# A Deep Learning-Based Real-time Seizure Detection System

*N. Shawki[1], T. Elseify[2], T. Cap[1], V. Shah[1], I. Obeid[1] and J. Picone[1]*

1. Neural Engineering Data Consortium, Temple University, Philadelphia, Pennsylvania, USA
2. Epic Systems, Verona, Wisconsin, USA
{nabilashawki, thao.cap, vinitshah, iobeid, picone}@temple.edu, tarek@epic.com

Electroencephalography (EEG) is a popular clinical monitoring tool used for diagnosing brain-related disorders such as epilepsy [1]. As monitoring EEGs in a critical-care setting is an expensive and tedious task, there is a great interest in developing real-time EEG monitoring tools to improve patient care quality and efficiency [2]. However, clinicians require automatic seizure detection tools that provide decisions with at least 75% sensitivity and less than 1 false alarm (FA) per 24 hours [3]. Some commercial tools recently claim to reach such performance levels, including the Olympic Brainz Monitor [4] and Persyst 14 [5].

In this abstract, we describe our efforts to transform a high-performance offline seizure detection system [3] into a low latency real-time or online seizure detection system. An overview of the system is shown in Figure 1. The main difference between an online versus offline system is that an online system should always be causal and has minimum latency which is often defined by domain experts. The offline system, shown in Figure 2, uses two phases of deep learning models with postprocessing [3]. The channel-based long short term memory (LSTM) model (Phase 1 or P1) processes linear frequency cepstral coefficients (LFCC) [6] features from each EEG channel separately. We use the hypotheses generated by the P1 model and create additional features that carry information about the detected events and their confidence. The P2 model uses these additional features and the LFCC features to learn the temporal and spatial aspects of the EEG signals using a hybrid convolutional neural network (CNN) and LSTM model. Finally, Phase 3 aggregates the results from both P1 and P2 before applying a final postprocessing step.

The online system implements Phase 1 by taking advantage of the Linux piping mechanism, multithreading techniques, and multi-core processors. To convert Phase 1 into an online system, we divide the system into five major modules: signal preprocessor, feature extractor, event decoder, postprocessor, and visualizer. The system reads 0.1-second frames from each EEG channel and sends them to the feature extractor and

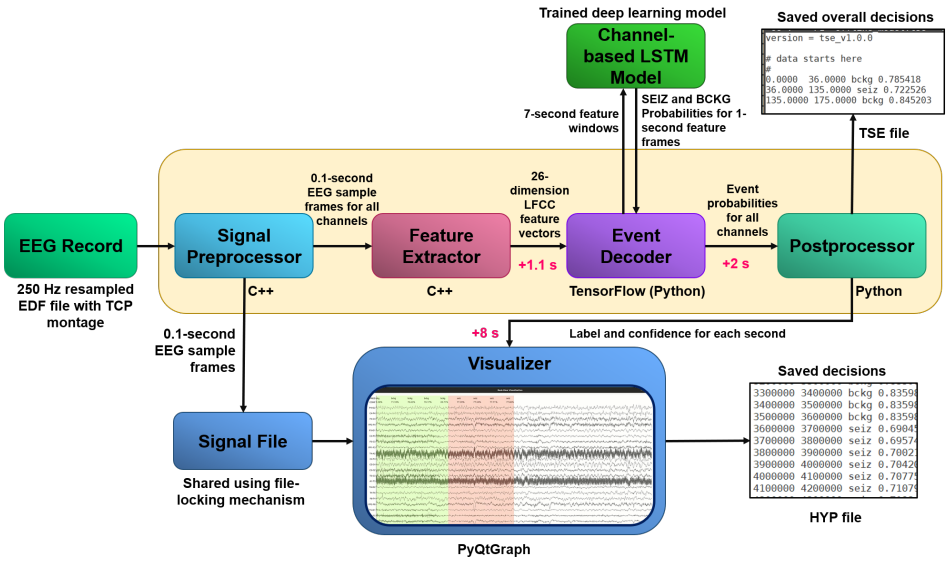


**Figure 1.** The block diagram of the real-time/online seizure detection system

the visualizer. The feature extractor generates LFCC features in real time from the streaming EEG signal. Next, the system computes seizure and background probabilities using a channel-based LSTM model and applies a postprocessor to aggregate the detected events across channels. The system then displays the EEG signal and the decisions simultaneously using a visualization module. The online system uses C++, Python, TensorFlow, and PyQtGraph in its implementation.

