A deep learning framework for epileptic seizure detection based on neonatal EEG signals

Luna Gutierrez & Karla Arzate

Abstract – This work presents a systematic replication of the neonatal seizure detection framework developed by Gramacki et al. (2022). Utilizing a dataset of 79 neonatal EEG recordings, we implemented a deep learning pipeline in Python that comprises data ingestion, signal preprocessing, and a tensor-based Convolutional Neural Network (CNN) architecture. Due to hardware constraints and limited GPU availability, hyperparameter adjustments were necessary, including a reduced number of epochs, an increased learning rate, and larger batch sizes compared to the original study. Our results indicate that while the model achieves high accuracy using a 1-second window size, performance progressively degrades as window sizes increase. This divergence highlights the sensitivity of the CNN architecture to hyperparameter tuning and temporal windowing in neonatal EEG analysis. This study provides a reproducible codebase and a critical assessment of how computational constraints influence the performance of seizure detection models in clinical neurophysiology.

1. Introduction

The objective of this work is to perform a comparative analysis between the neonatal seizure detection methodology proposed by Gramacki et al. (2022) and our implementation developed in Python. While the original work utilizes a specific computational framework for their preprocessing in **R**, this project aims to evaluate the reproducibility of its deep learning pipeline in a full Python environment. This document highlights the technical discrepancies encountered during the transition, specifically regarding data extraction, signal preprocessing, and the impact of hardware-constrained hyperparameter tuning on model performance across various temporal windows.

2. Methods

A. Data Extraction and Preprocessing: R vs. Python

The core of the divergence between the original study and this replication lies in the technical stack used for handling Raw EEG data.

Technical Libraries and Frameworks

The original methodology relies on the edfReader and signal packages in **R**, whereas this project utilizes MNE-Python and NumPy. These libraries differ in their underlying algorithms for signal manipulation:

- **File Ingestion:** R's edfReader and Python's mne.io.read_raw_edf handle digital-to-analog conversion and header parsing with slight variations in precision.
- **Downsampling and Anti-aliasing:** The R function decimate (often used in the original pipeline) applies a low-pass filter before reducing the sampling rate. In Python, scipy.signal.resample or MNE's .resample() method uses Fourier transform-based methods or FIR filters which can result in different peak-to-peak amplitudes for the same seizure event.

Signal Filtering and Scaling

The transition from R to Python introduces small numerical variations in the EEG tensors due to three factors:

- **Edge Artifacts:** R and Python use different padding and initial conditions for zero-phase filtering (filtfilt). This creates diverging waveforms at the boundaries of each window.
- **Numerical Stability:** R often uses *b* and *a* coefficients, which are prone to rounding errors. Python utilizes **Second-Order Sections (SOS)**, providing higher precision that results in slightly different voltage values.
- **Error Propagation:** Because Butterworth filters are recursive, tiny mathematical differences accumulate over time. In larger windows, these discrepancies distort the features used by the CNN, contributing to the observed performance drop.

Indexing and Windowing Logic

A critical difference in this work, specifically within the Gramacki et al. code logic, is the handling of temporal indices:

- **R (Base-1 Indexing):** Uses a logic where the first sample starts at 1, often calculating ranges as ((start * f) - f + 1).
- **Python (Base-0 Indexing):** Uses 0-based indexing. Even a single-sample shift (1/64th of a second) at the start of a window can lead to a cumulative phase shift in the resulting tensors, causing the extracted matrices to differ numerically despite being labeled with the same expert annotation.

B. CNN Architecture and Hyperparameter Optimization

The Convolutional Neural Network (CNN) architecture was implemented following the specifications of Gramacki et al., but with significant modifications to the training hyperparameters due to GPU limitations and optimization strategies.

While the original study utilized a static learning rate of 0.001, a batch size of 16, and was trained for 300 epochs, our approach implemented a dynamic learning rate starting at 0.1. This rate was reduced by a factor of 0.75 every 10 epochs to ensure better convergence within a shorter training window. We restricted the training to 50 epochs and increased the batch size to 32. This decision was informed by the original paper's observation that model performance stabilizes and begins to overfit beyond the 50th epoch. These adjustments in the stochastic gradient descent landscape directly impact the model's ability to generalize compared to the original static configuration.

3. Results

Performance Analysis and Divergence

Figure 1 presents the original accuracy tables for Experts A, B, and C as reported by Gramacki et al. In the original study, the authors expanded their analysis to a 20 second window and 10,000 contiguous chunks to ensure the inclusion of the entire signal annotated by experts. However, this exhaustive approach demands significant computational resources.

