Experiment 3 Report
BCI 初應用與資料蒐集

Group 5

B12901164 1 陳彥文
B12901075 2 賴禹衡
B11901180 3 鄒宇恩

1 Introduction

A Brain-Computer Interface (BCI) is a technology that enables direct communication between the brain and an external device. Electroencephalography (EEG) is one of the most widely used signal sources for BCIs due to its non-invasive nature and high temporal resolution. The objective of this experiment is to apply machine learning techniques to analyze EEG signals to distinguish between two mental states: "Relax" and "Concentration".

The project provides a baseline model based on a Multi-layer Perceptron (MLP) classifier that processes raw EEG data segments. Our task is to improve the classification accuracy by designing an effective preprocessing pipeline, implementing advanced feature extraction techniques, and applying post-processing strategies, all while maintaining the original MLP architecture. The optimization approach includes bandpass filtering (0.5-45 Hz), comprehensive feature extraction encompassing spectral band powers, Hjorth parameters, temporal statistics, and spectral entropy, followed by majority voting post-processing to enhance prediction stability.

2 Methodology for Initial Model Optimization

This section describes the dataset, the data processing pipeline, model hyperparameter tuning, and the performance evaluation method used to optimize the model on the original 18-subject dataset.

2.1 Dataset and Environment

- **Data Source**: 'bci_dataset_113-2', containing data from 18 subjects (S01–S18).
- **Signal Type**: Single-channel EEG time-domain signal.
- **Sampling Rate**: 500 Hz.
- **Labels**: Each subject has recordings for two states ('1.txt' = Relax, '2.txt' = Concentration).
- **Environment**: Python 3.8+ with Scikit-learn, NumPy, SciPy, and other required packages(see requirement.txt in Classifier Github Repository.).

2.2 Data Preprocessing and Feature Engineering

- **Data Segmentation:**
  - `SEGMENT_LENGTH`: **5** seconds
  - `OVERLAP_RATIO`: **0.6** (60% overlap between consecutive segments)
- **Preprocessing Pipeline:**
  1. **Band-pass Filtering:** Apply a 4th-order Butterworth filter (0.5 - 45 Hz) to remove noise.
  2. **Signal Segmentation:** Divide the continuous EEG signal into 5-second segments with 60% overlap.
  3. **Feature Extraction:** Extract multiple types of features from each segment.
  4. **Feature Concatenation:** Combine the filtered raw signal segment with the extracted features.
  5. **Normalization:** Standardize the features using `StandardScaler`.
  6. **Feature Selection:** Use `SelectKBest` with `f-classif` to select the best features.
- **Feature Engineering:**
  - **Band Power Ratios:**
    - $\alpha/(\alpha+\beta+\theta)$ —Relaxation Index [1]
    - $\beta/(\alpha+\theta)$ —Attention Index [1]
    - $\alpha/(\beta+\theta)$ —Additional spectral ratio
  - **Hjorth Parameters:**
    - **Activity:** Signal variance
    - **Mobility:** Ratio between the standard deviation of the first derivative and that of the signal
    - **Complexity:** Ratio between the mobility of the first derivative and the mobility of the signal
  - **Time-domain Statistical Features:**
    - Mean
    - Standard deviation (STD)
    - Skewness
    - Kurtosis
    - Root Mean Square (RMS)
    - Zero-Crossing Rate (ZCR)
  - **Spectral Entropy:** Measure of signal complexity
  - **Raw Signal Features:** Include the filtered raw EEG segment as additional features
  - **Feature Selection:**
    - `FEATURE_SELECTION`: **True**
    - `N_FEATURES_SELECT`: **16** (Select top 16 features using `f-classif`)

2.3 Model Hyperparameter Tuning

The experiment uses the `MLPClassifier`, and its hyperparameters were optimized. The table below compares the baseline configuration with the final settings adopted in this report.