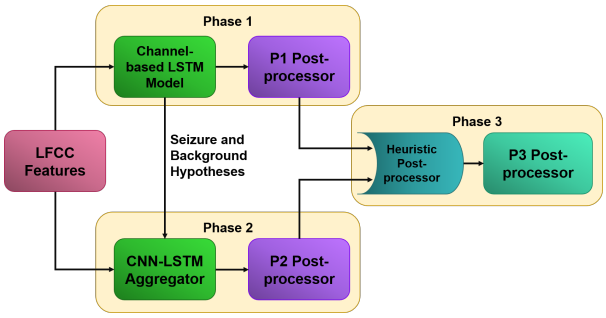


**Figure 2.** A system level block diagram of the offline seizure detection system

The online system accepts streamed EEG data sampled at 250 Hz as input. The system begins processing the EEG signal by applying a TCP montage [8]. Depending on the type of the montage, the EEG signal can have either 22 or 20 channels. To enable the online operation, we send 0.1-second (25 samples) length frames from each channel of the streamed EEG signal to the feature extractor and the visualizer. Feature extraction is performed sequentially on each channel. The signal preprocessor writes the sample frames into two streams to facilitate these modules. In the first stream, the feature extractor receives the signals using stdin. In parallel, as a second stream, the visualizer shares a user-defined file with the signal preprocessor. This user-defined file holds raw signal information as a buffer for the visualizer. The signal preprocessor writes into the file while the visualizer reads from it. Reading and writing into the same file poses a challenge. The visualizer can start reading while the signal preprocessor is writing into it. To resolve this issue, we utilize a file locking mechanism in the signal preprocessor and visualizer. Each of the processes temporarily locks the file, performs its operation, releases the lock, and tries to obtain the lock after a waiting period. The file locking mechanism ensures that only one process can access the file by prohibiting other processes from reading or writing while one process is modifying the file [9].

The feature extractor uses circular buffers to save 0.3 seconds or 75 samples from each channel for extracting 0.2-second or 50-sample long center-aligned windows. The module generates 8 absolute LFCC features where the zeroth cepstral coefficient is replaced by a temporal domain energy term. For extracting the rest of the features, three pipelines are used. The differential energy feature is calculated in a 0.9-second absolute feature window with a frame size of 0.1 seconds. The difference between the maximum and minimum temporal energy terms is calculated in this range. Then, the first derivative or the delta features are calculated using another 0.9-second window. Finally, the second derivative or delta-delta features are calculated using a 0.3-second window [6]. The differential energy for the delta-delta features is not included. In total, we extract 26 features from the raw sample windows which add 1.1 seconds of delay to the system.

We used the Temple University Hospital Seizure Database (TUSZ) v1.2.1 for developing the online system [10]. The statistics for this dataset are shown in Table 1. A channel-based LSTM model was trained using the features derived from the train set using the online feature extractor module. A window-based normalization technique was applied to those features. In the offline model, we scale features by normalizing using the maximum absolute value of a channel [11] before applying a sliding window approach. Since the online system has access to a limited amount of data, we normalize based on the observed window. The model uses the feature vectors with a frame size of 1 second

**Table 1.** TUSZ v1.2.1 Database Statistics

| | Train Set | Dev Set |
|---|---|---|
| Total Files | 1989 | 1015 |
| Files with Seizures | 384 | 273 |
| Total Duration (secs) | 1,188,313.00 | 617,102.00 |
| Seizure Duration (secs) | 78,838.09 | 58,322.37 |
| Patients | 264 | 50 |
| Patients with seizures | 118 | 38 |

and a window size of 7 seconds. We evaluated the model using the offline P1 postprocessor to determine the efficacy of the delayed features and the window-based normalization technique. As shown by the results of experiments 1 and 4 in Table 2, these changes give us a comparable performance to the offline model. The online event decoder module utilizes this trained model for computing probabilities for the seizure and background classes. These posteriors are then postprocessed to remove spurious detections.