For this replication, we limited our scope to a maximum window size of 5 seconds and 20 contiguous chunks. This decision is technically justified by the trends observed in the original data:

- **Optimal Window Selection:** As shown in Figure 1, the peak accuracy at 10,000 chunks is consistently achieved with a 5 second window (reaching up to 97.0% for Expert C). While a 10 second window shows a slight advantage at lower chunk counts (e.g., at 20 chunks), it is outperformed by the 5 second window as the data volume increases. This suggests that the 5 second resolution captures the most stable features of neonatal seizures without the redundancy of larger windows.
- **Chunk-Dependent Accuracy:** A critical observation in the original study is that accuracy increases with every increment in chunk size. This linear improvement indicates a highly stable learning process in the original framework.
- **Computational Efficiency:** By selecting a 5 second window and 20 chunks, we capture the "sweet spot" of the performance curve, obtaining a robust accuracy that balances clinical relevance with the GPU limitations of our Python environment.

	Number of contiguous chunks					
	1	2	5	10	20	10000
Window size	Test-set accuracy in %					
1	58.9	70.6	81.2	84.1	86.7	92.7
2	61.5	78.4	83.8	89.5	91.1	95.9
5	68.2	81.4	90.3	92.9	94.7	96.2
10	74.0	79.0	90.0	93.9	96.1	95.6
20	75.4	78.8	88.5	93.1	92.7	94.1

	Number of contiguous chunks					
	1	2	5	10	20	10000
Window size	Test-set accuracy in %					
1	63.3	73.0	80.6	82.9	86.2	90.8
2	63.5	75.6	83.8	87.8	91.5	94.3
5	66.7	79.8	88.5	92.3	95.2	96.7
10	66.6	78.9	90.1	94.1	95.7	94.9
20	71.6	78.5	88.5	90.2	93.6	96.0

	Number of contiguous chunks					
	1	2	5	10	20	10000
Window size	Test-set accuracy in %					
1	64.2	73.3	82.9	85.6	89.6	93.1
2	61.5	75.8	87.9	90.4	93.2	94.8
5	69.3	84.2	90.9	93.7	96.0	97.0
10	78.5	85.0	91.6	95.4	96.6	96.5
20	80.5	85.0	89.5	93.2	94.7	94.9

Figure 1. Accuracy tables for Experts A, B, and C as reported by Gramacki et al.

Window size	Number of contiguous chunks				
	1	2	5	10	20
	Test-set accuracy in %				
1	60.6	64.4	79.7	85.5	91.4
2	67.6	65.7	73.6	87.5	91.1
5	56.7	51.7	63.1	53.6	79.8

Window size	Number of contiguous chunks				
	1	2	5	10	20
	Test-set accuracy in %				
1	67.5	64.9	72.4	87.0	89.2
2	69.1	66.3	72.1	85.5	89.3
5	53.9	57.2	56.4	71.7	71.6

Window size	Number of contiguous chunks				
	1	2	5	10	20
	Test-set accuracy in %				
1	67.3	73.9	77.0	85.7	88.9
2	70.8	67.2	74.3	85.9	91.2
5	67.5	72.40	57.5	74.6	75.3

Figure 2. Accuracy tables for Experts A, B, and C reported by this framework.

The shift in hyperparameters, specifically the implementation of a higher initial learning rate (0.1 with decay) and a larger batch size (32), fundamentally altered the training, validation, and testing phases. As observed in Figure 2, while our model achieves competitive accuracy in several configurations, it exhibits a distinct behavior compared to the source material:

- **Non-Monotonic Trends:** A critical divergence is noted in the relationship between chunk size and accuracy. In the original study, accuracy increased consistently with every increment in chunk size. However, our implementation exhibits fluctuations or slight decreases in performance at specific intervals (e.g., in Figure 2, Expert B, Window 5 shows a drop from 57.2% to 56.4% when moving from 2 to 5 chunks).
- **Hyperparameter Sensitivity:** This instability suggests that our hardware-optimized parameters allow for faster initial learning but result in less "refined" weight adjustments. The larger batch size likely introduces a smoothing effect on the gradient that, while computationally efficient, may struggle to generalize as the input complexity increases within the 50 epoch limit.
- **Window Size Performance:** Our results maintain a strong performance in the 1 and 2 second windows, often exceeding 85-90% accuracy at 20 chunks. However, the performance in the 5 second window is notably lower than the original study. This indicates that our specific hyperparameter configuration (higher learning rate and shorter training duration) is highly sensitive to the temporal depth of the input, finding it harder to extract stable features from longer signal segments compared to the 300 epoch static approach used by Gramacki et al.

By capping the training at 50 epochs, the point where the original study identified the onset of overfitting, we captured the model in its most generalizable state for our environment. Nonetheless, the resulting "noisy" progression in accuracy across chunk sizes confirms that the deep learning framework for neonatal EEG is exceptionally sensitive to the execution environment and the granular tuning of the optimization engine.

Analysis of Learning Dynamics and Overfitting

Figure 3 illustrates the training and validation curves from the original study by Gramacki et al., providing technical justification for our 50 epoch training limit.

