Application of Machine Learning in Attentiveness detection from EEG signal

by

Sadia Sobhana Ridi
18301279
Jannatul Farzana Tandra
19101097
Mahmudul Hasan Emon
19101098
Md Ridwan Mahmud
19101104
Sumaiya Tabassum
19101113

A thesis submitted to the Department of Computer Science and Engineering
in partial fulfillment of the requirements for the degree of
B.Sc. in Computer Science and Engineering

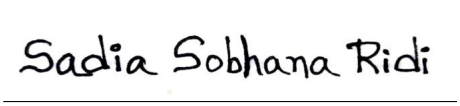
Department of Computer Science and Engineering
Brac University
September 2022

Declaration

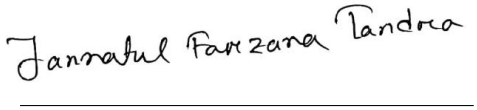
It is hereby declared that

1. The thesis submitted is my/our own original work while completing a degree at Brac University.
2. The thesis does not contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.
3. The thesis does not contain material which has been accepted, or submitted, for any other degree or diploma at a university or other institution.
4. We have acknowledged all main sources of help.

Student's Full Name & Signature:

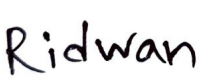


Sadia Sobhana Ridi
18301279

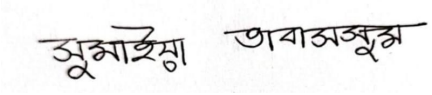


Jannatul Farzana Tandra
19101097

Mahmudul Hasan Emon
19101098



Md Ridwan Mahmud
19101104



Sumaiya Tabassum
19101113

Approval

The thesis titled "Application of Machine Learning in Attentiveness detection from EEG signal" submitted by

1. Sadia Sobhana Ridi (18301279)
2. Jannatul Farzana Tandra (19101097)
3. Mahmudul Hasan Emon (19101098)
4. Md Ridwan Mahmud (19101104)
5. Sumaiya Tabassum (19101113)

Of Summer, 2022 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of B.Sc. in Computer Science on January 21, 2022.

Examining Committee:

Supervisor:
(Member)

Faisal Bin Ashraf
Lecturer
Dept. of Computer Science and Engineering
Brac University

Program Coordinator:
(Member)

Md. Golam Rabiul Alam
Associate Professor
Dept. of Computer Science and Engineering
Brac University

Head of Department:
(Chair)

Dr. Sadia Hamid Kazi
Chairperson and Associate Professor
Department of Computer Science and Engineering
Brac University

Abstract

Brain-computer interface (BCI) spellers enable severely motor-impaired people to communicate through brain activity without the use of their muscles. Our brains precisely predict what we will think. If a human-readable character can be identified by its appearance, our issues may be resolved. Currently, human, machine, and brain communication based on machine learning is highly believable. In this study, we intend to employ the non-invasive brain stimulation technique, often known as EEG, for the treatment of these individuals. A Braincomputer interface system based on electroencephalography provides the optimal solution to this issue. It establishes a link between the brain and the computer system, allowing brain waves to control our actions. The objective is to determine if a person is paying attention by recognizing characters from a dataset of P300, which is an event-related potential (ERP) component, using a BCI design. If a character is identified as a person paying attention, the data is labelled as target class; otherwise, the data is displayed as non-target. Our study has resulted in a number of Machine Learning strategy techniques. In this study, we analyzed the performance of four different types of Machine Learning Algorithms, including Logistic Regression (LRR), Random Forest Classifier, AdaBoost classifier, and XGBoost Classifier, to determine the most accurate algorithm. Custom CNN achieved the highest accuracy among classifiers, at approximately 88.46%

Keywords: *Textual representation*, *Brain signal*, *BCI*, *EEG*, *LRR*, *CNN*, ERP

Dedication

It is our genuine gratefulness, and warmest regard that we dedicate this work to the people who are unable to speak or speak with involuntary pauses as well as repetition, to the people who's thoughts are beautifully arranged in their head, to the people who are equally talented but did not get the chance to participate, to the people who feel hesitant to speak considering that they might take a little bit extra time comparing others, to the people who lost their spontaneity, to the people who did not need any favor to achieve something big in their life and to prove every person who thought that the disable people should stop dreaming. There might be a significant change happening tomorrow.

Acknowledgement

In the first place, we would like to express our thankfulness to Allah the Almighty, who has provided us with guidance and His blessings, allowing us to complete our research on time and with the highest care.

Moreover, we'd want to express our gratitude to Mr. Faisal Bin Ashraf, our supervisor, who guided us through our project and helped us bring it to a successful conclusion.

Finally, but certainly not least, we would like to express our gratitude to our parents, who have been there for us throughout our entire lives to offer support and encouragement, and without whom we would never have even been able to get to the point where we are right now.

Table of Contents

List of Figures

List of Tables

Nomenclature

The next list describes several symbols & abbreviation that will be later used within the body of the document

$ACCW$ Word Accuracy

ASR Automated Speech Recognition

AUC Area Under the Curve

BW Bits Per Word

CNN Convolutional Neural Nwtwork

$ECoG$ Electrocorticography

EEG Electroencephalogram

ERP Event -Related Potential

GUI Graphical User Interface

LFP Local Field Potential

LRR Logistic Regression

$LSTM$ Long Short Term-Memory

MI Mutual Information

PCA Principal Component Analysis

PF Particle Algorithm

PSD Power Spectral Density

RNN Recurrent Neural Network

WAV Waveform Audio File Format

WPM Words Per Minute

Chapter 1

Introduction

The nervous system's command center is the human brain. It is composed of the spinal cord, the cerebrum, the brainstem, and the cerebellum. The most significant part of our brain, the cerebrum, is in charge of speech. The corpus callosum, a band of nerve fibers that connects the two hemispheres of the cerebrum, divides it into two halves. The left half of our brain is in charge of our speech. Speech-related brain activity occurs predominantly on the left side of the brain. These elements can be damaged or injured, resulting in speech difficulties. However, everyone wants to be ahead of the curve in this age of science and technology; no one wants to be left behind. According to Christopher and Dana Reeve Foundation, 1 out of every 50 persons is paralyzed, almost 5.4 million people worldwide [15].

People who are unable to interact due to any physical illness may be left behind as it is such a crucial component of society.To aid these individuals, we would like to present a method that predicts attentiveness based on character emphasis, with the hope that this effort will produce a new method of communication that does not require physical interaction. If a person's thoughts can be read by a computer, it will be a remarkable discovery that will prevent anyone from feeling alone even if they are unable to articulate their feelings in an usual human manner.

Brain computer interfaces, or BCIs, offer a significant possibility to improve the lives of persons who are unable to interact normally by translating brain signals into commands that are transmitted to an output device to execute desired outcome. In a BCI speller, P300 components of ERPs are used to figure out how the brain is processing a physical, auditory, or visual stimulus [21]. There will be a rectangular grid of alphanumeric characters ordered in row/column stimulus intensification by which the BCI system will detect the presence of P300 wave and choose targeted characters [1][21].It utilizes event-related potentials, which are components of EEG signals, to identify activity in the brain based on the mental state of the user [12][6][21].

Our study is based on a P300 dataset where a scalp-connected ERP signal has been used to determine if a patient is paying attention by focusing on a target character thinking or about something else [21].. We refer to it as "Target ERPs" when the patient is concentrated on the desired character. Otherwise, when a person perceives something other than a character designated as "Non Target ERPs" [21].

1.1 Problem Statement

Although the BCI system, which uses EEG signals from a patient's brain, has a lot of promise to improve our lives, there are several obstacles in this field, such as detecting a brain signal is a very complex technology, and the size of the speller has limitations. Every day, this system gets better, but it's not yet possible to decode people's thoughts and turn them into proper text due to the limitations and complexity of the brain. So, this system is working on one area at a time in the hopes of being able to translate people's thoughts someday.

The aim of our paper's research is to determine whether or not someone is paying attention by choosing characters from the alphanumeric grid. A person must predict the character in our study topic to demonstrate their attentiveness; else, something else will be predicted. When using a BCI speller to extract brain signals, it is not always possible to anticipate the intended results. He or she may instead choose irrelevant results. In our study, the phrase "target ERPs" refers to when a patient is paying attention by predicting character. In contrast, "non target ERPs" occur when a patient guesses something other than a character.