The online postprocessor receives and saves 8 seconds of class posteriors in a buffer for further processing. It applies multiple heuristic filters (e.g., probability threshold) to make an overall decision by combining events across the channels. These filters evaluate the average confidence, the duration of a seizure, and the channels where the seizures were observed. The postprocessor delivers the label and confidence to the visualizer. The visualizer starts to display the signal as soon as it gets access to the signal file, as shown in Figure 1 using the "Signal File" and "Visualizer" blocks. Once the visualizer receives the label and confidence for the latest epoch from the postprocessor, it overlays the decision and color codes that epoch. The visualizer uses red for seizure with the label SEIZ and green for the background class with the label BCKG. Once the streaming finishes, the system saves three files: a signal file in which the sample frames are saved in the order they were streamed, a time segmented event (TSE) file with the overall decisions and confidences, and a hypotheses (HYP) file that saves the label and confidence for each epoch. The user can plot the signal and decisions using the signal and HYP files with only the visualizer by enabling appropriate options.

For comparing the performance of different stages of development, we used the test set of TUSZ v1.2.1 database. It contains 1015 EEG records of varying duration. The any-overlap performance [12] of the overall system shown in Figure 2 is 40.29% sensitivity with 5.77 FAs per 24 hours. For comparison, the previous state-of-the-art model developed on this database performed at 30.71% sensitivity with 6.77 FAs per 24 hours [3]. The individual performances of the deep learning phases are as follows: Phase 1's (P1) performance is 39.46% sensitivity and 11.62 FAs per 24 hours, and Phase 2 detects seizures with 41.16% sensitivity and 11.69 FAs per 24 hours. We trained an LSTM model with the delayed features and the window-based normalization technique for developing the online system. Using the offline decoder and postprocessor, the model performed at 36.23% sensitivity with 9.52 FAs per 24 hours. The trained model was then evaluated with the online modules. The current performance of the overall online system is 45.80% sensitivity with 28.14 FAs per 24 hours.

**Table 2.** A comparison of performances of different models

| **Exp. No.** | Systems | Description | Sensitivity (%) | FA/24 Hours |
|---|---|---|---|---|
| **1** | The offline system | P1-The channel-based LSTM model | 39.46 | 11.62 |
| **2** | | P2-The CNN-LSTM aggregator | 41.16 | 11.69 |
| **3** | | P3 and final postprocessor | 40.29 | 5.77 |
| **4** | The online system (the online channel-based LSTM model) | Evaluated with the offline P1 decoder and postprocessor | 36.23 | 9.52 |
| **5** | | Evaluated with the online P1 postprocessor | **45.80** | **28.14** |

Table 2 summarizes the performances of these systems. The performance of the online system deviates from the offline P1 model because the online postprocessor fails to combine the events as the seizure probability fluctuates during an event. The modules in the online system add a total of 11.1 seconds of delay for processing each second of the data, as shown in Figure 3. In practice, we also count the time for loading the model and starting the visualizer block. When we consider these facts, the system consumes 15 seconds to display the first hypothesis. The system detects seizure onsets with an average latency of 15 seconds.