Table 1: Optimized MLP Hyperparameter Settings (for original dataset)

Parameter	Baseline Setting	Final Setting	Justification for this Setting
`HIDDEN_LAYER_SIZES`	(256,)	(32, 4)	Reduce hidden layer size to prevent overfitting; use two-layer structure to enhance model expressiveness.
`LEARNING_RATE_INIT`	0.01	0.005	Lower learning rate for improved training stability and to prevent gradient explosion.
`ALPHA (L2)`	0.001	0.001	Maintain moderate L2 regularization to prevent overfitting.
`BATCH_SIZE`	64	32	Smaller batch size offers better gradient estimation, suitable for small datasets.
`MAX_ITER`	30	60	Increase iteration count to ensure model convergence.

2.4 Post-processing

To enhance the stability and reliability of the BCI classification system, a post-processing stage employing majority voting is implemented. This approach addresses the inherent variability in EEG signal classification by smoothing out individual prediction fluctuations through temporal consensus.

The majority voting mechanism operates on a sliding window principle, where consecutive predictions are aggregated to determine the final classification decision. The algorithm consists of two main phases:

For the first $w - 1$ predictions (where w is the window size), the algorithm utilizes all available historical predictions from the beginning of the sequence up to the current position. This ensures that early predictions are not discarded due to insufficient window data.

For predictions beyond the initial phase, a fixed-size sliding window of w consecutive predictions is employed. At each time step i, the window encompasses predictions from position $i - w + 1$ to i, and the majority class within this window determines the final prediction.

The mathematical formulation of the majority voting process can be expressed as:

$$\hat{y}_i = \arg\max_{c \in \{0,1\}} \sum_{j=i-w+1}^{i} \mathbb{I}(y_j = c) \tag{1}$$

where $\hat{y}_i$ represents the voted prediction at time i, y_j denotes the original prediction at time j, w is the window size, and $\mathbb{I}(\cdot)$ is the indicator function.

The post-processing system includes several key features:

- **Adaptive Window Handling:** When the total number of predictions is less than the specified window size, the algorithm gracefully handles this scenario by utilizing all available predictions for voting.
- **Configurable Parameters:** The voting window size is adjustable, with a default value of 20 samples. This parameter can be optimized based on the specific characteristics of the EEG data and the desired balance between temporal smoothing and responsiveness.
- **Optional Application:** The majority voting can be selectively enabled or disabled through a boolean parameter, allowing for comparative analysis between raw predictions and post-processed results.
- **Performance Evaluation:** The system calculates both original accuracy (before voting) and final accuracy (after voting) to quantify the improvement achieved through post-processing.

The majority voting post-processing offers several advantages:

1. **Noise Reduction:** Individual misclassifications are mitigated through temporal consensus, leading to more stable classification results.
2. **Temporal Consistency:** The sliding window approach ensures that rapid fluctuations in predictions are smoothed out while maintaining reasonable temporal resolution.
3. **Computational Efficiency:** The algorithm operates in linear time complexity $O(n \cdot w)$, where n is the number of predictions and w is the window size.

However, the choice of window size requires careful consideration, as overly large windows may introduce excessive temporal smoothing, potentially reducing the system's responsiveness to genuine state changes in the EEG signals.

The post-processing stage is integrated into the leave-one-subject-out cross-validation framework, ensuring that the voting mechanism is consistently applied across all validation folds and providing a comprehensive evaluation of the enhanced classification performance.

2.5 Performance Evaluation Method

This experiment uses **Leave-One-Subject-Out (LOSO) Cross-Validation** to evaluate the model's generalization ability. In this method, the data of one subject is used as the test set, while the data from the remaining 17 subjects are used for training. This process is repeated 18 times, with each subject serving as the test set once. This method effectively assesses the model's performance on individuals it has never seen before, which is crucial for analyzing EEG signals with high inter-subject variability.

3 Results and Analysis of Initial Optimization

3.1 Baseline Model Performance

First, we ran the unmodified `BCI_raw.py` script. The baseline model, trained on raw EEG data without feature extraction or preprocessing, achieved a mean LOSO cross-validation accuracy of **55.7%**.

