SOFTWARE ENGINEERING PROJECT

Prestimulus EEG-Based Behavioral Prediction in the HBN-EEG Dataset

BY

Nantawat Suksirisunt
Naytitorn Chaovirachot

DEPARTMENT OF COMPUTER ENGINEERING
FACULTY OF ENGINEERING
KASETSART UNIVERSITY

Academic Year 2565

Prestimulus EEG-Based Behavioral Prediction in the HBN-EEG Dataset

BY

Nantawat Suksirisunt
Naytitorn Chaovirachot

This Project Submitted in Partial Fulfillment of the Requirement for Bachelor Degree of Engineering (Software Engineering)
Department of Computer Engineering, Faculty of Engineering KASERTSART UNIVERSITY
Academic Year 2565

Approved By:

Advisor . . . . . **Date** . . . . . .
(Asst. Prof.Dr. Thanawin Rakthanmanon)

Co-Advisor . . . . . **Date** . . . . . .
(Assoc. Prof. Dr. Theerawit Wilaiprasitporn)

Head of Department . . . . . **Date** . . . . . .
(Assoc. Prof.Dr. Punpiti Piamsa-Nga)

Abstract

Electroencephalography (EEG) research commonly relies on interactive notebook environments for data inspection and experimentation. While flexible, these workflows introduce hidden execution states, environment dependency, and limited reproducibility, making collaboration and systematic experimentation difficult. Furthermore, large-scale EEG datasets require significant computational resources that are not always available on user machines.

This project develops a pipeline for prestimulus EEG-based behavioral prediction on the Healthy Brain Network EEG (HBN-EEG) dataset. We focus on the Contrast Change Detection (CCD) task and use the 2-second inter-trial interval (ITI) segment as a constrained prestimulus window for modeling.

The proposed approach extracts steady-state and spectral features from CCD-ITI epochs after standard preprocessing (e.g., bandpass filtering, artifact mitigation, and normalization). We train lightweight models to predict reaction time (RT) and trial-level accuracy as regression targets, and evaluate performance using regression metrics (e.g., MAE/MSE).

To support reproducible experimentation, the system includes a lightweight web interface for configuring preprocessing and running analyses without relying on a full notebook environment or heavy local machine learning installations.

Acknowledgement

Put your acknowledgement paragraph here.

Nantawat Suksirisunt
Naytitorn Chaovirachot

Table of Contents

List of Tables

Page

List of Figures

Page

Chapter 1

Introduction

1.1 Background

This project aligns with the EEG Foundation Challenge (EEG2025), which calls for models that generalize across tasks and subjects using large-scale, high-density EEG. The Healthy Brain Network EEG (HBN-EEG) dataset provides 128-channel recordings across six tasks, with FAIR, BIDS, and HED annotations that enable reproducible analyses. The competition emphasizes zero-shot transfer to new tasks and subjects, and prediction of latent psychopathology factors.[1]

1.2 Problem Statement

Under the challenge's constraints, the test phase exposes only limited CCD segments (e.g., ITI), making behavior prediction difficult without full event-locked trials. We aim to establish a constrained approach using the 2-second CCD ITI to predict reaction time (RT) and accuracy.

1.3 Project Aim

Our goal is to build a principled framework that:

- Verifies SSVEP and P300 presence with a reproducible preprocessing pipeline (SNR spectra and evoked responses),
- Predicts CCD RT and accuracy using features derived strictly from ITI segments,

[1] See [1, 2, 3].

- Establishes a clear, reproducible approach under the competition constraints.

1.4 Terminology

- **SSVEP:** Steady-State Visual Evoked Potential, frequency-locked EEG response to periodic stimulation.
- **P300:** Positive ERP component around $\tilde{3}00$ ms after salient events, associated with attention and context updating.
- **ITI:** Inter-Trial Interval, rest period between trials; in CCD, flickering gratings persist during ITI.
- **BIDS/HED:** Data and event annotation standards supporting consistent, analysis-ready datasets.

Chapter 2

Literature Review and Related Work

2.1 Challenge and Dataset Context

The EEG Foundation Challenge proposes two tracks: (1) zero-shot decoding across new tasks and subjects, and (2) prediction of psychopathology factors from EEG. It leverages an unprecedented high-density, multi-task HBN-EEG dataset formatted in BIDS with HED annotations.[1]

2.2 HBN-EEG Resources

- **HBN-EEG FAIR Implementation (2024):** Presents analysis-ready EEG with integrated behavioral events and HED annotations, enabling reproducible analyses.[2]

- **Transdiagnostic Resource (2017):** Describes the HBN biobank's multimodal, large-scale and community-sampled design to support dimensional (transdiagnostic) research.[3]

- **EEG + Eye Tracking Dataset (2017):** Provides active and passive paradigms, including steady-state and contrast decision tasks, supporting developmental brain investigations.[4]

[1] [1, 2].
[2] [2].
[3] [3].
[4] [4].

2.3 Positioning of This Work

Our study focuses on a constrained but practical scenario: prestimulus behavior prediction (RT, accuracy) from CCD ITI-only segments in HBN-EEG. This targets the challenge setting of limited test data while keeping the methodology simple and reproducible using FAIR/BIDS/HED resources.

Chapter 3

Objectives and Method Overview

3.1 Research Objectives

- **CCD-ITI approach:** Predict reaction time (RT) and accuracy as regression targets using features derived strictly from the 2-second CCD ITI.
- **Verification:** Confirm presence of SSVEP and P300 components via SNR spectra and evoked response plots.

3.2 Dataset and Tasks

HBN-EEG provides six tasks: Resting State, Surround Suppression (SuS), Movie Watching, Sequence Learning, Contrast Change Detection (CCD), and Symbol Search. CCD includes periodic stimulation during ITI, yielding steady-state signals suitable for modeling under limited test segments.[1]

3.3 Preprocessing and Features

- **Preprocessing:** Bandpass filtering, epoching (CCD ITI: -2 to 0 s), artifact mitigation, and normalization.
- **Verification:** Compute SNR spectra at stimulation frequencies; plot evoked responses to confirm SSVEP/P300.
- **Features:** ITI-derived steady-state features (spectral power, harmonics, topographies) for RT/accuracy prediction.

[1] [2, 4].

Chapter 4

Software Architecture Design

4.1 Domain Model

The architecture reflects a research workflow tailored to the EEG2025 challenge: behavioral prediction using CCD-ITI segments. Components:

- **DataLoader:** Ingests BIDS/HED-formatted HBN-EEG and parses CCD events and metadata.[1]
- **PreprocessingPipeline:** Bandpass filters, epochs CCD-ITI, artifact mitigation, normalization.
- **PredictionModel:** Maps ITI-derived steady-state features to RT and accuracy as regression targets.
- **Evaluator:** Computes regression metrics (e.g., MAE/MSE) and produces SNR/ERP verification plots.
- **Visualizer:** Summarizes metrics and renders spectra/evoked responses.

4.2 Training and Evaluation Flow

Scenario: Workflow

1. Load CCD-ITI segments; preprocess and verify steady-state/evoked responses.
2. Train the model on CCD-ITI to predict RT/accuracy.
3. Evaluate metrics and visualize spectra/evoked responses.

[1] [2].