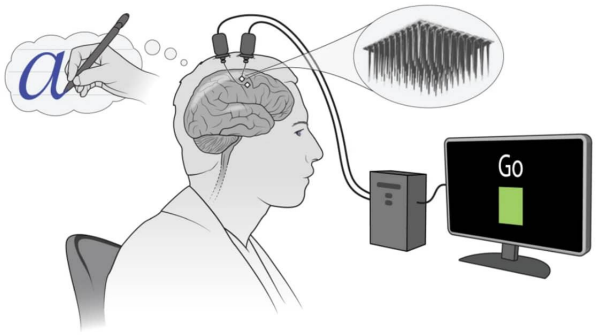


Figure 1.1: The patient is thinking of a character

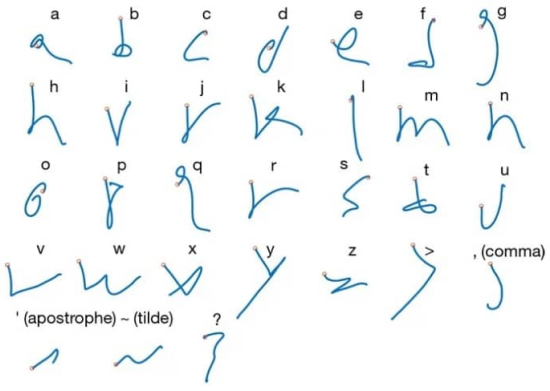


Figure 1.2: A list of characters

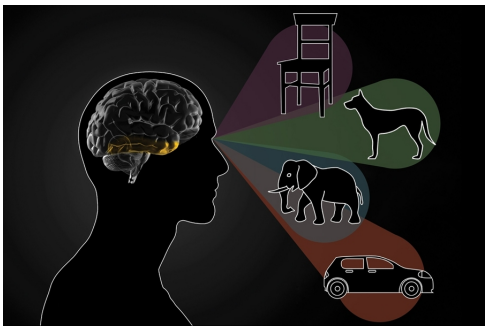


Figure 1.3: The patient is thinking of other things except for the characters

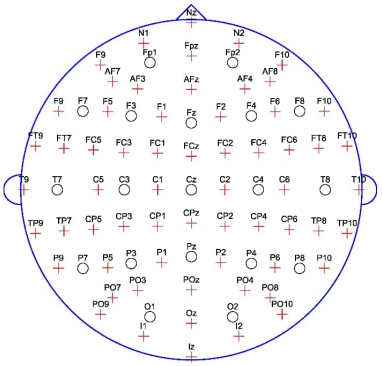


Figure 1.4: Target ERPs and Non-target ERPss

In figure 1.1, the patient is trying to identify a character from the list of characters in figure 1.2. Sometimes the patient can not recall the characters; instead, they identify any other else in figure 1.3. In figure 1.4, the electrode montage used in this study, as well as the target and non-target ERPs. Target ERPs are events in which the patient is attentive and identifies the target character, while non-target ERPs are all other occurrences [21].

1.2 Research objectives

In this work, we need to develop several machine learning algorithms for our case which are Logistic Regression, Random forest, XGBoost, AdaBoost and also a deep learning algorithm that can evaluate patient attentiveness using a CNN model. This would increase the effectiveness and appeal of the model. In our research paper, we will focus mostly on the following study objectives in order to construct the most effective model possible:

- By evaluating a patient's ability to predict the presence or absence of a target character, we can determine whether or not he or she is attentive.
- Again, we can find a model that best fits our training data.
- Moreover,to find a model that can handle overfitting, underfitting, and overlapping in any data.

1.3 Paper Orientation

This chapter primarily exposes the reader to the brain's function related speech generation and the suffering of speech-impaired individuals. It also includes a brief description of the issue statement and aims of the research. The remaining sections of the paper are structured as follows: The second chapter presents a summary of the background information for this study, introduces the concept of CNN, and justifies the usage of CNN in this research. The third chapter offers a literature review of previously published research on the application of machine learning to recognize desired characters/texts, etc. from patient's brain signals. The fourth chapter focuses on the dataset, its analysis, and its preprocessing for use in the research. This research's models are introduced in Chapter 5, and their outcomes are analyzed in Chapter 6. Chapter 7 discusses briefly the potential elements that contributed to the success of the custom CNN, as well as the limitations of the research and proposed enhancements. The eighth chapter summarizes the entire investigation and finishes the article. The paper concludes with a bibliography that contains a list of all the websites and periodicals cited throughout.

Chapter 2

Background

2.1 Convolutional Neural Network

CNN's are deep learning neural networks that can recognize and detect particular features in pictures and are frequently used to analyze visual images. It is a regularized version of multilayer perceptrons, and at at least one layer of this neural network, matrix multiplication is replaced by a mathematical operation known as convolution. A CNN architecture is built on the foundation of three layers, each with its unique parameter and activation function. The layers are the convolutional, pooling, fully connected, and a ReLU correction layer, mainly an activation function.

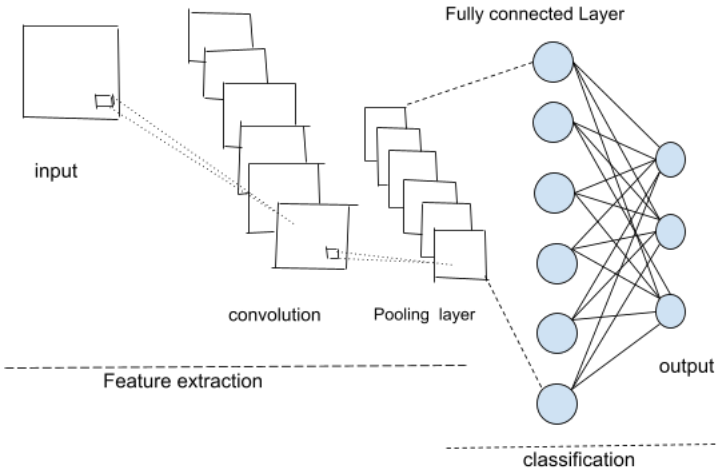


Figure 2.1: Basic Architecture of CNN

2.2 Building Blocks of CNN Architecture

2.2.1 The Convolutional Layer

The convolutional layer in convolutional neural networks is the first and most important layer. The goal of this layer is to locate a particular set of features in the

input images. Between the input image and a filter of a particular size FxF in this layer, the mathematical convolution process is performed. Sliding the filter over the image produces the dot product between the filter and the sections of the input image corresponding to the filter's size (F x F). As a result, the convolutional layer calculates the convolution of each image with each filter after receiving input from numerous images.The features we're looking for in the photographs are exactly what the filters match. The computation of the dot product provides us with a feature map, which identifies the locations of the features in the image. Rather than being pre-defined, the network learns features during the training phase within this layer .Filter kernels are the name given to the weights of the convolution layer. Usually, convolutional methods do not allow the kernel's center to overlap the input's outermost element, resulting in data loss in the feature map, which can be reduced by using a technique known as zero padding. The space between two sequential kernel places is called a stride, which is usually 1.

It can, however, be different on occasion. The following equations are used to achieve the output size in the convolution layer :

Output size =

$$\frac{N - F + 2P}{S} + 1 \tag{2.1}$$

where,
N = Size of the images
F = Size of the filter
S = Stride
P = The amount of padding

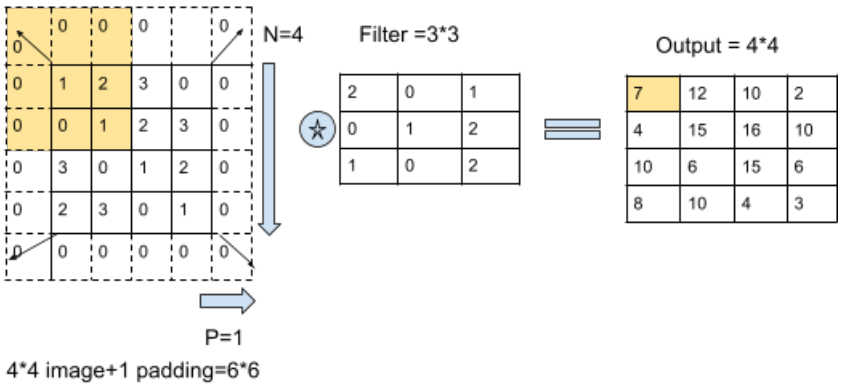


Figure 2.2: Convolutional operation with zero padding

2.2.2 Pooling Layer or Max Pool:

This layer lets feature maps be made smaller while still retaining their properties. Each map is pooled to reduce the amount of computing time required for each.

Various pooling processes exist, each with advantages and disadvantages, depending on the method employed. When CNNS is pooled, it is frequently done as a max pool. The maximum pool provides us with precisely the same number of feature maps we had before, but they are significantly smaller. When a size 2*2 max-pooling filter with a 2 stride is used, the feature map is reduced in size by half.

