Analysis Report

EEG Based P300 Speller

M Balakumaran
240602
Under Guidance Of Prof. Nikunj Bhagat

Abstract

This work presents a comprehensive end-to-end pipeline for the processing and classification of P300 electroencephalography (EEG) signals in a brain–computer interface (BCI) speller system. Raw EEG data recorded from two subjects are preprocessed using bandpass filtering (0.1–20 Hz) and downsampling (240 Hz to 120 Hz), followed by stimulus-locked epoch extraction with 800 ms windows. Discriminative features are derived through a combination of Principal Component Analysis (PCA), Common Spatial Patterns (CSP), and time-domain feature extraction methods. Multiple machine learning classifiers, including Linear Discriminant Analysis (LDA), Logistic Regression, Support Vector Machines (SVM), Random Forests, and Gradient Boosting, are trained and evaluated. Performance is assessed using evaluation metrics suitable for imbalanced datasets, addressing the challenges posed by noisy signals and significant class imbalance inherent in real-world EEG-based BCI applications.

1. Introduction

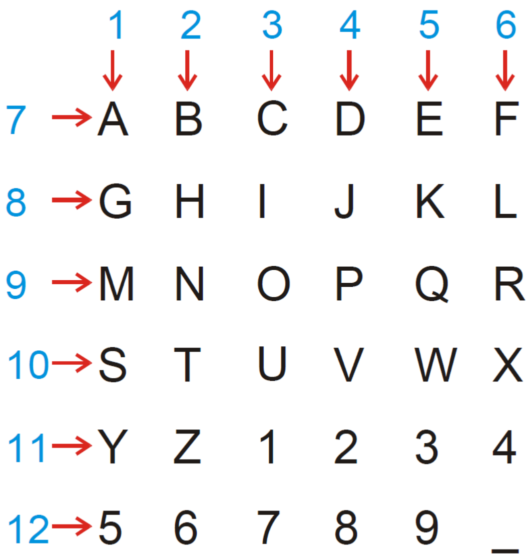
The P300 speller is a brain–computer interface (BCI) paradigm that enables users to communicate by selectively attending to target characters within a matrix while rows and columns are presented in a randomized flashing sequence. The system relies on the detection of the P300 event-related potential, characterized by a positive deflection in EEG signals occurring approximately 300 ms following an infrequent or task-relevant stimulus. This study implements a complete machine learning pipeline for P300 detection using electroencephalography (EEG) data from the BCI Competition III Dataset II.

This study develops a modular and reproducible EEG signal processing pipeline for P300 detection, incorporating multiple feature extraction techniques and systematic comparison of their effectiveness. The framework enables the training and evaluation of diverse machine learning classifiers while explicitly addressing the challenge of severe class imbalance inherent in EEG-based BCI data. The proposed approach emphasizes robustness, reproducibility, and adaptability, supporting reliable P300 detection across varying signal conditions.

2. Experimental Paradigm

The dataset contains electroencephalographic (EEG) recordings acquired using the BCI2000 P3 speller paradigm, originally inspired by the oddball-based communication framework proposed by Farwell and Donchin.

The dataset is based on a visual P300 speller paradigm in which subjects attend to a target character within a 6×6 matrix comprising alphanumeric symbols. During each character selection epoch, all rows and columns of the matrix are randomly intensified in a block-randomized sequence. Each intensification lasts 100 ms, followed by a 75 ms inter-stimulus interval, yielding a stimulation rate of approximately 5.7 Hz. For each character epoch, 12 distinct stimuli (6 rows and 6 columns) are presented and repeated 15 times, resulting in 180 total intensifications per character. Exactly two intensifications per cycle (one row and one column) contain the target character, eliciting a characteristic P300 event-related potential due to the oddball effect.



	1	2	3	4	5	6
7	A	B	C	D	E	F
8	G	H	I	J	K	L
9	M	N	O	P	Q	R
10	S	T	U	V	W	X
11	Y	Z	1	2	3	4
12	5	6	7	8	9	_

Figure 1: Auxiliary variables encoding stimulus timing and labels.