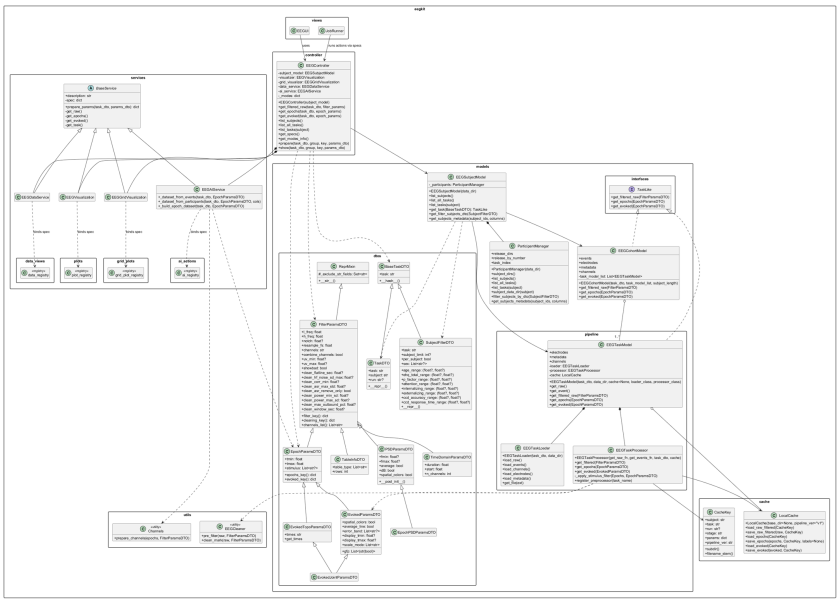

Figure 4.1: System architecture overview

4.3 Method Notes

- **Features:** Spectral power at stimulation frequencies and harmonics; topographic distributions.
- **Metrics:** Regression metrics (e.g., MAE/MSE) for RT and accuracy.
- **Plots:** SNR spectra and evoked response overlays for CCD.

Chapter 5

AI Component Design

5.1 Business Context and AI Integration

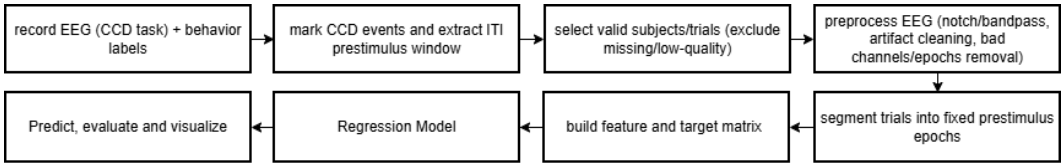

Figure 5.1: Workflow for prestimulus EEG-based behavioral prediction from CCD-ITI segments.

Diagram Explanation: The system is designed as an experimental EEG research platform integrating signal visualization, dataset inspection, and machine learning experimentation into a unified workflow. The frontend interface allows users to explore EEG signals, construct datasets, and execute AI models, while the backend handles preprocessing, model execution, and evaluation.

The workflow proceeds as:

signal inspection $\rightarrow$ dataset verification $\rightarrow$ model training $\rightarrow$ evaluation

This ensures that model results are directly connected to verified signal characteristics rather than blindly training on raw data.

Why is AI suitable for this problem?

EEG analysis involves high-dimensional temporal signals with complex spatial relationships across electrodes. Manual feature engineering is difficult because:

- patterns vary across subjects
- noise and artifacts are common
- signal distributions change across recording sessions

Machine learning models can learn robust mappings from EEG-derived features to behavioral targets more effectively than fixed heuristics.

Is the problem large, complex, or always changing?

Yes.

EEG data presents three major challenges:

- Large — continuous multi-channel time series
- Complex — spatial-temporal correlations
- Non-stationary — signals differ between subjects and sessions

Because of this variability, deterministic rule-based systems are unsuitable.

Can we accept an answer that's not 100% perfect?

Yes. This is a research system.

The system is intended for research experimentation rather than safety-critical decision making. Model predictions are used as analytical indicators, not medical diagnoses. Therefore approximate predictions with measurable error are acceptable.

5.2 Goal Hierarchy

Organizational Goal: Enable efficient experimentation in EEG machine learning research

System Goal: Provide an interactive platform that links signal inspection with machine learning experimentation

User Goal: Explore EEG signals, construct datasets, train models, and compare results without manual scripting

AI Model Goal: Learn representations that generalize across subjects and improve classification performance

Success Metrics:

- reduced prediction error (e.g., MAE/MSE) for RT and accuracy
- consistent performance on unseen subjects
- reduced experiment setup time
- reproducible experiment workflow

5.3 Task Requirements Analysis Using AI Canvas

5.3.1 AI Task Requirements

- **Requirements (REQ):** Predict behavioral outcomes from prestimulus CCD-ITI EEG segments in HBN-EEG.
- **Specifications (SPEC):** Evaluate models that map CCD-ITI features to reaction time (RT) and trial-level accuracy as regression targets.
- **Environment (ENV):** Evaluated on multi-subject EEG datasets with different recording conditions. (e.g., noise, subject variation).

5.3.2 AI Canvas Summary

- **Input:** Preprocessed EEG segments
- **Output:** Predicted RT and trial-level accuracy (regression)
- **Success Criteria:** Lower prediction error with stable generalization to unseen subjects.

5.3.3 Innovation

The system integrates visualization-driven dataset validation with model training inside a single interactive interface, reducing mismatch between inspected signals and training data.

5.4 User Experience Design with AI

The platform follows a research-oriented workflow where users first inspect signals before executing machine learning models. To support this process, the interface uses a **mode-action** structure.

Interface Overview: The interface contains four primary operational modes:

- **Plot Mode:** — detailed single-view signal inspection
- **Grid Plot Mode:** — comparative multi-condition visualization
- **Data Mode:** — structured data inspection and preparation
- **AI Mode:** — machine learning experimentation

Each mode exposes only relevant actions.

Visualization and Inspection Modes

5.4.1 Plot Mode

Provides detailed EEG inspection including sensor layout, time-domain plots, frequency plots, epoch plots, evoked responses, and SNR analysis.

Available actions include:

- sensor layout visualization
- time-domain signal plots
- frequency-domain plots
- epoch visualization
- evoked response plots
- evoked topography plots
- SNR spectrum analysis

5.4.2 Grid Plot Mode

Allows comparison across conditions using PSD, SNR, and evoked grids.

Available actions include:

- PSD Grid
- SNR Grid
- Evoked Grid

5.4.3 Data Mode

Displays structured EEG data tables to confirm dataset composition.

Available actions include:

- EEG sample tables
- epoch tables
- metadata

5.4.4 AI Interaction Mode

AI Mode enables machine learning experimentation using the prepared EEG data.

Available actions include:

- Training
- Prediction
- Evaluation
- Model Comparison

System Behavior: Upon user interaction, the backend automatically executes preprocessing, training, evaluation, and result generation. Logs, graphs, and final metrics are dynamically updated and available for export.

Feedback Loop: inspect → adjust → train → evaluate → compare → refine

Researchers can rapidly modify dataset selections and rerun experiments without rewriting scripts or managing computing environments. This significantly shortens the experimental cycle compared to notebook-based pipelines.

Interface Screenshots:

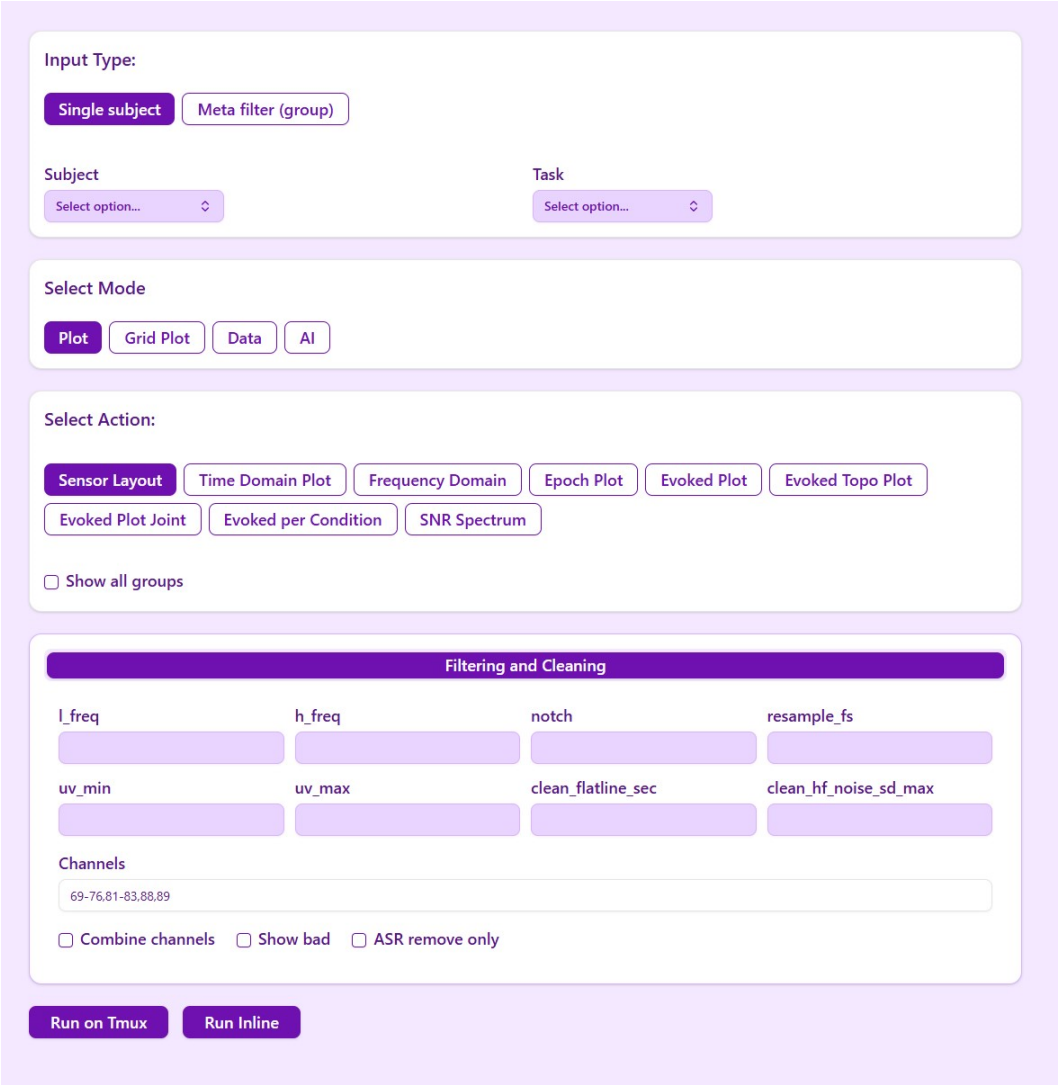

Figure 5.2: React Notebook Interface – Model visualization and execution

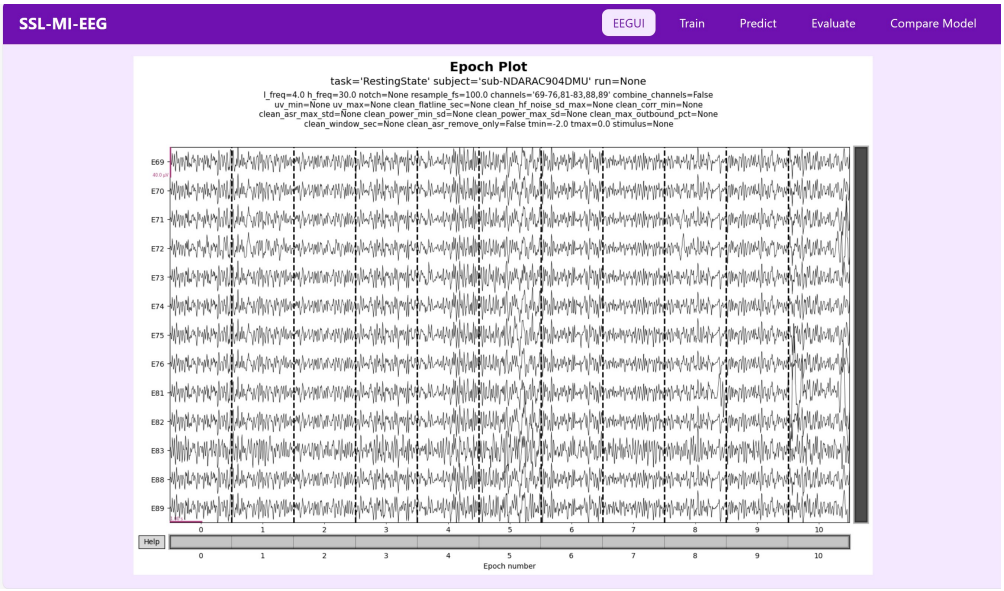

Figure 5.3: Epoch visualization

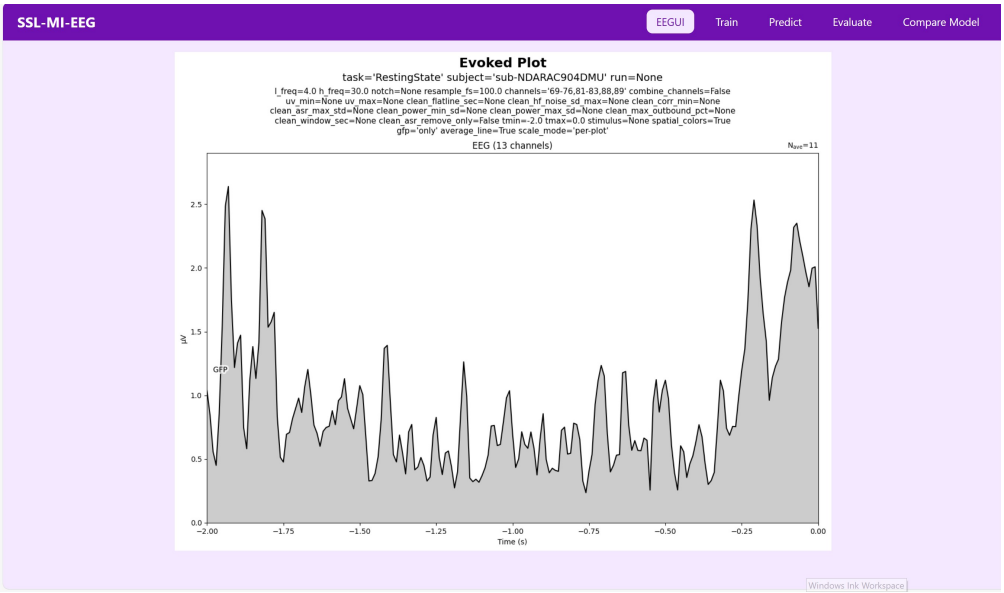

Figure 5.4: Evoked response

Figure 5.5: Evoked grid view