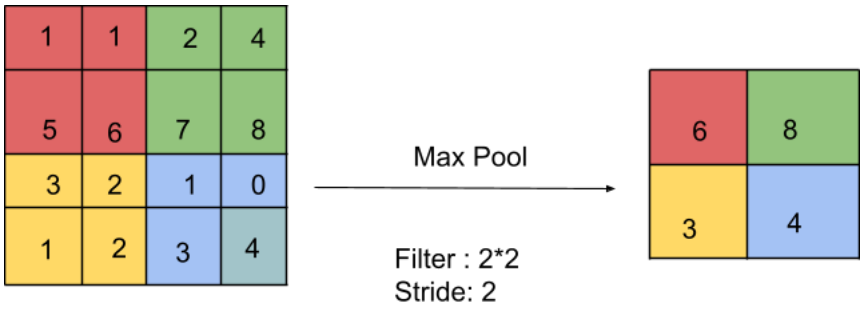


Figure 2.3: Max pooling layer operation

2.2.3 Fully Connected Layer:

CNN's last layer is called the "fully connected" layer since every input value is related to each output value. At this point, the process of classification begins. Consequently, the layer returns a vector of size H, which is the number of classes in the picture classification issue, as an input to the network. For instance, probabilities of classification or other outputs can be calculated by lowering the number of features in the convolution layer and then pooling them together.

2.2.4 ReLU Activation Layer:

The activation function is an essential parameter in the CNN model. The feature map from the convolutional layer is later passed on to a non-linear activation function. The ReLU, or rectified linear activation function, is common to activate neural networks, especially CNN.

If the input is positive, this activation layer will output it directly; otherwise, it will output zero, replacing all negative numbers received as inputs with zeros.

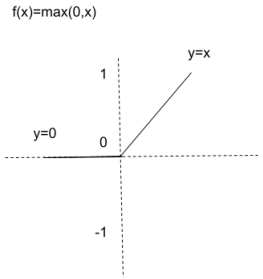


Figure 2.4: The activation function in CNN (ReLU)

2.3 Training a network

A network is trained to discover a combination of kernels in the convolution layer and weights in the fully connected layers that produces outputs with the smallest deviation from the labeled dataset.

2.3.1 Loss function:

A loss function is a function that determines the variance between the network output and the real output via forward propagation. This function determines the dissimilarity between the output of the algorithms and the target value.

Mainly, it considers the possibility or uncertainty of a prediction depending on how far it differs from its actual value after each optimization iteration. As a loss function, we used binary cross-entropy in our research. The actual class output is compared to each predicted probability in this particular loss function, which can be 0 or 1. It then calculates the score, penalizing the probabilities based on their difference from the predicted to the actual value.

2.3.2 Adam Optimizer:

To compile a model, we need an optimizer. Adam is a more robust and most popular optimizer in deep learning. It creates a new learning approach to maximize a range of neural networks by combining the benefits of two earlier stochastic gradient techniques, Adaptive Gradients and Root Mean Square Propagation (RMSprop). It squares the gradients like RMSprop and utilizes the gradient's moving average instead of the gradient like SGD with momentum to scale the learning rate. Adam uses adaptive moment estimation, which modifies the learning rates of each weight based on an estimate of the gradient's first and second moment. Adam additionally stores an average of prior gradients that decays exponentially. The following equations are used to achieve this

$$m^t = \frac{m^t}{(1 - \beta 1(t))} \tag{2.2}$$

$$v(t) = \frac{v(t)}{1 - \beta 2(t)} \tag{2.3}$$

where,
m(t) = first moment,
v(t) = second moment,
$\beta 1 = 0.9$,
$\beta 2 = 0.999$,
$t = batchnumber$

We employ Adam to update the weight. Here, adam optimizer uses the following formula:

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{v(t) + \epsilon}} \tag{2.4}$$

where,
θ = weight,
η = the learning rate,
ϵ = zero-avoidance parameter = 10^{-8}

2.3.3 Overfitting:

Overfitting is one of the problems that might arise in supervised machine learning. This prevents us from generalizing the models to accurately match both the training set's observed data and the testing set's unseen data [23].When the training accuracy is higher than the validation accuracy, an overfitting problem is evident. It has a detrimental effect on both test data and new data, which is a critical issue because test data are mainly used to evaluate a model's performance.

The common practice of tracking accuracy and loss on the training and validation sets allows for the detection of overfitting on training data [20]. Numerous factors, including the existence of noise, the complexity of the classifiers, and the small size of the training set, might cause overfitting [23]. The best way to reduce it is to increase the amount of training data, however there are other helpful methods that can also benefit in the overfitting issue, such as batch normalization, regularization with dropout, weight decay, early stopping, data augmentation, and architectural complexity reduction [20].

2.3.4 Dropout:

Dropout is a regularization procedure in which random activations are set to zero during training to eliminate them [20]. In other words, during the training phase, randomly selected neurons are not considered.

Dropouts are units that have dropped out of a neural network, both visible and hidden [13]. When a unit is "dropped out," the entire set of connections to and from it are temporarily removed from the network[13]. Dropout is employed in models to prevent overfitting, where it randomly removes some nodes according to the drop probability. For example, if a dropout is implemented with a drop probability of 0.5 and the hidden layers contain 100 neurons, 50 neurons will be discarded at random throughout each iteration.

2.3.5 BatchNormalization

A technique known as batch normalization can be used to normalize activations in deep neural networks' intermediary layers. Due to its capacity to increase accuracy and cut training time, it has become a popular deep learning technique [19]. To standardize the input or output of the sequential model, a layer is introduced. The model layer allows for its application at a number of places in between. It frequently comes just after the convolution and pooling layers. Since batch normalization introduces noise to each layer's inputs, it has a regularizing impact. As a result, overfitting is prevented since the model no longer generates deterministic values based only on a specific training sample.

2.3.6 Epoch:

A neural network is typically trained across several epochs. The number of epochs is a hyper-parameter that controls how many times the learning algorithm runs over the whole training dataset. A cycle across the entire training dataset is referred to as an epoch. The length of the epoch depends on the complexity and dataset size; it could be represented as seconds or hours. The more epochs are used, the more modified parameters, and the model perform better. However, overfitting may occur if there are too many epochs. When a model is overfitted, it performs well in training data but poorly in testing data. In our BCI dataset, we employ 100 epochs with an input size of (16,76).

2.4 Why we use CNN for this research?

While doing our research, to recognize characters, we use the Convolutional neural network (CNN), which uses convolutional layers and top pooling layers to extract higher-level features. CNN distinguishes the characters by analyzing their shapes and contrasting the qualities that differentiate them. Multiple convolutional layers are active in all layers of a CNN and scan the entire feature matrix while performing dimensionality reduction. As a result, CNN is an excellent model for classification and processing. However, the number of parameters in a neural network increases rapidly as the number of layers increases. This can make model training time-consuming. In this case, CNN effectively reduces the time required to tune these parameters. Moreover, even if only a tiny amount of training data is provided, a CNN can yield excellent recognition accuracy. These features of CNN are beneficial for recognizing characters. Our goal of this research is to know if the person writing the character or not, CNN can be more effective than other models. For these reasons, CNN was selected for this research.

Chapter 3

Literature review

The authors of this article [21] evaluated the performance of their Logistic Regression (LRR) models in forecasting the class variable using a collection of independent test datasets collected from the same participants. This implies that the Logistic Regression classifier outperforms all other classifiers regarding the area under the curve(AUC), demonstrating that LRR models accurately predict class variables [21]. On the High Dimensional and Amyotrophic Lateral Sclerosis datasets, the outcomes of sinusoidal signal modeling with principal component analysis(PCA), Waveform Audio File Format(WAV), and power spectral density(PSD) are compared. LRR is compared to several feature extraction algorithms in this graphv [21] [11] [8] [5] [3] [2]. This shows that the AUC of LRR models is consistently the highest. Here [21], a non-linear least square technique is used to determine the unknown parameters of a signal model, which has two advantages: signal spectrum elements are included in the collection of model parameters, and signal modeling dimensionality reduction is justified because the larger the sample size, the more precisely the classifier is trained. The amount of predicted dominant and discriminatory frequency components for each participant is determined by the quality of the classification process and the efficiency of the estimation method. In this paper [21], The Time Frequency representation was generated to check for missing frequencies and has the benefit of simultaneously modifying and distributing the chronological and frequency domain structure of subject-specific EEG recordings. For each class of event-related potential trials independently, this paper [21] discovered a sliding hanging taper with an adjustable time window of six cycles for each frequency in 20-millisecond steps for frequencies from 0.1 to 12 Hz. It also takes the absolute difference between target and non-target after grand averaging in the time-frequency domain, but the disadvantage is that it does not account for efficacy [21].
Furthermore, according to Farquhar et al [21] [10], the most discriminative frequency range is between 0.1 and 12 Hz, but other research extracts spectral characteristics using alternative frequency bands. It was recently proposed that the optimal frequency range is between 0.1 and 21.33 Hz, as stated in [17]. In this study, they found that spectral bands with a frequency of 6.4 Hz have the most substantial discriminative frequency for excellent decoding and can be used to develop a BCI speller algorithm [21].
In this signal modeling strategy, the inherent features of ERPs result incorrect mental state decoding models. These findings show that signal modeling-based approaches are more effective and accurate against variability than traditional mod-

els such as PCA, Wavelets, and PSDs. In most BCIs, the failure of a previously functioning BCI system is caused by irrelevant properties, high dimensionality, and enormous unpredictability, hence these findings are essential [21] [8]. The signal modeling method, in conjunction with logistic regression with the L2-ridge penalty, enabled us to extract underlying spectrum features of ERPs and construct unexpectedly specifically designed for predicting users' psychological intention in a BCI speller system in this study [21]. In addition, shorter waves, such as delta or theta-band activity, might broaden the uses of BCI, as was previously mentioned. This research indicates that BCI systems can create surprisingly accurate classifiers using only a few channels.