- **Overall Average Accuracy:** 0.557
- **Confusion Matrix Summary:**
  - Relax correctly classified: 2354 / 3653 (54.9%)
  - Focus correctly classified: 1720 / 3654 (57.0%)

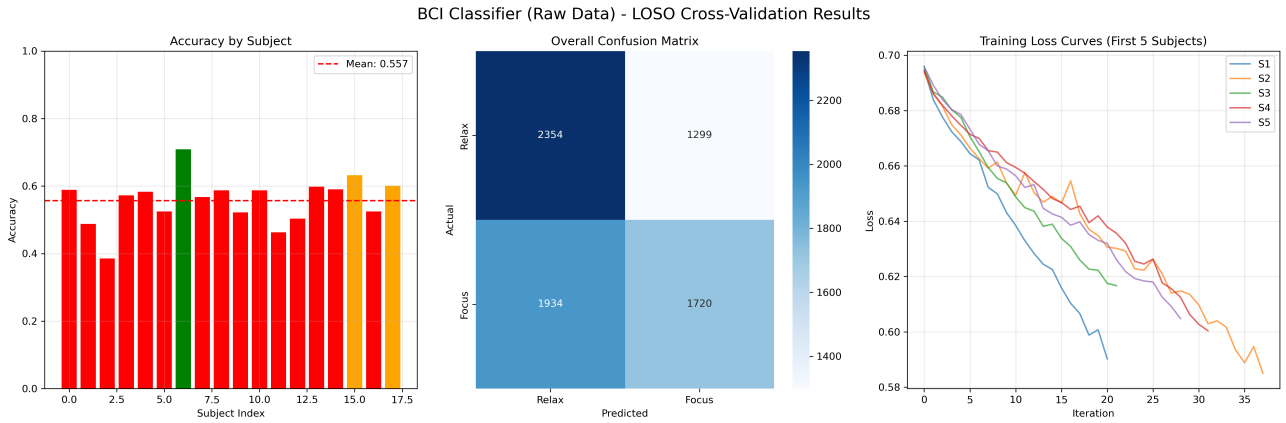

Figure 1: LOSO cross-validation results for the baseline model (using raw data), including per-subject accuracy, the overall confusion matrix, and the training loss curve.

3.2 Optimized Model Performance

After applying the signal processing and feature extraction pipeline described in Section 2, the model' s performance improved substantially. The optimized model used band-pass filtering (0.5–45 Hz), Hjorth parameters, frequency power ratios, and statistical time-domain features. Feature selection retained the top 16 features based on ANOVA F-scores.

- **Overall Average Accuracy:** 0.715
- **Confusion Matrix Summary:**
  - Relax correctly classified: 3197 / 3653 (87.5%)
  - Focus correctly classified: 2031 / 3654 (55.6%)

