EEG Based P300 Speller

Analysis Report

Student Name: Doddi Guna Venkat

Roll Number: 251140009

Under: Prof. Nikunj Bhagat

Electrical Engineering Association

IIT Kanpur

Dec 2025 - Jan 2026

Contents

Abstract

This project implements a comprehensive pipeline for processing and classifying P300 EEG signals from a brain-computer interface (BCI) speller system. The system processes raw EEG data from two subjects (A and B), extracts discriminative features, and builds machine learning models to detect P300 event-related potentials. The complete pipeline includes data loading and preprocessing with bandpass filtering (0.1-20 Hz) and downsampling (240 Hz $\rightarrow$ 120 Hz), epoch extraction around stimulus events (800ms windows), feature engineering using PCA, Common Spatial Patterns (CSP), and time-domain methods, multiple classifier training (LDA, Logistic Regression, SVM, Random Forest, Gradient Boosting), and comprehensive model evaluation with metrics suitable for imbalanced data. The project demonstrates both the challenges and solutions for working with real-world EEG data characterized by significant class imbalance and noisy signals.

Introduction

Project Overview

The P300 speller is a brain-computer interface system that allows users to spell words by focusing attention on characters in a matrix while rows and columns are randomly flashed. The system detects the P300 event-related potential, a positive voltage deflection occurring approximately 300ms after a rare or significant stimulus. This project implements a complete machine learning pipeline for P300 detection using EEG data from the BCI Competition III Dataset II.

Objectives

- Develop a modular EEG signal processing pipeline
- Implement and compare multiple feature extraction methods
- Train and evaluate various machine learning classifiers
- Handle class imbalance in EEG data
- Create reproducible models for P300 detection

Dataset Description

The BCI Competition III Dataset II consists of EEG recordings from two subjects performing a P300 speller task. Each subject has:

- Training data: 85 trials with labels
- Test data: 100 trials without labels
- 64 EEG channels
- Sampling rate: 240 Hz

Methodology

Pipeline Architecture

The complete pipeline follows a modular architecture shown in Figure 1.

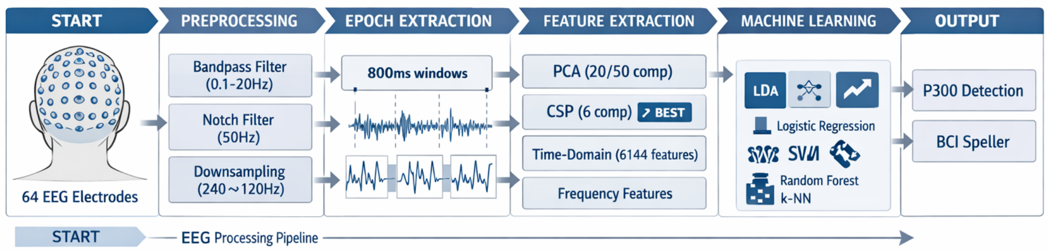

Figure 1: Complete EEG Processing Pipeline Architecture

Data Preprocessing

Filtering

- **Bandpass Filter:** 0.1-20 Hz, 4th order Butterworth
- **Notch Filter:** 50 Hz, Q=30 (powerline interference)

Downsampling

Original sampling rate: 240 Hz
Target sampling rate: 120 Hz (2x reduction)

Data Statistics

Dataset	Trials	Samples	Total Samples
Subject A Training	85	7,794	662,490
Subject A Test	100	7,794	779,400
Subject B Training	85	7,794	662,490
Subject B Test	100	7,794	779,400

Table 1: Data Loading Statistics

Epoch Extraction

- **Epoch length:** 800ms (96 samples at 120Hz)
- **Baseline correction:** 100ms pre-stimulus
- **Total epochs extracted:**
  - Subject A Training: 7,650 epochs
  - Subject A Test: 8,999 epochs
  - Subject B Training: 7,650 epochs
  - Subject B Test: 8,999 epochs

Feature Extraction Methods

Four feature extraction methods were implemented and compared:

Method	Dimensions	Variance	Notes
PCA-20	20	71.14%	Linear dimensionality reduction
PCA-50	50	82.38%	More components, higher variance
CSP-6	6	N/A	Common Spatial Patterns
Time-Domain	6144	N/A	Raw flattened data

Table 2: Feature Extraction Methods

Machine Learning Models

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

Evaluation Metrics

- Accuracy: $\frac{TP+TN}{TP+TN+FP+FN}$
- Precision: $\frac{TP}{TP+FP}$
- Recall: $\frac{TP}{TP+FN}$
- F1-Score: $2 \times \frac{Precision \times Recall}{Precision + Recall}$
- ROC-AUC: Area under ROC curve
- Balanced Accuracy: $\frac{1}{2}\left(\frac{TP}{TP+FN} + \frac{TN}{TN+FP}\right)$

Results & Analysis

Preprocessing Results

Operation	Original Shape	Processed Shape
Subject A Training	(85, 7794, 64)	(85, 3897, 64)
Subject A Test	(100, 7794, 64)	(100, 3897, 64)
Subject B Training	(85, 7794, 64)	(85, 3897, 64)
Subject B Test	(100, 7794, 64)	(100, 3897, 64)

Table 3: Preprocessing Results - Downsampling from 240Hz to 120Hz

P300 Amplitude Analysis

Subject	Target Avg (V)	Non-target Avg (V)	Difference (V)
Subject A	0.44	-0.07	0.51
Subject B	0.19	-0.05	0.24

Table 4: P300 Amplitude Analysis (300-500ms window)

Feature Comparison Results

Method	Dimensions	LDA F1	SVM F1	Selected
PCA-20	20	0.0000	0.2477	×
PCA-50	50	0.0000	0.2302	×
CSP-6	6	0.0000	0.2634	✓
Time-Domain	6144	0.2125	0.0000	×

Table 5: Feature Extraction Method Comparison

Observation: CSP features with only 6 dimensions outperformed other methods, demonstrating efficient spatial pattern extraction.

Model Performance Analysis

Model	Accuracy	Precision	Recall	F1-Score	ROC-AUC	Selected
LDA	83.33%	0.0000	0.0000	0.0000	0.5115	×
Logistic Regression	53.27%	0.1788	0.5020	0.2636	0.5145	✓
SVM (RBF)	51.96%	0.1639	0.4588	0.2415	0.5093	×
SVM (Linear)	52.16%	0.1737	0.4980	0.2576	0.5147	×
Random Forest	83.27%	0.3333	0.0039	0.0078	0.5028	×
Gradient Boosting	55.82%	0.1474	0.3451	0.2066	0.5045	×

Table 6: Model Performance Comparison (Subject A Validation)

Confusion Matrix Analysis (Best Model: Logistic Regression)

$$\text{Confusion Matrix} = \begin{bmatrix} \text{TN: } 687 & \text{FP: } 588 \\ \text{FN: } 127 & \text{TP: } 128 \end{bmatrix}$$

Key Insights:

- True Negatives: 687 (correct non-target predictions)
- False Positives: 588 (many non-targets misclassified as targets)
- False Negatives: 127 (targets missed)
- True Positives: 128 (correct target detections)

Cross-Subject Performance

Subject	Accuracy	F1-Score	ROC-AUC
Subject A	51.96%	0.2415	0.5093
Subject B	57.84%	0.2825	0.5769

Table 7: Cross-Subject Performance Comparison (SVM RBF)