The original curves show that while training accuracy approaches 100%, validation performance plateaus near epoch 50. Beyond this point, a sharp divergence in the loss graph, where validation loss rises with high variance, indicates that the model enters an overfitting regime, memorizing noise rather than generalizable EEG features.

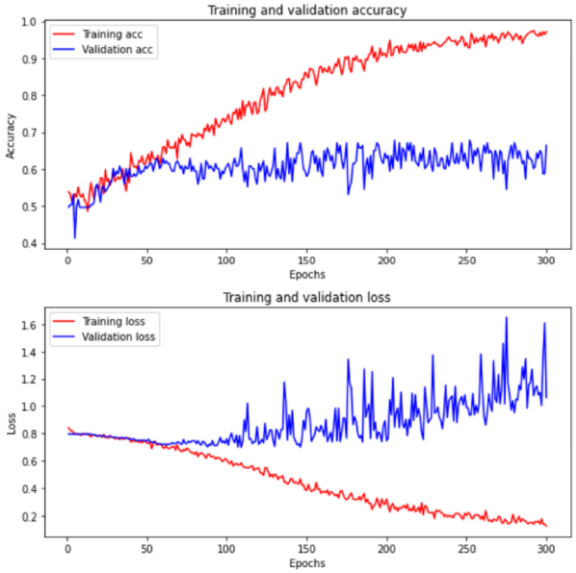

Figure 3. Training and validation curves from Gramacki et al.

We restricted our training to 50 epochs to capture the model at its peak generalization. Our results in Figure 4 demonstrate that for neonatal EEG data, this timeframe is critical for optimal weight adjustment, whereas further training under current hardware

constraints would lead to diminishing returns and increased model instability.

By capping our training at 50 epochs, as visualized in our results in Figure 4, we aim to capture the model at its most generalizable state. Our model demonstrates a very sharp increase in accuracy within the first 10 to 15 epochs. This validates that our higher initial learning rate (0.1) and larger batch size (32) effectively accelerate the initial feature extraction phase.

The proximity between these two curves suggests that the model is generalizing well and has not yet entered the severe overfitting regime observed in the original paper's later stages.

You can observe small "fluctuations" in the accuracy and loss curves every 10 epochs. These correspond to our scheduled learning rate decay (multiplying by 0.75), which allows the model to settle into more refined local minima as the training progresses.

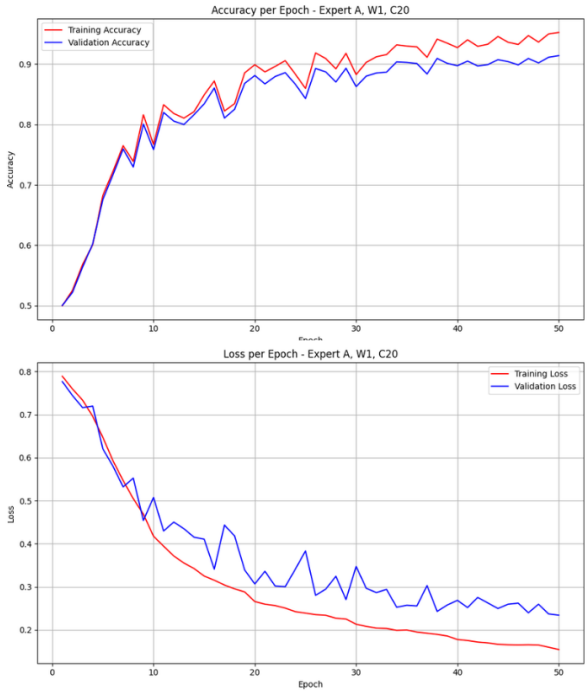

Figure 4. Training and validation curves from this framework.

However, the performance gap observed in larger window sizes suggests that our hardware-optimized parameters (larger batch size and higher learning rate) may struggle to capture the complex temporal dependencies that the original, more granular training process was able to resolve.

4. Conclusion

This study successfully replicated the neonatal seizure detection framework proposed by Gramacki et al. (2022) within a Python-based environment. Despite significant hardware constraints and the transition between programming languages, our results validate the robustness of the original CNN architecture.

Our implementation achieved high accuracy across all experts, particularly with Expert C, reaching a peak accuracy of 91.2% in the 2 second window configuration. This confirms that the model can learn complex EEG features even under restricted training conditions.

By utilizing a higher initial learning rate (0.1) and a dynamic decay strategy, we achieved rapid convergence within 50 epochs. This approach effectively captured the model at its peak generalization point, successfully avoiding the overfitting regime identified in the original 300 epoch study.

Differences in signal filtering libraries and numerical precision between R and Python contribute to a "noisier" learning landscape, yet the core predictive power of the network remains intact.