m_{max}$ | Model giving maximum accuracy |
| $d_{max}$ | Dataset giving maximum accuracy |
| $r$ | Random seed |
| $P_H$ | Dictionary containing hyperparameters of the model |
| $D$ | List of datasets |
| $M_N$ | List of model names |
| $V_N$ | List of validation methods |
| $A_m$ | Model accuracy |
| $S$ | Subjects |
| $E_S$ | Data of each subject |
| $n$ | Number of subjects |
| $m$ | Number of data of each subject |
| $d_{train}, d_{test}$ | Train and test sets for each fold |
| $F_{sel}$ | Selectes list of features |
| $F_{rank}$ | Ranking of features according to their gini impurity |
| $A_{FS}$ | Accuracy of dataset containing only important features |
| $P_{HT}$ | Optimal hyperparameters for model $m$ |
| $A_{op}$ | Optimal accuracy |
| $x$ | Features of the dataset |
| $y$ | Actual classes of the dataset |
| $\bar{y}$ | Predicted classes |
| $N_t$ | Number of trees used in Random Forest |
| $k_n$ | Number of nearest neighbor used in KNN |
| C | Penalty factor used in SVM |
| c | Number of 1D convolutional layer |
| $a_c$ | Number of hidden layers used in CNN |
| $\delta$ | Activation function |
| $a$ | Number of hidden layers used in ANN |
| $k$ | Number of folds in k-Fold cross-validation |
| D1, D2, D3, D4 | Datasets 1, 2, 3, 4, respectively |

**Algorithm 1:** Systematic methodology for LiHEA.

**Input:** $d$ (Dataset Containing EEG Signal Features and Their Respective Classes)
**Output:** $A_{op}$ (Optimal Performance of the Model)

**Function** `validateModel`$(d, M_N, V_N, P_H, r)$**:**
  **if** $V_N =$ *"K Fold CV"* **then**
    Divide $d$ into $k$ folds as per $r$
    $F_{train} = (n-1)k$
    $F_{test} = k$
  **end**
  **else**
    Divide d as per $S$
    **if** $V_N =$ *"Subject Independent"* **then**
      $F_{train} = (n-1)S$
      $F_{test} = S$
    **end**
    **else if** $V_N =$ *"Subject Specific"* **then**
      $F_{train} = (m-1)E_s$
      $F_{test} = E_s$
    **end**
  **end**
  **return** $F_{train}, F_{test}$
**end**
$A_{max}, m_{max}, d_{max} \leftarrow 0$
Initialize $P_H, r, D, M_N, V_N$
**if** *Preprocessing required* **then**
  Scale D in range $(r_0, r_1)$
**end**
**foreach** $d, m_n, v_n \in D, M_N, V_N$ **do**
  $d_{train}, d_{test} = validateModel(d, m_n, v_n, P_H)$
  Find $A_m$ using eq. (3.12)
  $A_{max} = max(A_m, A_{max})$
  $m_{max} = M[A_{max}]$
**end**
**foreach** $d$ *in* $D$ **do**
  $F_{sel}, F_{rank} = FeatureSelection(d)$
  $d_{train}, d_{test} = validateModel(d[F_{sel}], m_{max}, v_n, P_H, r)$
  Find $A_{FS}$ using eq. (3.12)
  $A_{max} = max(A_{max}, A_{FS})$
  $d_{max} = D[A_{max}]$
**end**
Initialize $P_H$ for $m_{max}$
$P_{HT}, A_{op} = TuneHyperparameters(d_{max}, m_{max}, P_H)$
**return** $A_{op}$

Table 4.2 Performance of K-nearest neighbors with various $k_n$ values.

| $k_n$ | Accuracy (%) | | | |
|---|---|---|---|---|
| | D1 | D2 | D3 | D4 |
| 3 | 50 | 66 | 60 | 67 |
| 5 | 58 | 57 | 58 | 67 |
| 7 | 60 | 61 | 61 | 62 |
| 9 | 57 | 63 | 58 | 60 |

Table 4.3 Overall performance of the AI models.

| Algorithms | Accuracy (%) | | | |
|---|---|---|---|---|
| | D1 | D2 | D3 | D4 |
| Random Forest | 61.9 | 80 | 78 | 73 |
| KNN | 60 | 66 | 61 | 67 |
| SVM | 65.9 | 68 | 65 | 70 |
| ANN | 52 | 54.9 | 45 | 41 |
| CNN | 49 | 48 | 44 | 40 |
| Logistic Regression | 56 | 62 | 66 | 60 |

Table 4.4 Performances of the candidate models using various validation techniques.

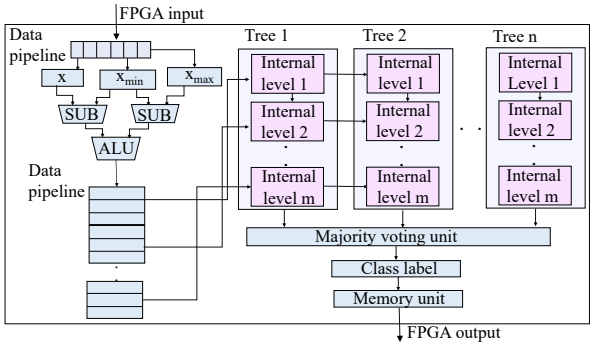
| Algorithms | Accuracy (%) | | |
|---|---|---|---|
| | 5 Fold CV | SIV | SSV |
| Random Forest | 80 | 75 | 83 |
| KNN | 66 | 65 | 58 |
| SVM | 68 | 63 | 46 |
| ANN | 54.9 | 55 | 49 |
| CNN | 48 | 47 | 45 |
| Logistic Regression | 62 | 62 | 41 |

Table 4.5 Top five important features.

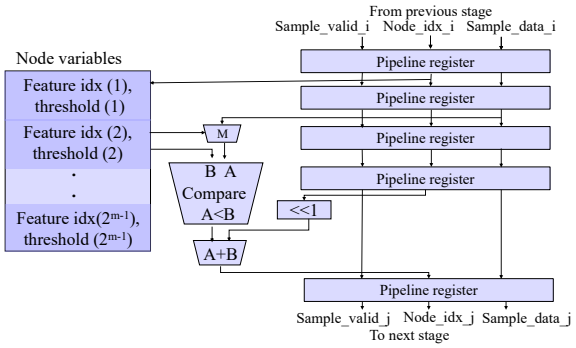
| Feature | Gini Impurity | Feature | Gini Impurity |
|---|---|---|---|
| Dataset 1 | | Dataset 2 | |
| Delta57 | 0.0047 | Delta | 0.145 |
| Delta3 | 0.0038 | Attention | 0.1374 |
| Theta98 | 0.0037 | Theta | 0.1337 |
| Theta16 | 0.0037 | Beta1 | 0.116 |
| Alpha1_85 | 0.0036 | Alpha1 | 0.0904 |
| Dataset 3 | | Dataset 4 | |
| Theta | 0.0898 | Delta0 | 0.0616 |
| Attention | 0.0866 | Theta0 | 0.0498 |
| Delta | 0.0852 | Attention0 | 0.0491 |
| Beta1 | 0.0716 | Beta1_0 | 0.0422 |
| Meditation_1 | 0.06 | Meditation1 | 0.042 |

Table 4.6 Grid search results.

| Hyperparameter | Values |
|---|---|
| Maximum Depth | 50 |
| Number of Trees | 57 |
| Minimum Sample Split | 2 |
| Bootstrap | True |




(a) FPGA implementation of random forest.




(b) Architecture of each internal level.

Fig. 4.8 FPGA implementation of the models.

### 4.6.5 FPGA Implementation of the Approaches.

Due to the satisfactory performance of random forest algorithm, we have analyzed the algorithm in terms of feasibilityIn order to achieve that, we have implemented it in Field Programmable Gate Arrays (FPGA). Fig. 4.8 shows the FPGA implementation of Random Forest model in order to represent the circuit configuration of the model. Initially EEG signals are collected through FPGA I/O panel where it is stored in a data pipeline. In random forest in Fig. 4.8(a), inspired from [120], the pipeline data is preprocessed using min-max feature scaling which is stored in another pipeline. The data from the second pipeline is then sent to the random forest model. The model consist of n trees, each tree having m levels containing the nodes of the tree. The computation of each level of a tree will be done in parallel, therefore saving valuable time which is represented by the horizontal arrows between the trees. The results of each tree is passed in the majority voting unit where the class label is determined. The generic computation of each level is described in Fig. 4.8(b) where multiple pipeline register has been used for non-blocking access to the local memory. The node index will enable tracking of the particular node in order to make the architecture behave accordingly.The computed class label is saved in memory unit and is then transferred to the I/O stream of FPGA from where the results can be attained by the user.

## 4.7 Enabling Robust Model Training based on EEG Data Augmentation

The results of the considered AI inference models using the four datasets revealed that the dataset-size could be a key bottleneck for more advanced deep learning-based algorithms. In this section, we aim to explore if data augmentation-based synthetic EEG samples generation could train a more robust model to further boost the performance of the considered machine as well deep learning methods.

### 4.7.1 Investigating DCGAN-Assisted EEG Data Augmentation

The availability of a variety of EEG data is challenging since data collection is time consuming and requires plenty of resources. Hence, it is often difficult in some domains to achieve a good accuracy of the models. To further investigate how a relatively larger dataset may impact the performance of our candidate models for LiHEA, a data augmentation framework is designed to account for a variety of data, required to train a more robust model. In this vein, we apply a deep convolutional generative adversarial network (DCGAN) model, which generates synthetic EEG traces by generating representative samples of EEG signals when the size of dataset is not sufficient.

The steps of the DCGAN algorithm used for this chapter are demonstrated in Algorithm 2. For each class ($c$), the function “Generate Data" (*line* 1) takes in as input the original data ($s$), the maximum number of iterations ($I$), batch size ($S_B$), and the number of samples to be generated ($N_S$). The models $M_G$ and $M_D$ are constructed based on the architecture shown in Fig. 3.2. In lines $6-10$, for each iteration ($i$), $N_S$ latent vectors ($\eta$) are generated from a Gaussian distribution that are passed to $M_G$ to generate new EEG signals. The generated signals are combined with random samples of the original EEG signals($D_{batch}$) of size $S_B$, and the combined signals $D_{combined}$ are assigned to the corresponding real and fake labels. Next, $M_D$ is trained using $D_{combined}$. In the subsequent steps, new latent vectors ($\eta_{new}$) are produced, and the generator is trained with $\eta_{new}$ while $W_D$ is fixed. This allows $W_G$ to be updated based on the output of $M_D$. $Loss_A$ denotes the adversarial loss while and $Loss_D$ indicates the discriminator loss, and they are used to update $W_D$ and $W_G$, respectively. The process continues for $I$ iterations, and the ultimate generated value ($D_{gen}$) for $c$ is returned by the function.

**Algorithm 2:** Proposed DCGAN Structure.

**Input:** $s$ (Original Samples of EEG Signals)
**Output:** $s'$ (Newly Generated Samples of EEG Signals)

**Function** `Generate Data`$(s, I, S_B, N_S)$**:**
  Construct $M_G$ model as illustrated in fig. 3.2
  Construct $M_D$ model (fig. 3.2)
  $start = 0$
  **for** $i$ *in* $I$ **do**
    Generate $N_S$ samples of $\eta$
    $D_{gen} = M_G(\eta)$
    $D_{batch} = s[start : S_B]$
    $D_{combined} = D_{batch} + D_{gen}$
    Add corresponding labels to $D_{combined}$(*Real or Fake*)
    Train $M_D$ using $D_{combined}$
    $Loss_D = M_D(D_{combined})$
    Generate random latent vectors ($\eta_{new}$)
    Make $W_D$ non-trainable
    $Loss_A = M_D(M_G(\eta_{new}))$
    Update $W_G$
    $start = (start + S_B)$ % $length(s)$
  **end**
  **return** $D_{gen}$
**end**
Initialize $I$, $S_B$, $N_S$
$s' = \emptyset$
**foreach** $c \in C$ **do**
  $s'_c = GenerateData(s_c, I, S_B, N_S)$
  Append class label $c$ to $s'_c$
  $s' += s'_c$
**end**
**return** $s'$

### 4.7.2 Results and Computational Analysis of DCGAN-Enabled AI Model

The amount of data generated by DCGAN in addition to the original EEG signals is classified using the top three candidate algorithms for LiHEA, i.e., Random Forest, ANN, and CNN, as depicted in Fig. 4.9. The results were computed for the dataset D2 since that dataset demonstrated the best results in the earlier section (without data augmentation) for the majority of the algorithms. Here, the horizontal axis represents the percentage of newly generated signals with respect to original signals. It can be observed that the performances of the deep learning models (ANN and CNN) are significantly increased by approximately 20% with the increase in EEG samples. By

increasing the training data by 400%, ANN achieved the highest accuracy. On the other hand, for CNN, the data had to be increased by approximately 800%. When 1600% new data was generated, the accuracy of both the algorithms started to deteriorate, making 800% the most suitable value to generate the synthetic traces. However, the performance of random forest slightly deteriorated with the increase in data. This result indicates that the performance of the random forest model does not get impacted by GAN in a significant manner. Thus, it can be concluded that with the availability of data in a larger scale, deep learning models can be used to extract profound features directly from a dataset with greater preciseness and less pre-processing steps. While the incorporation of DCGAN framework increases the training complexity [121] of LiHEA model significantly, it increases the performances of the deep learning models (i.e., ANN and CNN) by approximately 20%. However, the performance increase of these models still get overshadowed by the performance of an optimal random forest model which achieved an accuracy of 90%, reported in the preceding section. Therefore, the random forest emerges as the most viable choice to implement LiHEA in linear time without requiring any data augmentation techniques. To summarize, the top AI inference model (i.e., random forest), while requiring some pre-processing/manual feature extraction, results in a much higher accuracy for both basic and DCGAN-assisted training. On the other hand, while the deep learning models (i.e., ANN and CNN) require no pre-processing, they are rendered with a much lower accuracy without data augmentation, and a much improved (yet still lower than the random forest model performance) accuracy with the DCGAN-based data augmentation approach.