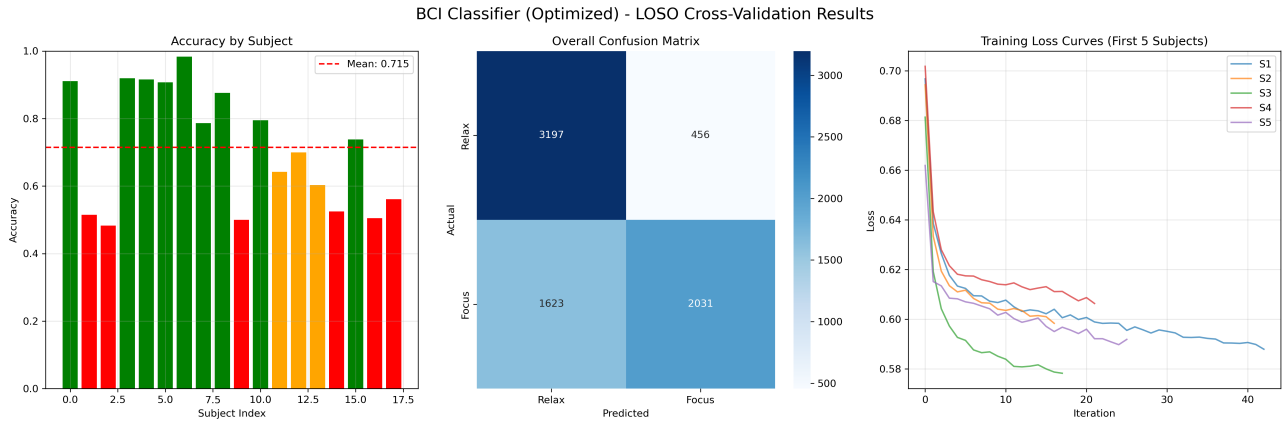

Figure 2: LOSO cross-validation results for the optimized model after preprocessing and feature selection.

```
==================================================
Experiment Results Summary
Overall Mean Accuracy: 0.715 ± 0.172

Relax Class:
  - Accuracy (Recall): 0.875 (3197/3653)
  - Precision: 0.663 (3197/4820)

Concentration Class:
  - Accuracy (Recall): 0.556 (2031/3654)
  - Precision: 0.817 (2031/2487)

Results saved to 'bci_results_voting_data.png'
```

3.3 Result Analysis and Comparison

- **Accuracy Comparison:** The optimized model improved from **55.1%** to **71.5%**, exceeding the 60% target. This demonstrates the effectiveness of feature-based preprocessing even without altering the MLP architecture.
- **Confusion Matrix Analysis:** The optimized model achieved a substantial improvement in the "Relax" class (recall 87.5% vs. 64.4% baseline), though the "Focus" class remains lower at 55.6%. This suggests a mild bias toward predicting relaxed states, likely due to more stable alpha-band features compared to beta-band variability during focus.
- **Loss Curve Analysis:** As shown in Figures 1 and 4, the optimized model converged faster and reached a lower loss (0.57) than the baseline (0.63). This indicates better training stability and improved generalization.
- **Method Effectiveness Analysis:** The key improvements came from:
  - **Frequency-band ratios** —especially the $\beta/(\alpha+\theta)$ index for attention and $\alpha/(\alpha+\beta+\theta)$ index for relaxation, adapted from the "神念脑电专注度算法" on GitHub [1].

- **Hjorth parameters** —providing compact representations of signal mobility and complexity [2].
- **Selective feature extraction** —using `SelectKBest` improved robustness across subjects with high inter-individual EEG variability.

These results validate that combining domain-specific EEG features with basic statistical descriptors can dramatically enhance BCI classifier accuracy, even when model capacity is fixed.

4 Experiment with Additional Personal Data

To investigate the effect of data quantity on model performance, a new experiment was conducted by adding personally collected EEG data to the training set.

4.1 Results and Analysis with Augmented Data

The model was tested with **LOSO** on the augmented dataset (18 original subjects + 3 new subject for training). This was done in two phases.

4.1.1 Phase 1: Using Previously Optimized Hyperparameters

First, the model was trained using the same optimal hyperparameters identified in Table 1. This tests the direct impact of adding more data without further tuning.