In this study [21], the authors used a BCI speller algorithm to measure an individual's brain activity and then turn it into communication directives by removing the connections between the central and peripheral nervous systems in this work. Event-related potentials, which are elements of EEG data, are utilized by BCI spellers to infer activity in the brain variations. Event-related potentials are time-locked brain reactions to an external sensory input event or an internal event related to the performance of motor activity. In this article [21], the authors examined seven patients who did not have a neurological condition to evaluate their mental status. They have used the row-column paradigm, which was made by Farwell and Danchin [21] [9], to improve the presentation of visual stimuli by using a rectangular grid of letters and numbers. Based on that row-column stimulation, they used a BCI system that infers a P300 waveform and picks sequential characters for spelling and communication for the user's mental purpose. According to Bostanov [21] [4], to get the accurate event-related potential features of P3, waves can be deconstructed using t-statistics. To evaluate the user's attentiveness [21], it was mentioned that target characters on the visual stimulus matrix were green for two seconds before flashing began, and participants were directed to count the target character's flashes where they had to visit five target letters. It was mentioned that Mowla et al they; employed a wavelet wavelet algorithm to identify stable Spatio-temporal properties through evaluating the time delays and variation in event-related potential waves in order to improve the performance of the BCI system [21][18].

The following research paper is [22], where the authors propose creating a BCI system to directly translate the text from brain signals to help disable patients speak again.However, implementing this in practical life can cause obstacles because of the speed and accuracy of the words. To address these flaws, the authors describe a solution technique in which a language model, an LSTM (Long short-term memory), and a Particle Algorithm (pf) are applied to patient-collected data [22]. In this paper [22] the authors used the LSTM to find all the possible outcomes of a phoneme at each time uttered by a subject and added it to the particle filtering algorithm to produce the efficiency text relating to the debugging word by including the prior knowledge of the English language model. The authors [22] applied similar techniques to the data of six patients by capturing signals and dividing each recording into time windows ranging from -166.67 to 100 MS relative to a single set of speaking stimuli. In Table 3.1 , this system's [22] input feature was concatenated with a matrix that contained a time domain signal, a highest z-scored frequency band for creating vowels, and an enclosing band that contained consonant information.

Subjects	Frequency Bands
1	150-200, 200-250
2	150-200, 1000-1150
3	70-150
4	200-250, 650-1150
5	200-250, 700-1150
6	150-400, 600-750

Table 3.1: The requisite frequency bands [22]

Moreover, in Table 3.1 , the authors achieved great results in Mutual Information (MI) 47.53 and Words per Minute (wpm) 26.67 compared to the other results which were cited by the author. On the other hand, the word accuracy (ACCW) and bits per word (BW) was respectively 32.16, 1.78 which was quite poor considering the other results.

In this paper [29] the authors approached to uses a customised GUI to convert thoughts to writing (GUI). This GUI displays numbers, alphabets, and special characters on a virtual keyboard based on the patient's mood. The sequence of character sets on the keyboard helps users choose from restricted characters, speeding up typing. The variable character set arrangement promotes the productivity of differently abled patients. The 14-channel Emotiv Epoc headset has been used to capture brain signals. EmoWrite uses proven classification methods to reduce training time.
Moreover, the proposed "EmoWrite" system uses BCI to help paralyzed individuals with speech impairments express their thoughts. EmoWrite has a customized keyboard. A customized keyboard with many numbers, letters, and special characters saves time and screen space. This system's sentiment-based thought-to-text translation and suggestions are not employed in other brain signal-to-text conversion systems. As keyboard design affects typing speed in this system, a circular design is employed to minimize traversing delay. The proposed solution uses 'machine-learning' to improve users' writing.

In this work [11], they studied amyotrophic lateral sclerosis (ALS) patients' attentiveness. This study examined whether ALS patients' attention and memory affected their control of a visual P300-based BCI. The P300 is a positive shift that shows up 300 ms after a stimulus is shown. The ability to pay attention to one aspect of the environment while ignoring or neglecting others is known as selective attention. Participants are shown a visual display and asked if a certain target is present.This task is a common technique to explore the effects of selective attention on visual search.In RSVP task, finding one or more objectives hidden among the stream of inputs is the goal. Participants were asked to report on two targets: T1 and T2. The experiment protocol involved two different sessions. A single letter flash at the beginning of each trial focused the target.EEG data from the first three runs (15 trials) was kept for BCI calibration.Due to the participants' motor impairments, they were requested to respond in binary (yes or no) to the operator via the remaining communication channel.
Three calibration runs were SWLDA to obtain classifier weights.The P300 peak amplitude for each test was calculated by averaging target and non-target epochs.

P300 potential in Cz was determined by comparing target and non-target waveforms in 250-700 ms. Sevenfold cross-validation tested each participant's classifier's binary accuracy. T1 and T2 were the two test subjects. At each repetition, they put the test data through a SWLDA. The average online accuracy for the BCI task was 97.5%[11]. The average offline binary accuracy was 87.4%The average online accuracy for the BCI task was 97.5% [11]. The average offline binary accuracy was 87.4%. The mean detection accuracy for T1 and T2 in the RSVP task was 77.2% and 67.7% [11].

Chapter 4

The Dataset

4.1 Data collection:

In order to gather P300 wave data from brain signals, the row/column ordered flashing approach initially demonstrated by Farwell and Donchen [1] is typically utilized. The EEG signals of seven healthy individuals and eight ALS patients were acquired from a publicly available dataset [21]. Following a 10-20 international system electrode was positioned on the subject's scalp. A 6*6 matrix with an alphanumeric grid was employed, and a green target letter was flashed, in order to determine a person's mental state and attention level by measuring screen flash time. The authors began the session by telling the participants to count the flashes of the target character [21].Participants were told to calculate the number of instances target letters flickered to determine their level of attentiveness [21]. Each target character was displayed five times, separated by a two-second interval [21].

4.2 Data Analysis and Pre-processing:

The p300 (p3) wave is a component of the event-related potential (ERP) that manifests when a decision must be made. It is called an endogenous potential because it has nothing to do with how a stimulus looks but with how a person responds to it. The raw features collected from the brain signals will be greater in size and more sensitive to noise, thus it is necessary to process the datasets adequately in order to fit them into our classification model.We retrieved a dataset that has been pre-processed in six steps [21] as part of our research.The EEG data are divided into target and non-target trials lasting 600 ms following stimulus onset in the first step [21], and observations between 200 and 500 ms are taken into consideration. In the subsequent detrending process, a total mean value was subtracted from each channel to eliminate noisy data caused by arbitrary offsets.In bad trial removal, eeg data trials were edited to remove artifacts by computing the mean absolute value of each trial and removing trials with values more than three standard deviations above the median trial[21]. For the purpose of bad channel removal, the total power of each channel, the mean channel power, and the variation in channel power throughout all epochs were calculated. Then, they eliminated and replaced channels with a power that surpassed three standard deviations[21].Once more, source mixing and volume-transfer noise were removed using spatial filtering. Electrodes were then linearly reweighed and the original data was transmitted using uncorrelated, equal-

power sensors[21]. In the final step of spectral filtering, the EEG signal was Fourier transformed and weighted to remove unwanted frequencies in order to band-pass the signal between 0.1 Hz and 15 Hz [21]. By following these processes they have pre processed their dataset and we retrieved it for our research work and used it on our classifications model. In our research, we measured subject1's attentiveness using 3982 events where target and non-target was a class label for positive and negative value respectively.

Chapter 5

The Models

Following the data preprocessing, we will describe the algorithms we used to measure accuracy in this section.