Implementing an automatic seizure detection model in real time is not trivial. We used a variety of techniques such as the file locking mechanism, multithreading, circular buffers, real-time event decoding,

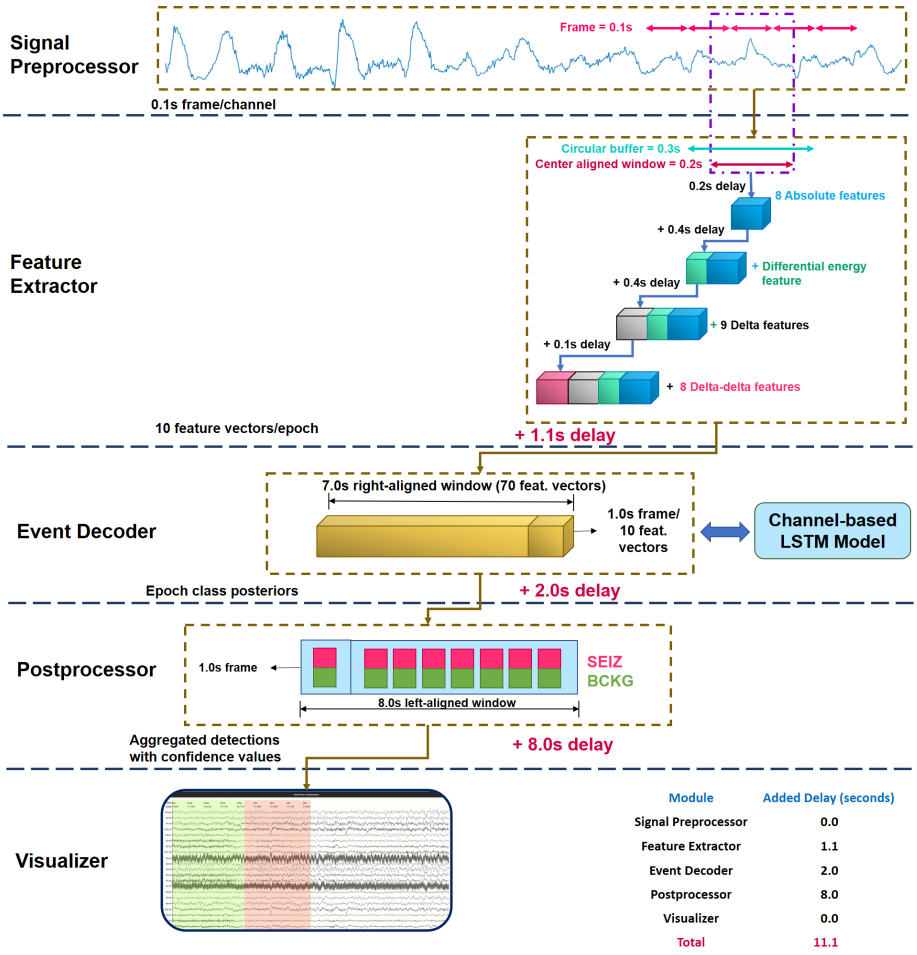


**Figure 3.** The data flow mechanism of the online (real-time) seizure detection system. Note that the figure is not drawn to scale.

and signal-decision plotting to realize the system. A video demonstrating the system is available at: *https://www.isip.piconepress.com/projects/nsf_pfi_tt/resources/videos/realtime_eeg_analysis/v2.5.1/video_2.5.1.mp4*. The final conference submission will include a more detailed analysis of the online performance of each module.

## ACKNOWLEDGMENTS

Research reported in this publication was most recently supported by the National Science Foundation Partnership for Innovation award number IIP-1827565 and the Pennsylvania Commonwealth Universal Research Enhancement Program (PA CURE). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the official views of any of these organizations.

## REFERENCES

[1] A. Craik, Y. He, and J. L. Contreras-Vidal, "Deep learning for electroencephalogram (EEG) classification tasks: a review," *J. Neural Eng.*, vol. 16, no. 3, p. 031001, 2019. *https://doi.org/10.1088/1741-2552/ab0ab5.*

[2] A. C. Bridi, T. Q. Louro, and R. C. L. Da Silva, "Clinical Alarms in intensive care: implications of alarm fatigue for the safety of patients," *Rev. Lat. Am. Enfermagem*, vol. 22, no. 6, p. 1034, 2014. *https://doi.org/10.1590/0104-1169.3488.2513.*