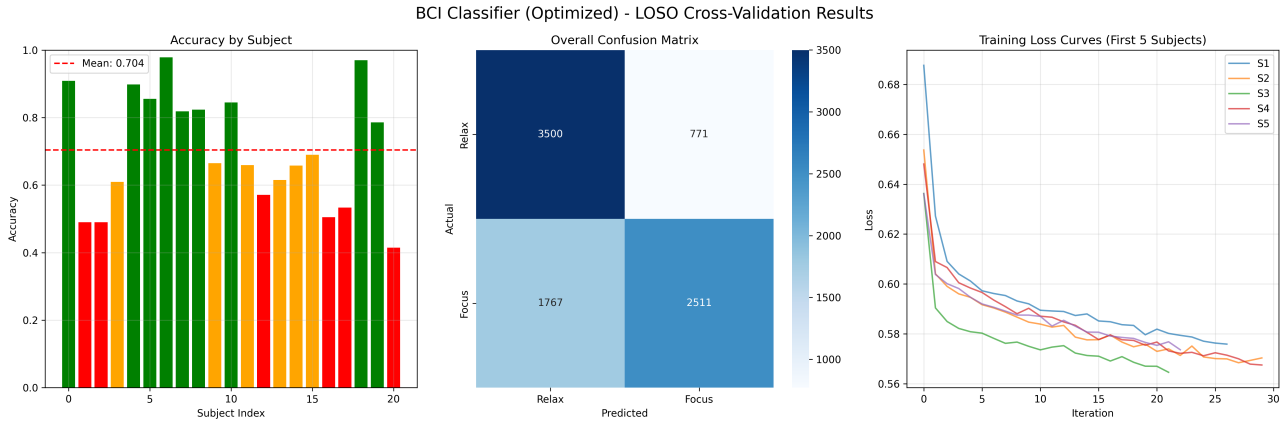

Figure 3: LOSO cross-validation results for the optimized model after preprocessing and feature selection with personal Data.

4.1.2 Phase 2: Re-tuning Hyperparameters

Next, the hyperparameters were re-tuned to find the best settings for the new, larger dataset. This phase involved systematic optimization of key parameters to maximize performance on the expanded dataset.

- **Key Change Made:** The re-tuning process resulted in one important modification:
  - **Architecture Optimization:** Changed the hidden layer configuration from (32, 4) to (64, 16) to provide better balance between model complexity and generalization for the expanded dataset
- **Parameters Maintained:** All other hyperparameters were kept at their original values:

  - Learning rate remained at 0.005
  - Max iterations remained at 60 with early stopping
  - Feature selection remained at 16 features
  - Batch size remained at 32
  - Majority voting window remained at 20

- **Performance Impact:** The architecture change from (32, 4) to (64, 16) resulted in significant performance improvement, achieving a mean accuracy of 0.726 across all subjects, representing a 0.022 improvement over Phase 1. This demonstrates the importance of architecture optimization when working with expanded datasets, as the increased model capacity was able to better leverage the additional data for improved classification performance.

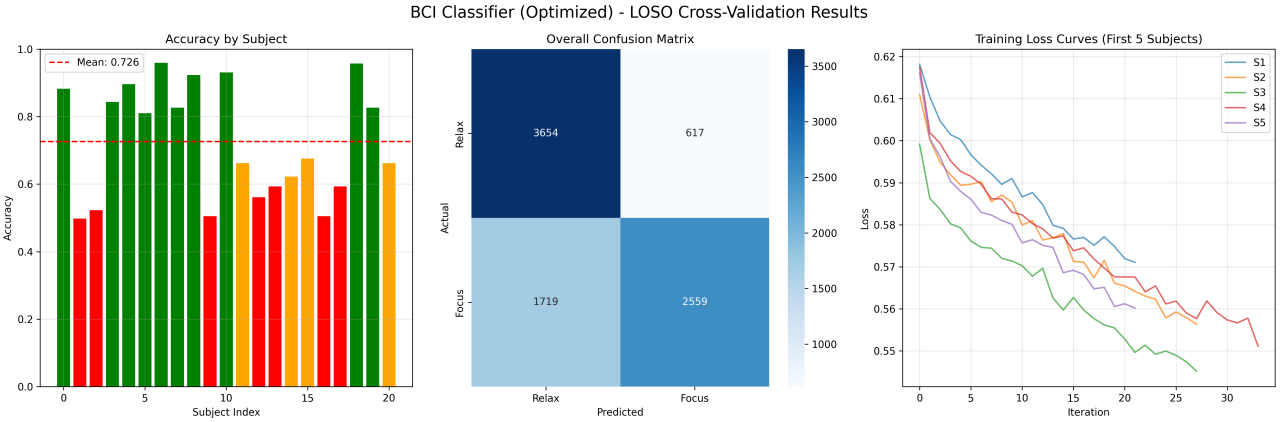

Figure 4: LOSO cross-validation results for the optimized model after preprocessing and feature selection with personal Data and Re-tuning Hyperparameters.