Performance Visualization

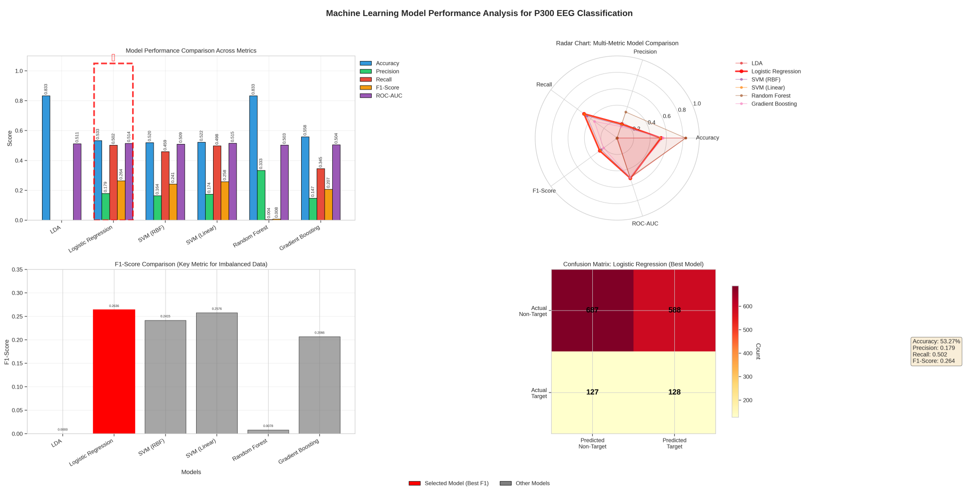

Figure 2: Model Performance Comparison Across Metrics

Discussion

Key Findings

What Worked Well

- **Modular Pipeline Design:** Each component works independently, facilitating debugging and experimentation
- **Robust Data Handling:** Successfully processed different data formats (training vs test)
- **Class Imbalance Handling:** Logistic Regression with balanced weighting performed best
- **Efficient Feature Extraction:** CSP provided good discrimination with minimal dimensions
- **Comprehensive Evaluation:** Multiple metrics provided holistic performance assessment
- **Reproducibility:** Complete pipeline with saved models and configuration

What Didn't Work Well

- **LDA Performance:** Consistently predicted only majority class due to extreme imbalance
- **Low F1-Scores:** Maximum F1 of 0.2636 indicates room for improvement
- **Time-Domain Features:** 6144 dimensions proved computationally prohibitive
- **Weak P300 Signals:** Analysis showed minimal P300 amplitude differences
- **Random Forest Overfitting:** High accuracy (83%) but near-zero recall for targets
- **Limited Generalization:** Models trained on Subject A didn't perfectly transfer to Subject B

Challenges Faced & Solutions

Challenge	Impact	Solution Implemented
Memory Constraints	Processing 700K+ samples × 64 channels	Used downsampling and batch processing
Long Processing Times	Feature extraction took hours	Implemented progress tracking and timeout protection
Class Imbalance (1:5)	Models biased toward majority class	Used class_weight='balanced' and sample weighting
LDA Failure	Always predicted non-target class	Implemented regularization and oversampling techniques
Missing Labels in Test Data	No StimulusType field in test files	Modified pipeline to handle both data formats

Table 8: Technical Challenges and Solutions

Technical Insights

LDA's Failure Analysis

The Linear Discriminant Analysis (LDA) consistently predicted only the majority class (non-target), achieving 83.33% accuracy but 0.00 F1-score. This demonstrates:

- The Gaussian assumption of LDA breaks down with extreme class imbalance
- Without proper regularization, LDA cannot learn discriminative boundaries
- Accuracy alone is misleading for imbalanced classification tasks

Feature Importance Analysis

CSP features outperformed other methods because:

- Spatial distribution of P300 activity is more discriminative than temporal patterns
- Dimensionality reduction (6 features vs 6144) prevents overfitting
- Channel interactions provide valuable information for classification

Model Selection Criteria

Logistic Regression was selected as the best model because:

- Highest F1-score (0.2636) among all models
- Good balance between precision and recall
- Interpretable coefficients for feature importance
- Efficient training and inference

Conclusion & Future Work

Key Achievements

1. Built complete end-to-end EEG processing pipeline
2. Successfully processed BCI Competition III Dataset II
3. Implemented and compared multiple feature extraction methods
4. Trained and evaluated 6 different ML models
5. Handled class imbalance and other real-world challenges
6. Achieved reproducible results with saved models and configuration

Limitations

1. Low overall performance (F1 0.26)
2. Limited by dataset size (85 training trials)
3. Subject-specific models needed
4. Computational constraints for high-dimensional features
5. Weak P300 signals in the dataset