## 4.8 Conclusion

In this chapter, we proposed the LiHEA framework to seamlessly integrate EEG analysis with commercially available, resource-constrained EEG headbands facilitating continuous monitoring and ultra-edge computing of brain signals. We considered a specific use-case for EEG classification to predict confusion states of students/users while attending online, academic video lectures. The LiHEA framework was designed to avoid sending unnecessary raw EEG data to the cloud for computation, hence saving precious network bandwidth and substantially reduce communication latency. We formulated a problem for finding the most suitable AI inference model to satisfy the lightweight property of LiHEA to be implemented in EEG headsets. In this vein, several machine/deep learning models, namely KNN, logistic regression, SVM, random forest, ANN, and CNN were considered for EEG classification to detect/predict confusion status of the users. We customized the dataset in four different variants to ascertain

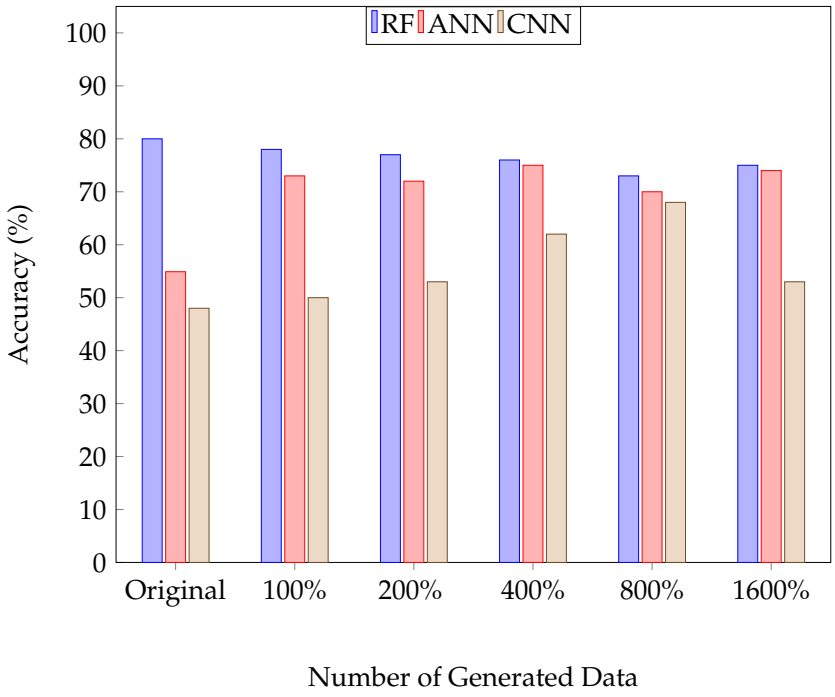



Fig. 4.9 Performance of various algorithms with DCGAN augmented data.

which variant represents the EEG data more elaborately and in which AI inference model. Random forest exhibited the highest accuracy (90%) on a dataset containing the mean of EEG signals for a datapoint, when grid search was applied to tune the hyper-parameters of a few selected features. This was the best performance attained by this dataset, compared to previous studies in the literature. The most important features were selected using the random forest feature selection method. Delta wave, attention, and theta wave were revealed to be the three most important features for detecting confusion. Additionally, the EEG analysis was carried out using three validation methods, k-fold cross-validation, subject-specific validation, and subject-independent validation. k-fold cross-validation yielded the best results for most of the algorithms and hence, was used as the validation method for all conducted analysis.

Due to the rather small size of the available EEG dataset, we carried out further experiments on the models to improve the performance by augmenting DCGAN-based EEG signal augmentation and introducing more variation in the dataset. Due to the synthetic samples, the performances of ANN and CNN, i.e., both deep learning models, significantly improve. However, they still were unable to outperform the performance of the initial random forest without data augmentation. Thus, the random forest is identified as the most viable AI inference model for the LiHEA framework, and its lightweight time and memory features demonstrate its suitability to be adopted for ultra-edge EEG analytics.

### 4.8.1 Future Work

In the future, this proof-of-concept random forest-based LiHEA framework can be implemented in various platforms, ranging from online courses, surgical monitoring, continuous monitoring of post-surgical patients under rehabilitation, sleep monitoring, and so forth, whereby brain signals analysis can be carried out effectively and promptly with portable and affordable IoT devices. This work can be regarded as a first step for developing logic-in-headband and smart IoT devices for carrying out edge analytics on the EEG data, and in the future, we will also consider the complexity of signal pre-processing and feature extraction.

# Chapter 5

# Enabling Lightweight Ultra-Edge EEG Analysis by Employing a Comparative Study on Various AI-based Techniques

## 5.1 Introduction

The recently emerging applications of the Internet of Things (IoT) and wearable devices are dramatically transforming the digital health landscape by offering continuous, efficient, portable as well as convenient monitoring systems [8, 9]. IoT-based health data gathering followed by cloud-based process and medical analytics have made the users more attentive toward their health. The main concepts of these IoT technologies usually include sending collected data through sensors that are then transferred to a central server or cloud over the internet for further processing and analysis. The entire process consumes precious bandwidth and adds communication delay while raising privacy as well as security concerns [10]. However, such concerns can significantly impact a person in biomedical applications, particularly in neurological use-cases. For instance, in order to analyze brain signals, Electroencephalogram (EEG) has acquired significant attention due to its cost effective, accurate representation of the signals. Previously, various edge/fog computing approaches were proposed to reduce data retrieval latency [11]. To facilitate a speedier and efficient data analysis method, the computation tasks need to be migrated to even further along the network edge, i.e., at the data generation point. Therefore, if a localized EEG analysis can be performed on the IoT device itself, the process of data collection and analysis can be made seamless, swift, and secure. However, this requires a lot of computational resources which poses a burden on the traditional IoT devices (i.e., ultra-edge nodes) which are typically resource-constrained. In this vein, we propose the concepts of a lightweight model

incorporated with the commercially available EEG headbands that will have inferring capability at the edge within limited resources. The EEG analytics will be carried out in the ultra-edge node thus avoiding any interaction with remote servers and the data will then be transferred to the edge devices such as smartphones, routers, etc.

In a traditional EEG analysis that is carried out in a cloud or edge server, the data is first processed with the help of various signal processing techniques such as noise and artefact elimination, signal transformation and so forth. The signals are usually first transformed to the spectral domain for ease of computation and also for a better representation of the data. This transformation, however, is usually very complex and requires heavy computational resources and time. Our main goal for the ultra-edge analysis is to reduce the computing steps towards EEG analysis as much as possible since the devices are resource constrained. Therefore, we explored several AI-based techniques to further reduce the complexity of the computation by eliminating some of the signal processing steps, to make the model more lightweight while still being able to perform robust computation. Various deep learning architectures are used now a days to perform complex computation in classifying and forecasting, by detecting various pattern or features in the data. However, with a deeper interpretation of the data, the complexity of the AI-based model also increases. Thus there arises a trade-off between better performance and simpler model. In this vein, we proposed a hybridized model, that will be able to employ more attributes of the collected EEG signals without greatly increasing the complexity of the model. The main concept of the hybridized approach was to add more information to the model, for a better interpretation of the EEG signals. In a traditional model, the signals are analysed to detect patterns for classification. However, there are various other minor information, that are not provided in the process such as the channel the data is coming from or the device from which the data is being collected and so forth. It is established that the EEG data coming from different devices might not have the same property. However, to make the model more generalized, to be used for various devices containing different channels, it is important for the model to be able to distinguish between them in order to provide a precise outcome. Although there are existing deep learning based architectures which considers the signals as 3D input having timesteps and channels of the signals, the models are limited to a particular device as the number of channels may vary from device to device. Thus in our hybrid model, we considered all such scenarios to make the EEG analysis on ultra-edge more universal and robust while not drastically increasing the complexity of the model in the process. Our proposed model was compared with several other AI-based classification algorithms to evaluate the performance. We considered the use-case of a motor imagery dataset where the participants were asked to think of four words, which are 'right','left','forward' and 'back'. The individual words were repeated

for 30 trials for each participant, and their corresponding EEG signals were recorded using a BioSemi ActiveTwo$^{TM}$ having 64 channels. This use-case was used, as it is an important indicator of how the EEG signals change with words. Therefore in the future, the experimentations can be carried out for more words to expand the relevance of the Logic-in-Sensor device such as a communication device for the speech impaired people within reasonable cost and efficiency.

In this chapter, we aim to conduct brainwave signal capturing and analysis directly on IoT and wearable devices, such as EEG-headsets/headbands. Due to their resource constraints, we proposed a deep learning based hybrid approach which will be able to detect patterns of an EEG signal with greater precision and less computational complexity. In the rest of the chapter, we have further discussed the gap in literature, the problem that is being addressed and our hybrid technique as a solution.

## 5.2 Related Work

Various machine learning and deep learning architectures have been used in the literature to analyse EEG data. However, most of these models only make their decision solely based on the EEG signals and no other relevant aspects of the signals. In this section, we present some of the research works that employs various feature extraction and state-of-the art machine learning and deep learning techniques to analyse EEG data.

The study in [122] proposed a deep evolutionary approach to extract features and classify the collected EEG signals. The model was applied on three use-cases to detect mental state, emotional state and digits where the features of the EEG signals were first extracted using Immune Clonal Algorithm and they claimed that the lightweight model can analyze the signals within 10% of the other deep learning models' time, with a slight trade-off in accuracy. In [123], discrete wavelet transform was first performed to extract the features followed by classification using Multilayer Perceptron. They achieved an accuracy of 93% using their model to classify between three emotions which are- happy, neutral and sad. Lawhern et al. proposed a compact model in [124], which employs depthwise and separable convolutions to classify the EEG signals. The study performed both cross-subject and within subject classification and their model was able to perform comparatively higher than other models such as Deep ConvNet and Shallow ConvNet for multiple datasets. In [125], an universal GRU-based model was proposed which employs 1D as well as 2D convolutions to extract features followed by a GRU layer, which was added due to the time series property of the EEG signals. The model was implemented using multiple dataset was validated using five datasets namely - SEED: Emotion Detection, MindBigData BMNIST: Digit Recognition, ThoughtViz:

Object Recognition, Sensory Motor Recognition: Task Identification and Feedback Error Related Negativity: Error Detection and it was able to perform better than CNN, EEGNet [124] and autoencoder-based models. Lastly, in [126], the EEG signals were preprocessed using principal component analysis (PCA), which was passed to stacked autoencoders to perform an emotion recognition classification. They achieved an accuracy of 54% using their proposed method while all the channels of the DEAP dataset were incorporated.

Apart from that, various types of hybrid models are being used to perform classification on the EEG data. In the next part of the section, we present various types of hybrid model that were presented in the literature to classify EEG signals. In [127], the author divided the EEG signal into smaller time windows. A spatial filtering was then performed in the smaller time windows using the one-versus rest filter bank common spatial pattern algorithm the results of which were passed to separate convolutional layer. The features of the different convolutional layers were concatenated and passed to the LSTM layer, for performing classification. The hybrid model performed significantly higher than existing models in the BCI competition dataset. A transfer learning-based deep neural network architecture was proposed by the same author in [128], where, the extracted features were concatenated and passed to two different layers - a CNN layer and an LSTM layer, the outputs of which were then passed to the fully connected layer. The same BCI competition dataset was used to evaluate the model and the model gave a good performance for multiple subjects in the dataset. Additionally, Budak et al. [129], in his paper, employed another hybrid architecture, which employs three types of feature extraction procedures in the EEG signals itself as well as the EEG spectrogram images. The extracted features are then sent tto three separate bi-directional LSTM model where the class label is selected based on majority vote. The model was able to achieve state-of-the-art accuracy for the MIT-BIH Polysomnographic database having an accuracy of 94.31%. In [130], a hybrid model was proposed which consist of a data augmentation unit utilizing the Generative Adversarial Network (GAN) and a deep learning fully connected layer model to classify the original and augmented EEG signals. The hybrid model obtained an accuracy of 98.4% accuracy which was slightly higher than the existing literature. Apart from hybrid architectures used in EEG data, there are other interesting hybrid models being incorporated in other domains. For instance, in [131], a unique hybrid architecture was proposed to detect false reading attacks using general multi-data-source deep hybrid learning-based model. Based on various correlations of the data, a CNN and GRU based model was constructed to extract features from the net meter reading. The extracted features were then concatenated with other features such as temperature and irradiance which was then passed to another block of GRU layers. Finally, the output of the second block was

concatenated with more features from the dataset and passed to the fully connected layer for robust classification. Various types of features were provided to the model at different stage, which contributed towards the robustness of the hybrid deep-learning model. Additionally, in [132], the audio and visual network were both used to construct two different CNN models containing 1D and 3D convolutional layers, respectively, for a audio-visual emotion recognition model. The extracted features from the CNN models were then passed to a deep belief network for emotion recognition. Lastly, Jaouedi et al., proposed a hybrid model in [133] that focused on various feature extraction techniques to track the motion of a person. The extracted features are then separately passed to GRU network to classify the action of a person due to the data being sequential. They claimed to have achieved a good performance with their proposed hybrid model.