[3] M. Golmohammadi, V. Shah, I. Obeid, and J. Picone, "Deep Learning Approaches for Automatic Seizure Detection from Scalp Electroencephalograms," in *Signal Processing in Medicine and Biology: Emerging Trends in Research and Applications*, 1st ed., I. Obeid, I. Selesnick, and J. Picone, Eds. New York, New York, USA: Springer, 2020, pp. 233–274. *https://doi.org/10.1007/978-3-030-36844-9_8.*

[4] "CFM Olympic Brainz Monitor." [Online]. Available: *https://newborncare.natus.com/products-services/newborn-care-products/newborn-brain-injury/cfm-olympic-brainz-monitor*. [Accessed: 17-Jul-2020].

[5] M. L. Scheuer, S. B. Wilson, A. Antony, G. Ghearing, A. Urban, and A. I. Bagic, "Seizure Detection: Interreader Agreement and Detection Algorithm Assessments Using a Large Dataset," *J. Clin. Neurophysiol.*, 2020. *https://doi.org/10.1097/WNP.0000000000000709.*

[6] A. Harati, M. Golmohammadi, S. Lopez, I. Obeid, and J. Picone, "Improved EEG Event Classification Using Differential Energy," in *Proceedings of the IEEE Signal Processing in Medicine and Biology Symposium*, 2015, pp. 1–4. *https://doi.org/10.1109/SPMB.2015.7405421.*

[7] V. Shah, C. Campbell, I. Obeid, and J. Picone, "Improved Spatio-Temporal Modeling in Automated Seizure Detection using Channel-Dependent Posteriors," *Neurocomputing*, 2021.

[8] W. Tatum, A. Husain, S. Benbadis, and P. Kaplan, *Handbook of EEG Interpretation*. New York City, New York, USA: Demos Medical Publishing, 2007.

[9] D. P. Bovet and C. Marco, *Understanding the Linux Kernel*, 3rd ed. O'Reilly Media, Inc., 2005. *https://www.oreilly.com/library/view/understanding-the-linux/0596005652/.*

[10] V. Shah et al., "The Temple University Hospital Seizure Detection Corpus," *Front. Neuroinform.,* vol. 12, pp. 1–6, 2018. *https://doi.org/10.3389/fninf.2018.00083.*

[11] F. Pedregosa et al., "Scikit-learn: Machine Learning in Python," *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, 2011. *https://dl.acm.org/doi/10.5555/1953048.2078195.*

[12] J. Gotman, D. Flanagan, J. Zhang, and B. Rosenblatt, "Automatic seizure detection in the newborn: Methods and initial evaluation," *Electroencephalogr. Clin. Neurophysiol.*, vol. 103, no. 3, pp. 356–362, 1997. *https://doi.org/10.1016/S0013-4694(97)00003-9.*

NEURAL ENGINEERING DATA CONSORTIUM
www.nedcdata.org

# A Deep Learning-Based Real-Time Seizure Detection System

N. Shawki, T. Elseify, T. Cap, V. Shah, I. Obeid and J. Picone
The Neural Engineering Data Consortium, Temple University

College of Engineering
Temple University

## Abstract


- Detecting seizures in a critical care setting is an expensive and tedious task.
- We describe our effort to establish a pipeline for a real-time seizure detection system.
- The real-time system consists of five modules: signal preprocessor, feature extractor, event decoder, postprocessor, and visualizer.
- The current performance of the online system is 45.80% sensitivity with 28.14 FAs per 24 hours. The performance of the research prototype is 40.29% sensitivity and 5.77 FAs per 24 hours.
- The real-time system, implemented in C++ and Python, takes advantage of the Linux piping mechanism, multithreading techniques and multi-core processors.