5.1 Classification Model:

5.1.1 Logistic Regression

A supervised machine learning model is Logistic Regression which is applied to anticipate the outcome of binary events, such as yes or no. Logistic regression can be used to predict the most influential variables used for the classification. That is why it is suitable for solving the classification problem. Here the sigmoid function is used to map predicted values into probabilities. Its value must be between 0 and 1, so it makes a graph of the 'S' shape.

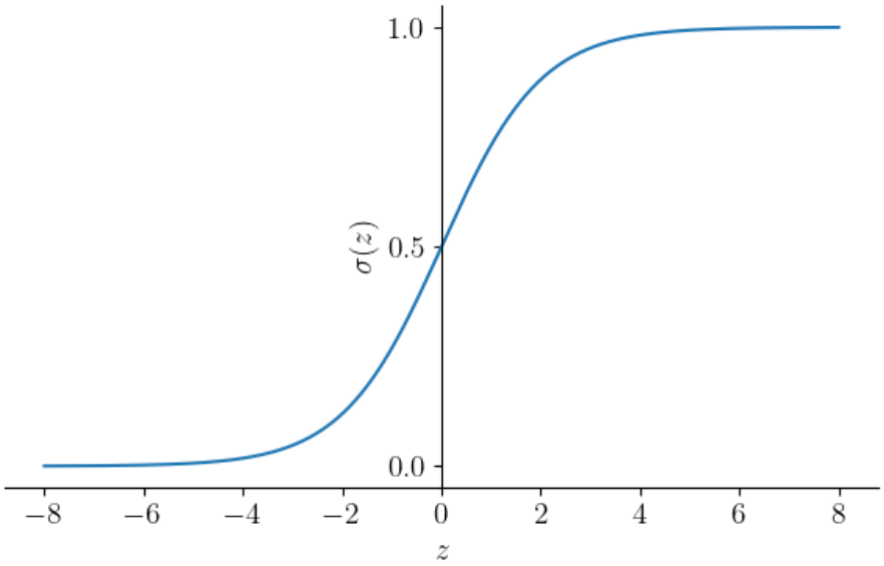


Figure 5.1: Logistic regression graph

To use the model for our binary classification, we have to fit this model for the training set. We extracted the dependent and independent variables. Then we

predict the result and give the test accuracy of the result.

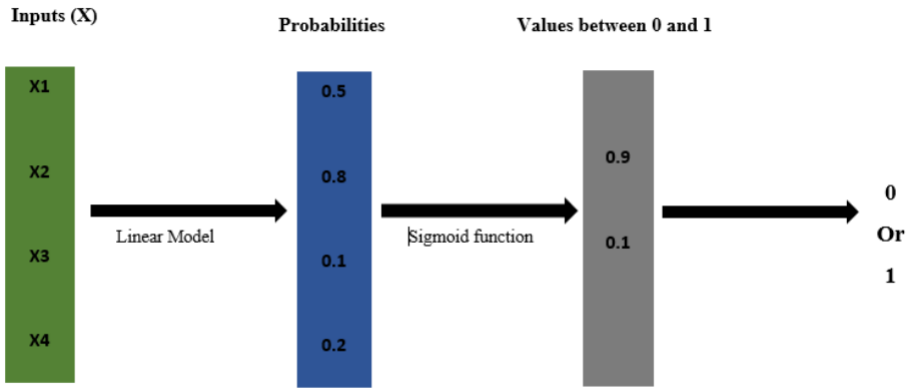


Figure 5.2: Logistic regression model

5.1.2 Random Forest

Random forest is a supervised machine learning algorithm that can be used for both classification and regression problems in Machine Learning. By merging several different learning models, it produces better results when applied to difficult issues. In addition, this approach requires significantly less training time than existing algorithms. The method constructs numerous decision trees and then combines those trees in order to obtain answers that are more accurate and consistent. The results will be significantly more trustworthy if there are more trees in the forest overall. While splitting a node, the algorithm seeks for the best characteristics from a random selection of features, which increases the model's variety and produces a superior model.

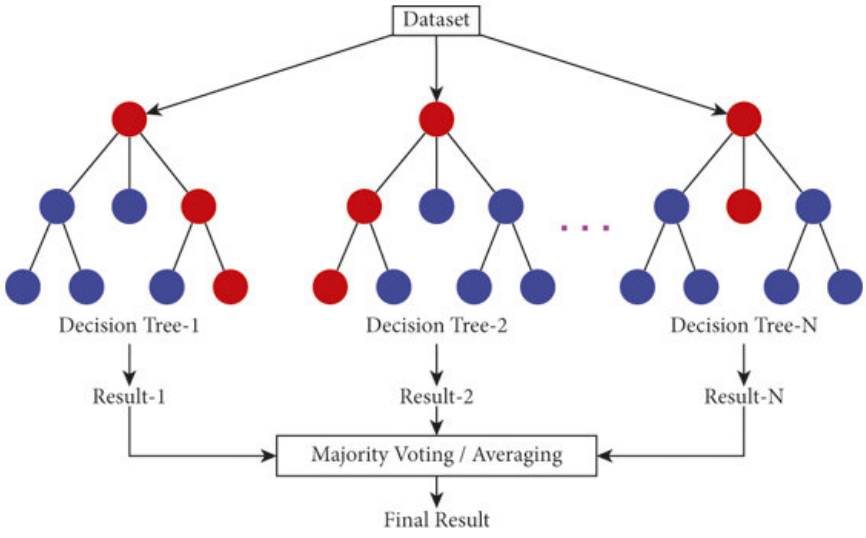


Figure 5.3: Random forest model
[27]

Random forest models first make a bootstrapped dataset from the original dataset randomly. In bootstrapped dataset, samples can be duplicated or also there can be missing samples. After that, it makes random decision trees. Different decision trees give different predictions. The prediction which is the highest number or average number is the final result.

5.1.3 XGBoost

Extreme Gradient Boosting (XGBoost) is an end-to-end tree boosting system with a high scalability function [25] [26]. This algorithm uses parallel and distributed processing to achieve faster solutions [16]. It maximizes the objective function by lowering the loss function by employing decision trees as base classifiers, where the loss function regulates the complexity of the tree [25]. It supports regularization, sparsity-aware algorithms, shrinkage and column subsampling blocks, cross-validation, cache-aware access, parallel learning blocks, out-of-core computations, greedy algorithms, etc. XGBoost's main prominent characteristics are:-

- To reduce model complexities and overfitting, the regularization method is utilized [25]
- XGBoost eliminates missing or zero-valued entries that induce sparsity from split candidates by computing their gains.[25]
- Enables out-of-core computation for data that cannot fit in main memory by dividing the data into several blocks and stores each block on disk [16].
- Cache optimization helps to switch the direct read/write dependency to a longer dependency by reducing the runtime overhead for large rows [16].
- Stores compressed and sorted data in memory blocks to minimize the tree sorting time [16].

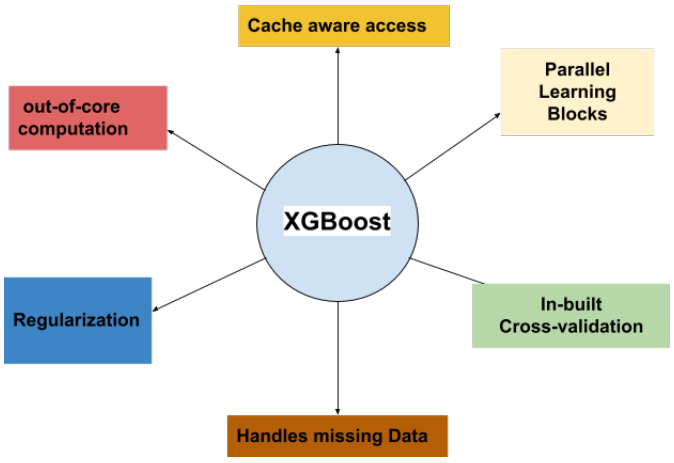


Figure 5.4: XGBoosts's Prominent Features

5.1.4 AdaBoost

A single classifier may not always be able to predict an item's class accurately. However, a robust model can be made by combining weak classifiers and letting each learn from the other's mistakes [24]. AdaBoost, also called Adaptive Boosting is a technique in Machine Learning that uses a sequential approach to train and deploy trees. The most frequent AdaBoost algorithm is decision trees with only 1 split. They're called Decision Stumps [28] Initially, some weights will be given to these data points. Then, identify the stump that best fits the new group of samples by figuring out their Gini Index and choosing the one with the lowest Gini Index. The next step is to figure out the 'Amount of Say' and 'Total error' to update the sample weights from before.

Using the following formula, the actual influence for this classifier,

$$\alpha_t = \frac{1}{2} \times ln \times \frac{(1 - TotalError)}{TotalError} \tag{5.1}$$

Here, α_t = Stumps influence in the final classification
TotalError = The total number of misclassifications for that set divided by the training set size.