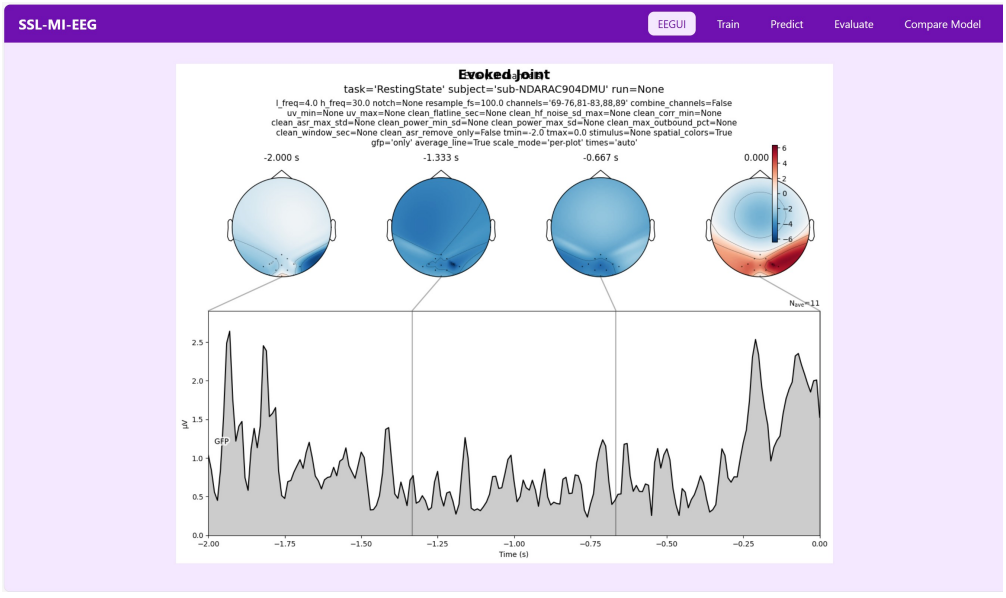

Figure 5.6: Evoked joint plot

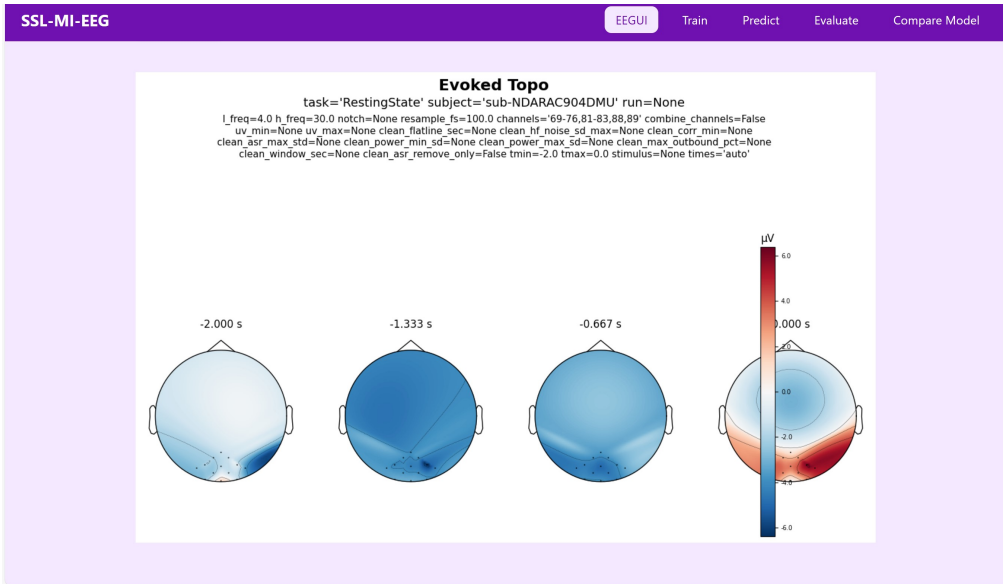

Figure 5.7: Evoked topomap

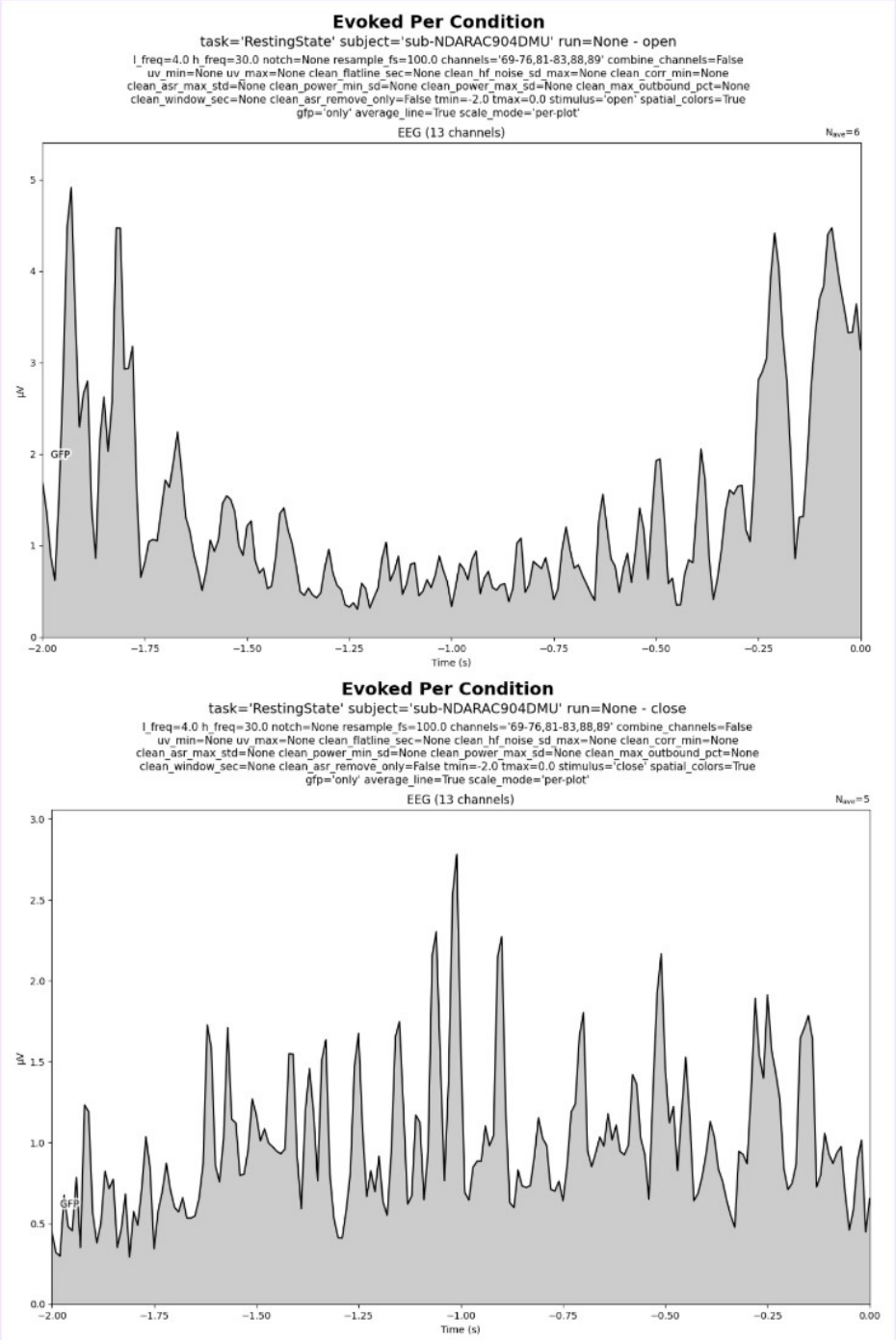

Figure 5.8: Evoked per condition

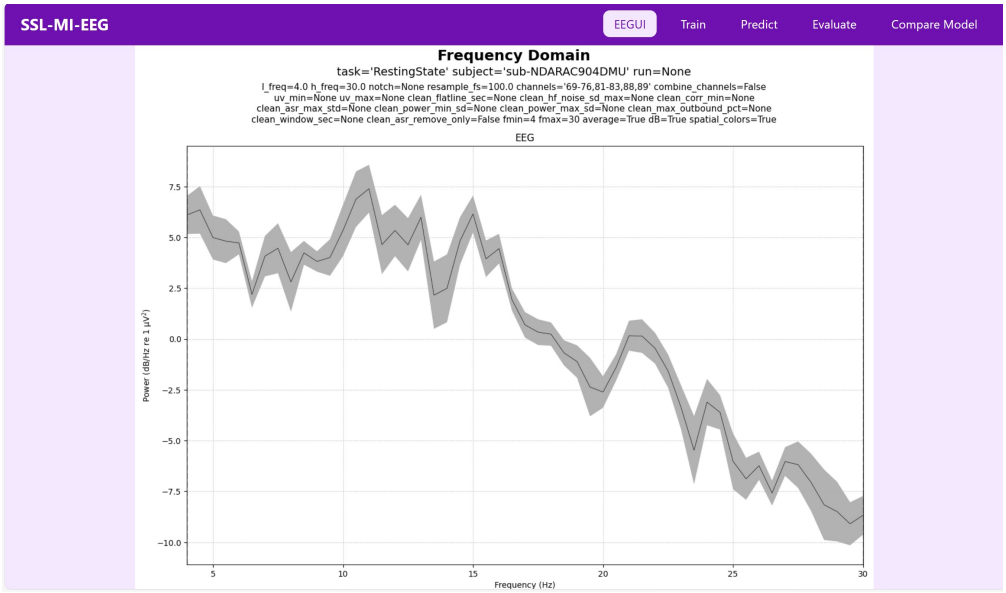

Figure 5.9: Frequency spectrum

Figure 5.10: Power spectral density grid