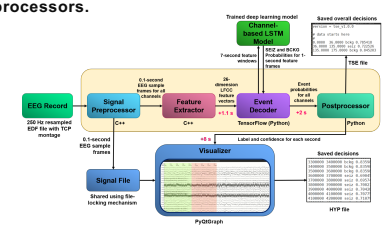


Figure 1. A block diagram of a real-time seizure detection system

## Signal Preprocessor

- The real-time system accepts streamed EEG data sampled at 250 Hz as input.
- The first module applies a TCP montage which also determines the channel count and order. The input signal can have either 20 or 22 channels.
- Each channel is processed in 0.1 second (25 sample) duration frames.
- Feature extraction and visualization receive a copy of the input signal in parallel so the waveform can be displayed in real time.

## Feature Extractor

- We produce 26 LFCC features: 8 absolute cepstral coefficients, a differential energy term, 9 first derivatives and 8 second derivatives.
- The differential energy term is excluded in the computation of the second derivatives.
- This module uses a 0.3 sec buffer (75 samples) from each channel to extract 0.2 sec (50 sample) center-aligned windows every 0.1 secs.
- The differential energy feature is calculated in a window of duration 0.9 secs.
- Computation of the first and second derivatives use 0.9 sec and 0.3 sec windows respectively.
- Total delay: 1.1 secs

## Event Decoder

- Using the features produced in the previous module, we trained a channel-based LSTM model in TensorFlow for the real-time system.
- The research system scales the feature across a channel before applying a sliding window approach. In the real-time system, we normalize each window since it has access to a limited amount of data.
- The model processes the feature vectors in a 7-second right-aligned window every 1 sec. The module adds 2 second delay.

## Postprocessor

- This module receives probabilities for two classes for each epoch and saves 8 epochs (8.0 secs) of data to make a decision.
- It applies multiple heuristic filters to make an overall decision by combining events across the channels.
- These filters use the average confidence, duration, and knowledge of the channel indices to make a decision.

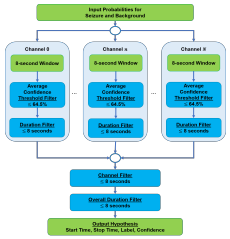


Figure 3. Postprocessing

## Visualizer

- The EEG signal is displayed as soon as signal data is available. Decisions are overlayed on the signal as soon as they are finalized (see Figure 4).
- The system uses the label SEIZ and color red for a seizure. Non-seizure events (background) are displayed using the label BCKG and the color green.
- When the streaming finishes, the system saves three files: a signal file in which the sample frames are saved in the order they were streamed, a time segmented event (TSE) file with the overall decisions and confidences, and a hypothesis (HYP) file that saves the label and confidence for each epoch.

## Performance

- The total system delay is 11.1 sec (see Figure 4). TensorFlow and model initialization requires about 20 secs, but this is not a factor in real-time performance.
- The system requires 1.81 secs to process 1 sec of data using a single CPU core, so it can easily run in real-time using multiple cores and/or GPUs.

Table 2. The delay in each module

| Module | Processing (secs) | Cumulative (secs) |
|---|---|---|
| Signal Preprocessor | 0.01 ( 0.55%) | 0.01 ( 0.55%) |
| Feature Extractor | 0.07 ( 3.87%) | 0.08 ( 4.42%) |
| Event Decoder | 1.56 ( 86.19%) | 1.64 ( 90.61%) |
| Postprocessor | 0.15 ( 8.29%) | 1.79 ( 98.90%) |
| Visualizer | 0.02 ( 1.10%) | 1.81 (100%) |
| Total | 1.81 (100%) | 1.81 (100%) |

- The decoding process can also be offloaded to GPUs if available and runs hyper real time.

## Overview

- The research prototype uses two phases of deep learning models and a postprocessor.