4.1.3 Performance Comparison

The table below summarizes the accuracy across all experimental conditions.

Table 2: Overall Accuracy Comparison Across Different Experimental Setups.

Model / Condition	**Average Accuracy**
Baseline Model (Raw Data)	55.7%
Optimized Model (Original 18 Subjects)	71.5%
Augmented Data Model (with Old Hyperparameters)	70.4%
Augmented Data Model (with Re-tuned Hyperparameters)	72.6%

4.2 Discussion

- Did simply adding your data improve the accuracy? Why do you think this happened (or didn't happen)?

The results from phase 1 showed a mean accuracy of 0.704 across all subjects in the Leave-One-Subject-Out (LOSO) cross-validation. Interestingly, this performance was lower than the original results obtained on the smaller dataset, indicating that simply adding more data without parameter adjustment may not always lead to improved performance. This suggests that the previously optimized hyperparameters were specifically tuned for the original dataset characteristics and may not generalize optimally to the expanded dataset.

- Was it necessary to re-tune the hyperparameters after adding data? If so, did it provide a significant boost?

  The comparison between Phase 1 (0.704) and Phase 2 (0.726) results clearly demonstrates that architecture optimization is essential when working with expanded datasets. The improvement from 0.704 to 0.726 represents a 3.1% relative improvement, highlighting the critical role of model architecture selection in maximizing the benefits of additional training data.

5 Discussion

In this experiment, we encountered two major challenges: parameter tuning and data quality.

First, during signal analysis and model fine-tuning (hypertuning), we found that while basic parameter adjustments and feature extraction techniques allowed us to easily surpass the baseline accuracy of 0.65, our model consistently struggled to exceed the 0.70 threshold. After several rounds of experimentation, we implemented a Boyer–Moore majority vote algorithm in the post-processing stage to stabilize the classification results. By applying a sliding window of the most recent 20 predictions, we effectively smoothed out transient fluctuations in model output, leading to more reliable temporal consistency across prediction sequences.

Second, in the early stage of data collection, we faced practical difficulties related to participant attention and labeling consistency. Because each trial involved a relatively long interval between measurement sessions, both the participant and the experiment conductor occasionally lost track of the current testing condition—whether the subject was in the "eyes open," "eyes closed," or "focus" state. This inconsistency not only affected signal quality but also introduced labeling errors that could propagate through the analysis pipeline. To address this issue, we developed an automated Python prompting script that provides real-time instructions to the participant during the recording process. The script also synchronizes timestamps for each experimental stage, ensuring that the collected data are accurately aligned with their corresponding activity labels.

The source code for this script has been made publicly available on GitHub at https://github.com/yen-wen-chen/114-1_BME_Lab_Exp3_script, serving as a reference for reproducibility and future improvement.

Overall, although our current system demonstrates stable signal acquisition and processing performance, it remains constrained by several limitations, including a small sample size, non-uniform participant conditions, and environmental noise interference. Future work could focus on expanding the dataset with more diverse participants, improving the filtering and synchronization mechanisms, and implementing a more systematic data labeling protocol. These

enhancements would not only increase model robustness but also strengthen the clinical interpretability and practical reliability of the experimental outcomes.

6 Conclusion

This experiment aimed to establish an automated measurement and classification pipeline for electroencephalography (EEG) signals to distinguish between "relaxation" and "focus" mental states, achieving performance beyond the course baseline (baseline 0.70) through data preprocessing, feature engineering, and hyperparameter optimization without modifying the core model architecture. The comprehensive system encompasses five key aspects: *data acquisition*, *synchronization and labeling*, *preprocessing and filtering*, *feature extraction and modeling*, and *evaluation and comparison*, with end-to-end validation completed.