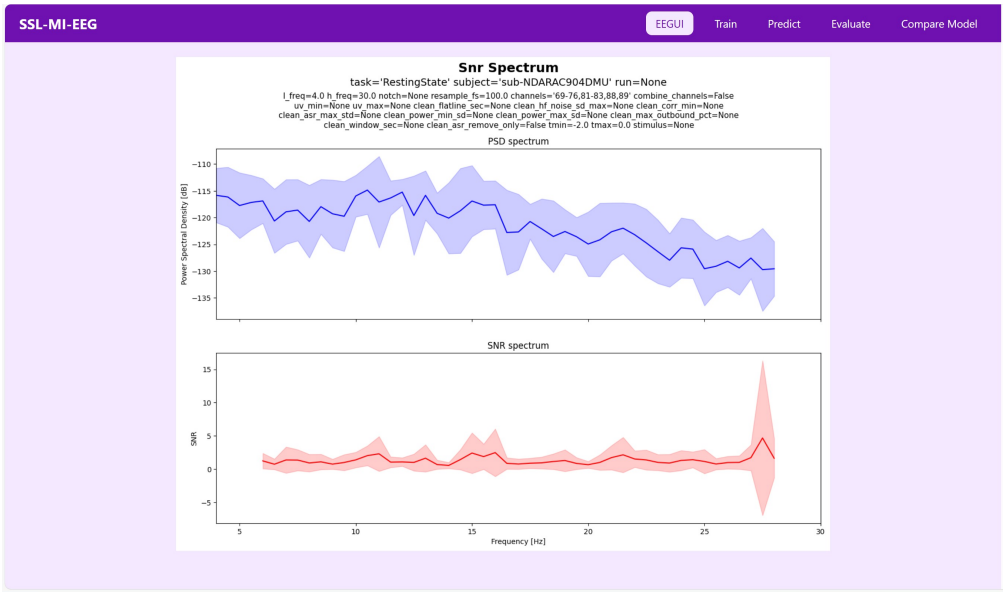

Figure 5.11: Signal-to-noise ratio spectrum

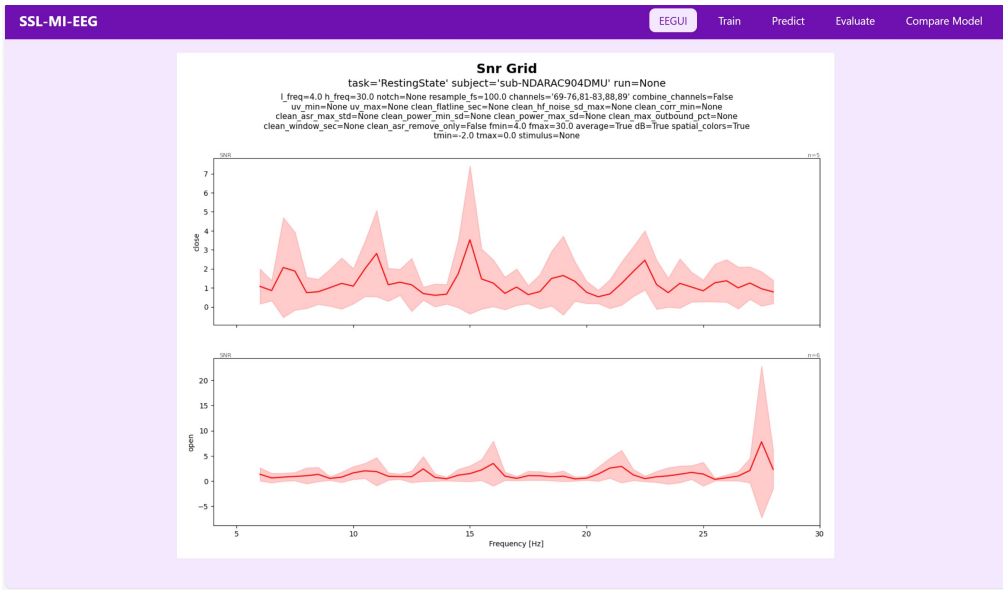

Figure 5.12: Signal-to-noise ratio grid

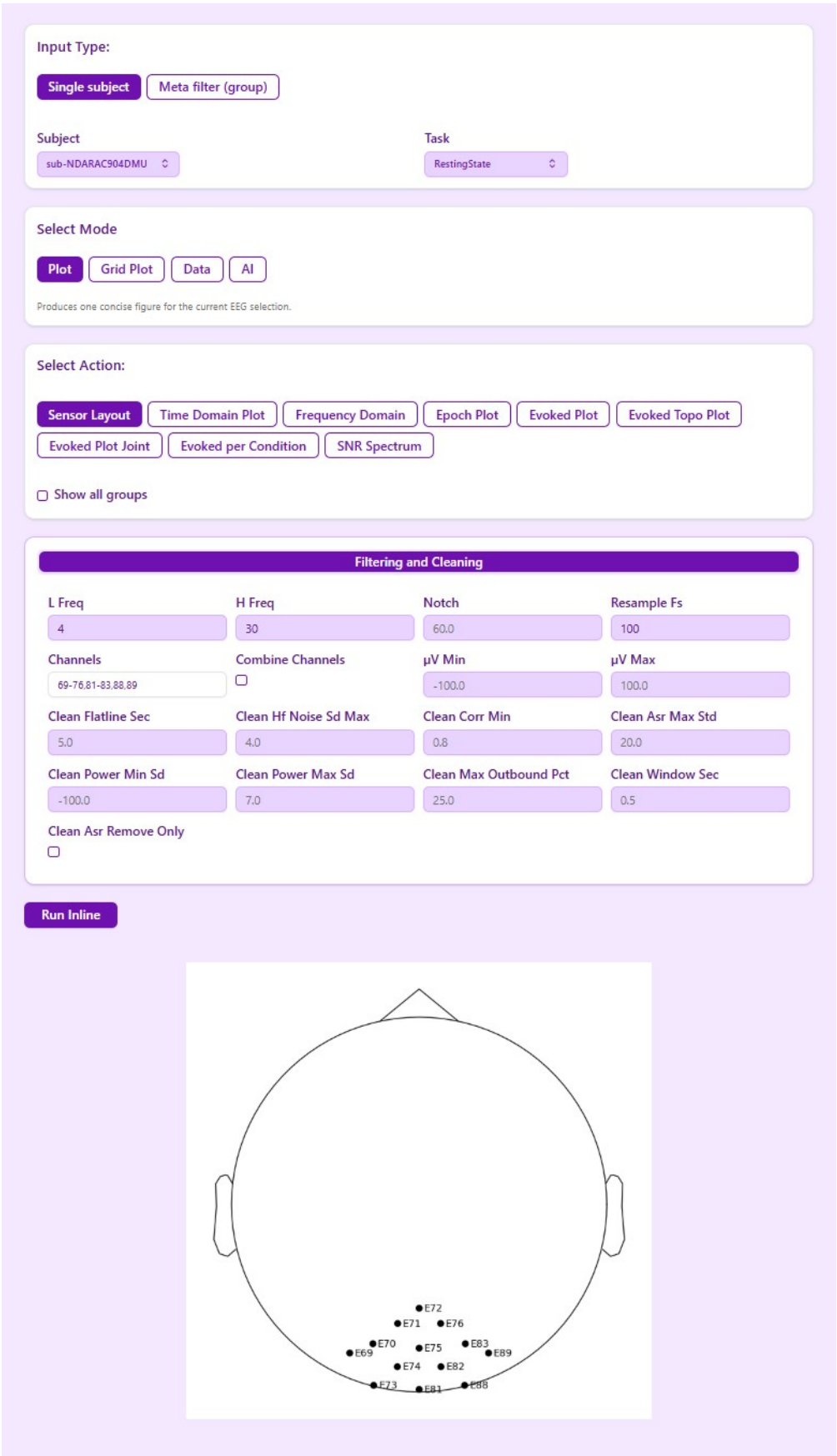

Figure 5.13: Sensor layout

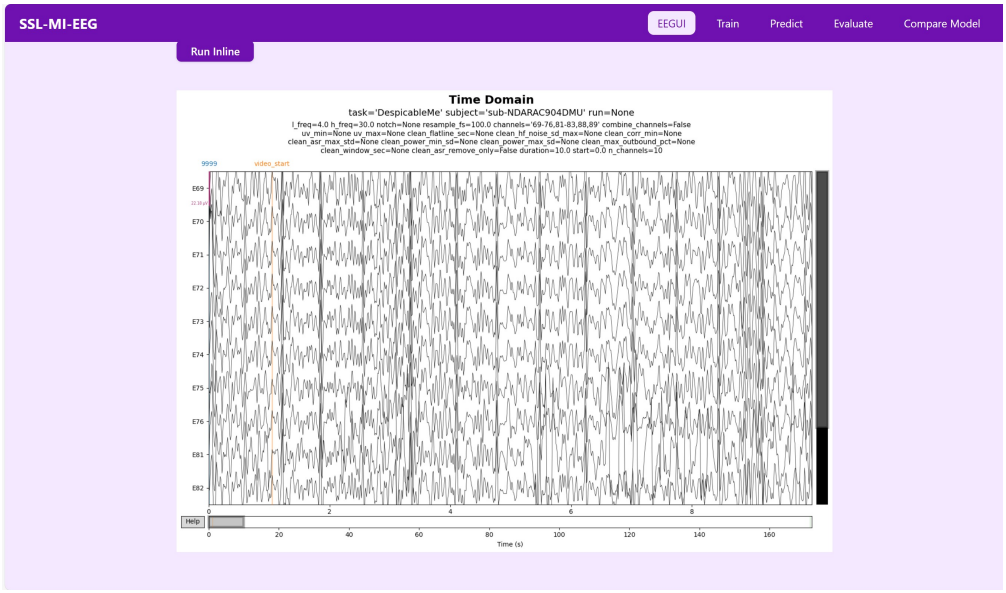