In addition, the total error will always be between 0 and 1. In the meantime, normalize the weights of the new samples. Lastly, the algorithm picks numbers at random from 0 to 1 and makes a new dataset.A sample whose classification by the previous tree was incorrect will have its weight increased so that the upcoming tree may categorize it effectively. Classification accuracy increases but overfitting and generalization capacity drops when inadequate classifiers are added to a structure in sequence. AdaBoost works well with skewed data but not with noise [24]

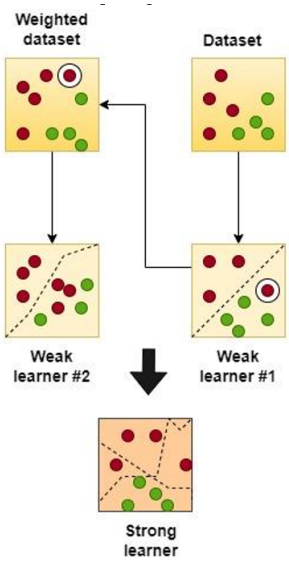


Figure 5.5: Implementation of AdaBoost classifier on a dataset that has two features and two classes

5.2 The Custom CNN model:

Our research aimed to develop models that were easy to access and simple to train, with fewer learning parameters but higher accuracy than the pre-trained models used in this research. As a result, we constructed a custom conv1D CNN model to improve efficiency rather than depending on pre-trained models. We build our customized CNN model as follows: For our research, before applying the model, we split our dataset into an 80:20 ratio where the training set is 80%, and the test set is 20%. Our network has 3 layers, namely the convolutional layer, max-pooling layer, and dense layer. We used 32 (3×3) kernel convolution layers with a stride of 1, RELU activation, and an input shape of (16,76). Similarly, 64, (3×3) a 1D convolutional layer containing half padding and a batch normalization layer with an activation function of RELU was applied. Along with that layer, we implemented a batch normalization layer with a RELU activation function and 128,(3×3) a 1D convolutional layer with half padding. Furthermore, a dropout layer with a value of 0.25 was introduced to improve the overfitting problem. Also, flattening layers help flatten the multi-dimensional layers into a 1-dimensional layer. At this time, another 64, (3×3) a 1D convolutional layer with half padding and a batch normalization layer with a RELU activation function were added to the model. After that, a max-pooling layer with an average kernel size of 1 and a stride of 1 was applied. Finally, we add a dense layer with the activation function of Sigmoid so that it can provide weights to all the nodes from previous layers. Most binary classification is done using sigmoid. For our model compiling, we execute adam as an optimizer, binary cross-entropy as loss function, and accuracy as matrices. To find validation data, we put 100 epochs to achieve reasonable accuracy.

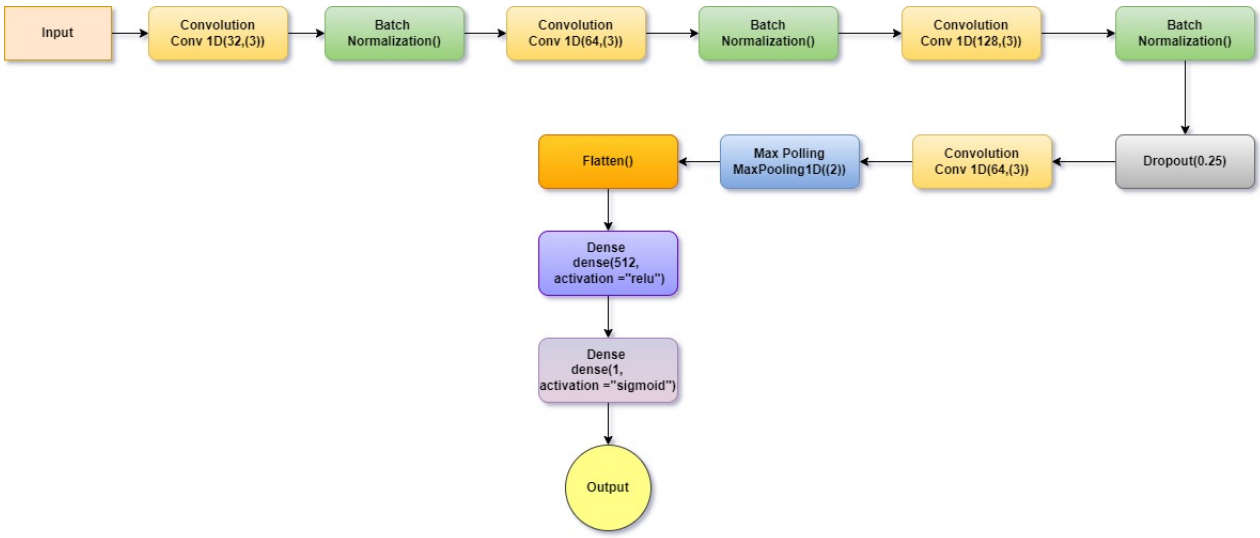


Figure 5.6: Architecture of custom CNN model

5.3 Model Evaluation Metrics

To evaluate our CNN model and machine learning classifier, we employed some metrics and functions in our algorithm and needed to learn about these metrics to understand our analysis at ease. This evaluation will help us to comprehend our result analysis.

5.3.1 Confusion Matrix:

The confusion matrix, which is also called the error matrix, is a table that shows or sums up the performance of a classification or supervised model. We used a binary classification in our research. As a result of employing this matrix, we were able to come up with four binary classification possibilities as below:

- True positives (TP): When the predicted and real outcomes are both positive.
- True negatives (TN): When both the authentic outcome and the prediction are negative.
- False positives (FP): When the actual outcome is negative, but the predicted outcome is Positive. This is also called as the Type 1 error.
- False negatives (FN): When prediction is negative but actual is positive. This is also called as type 2 error.

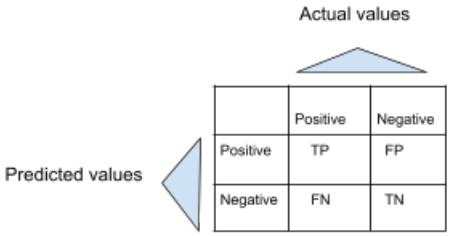


Figure 5.7: 2*2 confusion matrix for binary classification

5.3.2 Accuracy:

Accuracy is a way to measure how well a classification model works by looking at how often it makes correct predictions. It tells us how many of the positive predictions came true out of all of them, and it is usually given as a percentage. The following formulas are used to accomplish this:

Accuracy =

$$\frac{TP + TN}{TP + FP + FN + TN} \tag{5.2}$$

For our research, the proposed model is focused on determining whether a person can correctly predict the character or predicting something else is implemented in this section. After employing the algorithm we got our desired outcomes.

5.3.3 Precision

Precision is the capacity of a database to retrieve pertinent document records while excluding irrelevant ones [7]. Alternatively, precision is defined as the ratio of true positives to all positive model predictions.

$$Precision = \frac{TP}{FP + TP} \tag{5.3}$$

5.3.4 Recall

The ratio of all potentially corrected relevant items to the actual retrieved significant items is known as recall [7]. This indicator displays the percentage of all correctly predicted positive outcomes made by models that are actually true.
The mathematical representation of recall:

$$Recall = \frac{TP}{FN + TP} \tag{5.4}$$

5.3.5 F1-score

The harmonic average of recall and precision is what makes up the F1-score [14]. F1-score represents both classifiers by combining them into a single metric. One can calculate F1-score:

$$F1 - score = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{5.5}$$

The F1-score increases in direct proportion to a model's performance

Chapter 6

Result Analysis

To have a general understanding of the performance of the models, we have trained and tested each one. Consequently, it is now required to evaluate the results obtained from them in order to differentiate between the highest and lowest levels of accuracy from the table [Fig : Analysis of classification accuracies]. The accuracy percentage of Logistic Regression is 84.19%, Random Forest is 85.82%, XGBoost is 85.82%, AdaBoost is 81.81% and Custom CNN is 88.46%.