## 5.3 Problem Statement

The traditional EEG data collection process involves collecting the data in the time domain and applying various signal processing techniques to decompose them into frequency components such as alpha, beta, gamma, theta, delta, to analyze the signals in terms of their spectral characteristics. These techniques are used as the EEG signals are non-stationary and non-deterministic making the time domain analysis difficult. Some of the more popular signal processing techniques include Fourier Transform, Wavelet Decomposition, Gabor Transform etc. Among them, the Fourier Transform is considered as the most common approach to transform between frequency as well as time domain as it is time-shift invariant. The fourier $S(\omega)$ of a signal $s(t)$ in time domain and the inverse transformation of $S(\omega)$ can be calculated using eqns. (5.1) and (5.2) respectively.

$$S(\omega) = \int_{-\infty}^{+\infty} s(t) e^{-2\pi j \omega t} dt \tag{5.1}$$

$$s(t) = \frac{1}{2\pi} \int_{-\infty}^{+\infty} S(\omega) e^{2\pi j \omega t} d\omega \tag{5.2}$$

On the other hand, another very popular method of processing the signals is wavelet transform which was established as a comparatively better technique to represent the spectral characteristics of the EEG signals in [134]. The main concept of wavelet decomposition is to transform the signals into wavelets which are temporary oscillating function which can be both presented in frequency and time domain. The transformation can be both continuous ($CWT$) and discrete ($DWT$). In a CWT, the signals of the mother wavelets, $\phi(t)$ are translated and scaled to $\phi_{i,j}(t)$ as shown in eq. (5.3), where $i$ and $j$ are

the scale and translation parameters. On the other hand, for DWT, the parameters $x$ and $y$ are discretized (eq. (5.4)) where $i = 2^x$ and $j = y2^x$.

$$\phi_{i,j}(t) = \frac{1}{\sqrt{|i|}}\phi(\frac{t-j}{i}) \tag{5.3}$$

$$\phi_{x,y}(t) = 2^{-x/2}\omega(2^{-x}t - y) \tag{5.4}$$

However, this consumes a significant amount of time, which can be expensive for an ultra-edge IoT device, since the time complexity of the models will increase exponentially with greater number of inputs. There are various types of Fourier Transformations such as Fast Fourier Transformation (FFT) which computes the signals in a faster time. The complexity of a FFT can be represented as $O(nLog(n))$ which greatly reduces the computational time of the signals. However, with increasing number of data, the complexity will still increase in a decreasing rate, adding some complexity to the model. Additionally, the time complexity of a DWT can be represented as $O(n)$, which is slightly lower than FFT, however, will still increase with greater number of data. Hence, analysing the signals in polynomial time becomes difficult. It also becomes challenging for the ultra-edge IoT EEG headbands to make an inference in their limited capacity. Thus, it is important to keep the signal processing steps minimal. The AI-based techniques specially deep learning is proven to have the ability to extract various features from a raw data i.e EEG signals in their time domain with very minimal preprocessing in various scenarios. Therefore, in the next section, we explored various AI-based techniques to perform classification accurately, leaving out the signal processing steps.

## 5.4 Preliminaries

### 5.4.1 Traditional AI-based techniques

Various AI-based approaches are implemented in order to evaluate their performance in the time domain EEG signals. This includes, K-Nearest Neighbors (KNN), Random Forest, Artificial Neural Network(ANN) and Convolutional Neural Network(CNN). These methods have been claimed as effective models for evaluating EEG signals in the literature. The KNN is a lazy learning-based algorithm that determines the class label based on the majority class label of its $k_n$ neighbours. Additionally, the random forest is an ensemble classifier technique, that predicts the class labels based on the majority output of multiple decision trees. ANN is a deep learning technique that employs multiple fully connected layers which are able to make complex decisions of computing the class labels. The structure of the model consist of an input layer, multiple hidden

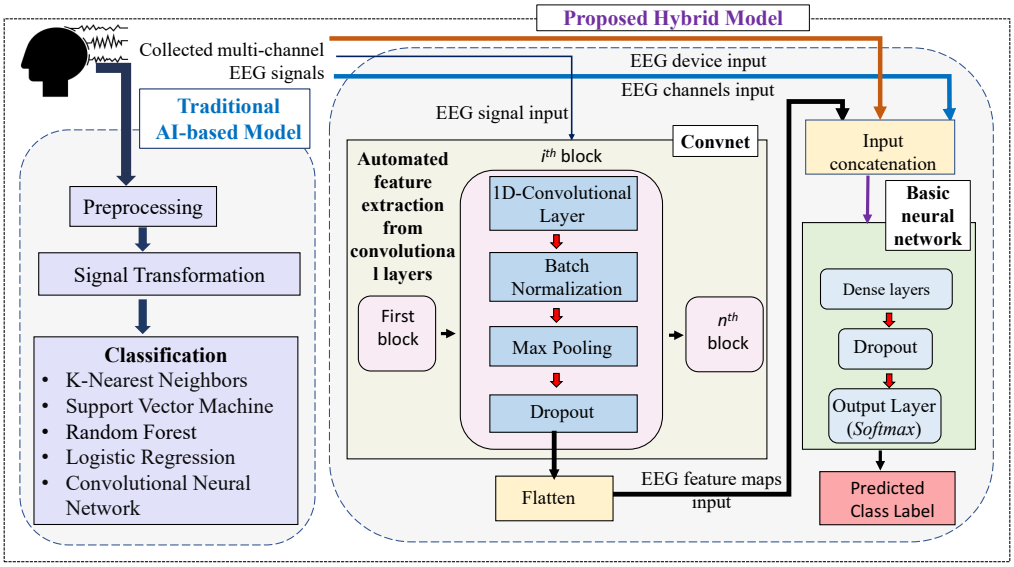



Fig. 5.1 The architecture of our proposed hybrid model vs. traditional AI-based algorithms.

layers and an output layer. Another 1D-convolutional model was also implemented as these models are able to find out trends or patterns in a dataset by extracting meaningful features. The convolutional layers are followed by a set of hidden layers for a robust computation. For both models, the output layer consists of neurons equal to the number of classes, $n_L$. Lastly, a combined CNN and Long Short-Term Memory (LSTM) model was also implemented, which had similar architecture as the CNN, and the features extracted from the CNN will be provided as an input to the LSTM model having $l$ LSTM layers and $u$ units in each layer. Since the LSTM is a recurrent neural network, it is proven to perform better for various time-series datasets. Therefore the combined CNN and LSTM model was also included in our systematic investigation.

The algorithms are able to perform well while analysing various types of EEG signals. However, they compute the class labels solely based on the EEG signals. The EEG of a person can vary based on various aspects such as the type of device used to collect the signal, the channels of the signal, the physical attributes of a person and so forth. These attributes are usually not included in the EEG analysis, since they do not hold any meaning on their own. However, if these features are combined with the EEG signals for the models, it is believed to achieve an even greater performance with the extra information. In this vein, we constructed a deep learning-based hybrid framework, which will be able to employ several features in addition to the EEG signals to perform classification. In the next section, we describe our proposed hybrid model along with its architecture.

### 5.4.2 A Deep Learning-based Hybrid Framework

We proposed a hybrid model by considering two types of input collected from the EEG sensors, the EEG signals and the channels of the signals. Since the convolutional layers are able to detect patterns in the EEG signals meticulously, the signals are first passed through the convolutional layers to extract crucial features and patterns of the signal. The EEG signals might vary from device to device, thus it is also important for the model to know the type of device being used and the specific channels of the signals while computing. In this vein, we considered those attributes of the signals as well in our computation, so that the model can be more generalizable. The EEG signals and the device and channel information were considered as two different inputs because, the device and channel information does not contain any patterns by itself. Thus the convolutional layers will not be able to extract any meaningful features/patterns from the input, making the model unnecessarily complex. In order to implement the model on the ultra-edge to satisfy our objective, we need to make the classification model precise and lightweight. Therefore, they were not added in the convolutional layers. On the other hand, the channel and device information, coupled with the extracted features of the EEG signals can be meaningful for the fully connected layers as the layers will have more information for their complex computation.

Our proposed hybrid model consist of $b$ convolutional blocks each having $c$ convolutional layers, followed by a batch normalization, a max pooling and a dropout layer. For each convolutional layer, the value of number of filters was increased in a geometric sequence such that if the first CNN layer has $\gamma$ filters, the second will have $2\gamma$, third will have $4\gamma$ and so forth. The number of kernels were increased at every layer so that the convolutional layer can extract more meaningful features from the feature maps of the signals. On the other hand, the number of filters were decreased in an arithmetic sequence - $\delta, \delta-1, \delta-2, \cdots, \delta-n$. As an activation function for the convolutional layer, $\alpha$ was used. Once the features were extracted from the $b^{th}$ convolutional block, the three dimensional feature maps were then flattened to make it two dimensions, so that it matches with the other inputs of the model. The channel and device inputs were then all concatenated with the extracted feature maps. This is a crucial step as the essence of hybridizaion lies in this step. The hybrid input is then passed to the Basic Neural Network (BNN) containing fully connected layers where robust computation is done on the data. The number of fully connected layers used was $f$ and at each layer, the number of neurons was varied in the geometric sequence - $\sigma, \sigma/2, \sigma/4, \cdots, \sigma/2^m$. $\theta$ was used as the optimizer of the hybrid model with a learning rate of $\Gamma$.

The algorithm of the hybrid architecture is represented in algorithm 3. The algorithm takes the EEG signals, $X_s$, channel information, $X_{ch}$ and device information, $X_d$ as input. The hyperparameters described above are initialized at first. Then the data is split

**Algorithm 3:** Algorithm of the proposed hybrid model

**Input** : $X_s$, $X_{ch}$, $X_d$, $y$
**Output**: $\partial$ (Performance of the Model)

**Function** `createHybridModel`($\alpha, n_L, n_{samples}$):
    Initialize $n$, $\gamma$,$\delta$
    $M_{CNN}$ = $n$ *Convnet* blocks with $n_{samples}$ (Fig. 5.1)
    $X_{CNN}$ = Extract features from $M_{CNN}$
    $X$ = concatenate($X_{CNN}$, $tensor(X_{ch})$, $tensor(X_d)$)
    Initialize $m$
    $M_{bnn}$ = $m$ BNN blocks with input $X$ (Fig. 5.1)
    $M_{hybrid}$ = add output layer with $n_L$ neurons
    **return** $M_{hybrid}$
Initialize $bs, e, \alpha, \theta, \Gamma, n_L$
$n_{samples}$ = Number of features in $X_s$
Initialize *kFold* **for** *k in kFold* **do**
    Split $X_s$, $X_{ch}$, $X_d$, $y$ into train and test set
    $M$ = $createHybridModel(\alpha, n_{label}, n_{samples})$
    Compile $M$ with $\theta, \Gamma$
    $M_{trained}$ = Train $M$ on train sets of $X_s$, $X_{ch}$, $X_d$ with $bs, e$
    $\partial_k$ = Evaluate $M_{trained}$ on test sets of $X_s$, $X_{ch}$, $X_d$
**end**
$\partial$ = $mean(\partial_k)$

into k folds for cross validation purposes and the hybrid model is created by calling the *createHybridModel* function. The $n_{samples}$ represents the number of features which is required as an input shape for the convnet blocks and $n_L$ are the number of class labels which is required to determine the output labels. *kFold* contains the set of folds created for the purpose of k-fold cross validation. In the *createHybridModel* function, the convnet blocks are then created, and the features from the $n_{th}$ blocks are extracted and concatenated with the tensors of $X_{ch}$ and $X_d$. The concatenated output is passed to the basic neural network blocks (BNN) and lastly an output layer with $n_{labels}$ to create the hybrid model, $M_{hybrid}$. Once the model is created, it is compiled with the stated $\theta$ with learning rate of $\Gamma$ and trained on the train sets of the input with batch size $bs$ for $e$ epochs. The trained model is then evaluated using the test set of the inputs.