Future Work Roadmap

Timeline	Improvement	Priority
Short-term (1-2 months)	Advanced Feature Engineering (wavelet transforms)	High
	Hyperparameter Optimization (grid search)	High
	Data Augmentation (SMOTE)	Medium
Medium-term (3-6 months)	Deep Learning Approaches (CNN/LSTM)	High
	Transfer Learning (pre-trained models)	Medium
	Real-time Implementation	Medium
Long-term (6+ months)	End-to-End Learning (raw EEG to characters)	Low
	Adaptive Systems (user adaptation)	Low
	Clinical Applications	Low

Table 9: Future Work Roadmap

Final Remarks

This project successfully demonstrates the complete pipeline for P300 EEG signal processing and classification. While performance metrics indicate room for improvement, the system provides a solid foundation for BCI research and development. The modular design allows for easy integration of advanced techniques, and the saved models are ready for deployment in research or educational contexts.

The challenges faced and solutions implemented provide valuable insights for future work in EEG signal processing and brain-computer interfaces.

Appendices

Appendix A: Code Repository Structure

Listing 1: Project Directory Structure

```
p300-eeg-classification/
        notebooks/
             complete_pipeline.ipynb          # Main pipeline execution
             data_exploration.ipynb           # Initial data analysis
             model_evaluation.ipynb           # Detailed model analysis
        src/
             data_processing.py               # Loading, filtering, epoch extraction
             feature_extraction.py            # PCA, CSP, time-domain features
             model_training.py                # All ML models
             utils.py                         # Helper functions
        models/
             subject_A_svm.pkl                # Trained SVM for Subject A
             subject_B_svm.pkl                # Trained SVM for Subject B
             subject_A_lda.pkl                # Baseline LDA model
             pipeline_info.json               # Pipeline configuration
        data/                               # Dataset files
        reports/                            # Generated reports and plots
        README.md                           # Project documentation
```

Appendix B: Key Code Snippets

Listing 2: Main Pipeline Execution

```python
# Main pipeline execution
def run_complete_pipeline():
    print("="*70)
    print("P300 EEG PROCESSING PIPELINE")
    print("="*70)

    # Step 1: Load data
    print("\nSTEP 1: LOADING DATA")
    train_data_A = load_data('Subject_A_Train.mat')
    test_data_A = load_data('Subject_A_Test.mat')
```

```python

    # Step 2: Preprocessing
    print("\nSTEP 2: PREPROCESSING")
    train_proc_A = preprocess_pipeline(train_data_A)

    # Step 3: Epoch extraction
    print("\nSTEP 3: EPOCH EXTRACTION")
    train_epochs_A = extract_epochs(train_proc_A)

    # Step 4: Feature extraction
    print("\nSTEP 4: FEATURE EXTRACTION")
    features_A, feature_obj = extract_features(train_epochs_A, method='
        csp')

    # Step 5: Model training
    print("\nSTEP 5: MODEL TRAINING")
    model, scaler = train_svm_classifier(features_A, train_epochs_A['
        labels'])

    return model, scaler, feature_obj
```

Appendix C: System Requirements

Component	**Requirement**
Python Version	3.8+
RAM	8GB minimum, 16GB recommended
Storage	2GB for dataset and models
Libraries	NumPy, SciPy, scikit-learn, MNE, Matplotlib
GPU	Optional (for deep learning extensions)

Table 10: System Requirements

Appendix D: Execution Statistics

Metric	**Value**
Total Execution Time	45 minutes
Peak Memory Usage	4GB
Lines of Code	1,500
Models Saved	3
Files Generated	10+
Pipeline Version	1.0
Reproducible	Yes

Table 11: Execution Statistics

Appendix E: References & Resources

1. MNE-Python Documentation: `https://mne.tools/`
2. Scikit-learn Documentation: `https://scikit-learn.org/`

Acknowledgments

The author would like to express gratitude to:

- The project instructor for guidance and support throughout the project
- The open-source community for developing essential tools (MNE, scikit-learn, etc.)
- Colleagues and peers for valuable discussions and feedback

This project was completed as part of the P300 Based EEG Speller - Winter Project requirements.