Figure 5.14: Time-domain signal

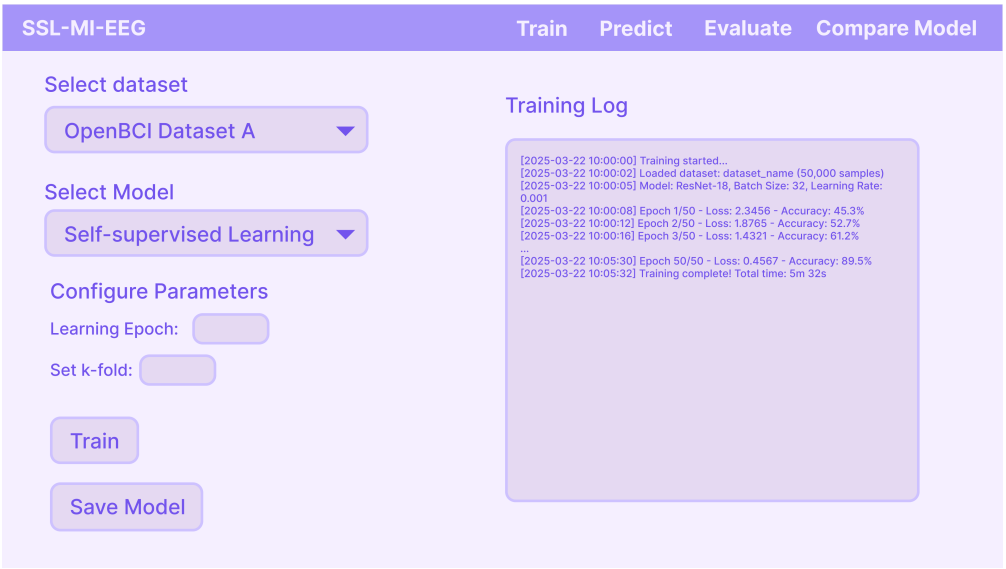

Figure 5.15: Training Interface – Dataset selection, model configuration, and training log

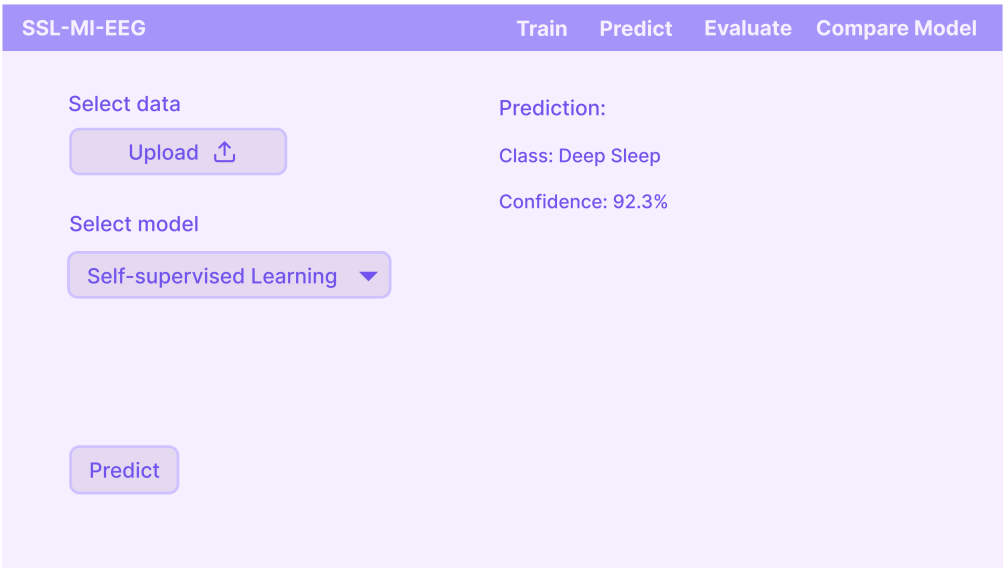

Figure 5.16: Prediction Interface – Uploading input data and viewing predicted RT and accuracy