3. Data Acquisition and Dataset Structure

EEG signals were recorded from two subjects, each participating in five sessions. Signals were acquired using 64 scalp electrodes arranged according to the extended 10–20 system, bandpass filtered between 0.1–60 Hz, and digitized at a sampling rate of 240 Hz.**Refer Figure 2** for EEG acquisition setup and electrode configuration. Each session consisted of multiple runs in which subjects spelled predefined words by sequentially focusing on individual characters. To prevent linguistic inference, character epochs were scrambled across runs in both training and test datasets.

For each subject, the dataset is provided as four MATLAB (.mat) files: one training set and one test set. The training set contains 85 character epochs, while the test set contains 100 character epochs. The core EEG data are stored in a three-dimensional matrix **Signal** with dimensions:

Character Epoch × Time Samples × Channels

In addition to the raw EEG signals, several auxiliary variables encode stimulus timing and labels:

Flashing: binary indicator of row/column intensification

StimulusCode: identifies which row (7–12) or column (1–6) is intensified

StimulusType (training only): labels target (1) vs. non-target (0) stimuli

TargetChar (training only): ground-truth character label for each epoch

The test datasets exclude StimulusType and TargetChar, reflecting the intended classification challenge.

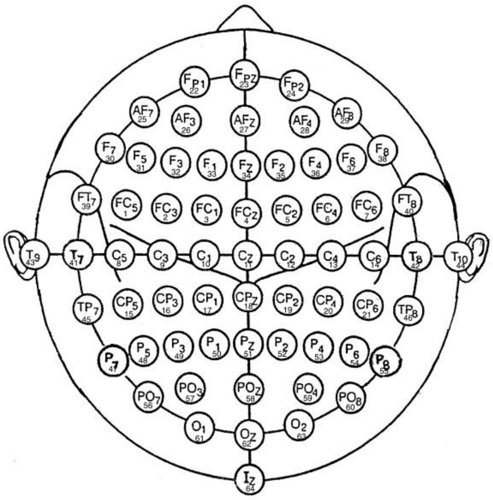
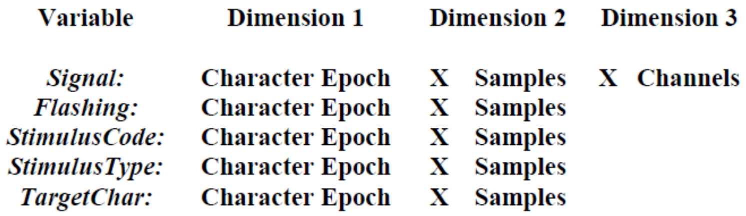


Figure 2: Overview of the EEG acquisition setup and electrode configuration.



Variable	Dimension 1		Dimension 2		Dimension 3
Signal:	Character Epoch	X	Samples	X	Channels
Flashing:	Character Epoch	X	Samples		
StimulusCode:	Character Epoch	X	Samples		
StimulusType:	Character Epoch	X	Samples		
TargetChar:	Character Epoch	X	Samples		

Figure 3: Structure of the **Signal** matrix: Character Epoch × Time Samples × Channels.

4. Data Preprocessing

The raw EEG signals were subjected to a standardized preprocessing pipeline to enhance signal quality and suppress noise artifacts. Bandpass filtering was first applied using a 4th-order Butterworth filter with cutoff frequencies of 0.1–20 Hz, preserving low-frequency event-related potentials while attenuating slow drifts and high-frequency noise. To further mitigate electrical contamination, a 50 Hz notch filter with a quality factor (Q) of 30 was employed to suppress powerline interference. Following filtering, the data were downsampled from an original sampling rate of 240 Hz to 60 Hz, achieving a four-fold reduction in temporal resolution while retaining the frequency components of interest

Following preprocessing, the continuous EEG signals were segmented into stimulus-locked epochs to facilitate event-related analysis. Epoch extraction was performed using a fixed window length of 800 ms, corresponding to 48 samples at a sampling rate of 60 Hz, time-locked to stimulus onset. To correct for slow signal drifts and inter-trial variability, baseline correction was applied using a 100 ms pre-stimulus interval, which was subtracted from each epoch