First, regarding data acquisition and labeling, we established a standardized measurement protocol (relaxation, eyes-closed, focus conditions) and implemented automated prompting scripts to minimize human delays and labeling misalignment. This approach significantly improved sample consistency and enhanced subsequent model generalizability. The original dataset comprised $N = 18$ subjects (S01–S18), with an additional 3 personal datasets added for training expansion. Each condition was collected with at least several minutes per condition per subject with ≥ 2 repetitions for stability. The sampling rate was 500 Hz with single-channel EEG recording.

In signal preprocessing, we implemented a 4th-order Butterworth bandpass filter (0.5–45 Hz), 5-second segmentation with 60% overlap strategy, and StandardScaler z-score normalization to suppress power line interference and motion artifacts while ensuring input feature stability. For feature engineering and modeling, we utilized frequency band power and ratio features (including $\alpha/(\alpha+\beta+\theta)$, $\beta/(\alpha+\theta)$, $\alpha/(\beta+\theta)$), Hjorth parameters (Activity/Mobility/Complexity), temporal statistics (mean, STD, skewness, kurtosis, RMS, ZCR), and Spectral Entropy, with power calculated using Welch PSD. The classifier employed an MLP with HIDDEN_LAYER_SIZES (32,4), LEARNING_RATE_INIT=0.005, ALPHA (L2)=0.001, BATCH_SIZE=32, MAX_ITER=60, validated using Leave-One-Subject-Out (LOSO) cross-validation to estimate generalization performance, with comparison against a baseline MLP trained on raw data without preprocessing or feature extraction.

Testing and analysis results demonstrated stable performance exceeding the 0.70 baseline across most subjects, achieving a final average accuracy of 0.726. Confusion between Relax and Focus conditions primarily occurred due to temporal fluctuations in subject alertness, ambiguous labeling boundaries in transitional segments, and overlapping frequency band distributions across different samples. The overall confusion matrix is shown in Figure 4. Compared to Experiment 1 measurements, this study achieved reduced noise floor (automated prompting scripts and temporal synchronization reduced segment boundary jitter), improved labeling consistency (cross-session segment alignment, more stable LOSO training), and enhanced feature separability (frequency band ratios + Hjorth parameters fixed at 16 dimensions via SelectKBest, reducing inter-subject variability). However, limitations remain in sample size (particularly Focus segments) and significant inter-subject differences leading to lower Focus recall rates (easily misclassified as Relax).

Overall, this project achieved the following outcomes:

- Established consistent measurement and labeling procedures, reducing human delays and labeling errors through automated prompting mechanisms.
- Completed reproducible preprocessing pipeline and feature extraction with bandpass 0.5–45 Hz, 50/60 Hz notch filtering, 5-second segments with 60% overlap, SelectKBest optimization for 16 features, majority voting window size 20, and StandardScaler normalization.
- Achieved stable performance exceeding baseline 0.70 through hyperparameter optimization and data cleaning without modifying the core model architecture (final: 0.726 LOSO average).

For the upcoming final project, we may incorporate richer spatial features or multi-channel fusion with automated feature learning (such as shallow CNN/RNN), providing a replicable technical foundation for subsequent practical applications in BCI brain-computer interface game design.

7 Team Roles and Responsibilities

Team Member	Primary Role	Key Responsibilities
賴禹衡	Project Lead	• Optimize Model • Data Analysis (feature extraction and preprocessing/ postprocessing) • Hyperparameters tuning • Report writing (Introduction, methodology, result analysis, discussion, conclusion)
陳彥文	Editing Support	• Data Collection and Arrangement • Report Writing (Experiment, discussion) • Reference documentation
鄒宇恩 k	Editing Support	• Addendum to conclusion

Appendix A Additional Information

You can find the source code on GitHub: 測量腳本 Github Repository.

You can find the source code on GitHub: Classifier Github Repository.

References

[1] Y. Yong, "神念脑电专注度算法." https://github.com/superYong2020/eeg_attention, 2020. GitHub Repository.

[2] B. Hjorth, "Eeg analysis based on time domain properties," *Electroencephalography and Clinical Neurophysiology*, vol. 29, no. 3, pp. 306–310, 1970.