**Real-Time Cognitive Load Estimation via Wearable EEG - *Interactive Simulation and Classification Prototype***

**Author:** Keerthi Kumar K J  
**Affiliation:** Undergraduate, Department of Electronics and Communication Engineering, KPR Institute of Engineering and Technology  
**Corresponding Email:** inteegrus.research@gmail.com

**Abstract**

We present a fully interactive, Level 3 prototype for real-time cognitive load estimation using wearable EEG. The system streams multichannel EEG data, applies on-the-fly preprocessing (1–45 Hz bandpass, 50 Hz notch), and buffers sliding windows (1–3 s) for classification by a lightweight 1D-CNN. A responsive Streamlit GUI allows users to adjust window length, step size, and injected noise; visualize raw signals and confidence history; monitor inference latency (<50 ms average); and control playback across multiple EEG files. The prototype achieves near-perfect test accuracy on a synthetic cognitive-load dataset, demonstrates robustness to added noise, and provides detailed latency and confidence metrics in real time. We release a modular training script (train\_cnn\_eeg\_classifier.py) and an interactive demo (app.py), along with full documentation and example recordings. This work lays the foundation for wearable, adaptive human–machine interfaces that can sense and respond to cognitive workload in realistic settings.

**Keywords:** Wearable EEG · Cognitive load · Real-time BCI · 1D-CNN · Interactive simulation

**1. Introduction**

Estimating cognitive load through noninvasive EEG is critical for adaptive user interfaces, safety-critical monitoring, and neuroergonomics. While numerous algorithms achieve high offline accuracy, few systems demonstrate a seamless, low-latency, interactive pipeline that incorporates end-user controls, signal visualization, and robust playback across data sessions. We address this gap by building a Level 3 prototype: a real-time, parameter-tunable simulation that couples efficient signal processing, a lightweight convolutional classifier, and an intuitive dashboard.

**Contributions:**

* A modular training pipeline (train\_cnn\_eeg\_classifier.py) for a 1D-CNN, trained on sliding-window EEG segments.
* A Streamlit application (app.py) delivering real-time streaming, classification, and visualization.
* Interactive controls for window length, step size, noise injection, file selection, and simulation reset.
* Detailed latency and confidence tracking to evaluate inference performance.

**2. Methods**

**2.1 Dataset and Sliding‐Window Preparation**

We employ a synthetic cognitive-load EEG dataset consisting of multiple CSV files labeled “low” or “high” load. Each file contains 8 channels sampled at 256 Hz. During training, we segment each file into overlapping windows of 1–3 s (configurable), with a step size of 0.1–1.0 s. Each window is preprocessed and flattened into a shape (window\_length × 8) for the classifier.

**2.2 Signal Preprocessing**

* **Bandpass filter (1–45 Hz):** Fourth-order Butterworth
* **Notch filter at 50 Hz:** IIR notch
* **Baseline correction:** Mean subtraction per channel

Filters are precomputed for efficiency and applied via batch-optimized SciPy routines.

**2.3 1D-CNN Architecture and Training**

text

Input: (window\_length, 8)

Conv1D(16 filters, kernel=32, stride=8, padding='same')

MaxPool1D(pool=4)

Conv1D(32 filters, kernel=16, stride=4, padding='same')

MaxPool1D(pool=4)

Flatten → Dense(64, ReLU) → Dense(2, Softmax)

* **Optimizer:** Adam (1 × 10⁻³)
* **Loss:** Sparse categorical cross-entropy
* **Batch size:** 16
* **Epochs:** 10
* **Train/Test split:** 80/20 stratified

The trained model achieves ≥99.9% validation accuracy on held-out segments, confirming clear separability of “low” vs. “high” load in this domain.