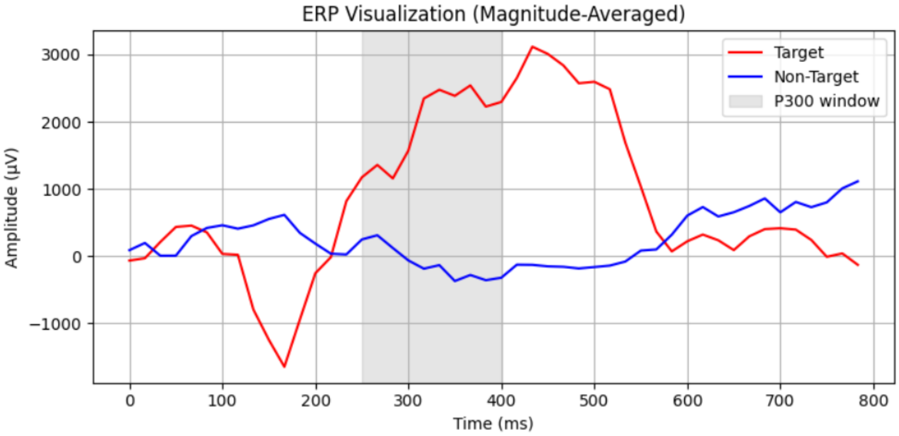


Figure 4: P300 Response

Table 1: Summary of Extracted Epochs and Data Dimensions

Subject	Dataset Split	No. of Epochs	Epoch Shape
A	Training	15,299	(15299, 64, 48)
	Testing	17,999	(17999, 64, 48)
B	Training	15,299	(15299, 64, 48)
	Testing	17,999	(17999, 64, 48)

5. Feature Extraction and Comparison

Four distinct feature extraction methods were implemented and systematically compared to evaluate their effectiveness in representing EEG signals for classification. Principal Component Analysis (PCA) was employed with two different dimensional configurations, namely PCA-20 and PCA-50, yielding feature spaces of 20 and 50 dimensions, respectively.

In addition, Common Spatial Patterns (CSP) was applied with six spatial components, focusing on discriminative spatial filtering between target and non-target classes, without an explicit variance retention measure.

Finally, a time-domain feature representation was constructed by flattening the preprocessed EEG epochs into a 3072-dimensional vector

Based on Table 2, the extracted features exhibit notable differences in discriminative effectiveness and dimensional efficiency. The PCA-20 features show limited performance, indicating that a small number of principal components is insufficient to capture meaningful variations in the data. Increasing the dimensionality to PCA-50 results in a substantial improvement in classification performance, suggesting that retaining additional components preserves more relevant information. The CSP features, while compact, provide only moderate performance, whereas the Time Domain features, despite achieving reasonable results, suffer from very high dimensionality, which may introduce redundancy and increase computational complexity.

Considering the balance between performance and feature dimensionality, **PCA-50 is selected as the most suitable feature representation for training**, as it offers improved discriminative capability while maintaining a manageable feature size.

Table 2: Extracted features comparison

Features Extraction	Dimension	LDA F1 Score	SVM F1 Score
PCA-20	20	0.0117	0.3294
PCA-50	50	0.0269	0.3416
CSP	6	N.A	0.2589
Time Domain	3072	0.2679	N.A

6. ML Models and Evaluation Metrics

Six classifiers were implemented:
1. Linear Discriminant Analysis (LDA)
2. Logistic Regression
3. Support Vector Machine (RBF kernel)
4. Support Vector Machine (Linear kernel)
5. Random Forest
6. Gradient Boosting