## 5.5 Complexity Analysis of the proposed hybrid architecture

### 5.5.1 Training Phase

In order to implement the hybrid framework, at first we need to account for the preprocessing steps that are required for classification. In this particular use-case, we performed EEG signal downsampling and PCA, before inputting to the hybrid model. The complexity of the downsampling have a bounded complexity of $O(nLog(n)$ where n is the number of samples and PCA has a time complexity of $O(m^2n + m^3)$ where $m$ is the number of features and $n$ is the number of data points. Therefore, the total time required in the preprocessing step can be represented as follows:

$$T(P) = O(nLog(n) + O(m^2n + m^3) \tag{5.5}$$

The proposed hybrid model can be divided into two parts which are the Convnet block and the basic neural network block as seen in Fig. 5.1. The complexity of the overall model can therefore be represented as eq. (5.6) where O(CNN) represents the time consumed in the CNN architecture and O(BNN) is the time complexity of the basic neural network or fully connected layers.

$$T(hybrid) = T(CNN) + T(BNN) \tag{5.6}$$

The complexity of the Convnet block is represented in eq. (5.7) where $n$ is the number of filters, $k$ is the size of kernel, $l$ is the length of input, $d$ is the depth dimension and $f$ is the number of filters.

$$T(CNN) = O(\sum_{i=0}^{n} k_i \times l_i \times d_i \times f_i) \tag{5.7}$$

Additionally, the concatenation of the feature maps with the new input tensors require a time of $O(1)$. On the other hand, for the fully connected layer, the time complexity can be determined based on eq. (5.8) where $m$ is the number of hidden layers, $W$ is the weight of each node, $n_x$ is the number of training data and $\beta$ is the bias. Since the gradient descent runs for $n_g$ iterations, the complexity of the backpropagation technique is multiplied by $n_g$.

$$T(BNN) = O(n_g \times \sum_{i=1}^{m} n_{xi} \times W_i + \beta_i) \tag{5.8}$$

Thus the overall complexity of the hybrid model can be computed by adding the complexities of the individual blocks as shown in eq. (5.9)

$$T(hybrid) = O(\sum_{i=0}^{n} k_i \times l_i \times d_i \times f_i) + O(n_g \times \sum_{i=1}^{m} n_{xi} \times W_i + \beta_i) \tag{5.9}$$

### 5.5.2 Inference Phase

To make an inference, the trained model is expected to only explore the feed forward part of the neural network, avoiding the iterations of the gradient descent. Thus for a newly collected set of data $x_r$, the inference complexity, T(I) will be:

$$T(I) = O(\sum_{i=1}^{m} x_{ri} \times W_i + \beta_i) \tag{5.10}$$

## 5.6 Dataset Preparation

To evaluate the performance of the AI models, we consider the use-case of motor imagery EEG data analytics collected using a 64-channel Biosemi ActiveTwo$^{TM}$ system [135]. This dataset was chosen as it contained the raw EEG signals of the subjects, which was relevant for our stated problem. There were 10 participants in the experiment. The participants were asked to think about 4 words: 'Right', 'Left', 'Forward', and 'Back' and their corresponding EEG signals were recorded. These four labels were chosen for the experimentation as they can greatly influence how the smart devices are controlled. This is also an indicator of how the EEG signals changes with various words. The dataset contains 2560 EEG records of the brain signals from the four words of ten participants over 64 channels, having a sampling rate of 2048 samples per second, and a data frequency content of 0-409 Hz. Each signal contained 30 trials of an activity (eg. 'right'). Therefore the raw input of the EEG signals were available in bioSemi data format (.bdf) files which were read using the pyedflib library in python. The data contained approximately 2 - 6 hundred thousand time points for each of the signals. Since the number of time points varied for the signals, the signals were all downsampled to the minimum number of features available, which in this case was around 2.9 hundred thousand time points or columns using the signal module of scipy library. An example of the downsampling of an EEG signals has been depicted in Fig. 5.2, which initially consisted of 3.3 hundred thousand features. It can be seen that the signals are slightly shifted towards left in the downsampled version represented with blue, thus contracting the signal.

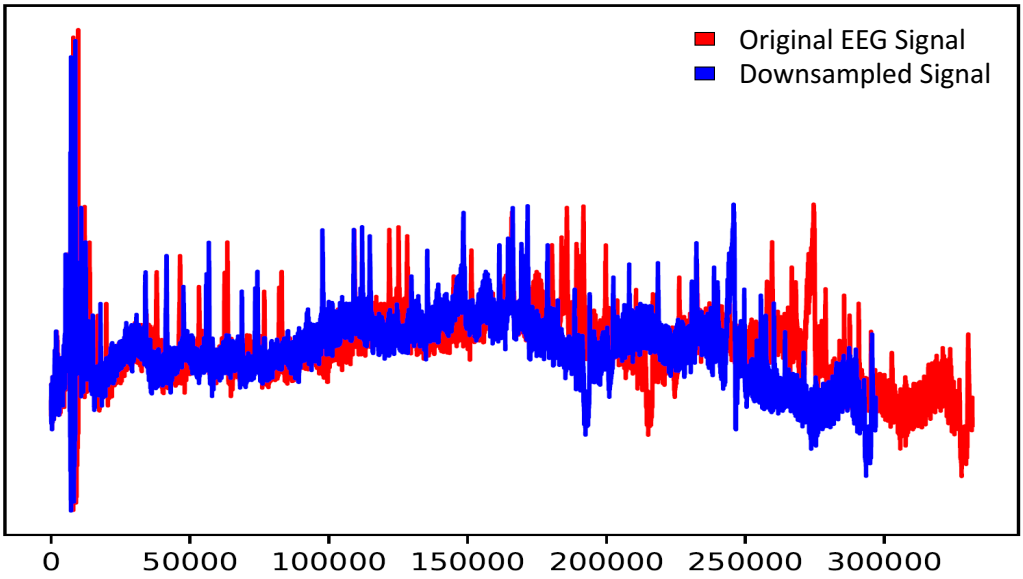



Fig. 5.2 Figure representing downsampling of an EEG signal

Since, there was a vast amount of time steps in this dataset, it required huge computational resources and time for performing the classification. In order to avoid this issue and restore the signals, the dimension of the signals was reduced using Principal Component Analysis (PCA). It is very commonly used in analysing EEG signals in various studies in the literature [136, 137]. The PCA converts the correlated signals to linearly uncorrelated principal components which are orthogonal variables. The first principal component always contains the maximum variance of the dataset. PCA aims to construct a set of features that is able to depict the maximum variance of the dataset using minimum number of principal components. For computing the principal components, at first, the covariance matrix, $C$ of the dataset, $X$ is calculated using $C = XX^T$ which is then decomposed using Eigenvalue decomposition. Then the variance of data captured by the PCA can be represented as a percentage of the dataset as stated in eq. (5.11) where $\lambda$ is the eigenvalue of the principal component. Thus most of the EEG signals can be restored within a very few features. In our case, we reduced the dimension to 1000 principal components for the dataset before inputting to the AI models.

$$\nu_{1:p} = \frac{\sum_{i=0}^{p} \lambda_i}{\sum_{i=0}^{n} \lambda_i} 100\% \tag{5.11}$$

This dataset was employed in [135] to compute the EEG signals within a subject. At first the signals were transformed to Gabor coefficient, which were broken down for the 30 trials of an individual where 29 trials were considered in the training set and

the leftover one was used as test set. This was repeated for all the 10 participants and the accuracy of the participants were averaged to compute the overall accuracy of the model. They achieved an accuracy of around 96% when they considered an epoch of 1s and around 94% when the epoch was 312ms.

## 5.7 Performance Evaluation

In this section, performance of the AI models discussed in Sec. 5.4 on the motor imagery dataset are presented. The results were validated using k-fold cross validation where the value of $k$ was considered as 5.

Since the Random Forest algorithm and the deep learning algorithms such as CNN has several hyperparameters, a hyperparameter tuning was performed in order to attain a robust performance of the models. As a part of tuning the hyperparameters of random forest, the depth of the tree was altered from 20 to 80 with an arithmetic progression of 20, along with the number of trees, which were altered between 100 and 500 with the $n^{th}$ number being $(n-1)^{th} + 100$. To tune the hyperparameters of our proposed hybrid model, at first the optimizer and the activation function values were altered to find a suitable combination. The activation function used in the tuning were, exponential, ReLU, Sigmoid, Softplus, Softsign, Tanh, SELU, and ELU. Additionally, the optimizers were altered between SGD, RMSprop, Adam, Adadelta, Adagrad, Adamax and Nadam. The optimizer and activation function that gave the highest accuracy was considered for further experimentation. Once the optimizer and activation function combination was found, the learning rate of the optimizer was altered for that combination to make the model more precise. The learning rate was altered from 1 to $10^{-4}$ in a geometric sequence where $n^{th}$ member of the sequence is $(n-1)^{th}/10$. The performance of a deep learning model is greatly dependent upon the learning rate as it helps to find the local minima in gradient descent. In order to perform a unbiased comparison of the deep learning models with our proposed hybrid approach, the architecture of the CNN used in the hybrid model was kept the same as the individual CNN as well as the combined CNN & LSTM model.

Fig. 5.3 presents the hyperparameter tuning results of the random forest model. It can be seen that with increasing number of trees, the accuracy of the models increased, however, it altered slightly with various depth of the tree values. For a tree depth of 20 and number of trees as 500, the model achieved the highest accuracy of approximately 95.8%. On the other hand, table 5.1 represents the results of activation function and optimizer tuning of the hybrid model. The optimizer Adagrad and Nadam performed well for most of the activation functions and the activation function ReLU, SELU and ELU performed well for most of the optimizers. Since the model with activation

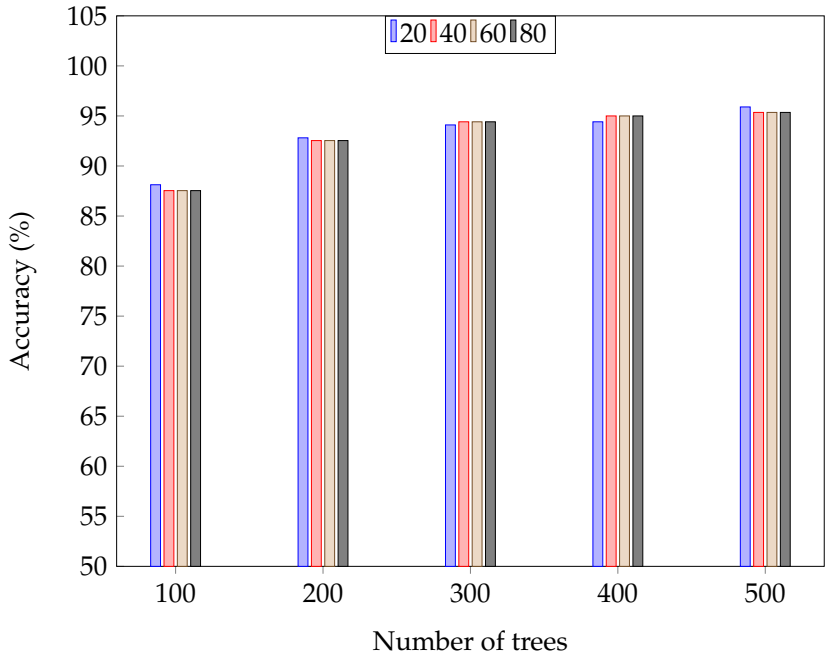



Fig. 5.3 Accuracy of random forest with various number of trees and maximum depth. The maximum depth of the tree was considered as 20,40,60,80.

function of 'ReLU' and an optimizer of 'Adamax' performed the best among all the other models, this combination was considered for our final hybrid model. Additionally, with the selected parameters, the learning rate was altered as shown in Fig. 5.4. The rates 0.1, $10^{-2}$, & $10^{-3}$ performed better than the rest with $10^{-3}$ being the best. This means that the gradient descent algorithm was able to find the optimal set of weights with a learning rate of $10^{-3}$. Therefore for the deep learning models, $\alpha$ = ReLU, *theta* = Adagrad and $\Gamma = 10^{-3}$ was used. After finding the optimal hyperparameters, 5 fold cross validation was applied in the models to make the model more robust.

The cross validated results of the discussed algorithms are presented in Fig. 5.5. For KNN, the value of $k_n$ was considered as 5. In the combined CNN and LSTM model, one LSTM layer was added with $u = 200$. For our proposed hybrid approach, in the CNN architecture, $b$ was taken as 4, the $\gamma$ value was taken as 25 and the $\delta$ was considered as 4. In the basic neural network block of the hybrid model, $f$ was 2 and $\sigma$ was 64. As mentioned earlier, the same architecture without the input concatenation part was used in the CNN algorithm. Additionally, since in our dataset, the signals were all collected from the same device, the device value was not considered individually, however, the channels with the EEG signals. A *bs* of 4 and $e = 500$ was taken for all the experimentation. It can be seen than our proposed hybrid model, random forest and CNN has performed comparatively better among all the algorithms. The hybrid model performed better compared to the individual CNN using the additional information as the model can make a decision based on the channel and the device the data is coming

Table 5.1 Hyperparameter tuning of the proposed hybrid model with various activation functions and optimizers.

| **Optimizer** | **Accuracy (%)** | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | **Activation Function** | | | | | | | |
| | Exponential | ReLU | Sigmoid | Softplus | Softsign | Tanh | Selu | Elu |
| SGD | 24.48 | 94.80 | 44.41 | 89.07 | 89.20 | 89.07 | 92.71 | 93.36 |
| RMSprop | 24.48 | 90.63 | 84.90 | 80.21 | 90.37 | 88.16 | 92.06 | 92.71 |
| Adam | 24.48 | 91.54 | 88.68 | 90.63 | 86.85 | 91.54 | 89.98 | 93.23 |
| Adadelta | 24.48 | 90.37 | 36.98 | 76.96 | 86.85 | 87.37 | 90.5 | 91.28 |
| Adagrad | 24.48 | **95.97** | 78.26 | 88.42 | 90.89 | 89.72 | 93.36 | 94.93 |
| Adamax | 24.48 | 24.48 | 24.48 | 24.48 | 24.48 | 24.48 | 24.48 | 24.48 |
| Nadam | 24.48 | 92.84 | 93.23 | 91.02 | 89.72 | 92.32 | 92.06 | 88.94 |