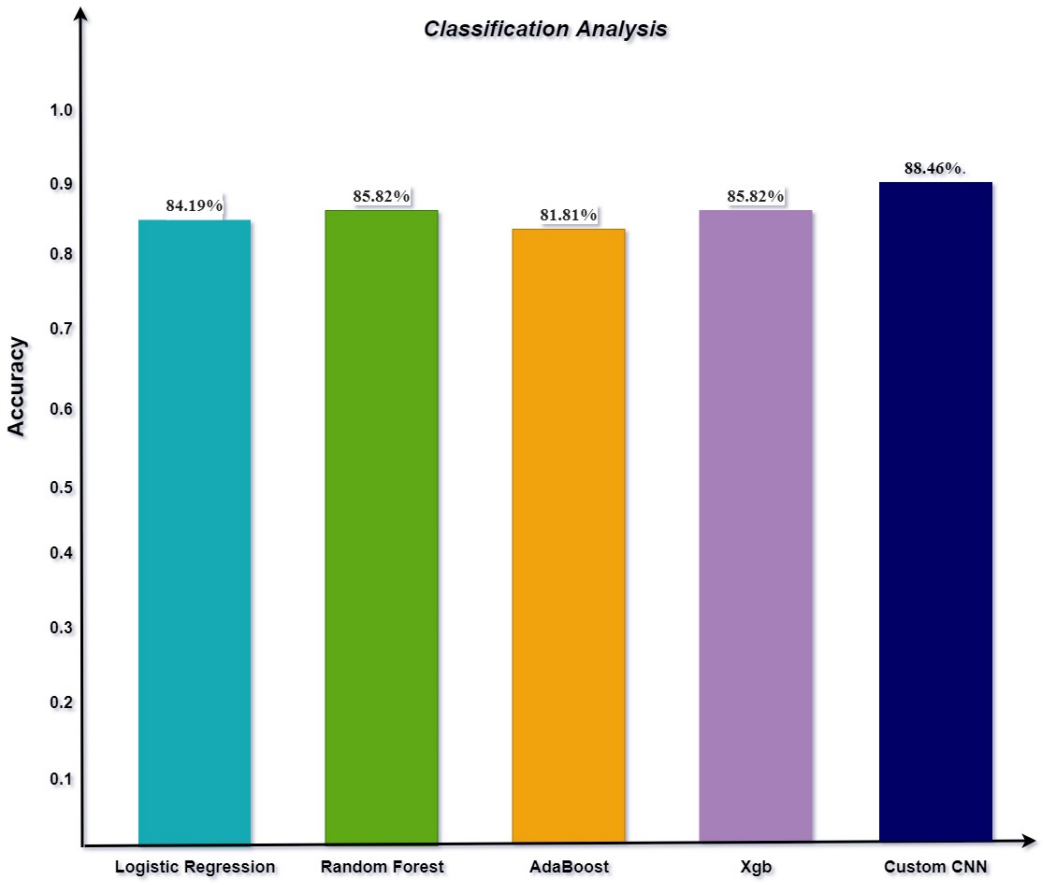


Figure 6.1: Analysis of classification accuracies

Test accuracy portrays the models' ability to differentiate between target and non-target using unseen data. High test accuracy indicates that the model can distinguish between targets and non-targets with more accuracy. In this table Custom

CNN has a higher accuracy of 0.8846 which indicates that the model accurately predicted the test data 88.46% of the time.The accuracy ratings show that Custom CNN has the maximum accuracy of 88.46% and the lowest test accuracy of 81.81% is achieved by AdaBoost.

```
100/100 [==============================] - 8s 38ms/step - loss: 0.5416 - accuracy: 0.8273 - val_loss: 0.5146 - val_accuracy: 0.8846
Epoch 2/100
100/100 [==============================] - 3s 34ms/step - loss: 0.3891 - accuracy: 0.8462 - val_loss: 0.4306 - val_accuracy: 0.8795
Epoch 3/100
100/100 [==============================] - 3s 34ms/step - loss: 0.3516 - accuracy: 0.8597 - val_loss: 0.3948 - val_accuracy: 0.8745
Epoch 4/100
100/100 [==============================] - 2s 18ms/step - loss: 0.3086 - accuracy: 0.8763 - val_loss: 0.3916 - val_accuracy: 0.8670
Epoch 5/100
100/100 [==============================] - 2s 19ms/step - loss: 0.2648 - accuracy: 0.8920 - val_loss: 0.5158 - val_accuracy: 0.7716
Epoch 6/100
100/100 [==============================] - 2s 18ms/step - loss: 0.2009 - accuracy: 0.9196 - val_loss: 0.5772 - val_accuracy: 0.7967
Epoch 7/100
100/100 [==============================] - 2s 19ms/step - loss: 0.1675 - accuracy: 0.9312 - val_loss: 0.5785 - val_accuracy: 0.8005
Epoch 8/100
100/100 [==============================] - 2s 19ms/step - loss: 0.1277 - accuracy: 0.9488 - val_loss: 0.7248 - val_accuracy: 0.7829
Epoch 9/100
100/100 [==============================] - 2s 19ms/step - loss: 0.1068 - accuracy: 0.9592 - val_loss: 0.7356 - val_accuracy: 0.7829
Epoch 10/100
100/100 [==============================] - 2s 19ms/step - loss: 0.1137 - accuracy: 0.9545 - val_loss: 0.8191 - val_accuracy: 0.8570
```

Figure 6.2: First 10 epochs of CNN accuracy

For deep learning, a customized 1D CNN model has been proposed. To fit our model, we arbitrarily utilized 100 epochs to evaluate its performance. After training and evaluating the model, the highest level of validation accuracy is achieved, which is 88.46%.

In our custom CNN model for some cases, the accuracy graph revealed that our model is overfitted by comparing training accuracy to validation accuracy. On the other hand, the validation loss rate is much higher than the training loss rate, which may indicate that the model is overfitting. As a result, we employed the regularized dropout method, which assists in reducing overfitting, along with batch normalization. In our model, we utilized the Batch Normalization Layer and Dropout Layer with a value of 0.25 for this purpose. After that, we had more accurate results than we had previously. By training and testing our model with an input set and multiple layers, we were able to get the highest accuracy for our CNN model.

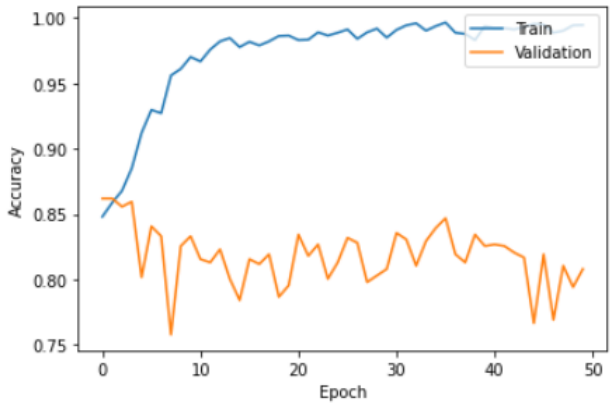


Figure 6.3: Train accuracy vs validation accuracy

The figure 6.3 above shows train accuracy vs validation accuracy. Here, the train accuracy increases with the increasing number of epochs on average. On the other hand, in terms of validation accuracy, when epoch is 0 it is the highest. After that, with the increase of epochs it's getting to a lower value on average.

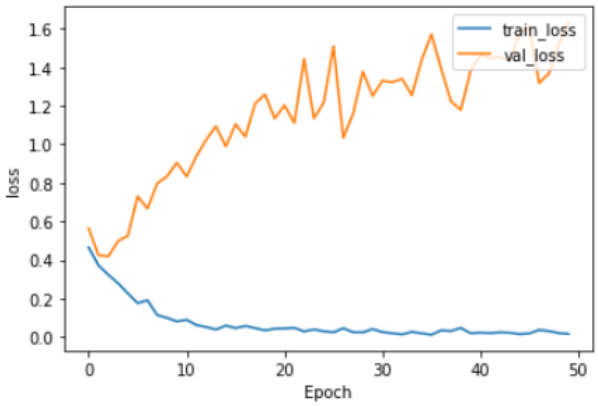


Figure 6.4: Train loss vs validation loss

The figure 6.4 shows the train loss vs validation loss. In terms of validation loss, it is increasing as the number of epoch increases step by step. On the other hand, in terms of train loss when epoch is increasing the training loss is decreasing on average.

For our four classification models, the average precision, recall, and F1 scores are used as evaluation measures, as indicated in the table below [Table 6.1],

Model	**Avg. Precision**	**Avg. Recall**	**Avg. F1-score**
Logistic Regression	0.79	0.84	0.80
Random Forest	0.73	0.86	0.79
AdaBoost	0.77	0.82	0.79
XGBoost	0.73	0.86	0.79

Table 6.1: Average Precision, Recall and F1- Scores of the Models

According to the [Table 6.1], all models have a greater Recall than Precision and F1-score. Approximately 86 percent of the target label can be identified using Random Forest and XGBoost classifiers as they have the highest recall values. Moreover, among the models, logistic regression has the highest precision values, meaning it would detect fewer non-target labels and measure attentiveness with a higher degree of accuracy (0.79 %). The F1 score is the harmonic mean of recall and precision, hence a classifier with a high F1 score is superior. As a result of its high precision and recall, logistic regression receives a high F1 score about 0.80%, but the other three models have a constant F1 score.