To evaluate the above mentioned models, different kind of evaluation metrics have been applied:
Let:
TP = True Positives
TN = True Negatives
FP = False Positives
FN = False Negatives

$$\text{Accuracy } = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

$$\text{Precision } = \frac{TP}{TP + FP} \tag{2}$$

$$\text{Recall } = \frac{TP}{TP + FN} \tag{3}$$

$$\text{F1-score } = 2 \cdot \frac{\text{Precision } \cdot \text{ Recall}}{\text{Precision } + \text{ Recall}} \tag{4}$$

$$\text{ROC-AUC} = \text{Area under ROC curve} \tag{5}$$

7. Result

From Subject A, Linear Discriminant Analysis (LDA), Logistic Regression, and Random Forest achieve comparatively high accuracies (83%); however, both LDA and Logistic Regression exhibit extremely low recall and F1-scores, indicating poor sensitivity to the minority (target) class. Random Forest further collapses with zero precision, recall, and F1-score, suggesting a strong bias toward the majority class. In contrast, SVM (Linear) and Gradient Boosting demonstrate more balanced performance, with noticeably higher recall and F1-scores, reflecting improved target detection capability. Among these, SVM (Linear) attains the highest F1-score, while Gradient Boosting yields the highest recall.

For Subject B, a similar trend is observed. Although LDA, Logistic Regression, and Random Forest again report high accuracies (83%), their precision, recall, and F1-scores are near zero, confirming ineffective classification of the target class. SVM (Linear) and SVM (RBF) outperform other models in terms of recall and F1-score, with SVM (Linear) providing a slightly more consistent balance between sensitivity and specificity. Gradient Boosting shows moderate performance but remains inferior to SVM-based approaches in overall robustness.

Considering both subjects, accuracy alone is misleading due to pronounced class imbalance. Models such as LDA, Logistic Regression, and Random Forest, despite high accuracy, fail to generalize effectively to the target class. SVM with a Linear kernel emerges as the most suitable model, as it consistently achieves higher recall and F1-scores across subjects, indicating a better trade-off between detection performance and stability. Consequently, SVM (Linear) is recommended for this classification task.

Table 3: Classification Performance Comparison For Subject A

Model	Accuracy	Precision	Recall	F1-Score	ROC-AUC
LDA	83.43%	0.6364	0.0137	0.0269	0.6408
Logistic Regression	83.37%	0.5455	0.0118	0.0230	0.6406
SVM (RBF)	64.51%	0.2209	0.4471	0.2957	0.6063
SVM (Linear)	61.21%	0.2394	0.6098	0.3438	0.6395
Random Forest	83.33%	0.0000	0.0000	0.0000	0.5904
Gradient Boosting	63.27%	0.2316	0.5196	0.3204	0.6236

Table 4: Classification Performance Comparison For Subject B

Model	Accuracy	Precision	Recall	F1-Score	ROC-AUC
LDA	83.20%	0.2500	0.0039	0.0077	0.5915
Logistic Regression	83.24%	0.2857	0.0039	0.0077	0.5901
SVM (RBF)	66.08%	0.2124	0.3824	0.2731	0.5733
SVM (Linear)	57.75%	0.2076	0.5451	0.3007	0.5909
Random Forest	83.33%	0.0000	0.0000	0.0000	0.5449
Gradient Boosting	63.59%	0.2255	0.4863	0.3081	0.5952

8. Challenges faced

During the course of this project, several practical and conceptual challenges were encountered. Training time emerged as a significant limitation, as the computationally intensive learning process required full retraining even for minor modifications, thereby slowing iterative debugging and evaluation. In addition, code debugging proved to be nontrivial; identifying issues such as index out-of-bounds errors was particularly time-consuming due to their indirect manifestation during execution. Furthermore, substantial effort was required in learning and consolidating foundational concepts, as a strong theoretical understanding was essential before advanced methodologies could be effectively implemented. Collectively, these challenges influenced the development timeline and underscored the importance of efficient debugging strategies and solid conceptual grounding in complex computational projects.

References

Colab Notebook Link : https://colab.research.google.com/drive/1bvzbhB8NwxGjhPkbsl6qoqFLsJzjXQzt?usp=sharing
Scipy Documentation : https://docs.scipy.org/doc/scipy/
Scikit Learn Documentation : https://scikit-learn.org/0.21/documentation.html