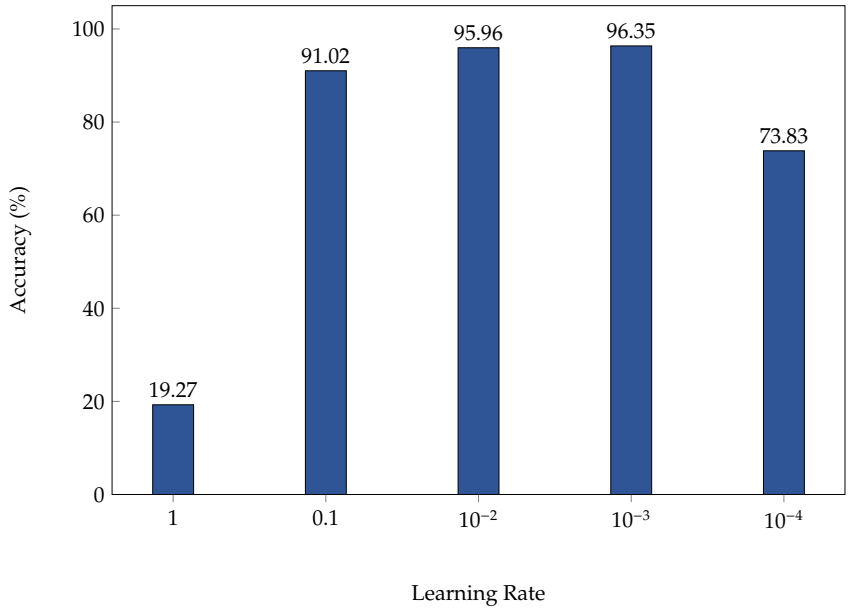



Fig. 5.4 Hyperparameter tuning of the proposed hybrid model with various learning rates.

from. Therefore, the model can find particular trends for an individual channel's signal and hence improve the performance of the model. It can also be noticed that when the LSTM layers are added to the CNN model, the performance of the model deteriorates. The confusion matrix of the three best performing approaches are presented in Figs. 5.6, 5.7 & 5.8. The hybrid model has been able to compute the class labels almost accurately for all four classes and random forest and CNN had a similar performance, in terms of detecting the class labels. Our hybrid model classification accuracy also surpassed the highest performance in the existing literature of 96% which was computed using Gabor transform and LDA. Thus our proposed novel hybrid model has been able to employ

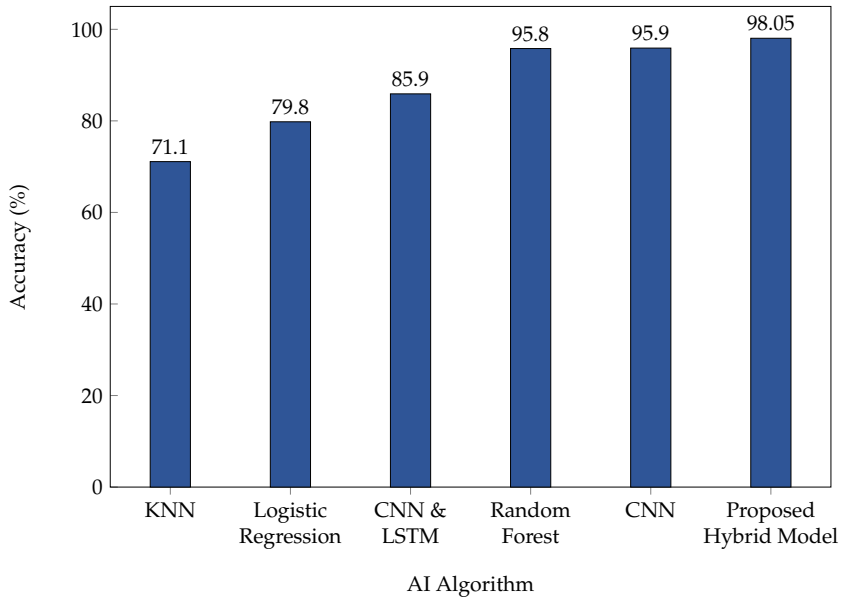



Fig. 5.5 Overall performance of the AI-based models with 5 Fold cross validation.

the EEG signals to give a remarkable performance for classifying the EEG signals in their time domain.

The time and memory consumption of the two best performing models were also analysed to check the compatibility of the model in an ultra-edge IoT node. These two properties can be considered as crucial indicator while determining the lightweight condition of the AI models, as it is important to deploy a model that will consume lower memory and time while operating so that it is not computationally expensive. The percentage of memory was computed based on how much memory is required to make an inference with respect to the total available memory of the device. The computation was performed in a base device which had an Intel Core i7, 3.00 GHz central processing unit (CPU), 16GB random access memory (RAM), powered by NVIDIA RTX 2060 graphics processing unit (GPU). The model was also tested in other IoT devices like Raspberry Pi 3 (Quad-core Cortex-A53 @ 1.4GHz and 1GB RAM), Raspberry Pi 4 (Quad-core Cortex-A72 @ 1.5GHz and and 2GB RAM) and NVIDIA Jetson Nano (Quad-core ARM Cortex-A57 @ 1.43GHz and 4GB RAM) to support our proof-of-concept. The analysis was performed including all the preprocessing steps such as downsampling and PCA, as well as inference of the EEG signals. Fig. 5.10 represents the time consumed by the random forest and the proposed hybrid model to make an inference. The random forest is taking around 1.5s with the lowest configuration device which is the Raspberry Pi 3 whereas for the hybrid model the inference time is about 4.5s for the same scenario. Additionally, in terms of memory consumption, as shown in Fig. 5.9, the random forest is taking slightly lower memory than the hybrid approach. Therefore, while the performance of hybrid model is the state of the art accuracy for

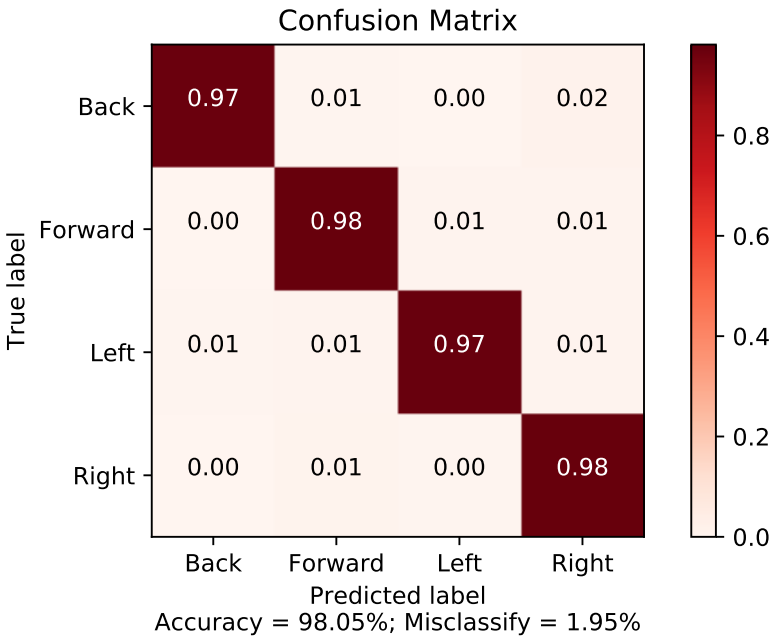



Fig. 5.6 Confusion matrix representing the performance of the proposed hybrid model


Fig. 5.7 Confusion matrix representing the performance of the random forest

this use-case, it is computationally more expensive than a random forest model. In order to implement a lightweight inference model in our ultra-edge, Random Forest is the more viable choice as it is still providing a decent performance in terms of accuracy while making an inference in a relatively lower time.

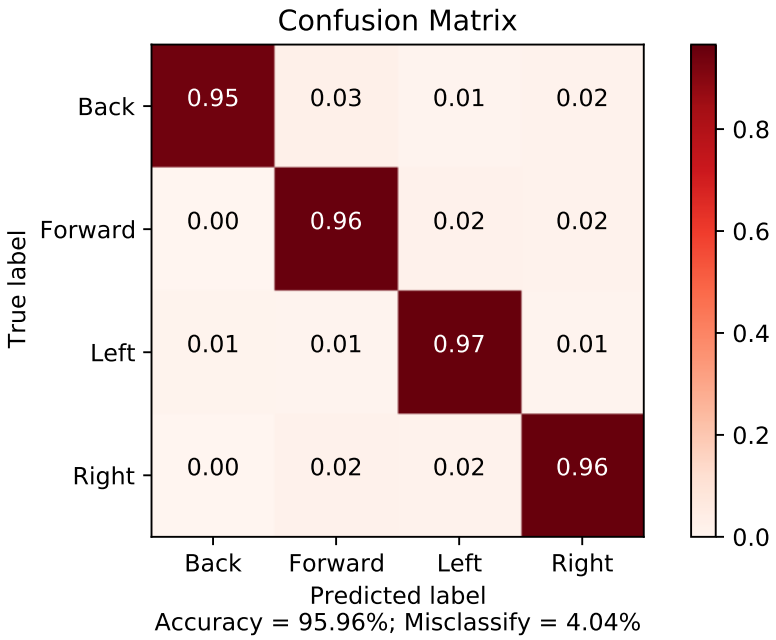



Fig. 5.8 Confusion matrix representing the performance of the individual CNN model

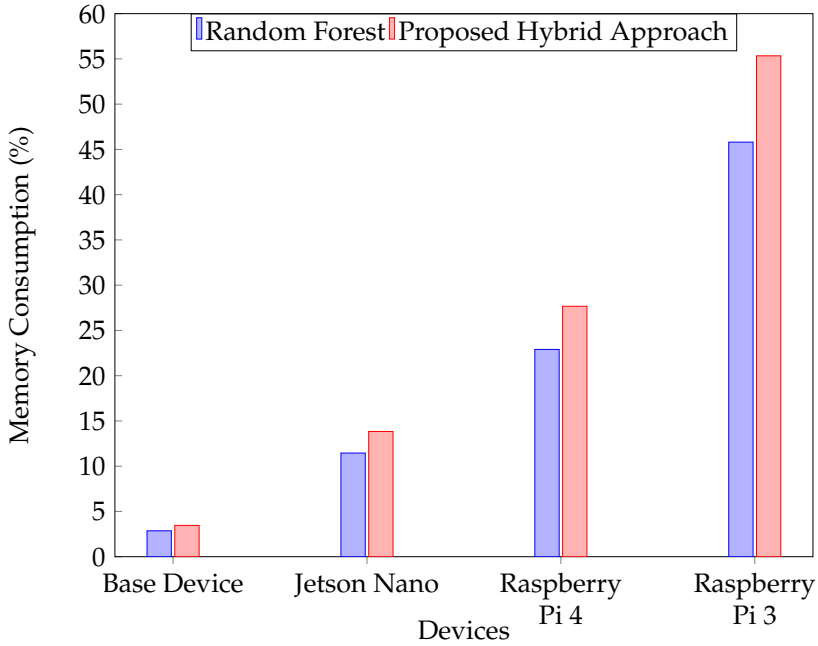



Fig. 5.9 Numeric analysis of memory consumption of two best performing algorithms on various ultra-edge devices.

## 5.8 Conclusion

A typical EEG analysis is usually done using a series of steps starting from signal collection to processing and transformation to classification. The signal transformation

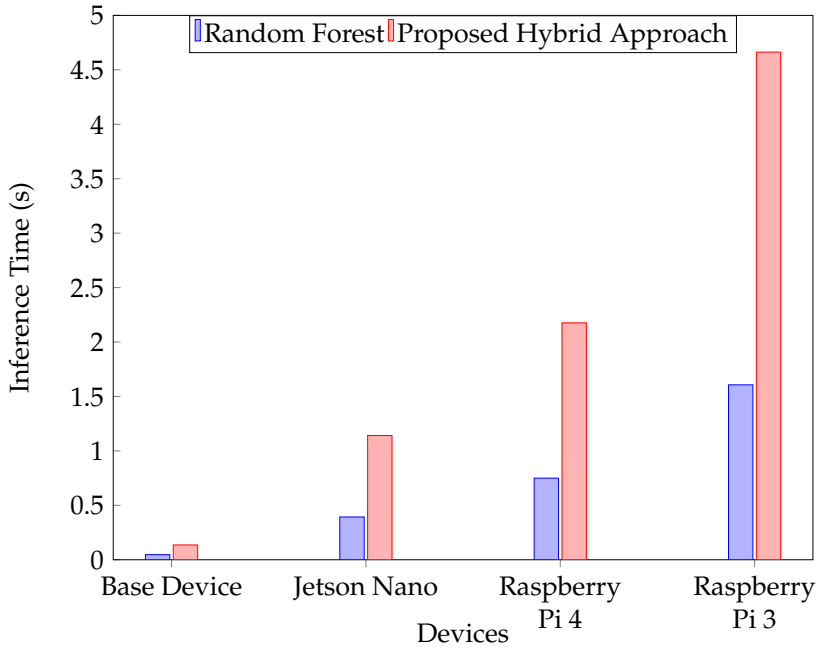



Fig. 5.10 Numeric analysis of inference time of two best performing algorithms on various ultra-edge devices.

steps are carried out to get some extra information about the EEG signals while reducing the dimension of the signals. However, this particular step takes an exponential time to execute i.e with increasing number of data, the complexity of the model increases exponentially. Thus we tried to employ the EEG signals in their time domain or raw EEG signals to carry out the EEG analysis, hence, eliminating this transformation step. In this chapter, we proposed a novel deep learning-based hybrid algorithm which was able to analyse the EEG signals directly collected from the sensors with a promising accuracy. We used the use-case of a motor imagery data that contained EEG records of 10 participants for the four classes collected over 64 channels. In our designed hybrid AI model, we considered additional information such as the channels of the EEG signals in addition to the EEG signal recording device information to make the AI model more generalizable and provide the model with some extra information in addition. However, to reduce the computational load of the model, we concatenated the additional channel and device information to the EEG signal features while passing them to the fully connected layer. This was done so that the convolutional layers can extract the patterns in the signals individually and the fully connected layers where complex computations are performed, also gets more information other than just the signals as input. After a computational complexity analysis was done, it was found that the proposed hybrid model is able to make an inference in polynomial time. However, the inferencing overhead is slightly higher than that of random forest, which gives the second best performance. Therefore, to make the model lightweight, random forest can

be used with a slight trade-off in accuracy rather than the hybrid model which gives state-of-the-art performance. Thus, the proposed logic in headbands based EEG analysis avoids sending unnecessary raw EEG data to the centralized cloud for computation while also reducing the computational resources of the analysis.