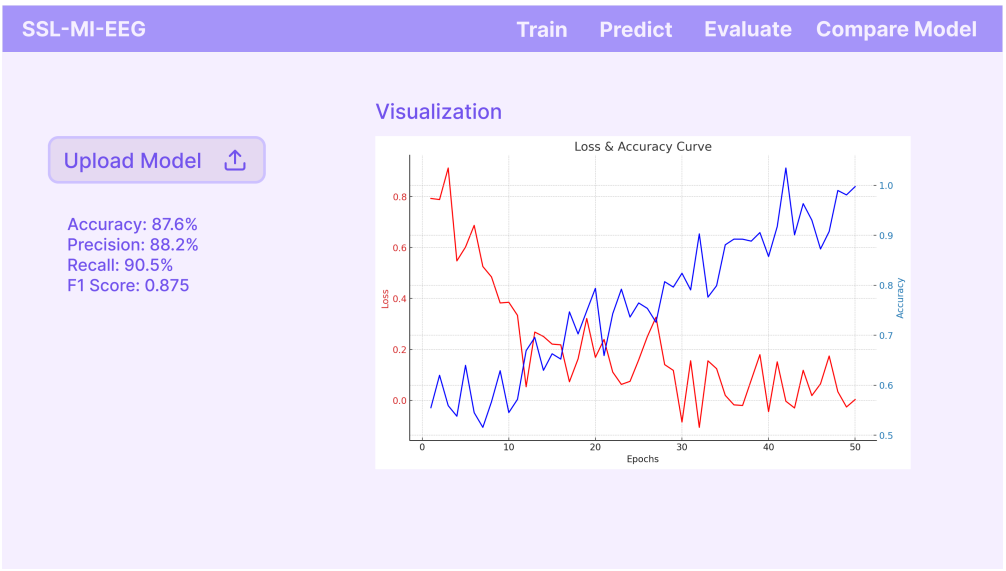

Figure 5.17: Evaluation Interface – Upload trained model and view performance metrics and curve

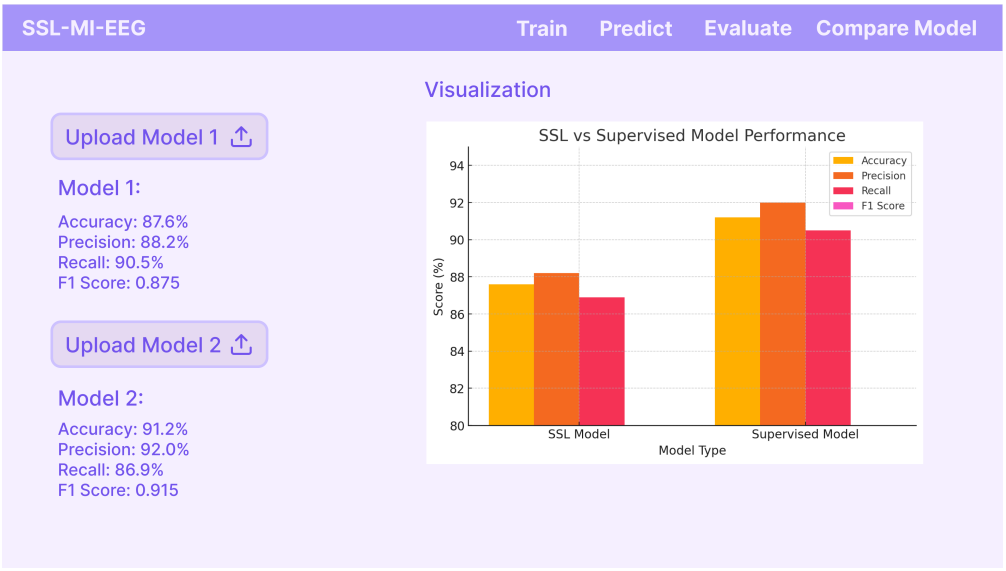

Figure 5.18: Comparison Interface – Visual comparison between model configurations

5.5 Deployment Strategy

5.5.1 Deployment Plan

The AI component is deployed in a research environment using both local execution and cloud notebooks.

The system consists of:

- Frontend — interactive research interface
- Backend — REST API service
- Model — behavioral prediction model

Technologies used:

- React — interactive frontend framework
- FastAPI — model service endpoints
- TensorFlow — neural network execution
- scientific Python libraries — preprocessing and evaluation

The frontend web system replaces manual notebook execution and allows controlled interaction with the EEG pipeline, and the backend exposes endpoints for training, prediction, and evaluation, allowing modular integration and testing.

The model in our system targets prestimulus behavioral prediction from CCD-ITI segments by mapping EEG-derived features to reaction time (RT) and trial-level accuracy as regression targets.

5.5.2 Proof of Concept

The model pipeline is validated under two evaluation settings:

- **Subject-Dependent Setting:** Model is trained and tested on individual subjects.
- **Subject-Independent Setting:** Model is trained on a group of subjects and tested on unseen individuals.

The system successfully supports end-to-end workflow:
visualization → dataset selection → training → inference → evaluation

Results can be accessed both through the interface and API endpoints.

5.6 Reflection and Future Development

Lessons Learned

- Prestimulus CCD-ITI segments contain steady-state information that can support behavioral prediction.
- A consistent preprocessing and verification workflow improves reproducibility of results.
- Integrated workflows reduce experimentation overhead

Challenges

- subject variability affects generalization
- interpreting learned representations remains difficult
- noisy EEG signals impact performance

Future Work

- transformer-based temporal modeling
- real-time prediction interface
- automated hyperparameter search
- live experiment integration

Chapter 6

Software Development

6.1 Software Development Methodology

This project was developed using an iterative, milestone-driven workflow. The core goal was to implement a reproducible pipeline for prestimulus EEG-based behavioral prediction on the HBN-EEG dataset (CCD ITI segments), and to support rapid experimentation with preprocessing and modeling choices.

The development process followed four repeating steps:

M1. Define a measurable milestone: e.g., extract CCD ITI prestimulus windows, compute features, train a regressor, or produce evaluation plots.

M2. Implement and validate: develop the smallest working version of the milestone, then validate with quick sanity checks (shape checks, trial counts, summary statistics, and train/validation splits).

M3. Evaluate and document: run a consistent evaluation protocol and record results, assumptions, and limitations.

M4. Refine: address issues found in evaluation (data leakage risks, unstable preprocessing, model underfitting/overfitting) before moving to the next milestone.

To keep the work aligned with the project scope, the model development emphasized:

- **Clear data boundaries** between subjects/sessions when creating splits.
- **Simple models first** (feature-based regressors) before attempting more complex architectures.

- **Reproducibility** through deterministic preprocessing parameters, fixed random seeds, and consistent evaluation scripts.

6.1.1 Architecture and Patterns

The software is organized to keep experiment logic, UI interaction, and data processing decoupled. In practice, the implementation follows a layered structure and applies a small set of design patterns that make experimentation and extension safer:

- **Layered architecture:** controller $\rightarrow$ services $\rightarrow$ models $\rightarrow$ pipeline $\rightarrow$ cache.
- **MVC-ish separation:** Views (EEG UI) focus on interaction and rendering; the Controller orchestrates actions; Models represent cohorts/tasks and their derived artifacts.
- **Facade pattern:** an `EEGController` provides a single entry point for common workflows (load data, preprocess, featurize, train, evaluate), hiding internal complexity.
- **DTOs (Data Transfer Objects):** small, explicit data containers are used when passing results between layers (e.g., feature tables, prediction outputs, and metric summaries).
- **Dependency Injection / Inversion of Control:** the controller injects service dependencies (e.g., via callbacks and class factories), improving testability and enabling swapping implementations.
- **Registry/Plugin + Command-style actions:** actions are decorator-registered and invoked by key, enabling extensible UI commands without hard-coding control flow.
- **Strategy pattern:** task-specific preprocessors are implemented per task type, so CCD ITI rules can differ from other task types without branching throughout the codebase.
- **Composite/Aggregation:** an `EEGCohortModel` aggregates multiple `EEGTaskModel` instances to support cohort-level evaluation.

- **Interfaces/Protocols:** a `TaskLike` interface (or protocol) standardizes what a task model must provide to downstream pipeline components.
- **Factory pattern:** loaders/processors are created through injectable factories, allowing dataset variants and preprocessing configurations to be selected at runtime.
- **Cache-aside with versioned file cache:** `CacheKey`/`LocalCache` implement a cache-aside policy so expensive steps (e.g., extracted windows/features) are reused safely across runs.
- **Lazy loading:** raw signals/events are loaded on demand; derived objects (e.g., epochs/features) are computed only when required.
- **Separation of concerns:** loader vs. processor responsibilities are split; UI vs. batch runs share the same underlying services.