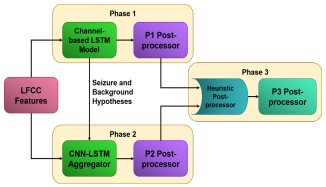


Figure 2. A system level block diagram of the offline seizure detection system

- In Phase 1 (P1), a long short term memory (LSTM) model processes linear frequency cepstral coefficients (LFCC) features derived from each EEG channel separately (per-channel features).
- For Phase 2 (P2), we generate additional features using the hypotheses produced in P1. A hybrid convolutional neural network (CNN) and LSTM model is then applied to learn the temporal and spatial aspects of the EEG signal.
- Phase 3 (P3) aggregates the results from P1 and P2 before applying a final postprocessing step.
- Data processing is parallelized and accelerated via Linux pipes. Efficient data structures such as circular buffers and queues are used.
- A network file locking mechanism is used to prevent file and data access collisions.
- Only P1 is part of the real-time system because P2 and P3 add excessive amounts of latency.

## The Data Flow Pipeline

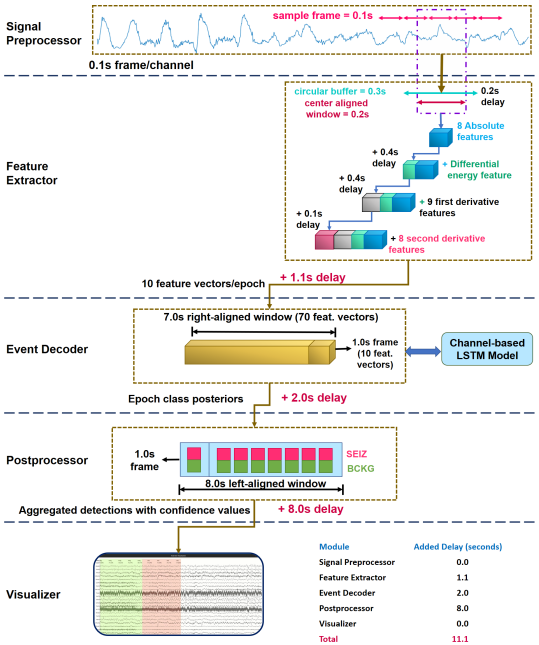


Figure 4: Data flow in the real-time system

## Results

Table 1. Performance comparison

| Exp. No. | Systems | Description | Sens. (%) | FA/24 Hours |
|---|---|---|---|---|
| 1 | Research Prototype | P1 – Channel-based LSTM (infinite latency) | 39.46 | 11.62 |
| 2 | | P2 – CNN-LSTM aggregation | 41.16 | 11.69 |
| 3 | | P3 – Final Postprocessing | 40.29 | 5.77 |
| 4 | Real-Time Channel-Based LSTM | Feature extraction + non-real-time P1 | 36.23 | 9.52 |
| 5 | | Feature extraction + real-time P1 | 45.80 | 28.14 |

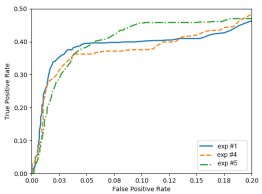


Figure 5: ROC comparison of performance

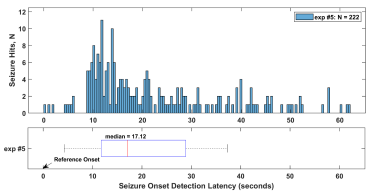


Figure 6: Seizure onset detection latency

## Conclusions

- Developed a low-latency (11.1 secs) real-time seizure detection system suitable for critical care applications.
- Compared to the offline model, the sensitivity increased by 6.34% absolute while the false alarms increased by 16.34 per 24 hours.
- Detects seizure onsets with a median latency of 17.1 secs and offsets with a latency of 11.1 secs.
- Our future research is focused on reducing the false alarm rate by exploring pretrained models such as ResNets and using more advanced architectures.
- For a video demonstration, see:
  https://www.isip.piconepress.com/projects/nsf_pfi_tt/resources/videos/realtime_eeg_analysis/current/video_2.5.1.mp4

## Acknowledgements

Research reported in this publication was most recently supported by the National Science Foundation Partnership for Innovation award number IIP-1827565 and the Pennsylvania Commonwealth Universal Research Enhancement Program (PA CURE). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the official views of any of these organizations.