# Chapter 6

# Conclusion

In this dissertation, the concept of lightweight ultra-edge EEG analytics model fused with AI logic has been proposed. Based on the previous works about EEG analysis, there was a need of a proactive technique that will be able to analyze EEG data efficiently and within a short period of time, requiring minimal resources. Therefore, the idea of the proposed model is to conduct EEG analysis instantaneously on the ultra-edge by coupling commercially available EEG headbands with optimized machine learning models to conduct a precise inference on the device itself instead of sending the data to a remote server. We made an assumption that the ultra-edge EEG analysis devices will have the same configuration as the standard IoT devices. In a conventional EEG analysis methodology, there are a series of steps that is carried out to ensure an accurate analysis. However, the steps bring complexity along with its advantages which makes the AI model computationally heavy. In order to incorporate the model in the ultra-edge IoT nodes, we need to ensure a lightweight classification model is used. The traditional classification methods such as matrix and tensor based classifiers as well as adaptive classifiers have some disadvantages while it comes to computation of the EEG signals which makes it difficult to be integrated with our proposal. In this vein, we explored various AI-based classification techniques to reduce the complexity of the models. Several machine learning algorithms were systematically explored to evaluate the efficacy of the algorithms in the proposed concept. At first we tried to select a lightweight classification model for the EEG analysis. We used an use-case of confusion detection and after a systematic analysis, Random Forest was voted as the most efficient model in terms of performance and computational overheads. Next we tried to reduce the complexity of our proposed system further by directly classifying the raw EEG signals, thus reducing some preprocessing steps. We proposed a deep learning based hybrid model to enable classification of the raw EEG data directly and the model was able to achieve a notable performance.

This paradigm shift toward ultra-edge computing saves precious network bandwidth and communication time with the far away cloud servers. The AI-based algorithms are established as a efficacious technique to construct an architecture that will be able to perform EEG analysis on the ultra-edge in a rational time with considerably reasonable accuracy. Thus any brain abnormalities and activities can be detected in a simplistic manner more conveniently. Our proof-of-concept model aims at encouraging the sensor foundries to integrate AI logic with wearable sensors. If this type of AI models are directly combined with the sensors, the localized EEG analysis can be conducted on the ultra-edge devices more practically. Additionally it will be cost effective and scalable since this requires lower bandwidth and computational resources. In this thesis, we have only investigated the concepts of logic in headbands using two use-cases. There are several advantage of conducting the analysis on the ultra-edge. Since the data is not required to be sent to another node, the privacy and security of the collected data can also be restored in addition to a speedy analysis. Integrating the model with the sensors will make it easily within the reach of the people, thus people will be able to use the EEG devices beyond hospital settings to keep track of various neural activities. These AI-based models can be deployed on various platforms for a plethora of tasks, which may range from smart device control, words prediction for the speech impaired people, assessing concentration level while taking part in online courses, continuous and non-intrusive monitoring of post-surgical patients while rehabilitating at home, non-intrusive sleep monitoring, etc. This system can therefore be considered as a stepping stone to extend the horizons of smart health industry towards the ultra-edge.

### 6.0.1 Limitations and Future Work

In our proposed ultra-edge EEG analysis study, there were a few limitations. First of all, the study was performed using offline data, and hence the performance of the models for real time analysis have not been tested. Additionally, since the data was collected from different sources, it is difficult to ensure that there is no biasness in the EEG signal collection process. Furthermore, in this dissertation, we only focused on the EEG signal analysis part of the ultra-edge system, however, the security and privacy concerns were not scrutinized. Lastly, the data storage part has not also been studied in our thesis, as we focused on providing an inferred decision to the user in a small amount of time. The EEG signal data storage in the ultra-edge device system can be studied in the future, if there is a necessity for storing the raw EEG data for future use. Furthermore, the security and privacy aspects of the ultra-edge EEG analysis system can be studied in depth to ensure the data is securely used and stored in the IoT device. Also, the complexity of the models can be reduced even further in the future while keeping the performance steady to enable a more efficient ultra-edge analysis.

# References


[1] J. W. Kam, S. Griffin, A. Shen, S. Patel, H. Hinrichs, H.-J. Heinze, L. Y. Deouell, and R. T. Knight, "Systematic comparison between a wireless eeg system with dry electrodes and a wired eeg system with wet electrodes," *NeuroImage*, vol. 184, pp. 119–129, 2019.

[2] R. Leeb, D. Friedman, G. R. Müller-Putz, R. Scherer, M. Slater, and G. Pfurtscheller, "Self-paced (asynchronous) bci control of a wheelchair in virtual environments: a case study with a tetraplegic," *Computational intelligence and neuroscience*, vol. 2007, 2007.

[3] R. Colombo, F. Pisano, S. Micera, A. Mazzone, C. Delconte, M. C. Carrozza, P. Dario, and G. Minuco, "Robotic techniques for upper limb evaluation and rehabilitation of stroke patients," *IEEE transactions on neural systems and rehabilitation engineering*, vol. 13, no. 3, pp. 311–324, 2005.

[4] C. Guger, J. d. R. Millán, D. Mattia, J. Ushiba, S. R. Soekadar, V. Prabhakaran, N. Mrachacz-Kersting, K. Kamada, and B. Z. Allison, "Brain-computer interfaces for stroke rehabilitation: summary of the 2016 bci meeting in asilomar," *Brain-Computer Interfaces*, vol. 5, no. 2-3, pp. 41–57, 2018.

[5] T. O. Zander and C. Kothe, "Towards passive brain–computer interfaces: applying brain–computer interface technology to human–machine systems in general," *Journal of neural engineering*, vol. 8, no. 2, p. 025005, 2011.

[6] M. Clerc, L. Bougrain, and F. Lotte, *Brain-computer interfaces 2: technology and applications*. John Wiley & Sons, 2016.

[7] J. Van Erp, F. Lotte, and M. Tangermann, "Brain-computer interfaces: beyond medical applications," *Computer*, vol. 45, no. 4, pp. 26–34, 2012.

[8] Y. Shaikh, V. Parvati, and S. Biradar, "Survey of smart healthcare systems using internet of things (iot)," in *2018 International Conference on Communication, Computing and Internet of Things (IC3IoT)*. Chennai, India, India: IEEE, February 2018, pp. 508–513.

[9] R. K. Kodali, G. Swamy, and B. Lakshmi, "An implementation of iot for healthcare," in *2015 IEEE Recent Advances in Intelligent Computational Systems (RAICS)*, IEEE. Trivandrum, India: IEEE, December 2015, pp. 411–416.

[10] K. S. Awaisi, S. Hussain, M. Ahmed, A. A. Khan, and G. Ahmed, "Leveraging iot and fog computing in healthcare systems," *IEEE Internet of Things Magazine*, vol. 3, no. 2, pp. 52–56, June 2020.

[11] X. Wang and Y. Li, "Fog-assisted content-centric healthcare iot," *IEEE Internet of Things Magazine*, pp. 1–4, August 2020.

[12] A. Mohsen, M. Al-Mahdawi, M. M. Fouda, M. Oogane, Y. Ando, and Z. M. Fadlullah, "AI aided noise processing of spintronic based IoT sensor for magnetocardiography application," in *ICC 2020 - 2020 IEEE International Conference on Communications (ICC)*, 2020, doi: 10.1109/ICC40277.2020.9148617.

[13] M. Congedo, A. Barachant, and R. Bhatia, "Riemannian geometry for eeg-based brain-computer interfaces; a primer and a review," *Brain-Computer Interfaces*, vol. 4, no. 3, pp. 155–174, 2017.

[14] I. Winkler, S. Haufe, and M. Tangermann, "Automatic classification of artifactual ica-components for artifact removal in eeg signals," *Behavioral and Brain Functions*, vol. 7, no. 1, pp. 1–15, 2011.

[15] E. M. ter Braack, B. de Jonge, and M. J. Van Putten, "Reduction of tms induced artifacts in eeg using principal component analysis," *IEEE transactions on neural systems and rehabilitation engineering*, vol. 21, no. 3, pp. 376–382, 2013.

[16] N. Robinson, A. P. Vinod, K. K. Ang, K. P. Tee, and C. T. Guan, "Eeg-based classification of fast and slow hand movements using wavelet-csp algorithm," *IEEE Transactions on Biomedical Engineering*, vol. 60, no. 8, pp. 2123–2132, 2013.

[17] R. Kher and R. Gandhi, "Adaptive filtering based artifact removal from electroencephalogram (eeg) signals," in *2016 International Conference on Communication and Signal Processing (ICCSP)*. IEEE, 2016, pp. 0561–0564.

[18] M. R. Lakshmi, T. Prasad, and D. V. C. Prakash, "Survey on eeg signal processing methods," *International Journal of Advanced Research in Computer Science and Software Engineering*, vol. 4, no. 1, 2014.

[19] T. Zeman, "Bss-preprocessing steps for separation improvement," *CTU FEE. Dept of Circuit Theory*, 2000.

[20] F. Lotte and C. Guan, "Regularizing common spatial patterns to improve bci designs: unified theory and new algorithms," *IEEE Transactions on biomedical Engineering*, vol. 58, no. 2, pp. 355–362, 2010.

[21] C. Michel, D. Lehmann, B. Henggeler, and D. Brandeis, "Localization of the sources of eeg delta, theta, alpha and beta frequency bands using the fft dipole approximation," *Electroencephalography and clinical neurophysiology*, vol. 82, no. 1, pp. 38–44, 1992.

[22] H. Ocak, "Automatic detection of epileptic seizures in eeg using discrete wavelet transform and approximate entropy," *Expert Systems with Applications*, vol. 36, no. 2, pp. 2027–2036, 2009.

[23] L. Chen, E. Zhao, D. Wang, Z. Han, S. Zhang, and C. Xu, "Feature extraction of eeg signals from epilepsy patients based on gabor transform and emd decomposition," in *2010 Sixth International Conference on Natural Computation*, vol. 3. IEEE, 2010, pp. 1243–1247.

[24] W. Ting, Y. Guo-Zheng, Y. Bang-Hua, and S. Hong, "Eeg feature extraction based on wavelet packet decomposition for brain computer interface," *Measurement*, vol. 41, no. 6, pp. 618–625, 2008.

[25] H. Nai-Jen and R. Palaniappan, "Classification of mental tasks using fixed and adaptive autoregressive models of eeg signals," in *The 26th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, vol. 1. IEEE, 2004, pp. 507–510.

[26] A. Vuckovic and F. Sepulveda, "Delta band contribution in cue based single trial classification of real and imaginary wrist movements," *Medical & biological engineering & computing*, vol. 46, no. 6, pp. 529–539, 2008.

[27] F. Schettini, F. Aloise, P. Aricò, S. Salinari, D. Mattia, and F. Cincotti, "Self-calibration algorithm in an asynchronous p300-based brain–computer interface," *Journal of neural engineering*, vol. 11, no. 3, p. 035004, 2014.

[28] H. Woehrle, M. M. Krell, S. Straube, S. K. Kim, E. A. Kirchner, and F. Kirchner, "An adaptive spatial filter for user-independent single trial detection of event-related potentials," *IEEE Transactions on Biomedical Engineering*, vol. 62, no. 7, pp. 1696–1705, 2015.

[29] J. W. Yoon, S. J. Roberts, M. Dyson, and J. Q. Gan, "Adaptive classification for brain computer interface systems using sequential monte carlo sampling," *Neural Networks*, vol. 22, no. 9, pp. 1286–1294, 2009.

[30] C. Vidaurre, A. Schlogl, R. Cabeza, R. Scherer, and G. Pfurtscheller, "Study of on-line adaptive discriminant analysis for eeg-based brain computer interfaces," *IEEE Transactions on Biomedical Engineering*, vol. 54, no. 3, pp. 550–556, 2007.

[31] M. K. Hazrati and A. Erfanian, "An online eeg-based brain–computer interface for controlling hand grasp using an adaptive probabilistic neural network," *Medical engineering & physics*, vol. 32, no. 7, pp. 730–739, 2010.

[32] J. Li and L. Zhang, "Bilateral adaptation and neurofeedback for brain computer interface system," *Journal of neuroscience methods*, vol. 193, no. 2, pp. 373–379, 2010.

[33] J. Liu, S. Qu, W. Chen, J. Chu, and Y. Sun, "Online adaptive decoding of motor imagery based on reinforcement learning," in *2019 14th IEEE Conference on Industrial Electronics and Applications (ICIEA)*. IEEE, 2019, pp. 522–527.

[34] J. Blumberg, J. Rickert, S. Waldert, A. Schulze-Bonhage, A. Aertsen, and C. Mehring, "Adaptive classification for brain computer interfaces," in *2007 29th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*. IEEE, 2007, pp. 2536–2539.