6.2 Technology Stack

The implementation combines a Python-based machine learning pipeline with a lightweight user-facing interface and a LaTeX report workflow.

- **Programming language:** Python (data processing, feature extraction, modeling, evaluation).
- **Scientific computing:** NumPy and Pandas for array/dataframe operations.
- **EEG processing:** MNE-Python for signal handling and standard EEG preprocessing utilities.
- **Machine learning:** scikit-learn for classical models and metrics; PyTorch for neural models (when applicable).
- **Visualization:** Matplotlib/Seaborn for static plots; Plotly for interactive exploration.
- **User interface (documentation/prototype):** a React-based notebook-style UI used to visualize signals, preview tabular features, and record experiment notes.

- **Version control:** Git for change tracking and branching.

- **Report tooling:** LaTeX (TeX Live) with `latexmk` to automate multi-pass builds (cross-references and bibliography).

6.3 Coding Standards

The codebase follows a small set of conventions to keep experiments easy to reproduce and results easy to interpret.

- **Modular structure:** preprocessing, feature extraction, training, and evaluation are separated into distinct modules to reduce coupling.

- **Configuration over hard-coding:** key parameters (window length, frequency bands, feature sets, split strategy, model hyperparameters) are stored in configuration objects/files.

- **Reproducible runs:** fixed random seeds, explicit train/validation/test split definitions, and logging of dataset filters and preprocessing parameters.

- **Consistent naming:** functions and variables are named after domain concepts (trial, epoch, prestimulus window, RT target, accuracy target) rather than implementation details.

- **Input validation:** early checks for missing channels, unexpected sampling rates, and mismatched trial labels to avoid silent failures.

- **Documentation:** concise docstrings for public functions, and short experiment notes describing what changed between runs.

For evaluation outputs, plots and metrics are produced by scripts that can be rerun end-to-end, ensuring that reported results can be regenerated from the same inputs.

6.4 Progress Tracking Report

Work was tracked by incremental milestones, with each milestone producing a tangible artifact (code, plots, tables, or document sections). Table 6.1 summarizes the main development stages.

Phase	Activities	Outputs
1: Setup & scoping	Repository setup, project scope definition (CCD ITI prestimulus), and report structure alignment	Buildable LaTeX document, chapter plan
2: Data understanding	Explore HBN-EEG metadata/task structure; identify CCD ITI segments and behavioral targets (RT, accuracy)	Dataset summary notes, initial sanity plots
3: Preprocessing pipeline	Implement prestimulus window extraction and consistent preprocessing parameters; verify trial counts and split boundaries	Reusable preprocessing scripts, cached intermediate outputs
4: Model development	Train regressors for RT and accuracy (as regression targets); compare simple model families and features	Evaluation metrics (e.g., MAE/MSE), prediction vs. ground-truth plots
5: Evaluation & reporting	Define evaluation protocol, error analysis, and limitations; integrate key results into deliverables chapter	Evaluation tables/figures, updated Chapters 6–7
6: UI documentation	Document the notebook-style interface used for exploration and experiment logging	UI screenshots and usage description

Table 6.1: Progress tracking summary for the prestimulus behavioral prediction project.

Chapter 7

Deliverables and Evaluation

7.1 Artifacts

- **Project repository:** Contains preprocessing scripts and modeling code. [Link: to be added]
- **Figures:** SNR spectra and evoked response plots verifying steady-state/evoked components for CCD; updated class diagram.
- **Logs:** Preprocessing and training logs for reproducibility.

7.2 Evaluation Protocol

- **CCD-ITI approach:** Train on available CCD-ITI segments; report MAE/MSE for RT and trial-level accuracy as regression targets.
- **Verification:** Visualize SNR spectra and evoked responses to confirm signal components.

Chapter 8

Conclusion and Discussion

8.1 Reflection

Summarize the key outcomes and contributions of the project.

8.2 Challenges

List the main technical and non-technical challenges encountered.

- Challenge 1: ...
- Challenge 2: ...

8.3 Future Work

Outline planned next steps, improvements, and open questions.

- Next step 1: ...
- Next step 2: ...

Reference

Bibliography

[1] B. Aristimunha, D. Truong, P. Guetschel, S. Y. Shirazi, I. Guyon, A. R. Franco, M. P. Milham, A. Dotan, S. Makeig, A. Gramfort, J.-R. King, M.-C. Corsi, P. A. Valdés-Sosa, A. Majumdar, A. Evans, T. J. Sejnowski, O. Shriki, S. Chevallier, and A. Delorme, "Eeg foundation challenge: From cross-task to cross-subject eeg decoding," NeurIPS 2025 Competition Proposal (preprint), 2025.

[2] S. Y. Shirazi, A. Franco, M. S. Hoffmann, N. B. Esper, D. Truong, A. Delorme, M. P. Milham, and S. Makeig, "Hbn-eeg: The fair implementation of the healthy brain network (hbn) electroencephalography dataset," bioRxiv, 2024.

[3] L. M. Alexander, J. Escalera, L. Ai, C. Andreotti, K. Febre, A. Mangone, N. Vega-Potler, N. Langer, A. Alexander, M. Kovacs, S. Litke, B. O'Hagan, J. Andersen, B. Bronstein, A. Bui, M. Bushey, H. Butler, V. Castagna, N. Camacho, E. Chan, D. Citera, J. Clucas, S. Cohen, S. Dufek, M. Eaves, B. Fradera, J. Gardner, N. Grant-Villegas, G. Green, C. Gregory, E. Hart, S. Harris, M. Horton, D. Kahn, K. Kabotyanski, B. Karmel, S. P. Kelly, K. Kleinman, B. Koo, E. Kramer, E. Lennon, C. Lord, G. Mantello, A. Margolis, K. R. Merikangas, J. Milham, G. Minniti, R. Neuhaus, A. Levine, Y. Osman, L. C. Parra, K. R. Pugh, A. Racanello, A. Restrepo, T. Saltzman, B. Septimus, R. Tobe, R. Waltz, A. Williams, A. Yeo, F. X. Castellanos, A. Klein, T. Paus, B. L. Leventhal, R. C. Craddock, H. S. Koplewicz, and M. P. Milham, "An open resource for transdiagnostic research in pediatric mental health and learning disorders," *Scientific Data*, vol. 4, no. 170181, 2017.

[4] N. Langer, E. J. Ho, L. M. Alexander, H. Y. Xu, R. K. Jozanovic, S. Henin, A. Petroni, S. Cohen, E. T. Marcelle, L. C. Parra, M. P. Milham, and S. P. Kelly, "A resource for assessing information processing

in the developing brain using eeg and eye tracking," *Scientific Data*, vol. 4, no. 170040, 2017.

[5] Overleaf, "Learn latex in 30 minutes," https://www.overleaf.com/learn/latex/Learn_LaTeX_in_30_minutes.

Appendix A

Appendix A: Example

<TIP: Put additional or supplementary information/data/figures in appendices. />

Appendix B

Appendix B: About LaTeX

LaTeX (stylized as LaTeX) is a software system for typesetting documents. LaTeX markup describes the content and layout of the document, as opposed to the formatted text found in WYSIWYG word processors like Google Docs, LibreOffice Writer, and Microsoft Word. The writer uses markup tagging conventions to define the general structure of a document, to stylize text throughout a document (such as bold and italics), and to add citations and cross-references.

LaTeX is widely used in academia for the communication and publication of scientific documents and technical note-taking in many fields, owing partially to its support for complex mathematical notation. It also has a prominent role in the preparation and publication of books and articles that contain complex multilingual materials, such as Arabic and Greek.

Overleaf has also provided a 30-minute guide on how you can get started on using LaTeX. [5]