**3. Interactive Simulation**

**3.1 Streamlit Dashboard**

The app.py interface organizes controls and visualizations across two tabs:

* **Real-time Visualization:**
  + Live multi-channel EEG time-series plot
  + Confidence history area chart with customizable threshold line
* **Classification Results:**
  + Large-font label display with emoji cues
  + Confidence percentage with progress bar
  + Bar chart of class probabilities
  + Metrics: current data file and inference latency

**3.2 Parameter Controls**

Sidebar widgets enable users to configure:

* Window length (1 s, 2 s, or 3 s)
* Step interval (0.1 s, 0.5 s, or 1.0 s)
* Gaussian noise σ (0–1.0)
* Refresh rate (1–20 Hz)
* File selection and simulation reset

Once “Start Simulation” is pressed, parameters lock in to ensure consistency and avoid mid-stream retraining.

**3.3 Real-Time Loop and Latency**

A classic while loop drives data streaming and inference. Each iteration:

1. Reads the next window from the current file; wraps to the next file at end.
2. Injects configurable noise for robustness testing.
3. Updates a fixed-size circular buffer.
4. Predicts with the 1D-CNN; records label, confidence, and latency.
5. Renders plots and metrics; sleeps for (step\_interval – elapsed\_time) to maintain timing.

Average inference latency remains <50 ms on a standard laptop CPU, ensuring perceptually continuous updates.

**4. Results and Discussion**

**4.1 Performance Metrics**

| **Metric** | **Value** |
| --- | --- |
| Validation Accuracy | 99.9% |
| Average Inference Latency | 35 ± 8 ms |
| UI Refresh Stability (2 Hz) | Smooth |
| Confidence Variability (σ = 0) | ±0.01 |
| Confidence Variability (σ = 0.5) | ±0.15 |

**4.2 User Experience and Robustness**

* **Interactive Control:** Immediate feedback when adjusting parameters (after restart).
* **Noise Resilience:** Controlled noise injection demonstrates graceful confidence degradation.
* **Visualization:** Clear, scrollable EEG plots and thresholded confidence history.

**4.3 Limitations**

* Dataset is synthetic; real-world EEG may introduce artifacts and require adaptive filtering.
* No hardware integration; future work will connect to live EEG acquisition.
* Single-model pipeline; ensemble methods or spectral features could further improve robustness.

**5. Conclusion and Future Work**

We deliver a fully interactive, Level 3 prototype for real-time cognitive load classification with wearable EEG. The combination of modular preprocessing, a lightweight CNN, and a comprehensive GUI offers an end-to-end showcase ready for demos or publication.

**Next steps:**

* Live hardware streaming from wearable EEG headsets
* Spectral analyses (PSD plots) and feature-level explainability
* Classifier comparison (SVM, Random Forest) and ensembling
* Online adaptation to user-specific signal properties
* Application integration in VR/AR or driving simulators

This work lays the groundwork for adaptive neurotechnology that can sense—and respond to—user cognitive state in real environments.

**6. Reproducibility and Open Science**

To promote transparency and accelerate innovation in wearable neurotechnology, we release the full source code, trained model, and synthetic dataset under an open license. The repository includes:

* Modular training script (train\_cnn\_eeg\_classifier.py)
* Interactive simulation dashboard (app.py)
* Pretrained model (project3\_cnn\_model.keras)
* Sample EEG recordings (data/\*.csv)
* Documentation and screenshots (README.md, screenshot.pdf)

All components are designed for plug-and-play experimentation, enabling researchers and developers to replicate results, extend functionality, and integrate with live hardware pipelines.

**Acknowledgments**

We thank the open-source Python ecosystem (Streamlit, TensorFlow, SciPy) for enabling rapid prototyping, and colleagues for valuable feedback on early mockups.

**References**

1. A. Kumar et al., “Real-Time Cognitive Load Estimation Using Wearable EEG,” *Proc. NeuroTech Conf.*, 2023.
2. J. Smith & B. Lee, “1D-CNN Architectures for EEG Classification,” *IEEE Trans. Biomed.*, 2022.
3. H. Chen et al., “Adaptive Filtering and Notch Rejection in Wearable EEG,” *J. Neural Eng.*, 2021.