[35] G. Liu, G. Huang, J. Meng, D. Zhang, and X. Zhu, "Improved gmm with parameter initialization for unsupervised adaptation of brain–computer interface," *International Journal for Numerical Methods in Biomedical Engineering*, vol. 26, no. 6, pp. 681–691, 2010.

[36] C. Vidaurre, M. Kawanabe, P. von Bünau, B. Blankertz, and K.-R. Müller, "Toward unsupervised adaptation of lda for brain–computer interfaces," *IEEE Transactions on Biomedical Engineering*, vol. 58, no. 3, pp. 587–597, 2010.

[37] A. Llera, V. Gómez, and H. J. Kappen, "Adaptive multiclass classification for brain computer interfaces," *Neural computation*, vol. 26, no. 6, pp. 1108–1127, 2014.

[38] M. Congedo, P. L. C. Rodrigues, and C. Jutten, "The riemannian minimum distance to means field classifier," in *8th Graz Brain-Computer Interface Conference 2019*, 2019.

[39] S. Kumar, F. Yger, and F. Lotte, "Towards adaptive classification using riemannian geometry approaches in brain-computer interfaces," in *2019 7th International Winter Conference on Brain-Computer Interface (BCI)*. IEEE, 2019, pp. 1–6.

[40] P.-A. Absil, R. Mahony, and R. Sepulchre, *Optimization algorithms on matrix manifolds*. Princeton University Press, 2009.

[41] A. Edelman, T. A. Arias, and S. T. Smith, "The geometry of algorithms with orthogonality constraints," *SIAM journal on Matrix Analysis and Applications*, vol. 20, no. 2, pp. 303–353, 1998.

[42] A. Barachant, S. Bonnet, M. Congedo, and C. Jutten, "Multiclass brain–computer interface classification by riemannian geometry," *IEEE Transactions on Biomedical Engineering*, vol. 59, no. 4, pp. 920–928, 2011.

[43] E. K. Kalunga, S. Chevallier, Q. Barthélemy, K. Djouani, E. Monacelli, and Y. Hamam, "Online ssvep-based bci using riemannian geometry," *Neurocomputing*, vol. 191, pp. 55–68, 2016.

[44] L. Mayaud, S. Cabanilles, A. Van Langhenhove, M. Congedo, A. Barachant, S. Pouplin, S. Filipe, L. Pétégnief, O. Rochecouste, E. Azabou *et al.*, "Brain-computer interface for the communication of acute patients: a feasibility study and a randomized controlled trial comparing performance with healthy participants and a traditional assistive device," *Brain-Computer Interfaces*, vol. 3, no. 4, pp. 197–215, 2016.

[45] A. Barachant, S. Bonnet, M. Congedo, and C. Jutten, "Classification of covariance matrices using a riemannian-based kernel for bci applications," *Neurocomputing*, vol. 112, pp. 172–178, 2013.

[46] S. Bhattacharyya, A. Khasnobish, S. Chatterjee, A. Konar, and D. Tibarewala, "Performance analysis of lda, qda and knn algorithms in left-right limb movement classification from eeg data," in *2010 international conference on systems in medicine and biology*. IEEE, 2010, pp. 126–131.

[47] R. M. Mehmood and H. J. Lee, "Emotion classification of eeg brain signal using svm and knn," in *2015 IEEE international conference on multimedia & expo workshops (ICMEW)*. IEEE, 2015, pp. 1–5.

[48] F. Bahari and A. Janghorbani, "Eeg-based emotion recognition using recurrence plot analysis and k nearest neighbor classifier," in *2013 20th Iranian Conference on Biomedical Engineering (ICBME)*. IEEE, 2013, pp. 228–233.

[49] A. Schlögl, F. Lee, H. Bischof, and G. Pfurtscheller, "Characterization of four-class motor imagery eeg data for the bci-competition 2005," *Journal of neural engineering*, vol. 2, no. 4, p. L14, 2005.

[50] B. Blankertz, G. Curio, and K.-R. Muller, "Classifying single trial eeg: Towards brain computer interfacing," *Advances in neural information processing systems*, vol. 1, pp. 157–164, 2002.

[51] J. F. Borisoff, S. G. Mason, A. Bashashati, and G. E. Birch, "Brain-computer interface design for asynchronous control applications: improvements to the lf-asd asynchronous brain switch," *IEEE Transactions on Biomedical Engineering*, vol. 51, no. 6, pp. 985–992, 2004.

[52] M. V. Yeo, X. Li, K. Shen, and E. P. Wilder-Smith, "Can SVM be used for automatic EEG detection of drowsiness during car driving?" *Safety Science*, vol. 47, no. 1, pp. 115–124, 2009, doi: 10.1016/j.ssci.2008.01.007.

[53] I. Guler and E. D. Ubeyli, "Multiclass support vector machines for eeg-signals classification," *IEEE transactions on information technology in biomedicine*, vol. 11, no. 2, pp. 117–126, 2007.

[54] Y. Liu, W. Zhou, Q. Yuan, and S. Chen, "Automatic seizure detection using wavelet transform and svm in long-term intracranial eeg," *IEEE transactions on neural systems and rehabilitation engineering*, vol. 20, no. 6, pp. 749–755, 2012.

[55] A. K. Jain, R. P. W. Duin, and J. Mao, "Statistical pattern recognition: A review," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 22, no. 1, pp. 4–37, 2000.

[56] X. Wang, G. Gong, N. Li, and S. Qiu, "Detection analysis of epileptic eeg using a novel random forest model combined with grid search optimization," *Frontiers in human neuroscience*, vol. 13, p. 52, 2019.

[57] C. Donos, M. Dümpelmann, and A. Schulze-Bonhage, "Early seizure detection algorithm based on intracranial eeg and random forest classification," *International journal of neural systems*, vol. 25, no. 05, p. 1550023, 2015.

[58] T. Zhang, W. Chen, and M. Li, "Ar based quadratic feature extraction in the vmd domain for the automated seizure detection of eeg using random forest classifier," *Biomedical Signal Processing and Control*, vol. 31, pp. 550–559, 2017.

[59] D. R. Edla, K. Mangalorekar, G. Dhavalikar, and S. Dodia, "Classification of eeg data for human mental state analysis using random forest classifier," *Procedia computer science*, vol. 132, pp. 1523–1532, 2018.

[60] Y. Roy, H. Banville, I. Albuquerque, A. Gramfort, T. H. Falk, and J. Faubert, "Deep learning-based electroencephalography analysis: A systematic review," *Journal of Neural Engineering*, vol. 16, no. 5, article no. 051001, 2019, doi: 10.1088/1741-2552/ab260c.

[61] R. San-Segundo, M. Gil-Martín, L. F. D'Haro-Enriquez, and J. M. Pardo, "Classification of epileptic eeg recordings using signal transforms and convolutional neural networks," *Computers in biology and medicine*, vol. 109, pp. 148–158, 2019.

[62] Y. Hao, H. M. Khoo, N. von Ellenrieder, N. Zazubovits, and J. Gotman, "Deepied: An epileptic discharge detector for eeg-fmri based on deep learning," *NeuroImage: Clinical*, vol. 17, pp. 962–975, 2018.

[63] X. Tian, Z. Deng, W. Ying, K.-S. Choi, D. Wu, B. Qin, J. Wang, H. Shen, and S. Wang, "Deep multi-view feature learning for eeg-based epileptic seizure detection," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 27, no. 10, pp. 1962–1972, 2019.

[64] U. R. Acharya, S. L. Oh, Y. Hagiwara, J. H. Tan, H. Adeli, and D. P. Subha, "Automated eeg-based screening of depression using deep convolutional neural network," *Computer methods and programs in biomedicine*, vol. 161, pp. 103–113, 2018.

[65] S. Roy, I. Kiral-Kornek, and S. Harrer, "Chrononet: a deep recurrent neural network for abnormal eeg identification," in *Conference on Artificial Intelligence in Medicine in Europe*. Springer, 2019, pp. 47–56.

[66] J. Chen, D. Jiang, and Y. Zhang, "A hierarchical bidirectional gru model with attention for eeg-based emotion classification," *IEEE Access*, vol. 7, pp. 118 530–118 540, 2019.

[67] E. J. Choi and D. K. Kim, "Arousal and valence classification model based on long short-term memory and deap data for mental healthcare management," *Healthcare informatics research*, vol. 24, no. 4, p. 309, 2018.

[68] T. Zhang, W. Zheng, Z. Cui, Y. Zong, and Y. Li, "Spatial–temporal recurrent neural network for emotion recognition," *IEEE transactions on cybernetics*, vol. 49, no. 3, pp. 839–847, 2018.

[69] D. Ahmedt-Aristizabal, C. Fookes, K. Nguyen, and S. Sridharan, "Deep classification of epileptic signals," in *2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*. IEEE, 2018, pp. 332–335.

[70] Y. Yuan, G. Xun, K. Jia, and A. Zhang, "A multi-view deep learning framework for eeg seizure detection," *IEEE journal of biomedical and health informatics*, vol. 23, no. 1, pp. 83–94, 2018.

[71] H. Daoud and M. A. Bayoumi, "Efficient epileptic seizure prediction based on deep learning," *IEEE transactions on biomedical circuits and systems*, vol. 13, no. 5, pp. 804–813, 2019.

[72] D. Ahmedt-Aristizabal, C. Fookes, S. Denman, K. Nguyen, S. Sridharan, and S. Dionisio, "Aberrant epileptic seizure identification: A computer vision perspective," *Seizure*, vol. 65, pp. 65–71, 2019.

[73] S. Roy, I. Kiral-Kornek, and S. Harrer, "Deep learning enabled automatic abnormal eeg identification," in *2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*. IEEE, 2018, pp. 2756–2759.

[74] M. Golmohammadi, A. H. Harati Nejad Torbati, S. Lopez de Diego, I. Obeid, and J. Picone, "Automatic analysis of eegs using big data and hybrid deep learning architectures," *Frontiers in Human Neuroscience*, vol. 13, p. 76, 2019. [Online]. Available: https://www.frontiersin.org/article/10.3389/fnhum.2019.00076

[75] K. G. Hartmann, R. T. Schirrmeister, and T. Ball, "EEG-GAN: Generative adversarial networks for electroencephalograhic (EEG) brain signals," *arXiv preprint arXiv:1806.01875*, 2018.

[76] Y. Luo and B. L. Lu, "EEG data augmentation for emotion recognition using a conditional wasserstein GAN," in *2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, 2018, pp. 2535–2538, doi: 10.1109/EMBC.2018.8512865.

[77] Y. Jiao, Y. Deng, Y. Luo, and B.-L. Lu, "Driver sleepiness detection from eeg and eog signals using gan and lstm networks," *Neurocomputing*, vol. 408, pp. 100–111, 2020.

[78] M.-P. Hosseini, H. Soltanian-Zadeh, K. Elisevich, and D. Pompili, "Cloud-based deep learning of big eeg data for epileptic seizure prediction," in *2016 IEEE global conference on signal and information processing (GlobalSIP)*. IEEE, 2016, pp. 1151–1155.

[79] E. Džaferović, S. Vrtagić, L. Bandić, J. Kevric, A. Subasi, and S. M. Qaisar, "Cloud-based mobile platform for eeg signal analysis," in *2016 5th international conference on electronic devices, systems and applications (ICEDSA)*. IEEE, 2016, pp. 1–4.

[80] H. Ke, D. Chen, T. Shah, X. Liu, X. Zhang, L. Zhang, and X. Li, "Cloud-aided online eeg classification system for brain healthcare: A case study of depression evaluation with a lightweight cnn," *Software: Practice and Experience*, vol. 50, no. 5, pp. 596–610, 2020.

[81] M. V. R. Blondet, A. Badarinath, C. Khanna, and Z. Jin, "A wearable real-time bci system based on mobile cloud computing," in *2013 6th International IEEE/EMBS Conference on Neural Engineering (NER)*. IEEE, 2013, pp. 739–742.

[82] Z. Zhang, T. Wen, W. Huang, M. Wang, and C. Li, "Automatic epileptic seizure detection in eegs using mf-dfa, svm based on cloud computing," *Journal of X-ray Science and Technology*, vol. 25, no. 2, pp. 261–272, 2017.

[83] M. Alhussein, G. Muhammad, M. S. Hossain, and S. U. Amin, "Cognitive iot-cloud integration for smart healthcare: case study for epileptic seizure detection and monitoring," *Mobile Networks and Applications*, vol. 23, no. 6, pp. 1624–1635, 2018.

[84] M.-P. Hosseini, D. Pompili, K. Elisevich, and H. Soltanian-Zadeh, "Random ensemble learning for eeg classification," *Artificial intelligence in medicine*, vol. 84, pp. 146–158, 2018.

[85] P. Jezek and L. Vareka, "Cloud infrastructure for storing and processing eeg and erp experimental data." in *ICT4AWE*, 2019, pp. 274–281.

[86] B. S. Prabakaran, A. G. Jiménez, G. M. Martínez, and M. Shafique, "Emap: A cloud-edge hybrid framework for eeg monitoring and cross-correlation based real-time anomaly prediction," in *2020 57th ACM/IEEE Design Automation Conference (DAC)*. IEEE, 2020, pp. 1–6.