To sum up , data analysis reveals that the custom CNN is the most accurate model for predicting attentiveness among the patients. Compared to the other machine learning classifiers we have tried, it is faster, easier to use, and more accurate in classifying attention.

Chapter 7

Discussion

The preceding [figure 6.1] depict the classification performance of our various approaches (Logistic Regression, Random Forest, XGBoost, and AdaBoost) to detect focused attention.We achieved an accuracy of 88.46% with our custom CNN 1D model for binary classification. In related works [11], Riccio et al. demonstrated a stepwise linear discriminant analysis (SWLDA) which was performed for determining weights of the classifier. They had two targets- T1 and T2. The mean accuracy of RSVP task for detection for T1 was 77.2% and for T2 67.7%. The BCI tasks' binary accuracy was on average 87.4%. In addition, [1] Farwell et al. established accuracy of 80% and 95% using their quickest tested algorithm SWDA. Here, they showed how the P300 part of the ERP, which measures how the brain reacts to events, works when mental prostheses are used.

Our research also uses the P300 dataset, therefore we tested a few alternative approaches from those studies.We tried a few different ways from those papers as our research is also based on the P300 dataset. To determine whether a person is attentive or not, we proposed a revolutionary tailored CNN 1D model combined with classifications based on machine learning. In addition, the CNN model provided us with a greater accuracy rate of 88.46%, thereby revealing a new research direction. A binary classification problem arises when a machine learning model needs to be trained to classify between two classes. Due to the fact that our dataset consists of two parts—train and test—binary classification is an efficient method for achieving a superior result. In this work, we also applied binary classification to determine a subject's degree of concentration using a BCI system. Due to the complexities of the BCI system and related work, the machine learning algorithm is not always up to the mark; therefore, we will also attempt to enhance the performance of our applied models.

Chapter 8

Conclusion and Future Plan

8.1 Conclusion

The BCI technology allows people to express themselves using machine translation. Our research relies on different approachs of machine learning while using the P300 dataset and EEG to monitor the patient's brain activity, we can determine if they are attentive and able to detect the target characters or if they are preoccupied. Logistic regression, Random forest, XGBoost, AdaBoost, and a Custom CNN were used in this investigation. Custom CNN There are numerous distinct EEG signal applications. People who are lagging behind in their ability to interact with others can profit greatly from our method as it will create an opportunity for them to later detect character. These machine learning algorithms can also detect brain signal problems. People who are lagging behind in their ability to interact with others can profit greatly from our method as it will create an opportunity for them to later detect character.

8.2 Future Plan

Following a patient attention test, the next stage of our research involves using the P300 speller system to recognize a specific character from a 6*6 matrix of 26 letters (A-Z) and 10 numbers (0-9) that are randomly flashed on a computer screen in rows and columns. We'll also attempt to use more algorithms to get the best possible precision and accuracy.

Bibliography

[1] L. A. Farwell and E. Donchin, "Talking off the top of your head: Toward a mental prosthesis utilizing event-related brain potentials," *Electroencephalography and clinical Neurophysiology*, vol. 70, no. 6, pp. 510–523, 1988.

[2] S. Treitel, "Spectral analysis for physical applications: Multitaper and conventional univariate techniques," *American Scientist*, vol. 83, no. 2, pp. 195–197, 1995.

[3] P. Flandrin, "Time-frequency/time-scale analysis. academic press, san diego.," *Time-frequency/time-scale analysis. Academic Press, San Diego.*, 1999.

[4] V. Bostanov, "Bci competition 2003-data sets ib and iib: Feature extraction from event-related brain potentials with the continuous wavelet transform and the t-value scalogram," *IEEE Transactions on Biomedical engineering*, vol. 51, no. 6, pp. 1057–1061, 2004.

[5] M. Mørup, L. K. Hansen, C. S. Herrmann, J. Parnas, and S. M. Arnfred, "Parallel factor analysis as an exploratory tool for wavelet transformed event-related eeg," *NeuroImage*, vol. 29, no. 3, pp. 938–947, 2006.

[6] A. Bashashati, M. Fatourechi, R. K. Ward, and G. E. Birch, "A survey of signal processing algorithms in brain–computer interfaces based on electrical brain signals," *Journal of Neural engineering*, vol. 4, no. 2, R32, 2007.

[7] W. H. Walters, "Google scholar search performance: Comparative recall and precision," *portal: Libraries and the Academy*, vol. 9, no. 1, pp. 5–24, 2009.

[8] H. Abdi and L. J. Williams, "Principal component analysis," *Wiley interdisciplinary reviews: computational statistics*, vol. 2, no. 4, pp. 433–459, 2010.

[9] R. Fazel-Rezai and W. Ahmad, "P300-based brain-computer interface paradigm design," *Recent advances in brain-computer interface systems*, pp. 83–98, 2011.

[10] J. Farquhar and N. J. Hill, "Interactions between pre-processing and classification methods for event-related-potential classification," *Neuroinformatics*, vol. 11, no. 2, pp. 175–192, 2013.

[11] A. Riccio, L. Simione, F. Schettini, *et al.*, "Attention and p300-based bci performance in people with amyotrophic lateral sclerosis," *Frontiers in human neuroscience*, vol. 7, p. 732, 2013.

[12] J. R. Wolpaw, "Brain–computer interfaces," in *Handbook of Clinical Neurology*, vol. 110, Elsevier, 2013, pp. 67–74.

[13] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[14] D. Zhang, J. Wang, and X. Zhao, "Estimating the uncertainty of average f1 scores," in *Proceedings of the 2015 International conference on the theory of information retrieval*, 2015, pp. 317–320.

[15] B. S. Armour, E. A. Courtney-Long, M. H. Fox, H. Fredine, and A. Cahill, "Prevalence and causes of paralysis—united states, 2013," *American journal of public health*, vol. 106, no. 10, pp. 1855–1857, 2016.

[16] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, 2016, pp. 785–794.

[17] H. Cecotti and A. J. Ries, "Best practice for single-trial detection of event-related potentials: Application to brain-computer interfaces," *International Journal of Psychophysiology*, vol. 111, pp. 156–169, 2017.

[18] M. R. Mowla, J. E. Huggins, and D. E. Thompson, "Enhancing p300-bci performance using latency estimation," *Brain-Computer Interfaces*, vol. 4, no. 3, pp. 137–145, 2017.

[19] N. Bjorck, C. P. Gomes, B. Selman, and K. Q. Weinberger, "Understanding batch normalization," *Advances in neural information processing systems*, vol. 31, 2018.

[20] R. Yamashita, M. Nishio, R. K. G. Do, and K. Togashi, "Convolutional neural networks: An overview and application in radiology," *Insights into imaging*, vol. 9, no. 4, pp. 611–629, 2018.

[21] B. Abibullaev and A. Zollanvari, "Learning discriminative spatiospectral features of erps for accurate brain–computer interfaces," *IEEE Journal of Biomedical and Health Informatics*, vol. 23, no. 5, pp. 2009–2020, 2019.

[22] J. Sheth, A. Tankus, M. Tran, N. Pouratian, I. Fried, and W. Speier, "Translating neural signals to text using a brain-machine interface," *arXiv preprint arXiv:1907.04265*, 2019.

[23] X. Ying, "An overview of overfitting and its solutions," in *Journal of physics: Conference series*, IOP Publishing, vol. 1168, 2019, p. 022 022.

[24] S. Misra, H. Li, and J. He, "Noninvasive fracture characterization based on the classification of sonic wave travel times," *Machine Learning for Subsurface Characterization*, pp. 243–287, 2020.

[25] C. Bentéjac, A. Csörgő, and G. Martınez-Muñoz, "A comparative analysis of gradient boosting algorithms," *Artificial Intelligence Review*, vol. 54, no. 3, pp. 1937–1967, 2021.

[26] Y. Guang, "Generalized xgboost method," *arXiv preprint arXiv:2109.07473*, 2021.

[27] M. Y. Khan, A. Qayoom, M. Nizami, M. S. Siddiqui, S. Wasi, and K.-U.-R. R. Syed, "Automated prediction of good dictionary examples (gdex): A comprehensive experiment with distant supervision, machine learning, and word embedding-based deep learning techniques," *Complexity*, Sep. 2021. DOI: 10.1155/2021/2553199.

[28] A. Saini, *Adaboost algorithm - a complete guide for beginners*, Sep. 2021. [Online]. Available: http://www.analyticsvidhya.com/blog/2021/09/adaboost-algorithm-a-complete-guide-for-beginners.

[29] A. Shahid, I. Raza, and S. A. Hussain, "Emowrite: A sentiment analysis-based thought to text conversion," *arXiv preprint arXiv:2103.02238*, 2021.