[87] M.-P. Hosseini, T. X. Tran, D. Pompili, K. Elisevich, and H. Soltanian-Zadeh, "Multimodal data analysis of epileptic eeg and rs-fmri via deep learning and edge computing," *Artificial Intelligence in Medicine*, vol. 104, p. 101813, 2020.

[88] A. A. Abdellatif, A. Mohamed, C. F. Chiasserini, M. Tlili, and A. Erbad, "Edge computing for smart health: Context-aware approaches, opportunities, and challenges," *IEEE Network*, vol. 33, no. 3, pp. 196–203, 2019, doi: 10.1109/MNET.2019.1800083.

[89] E. Konstantinidis, N. Conci, G. Bamparopoulos, E. Sidiropoulos, F. De Natale, and P. Bamidis, "Introducing neuroberry, a platform for pervasive eeg signaling in the iot domain," in *Proceedings of the 5th EAI international conference on wireless mobile communication and healthcare*, ser. MOBIHEALTH'15. London, Great Britain: ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), December 2015, pp. 166–169.

[90] M. Joneidi, P. Ahmadi, M. Sadeghi, and N. Rahnavard, "Union of low-rank subspaces detector," *IET Signal Processing*, vol. 10, no. 1, pp. 55–62, 2016.

[91] S. Minaee and Y. Wang, "Fingerprint recognition using translation invariant scattering network," in *2015 IEEE Signal Processing in Medicine and Biology Symposium (SPMB)*. IEEE, 2015, pp. 1–6.

[92] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," *arXiv preprint arXiv:1406.2661*, 2014.

[93] K. Rasheed, J. Qadir, T. J. O'Brien, L. Kuhlmann, and A. Razi, "A generative model to synthesize eeg data for epileptic seizure prediction," *arXiv preprint arXiv:2012.00430*, 2020.

[94] J. V. Tu, "Advantages and disadvantages of using artificial neural networks versus logistic regression for predicting medical outcomes," *Journal of clinical epidemiology*, vol. 49, no. 11, pp. 1225–1231, 1996.

[95] G. Li, C. H. Lee, J. J. Jung, Y. C. Youn, and D. Camacho, "Deep learning for eeg data analytics: A survey," *Concurrency and Computation: Practice and Experience*, vol. 32, no. 18, p. e5199, 2020.

[96] R. T. Schirrmeister, J. T. Springenberg, L. D. J. Fiederer, M. Glasstetter, K. Eggensperger, M. Tangermann, F. Hutter, W. Burgard, and T. Ball, "Deep learning with convolutional neural networks for eeg decoding and visualization," *Human brain mapping*, vol. 38, no. 11, pp. 5391–5420, 2017.

[97] R. Shea, F. Wang, H. Wang, and J. Liu, "A deep investigation into network performance in virtual machine based cloud environments," in *IEEE INFOCOM 2014 - IEEE Conference on Computer Communications*, 2014, doi: 10.1109/INFOCOM.2014.6848061.

[98] Z. Xiao and Y. Xiao, "Security and privacy in cloud computing," *IEEE Communications Surveys Tutorials*, vol. 15, no. 2, pp. 843–859, 2013, doi: 10.1109/SURV.2012.060912.00182.

[99] H. Wang, Y. Li, X. Hu, Y. Yang, Z. Meng, and K.-m. Chang, "Using EEG to improve massive open online courses feedback interaction," in *Proceedings of the 1st Workshop on Massive Open Online Courses at the 16th Annual Conference on Artificial Intelligence in Education, Memphis, TN*, 2013.

[100] Z. Ni, A. C. Yuksel, X. Ni, M. I. Mandel, and L. Xie, "Confused or not confused? disentangling brain activity from EEG data using bidirectional LSTM recurrent neural networks," in *Proceedings of the 8th ACM International Conference on Bioinformatics, Computational Biology,and Health Informatics*, 2017, doi: 10.1145/3107411.3107513.

[101] A. Tahmassebi, A. H. Gandomi, and A. Meyer-Baese, "An evolutionary online framework for MOOC performance using EEG data," in *2018 IEEE Congress on Evolutionary Computation (CEC)*, 2018, doi: 10.1109/CEC.2018.8477862.

[102] Y. Zhou, T. Xu, S. Li, and S. Li, "Confusion state induction and EEG-based detection in learning," in *2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, 2018, doi: 10.1109/EMBC.2018.8512943.

[103] H. Wang, Z. Wu, and E. P. Xing, "Removing confounding factors associated weights in deep neural networks improves the prediction accuracy for healthcare applications," in *Pacific Symposium on Biocomputing*, 2019, doi: 10.1142/9789813279827_0006.

[104] A. Subasi and M. Ismail Gursoy, "EEG signal classification using PCA, ICA, LDA and support vector machines," *Expert Systems with Applications*, vol. 37, no. 12, pp. 8659–8666, 2010, doi: 10.1016/j.eswa.2010.06.065.

[105] A. Barachant, S. Bonnet, M. Congedo, and C. Jutten, "Riemannian geometry applied to BCI classification," in *Latent Variable Analysis and Signal Separation*, 2010, doi: 10.1007/978-3-642-15995-4_78.

[106] S. Mousavi, F. Afghah, and U. R. Acharya, "SleepEEGNet: Automated sleep stage scoring with sequence to sequence deep learning approach," *PLOS ONE*, vol. 14, no. 5, pp. 1–15, 2019, doi: 10.1371/journal.pone.0216456.

[107] M. Dai, D. Zheng, R. Na, S. Wang, and S. Zhang, "EEG classification of motor imagery using a novel deep learning framework," *Sensors*, vol. 19, no. 3, article no. 551, 2019, doi: 10.3390/s19030551.

[108] Q. Zhang and Y. Liu, "Improving brain computer interface performance by data augmentation with conditional deep convolutional generative adversarial networks," *arXiv preprint arXiv:1806.07108*, 2018.

[109] M. Alhussein, "Monitoring parkinson's disease in smart cities," *IEEE Access*, vol. 5, pp. 19 835–19 841, 2017, doi: 10.1109/ACCESS.2017.2748561.

[110] L. S. Vidyaratne and K. M. Iftekharuddin, "Real-time epileptic seizure detection using EEG," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 25, no. 11, pp. 2146–2156, 2017, doi: 10.1109/TNSRE.2017.2697920.

[111] M. A. Hassan, A. S. Malik, D. Fofi, N. Saad, B. Karasfi, Y. S. Ali, and F. Meriaudeau, "Heart rate estimation using facial video: A review," *Biomedical Signal Processing and Control*, vol. 38, pp. 346–360, 2017, doi: 10.1016/j.bspc.2017.07.004.

[112] H. H. Asada, P. Shaltis, A. Reisner, Sokwoo Rhee, and R. C. Hutchinson, "Mobile monitoring with wearable photoplethysmographic biosensors," *IEEE Engineering in Medicine and Biology Magazine*, vol. 22, no. 3, pp. 28–40, 2003, doi: 10.1109/MEMB.2003.1213624.

[113] G. Yang, L. Xie, M. Mäntysalo, X. Zhou, Z. Pang, L. D. Xu, S. Kao-Walter, Q. Chen, and L. Zheng, "A health-IoT platform based on the integration of intelligent packaging, unobtrusive bio-sensor, and intelligent medicine box," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 4, pp. 2180–2191, 2014, doi: 10.1109/TII.2014.2307795.

[114] K. Mankodiya, Y. A. Hassan, S. Vogt, H. Gehring, and U. Hofmann, "Wearable ECG module for long-term recordings using a smartphone processor," in *Proceedings of the 5th International Workshop on Ubiquitous Health and Wellness, Copenhagen, Denmark*, vol. 2629, 2010.

[115] F. Seoane, J. Ferreira, L. Alvarez, R. Buendia, D. Ayllón, C. Llerena, and R. Gil-Pita, "Sensorized garments and textrode-enabled measurement instrumentation for ambulatory assessment of the autonomic nervous system response in the atrec project," *Sensors*, vol. 13, no. 7, pp. 8997–9015, 2013, doi: 10.3390/s130708997.

[116] F. Yger, M. Berar, and F. Lotte, "Riemannian approaches in brain-computer interfaces: A review," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 25, no. 10, pp. 1753–1762, 2017, doi: 10.1109/TNSRE.2016.2627016.

[117] H. Wang, "Confused student EEG brainwave data," https://www.kaggle.com/wanghaohan/confused-eeg/version/6.

[118] T. Harmony, "The functional significance of delta oscillations in cognitive processing," *Frontiers in Integrative Neuroscience*, vol. 7, p. 83, 2013, doi: 10.3389/fnint.2013.00083.

[119] J. Mari-Acevedo, K. Yelvington, and W. O. Tatum, "Chapter 9 - normal eeg variants," in *Clinical Neurophysiology: Basis and Technical Aspects*, ser. Handbook of Clinical Neurology, K. H. Levin and P. Chauvel, Eds. Elsevier, 2019, vol. 160, pp. 143–160, doi: 10.1016/B978-0-444-64032-1.00009-6.

[120] B. Van Essen, C. Macaraeg, M. Gokhale, and R. Prenger, "Accelerating a random forest classifier: Multi-core, gp-gpu, or fpga?" in *2012 IEEE 20th International Symposium on Field-Programmable Custom Computing Machines*. Toronto, ON, Canada: IEEE, Apr.–May 2012, pp. 232–239.

[121] S. Sakib, T. Tazrin, M. M. Fouda, Z. M. Fadlullah, and M. Guizani, "DL-CRC: Deep learning-based chest radiograph classification for COVID-19 detection: A novel approach," *IEEE Access*, vol. 8, pp. 171 575–171 589, 2020, doi: 10.1109/ACCESS.2020.3025010.

[122] J. J. Bird, D. R. Faria, L. J. Manso, A. Ekárt, and C. D. Buckingham, "A deep evolutionary approach to bioinspired classifier optimisation for brain-machine interaction," *Complexity*, vol. 2019, 2019.

[123] P. H. Nguyen, T. H. Nguyen, T. M. Duong, and T. M. T. Duong, "Combination of wavelet and mlp neural network for emotion recognition system," *Delta*, vol. 4, p. 8, 2018.

[124] V. J. Lawhern, A. J. Solon, N. R. Waytowich, S. M. Gordon, C. P. Hung, and B. J. Lance, "Eegnet: a compact convolutional neural network for eeg-based brain–computer interfaces," *Journal of neural engineering*, vol. 15, no. 5, p. 056013, 2018.

[125] B. L. K. Jolly, P. Aggrawal, S. S. Nath, V. Gupta, M. S. Grover, and R. R. Shah, "Universal eeg encoder for learning diverse intelligent tasks," in *2019 IEEE Fifth International Conference on Multimedia Big Data (BigMM)*. IEEE, 2019, pp. 213–218.

[126] S. Jirayucharoensak, S. Pan-Ngum, and P. Israsena, "Eeg-based emotion recognition using deep learning network with principal component based covariate shift adaptation," *The Scientific World Journal*, vol. 2014, 2014.

[127] R. Zhang, Q. Zong, L. Dou, and X. Zhao, "A novel hybrid deep learning scheme for four-class motor imagery classification," *Journal of neural engineering*, vol. 16, no. 6, p. 066004, 2019.

[128] R. Zhang, Q. Zong, L. Dou, X. Zhao, Y. Tang, and Z. Li, "Hybrid deep neural network using transfer learning for eeg motor imagery decoding," *Biomedical Signal Processing and Control*, vol. 63, p. 102144, 2021.

[129] U. Budak, V. Bajaj, Y. Akbulut, O. Atila, and A. Sengur, "An effective hybrid model for eeg-based drowsiness detection," *IEEE Sensors Journal*, vol. 19, no. 17, pp. 7624–7631, 2019.

[130] S. Chang and H. Jun, "Hybrid deep-learning model to recognise emotional responses of users towards architectural design alternatives," *Journal of Asian Architecture and Building Engineering*, vol. 18, no. 5, pp. 381–391, 2019.

[131] M. M. Badr, M. I. Ibrahem, M. Mahmoud, M. M. Fouda, and W. Alasmary, "Detection of false-reading attacks in the ami net-metering system," *arXiv preprint arXiv:2012.01983*, 2020.

[132] S. Zhang, S. Zhang, T. Huang, W. Gao, and Q. Tian, "Learning affective features with a hybrid deep model for audio–visual emotion recognition," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 28, no. 10, pp. 3030–3043, 2017.

[133] N. Jaouedi, N. Boujnah, and M. S. Bouhlel, "A new hybrid deep learning model for human action recognition," *Journal of King Saud University-Computer and Information Sciences*, vol. 32, no. 4, pp. 447–453, 2020.

[134] M. Akin, "Comparison of wavelet transform and fft methods in the analysis of eeg signals," *Journal of medical systems*, vol. 26, no. 3, pp. 241–247, 2002.

[135] A. Jahangiri and F. Sepulveda, "The relative contribution of high-gamma linguistic processing stages of word production, and motor imagery of articulation in class separability of covert speech tasks in eeg data," *Journal of medical systems*, vol. 43, no. 2, pp. 1–9, 2019.

[136] F. Artoni, A. Delorme, and S. Makeig, "Applying dimension reduction to eeg data by principal component analysis reduces the quality of its subsequent independent component decomposition," *NeuroImage*, vol. 175, pp. 176–187, 2018.

[137] S. Lekshmi, V. Selvam, and M. P. Rajasekaran, "Eeg signal classification using principal component analysis and wavelet transform with neural network," in *2014 International Conference on Communication and Signal Processing*. IEEE, 2014, pp. 687–690.