Testing Document

Project Name:

Unsupervised Learning of EEG States for the Application of Seizure Detection in Epilepsy

Project Number:

2022659

Group members:

Mehak Kalra

Fatima Siddiqui

Khalil Scott

Mina Assaad

1.0 Executive Summary

This document overviews the testing procedures for the requirements of the software toolbox the team has created. The three machine learning (ML) approaches implemented are clustering, stacked autoencoder and convolutional neural network (CNN). The stacked autoencoder and CNN have similar signal processing methods, so they will share the same testing procedures; on the other hand, clustering has different signal processing methods, so it will have its own procedures for testing. The requirements presented in the original proposal and implementation plan did not have proper measurable criteria. As such, the team has outlined the updated requirements in this testing document, allowing for a more detailed breakdown of the tests to be executed.

The Requirements for Testing section outlines these updates in a table. The functional requirements now include a seizure/non-seizure labelling accuracy of greater than 80%, a labelling speed faster than the signal sampling rate of 256 Hz, and the display of extracted features to provide insights on how the labels were determined by the ML algorithm. The Description for Testing section outlines the parameters used for each test, including applying wavelet transformations on 5-second signal segments, recording timestamps for speed measurements, and recording extracted features. The Test Plan section outlines the environment to conduct the tests, which includes using Windows 10 machines with CUDA compatible Nvidia GPUs, Google Colab, Jupyter Notebook, and the CHB-MIT EEG and Native Data datasets.

The Test Method section describes the step-by-step procedures for testing each of our requirements by obtaining seizure and non-seizure labelling accuracies, labelling speeds, displaying extracted features for all signal segments, and running signal extraction and labelling on NWB files without any errors. Finally, the Success Criteria section displays a table similar to the one in Requirements for Testing, but with a column including the pass/fail criterion for each requirement.

2.0 Requirements for Testing

This section outlines the requirements to be tested, as well as a summary of changes made to our original design specifications. The reason for these changes was that our original functional requirements were either not testable, having no measurable criteria to verify their success, or were no longer considered relevant to the design. As a result, we altered our original objective and functional requirements, all of which now have measurable success criteria assigned to them.

Table 1 below shows all of the updated functions and objectives. It also represents the entire pool of requirements to be tested, i.e., all functions and objectives will be tested, but only one out of three of our

constraints will be tested. The other two are not testable. Each requirement will have one test assigned to it with the same ID field.

Table 1. Requirements for Testing

No. #	Requirement	Type	Requirement for Testing
1	Generate correct labels for classifying input data between seizure and non-seizure	Function	A system with at least 4 GB of RAM and 4 GB of VRAM, and capable of running Python 3. Test set and ground truth labels will be extracted from part of the CHB-MIT EEG dataset.
2	The model should be accurate	Objective	Same as Requirement #1.
3	Label data faster than signal sampling rate	Function	Same as Requirement #1 but the machine must also be rebooted with no extraneous background processes running.
4	Provide clinical insights into how labels are determined	Function	Same as Requirement #1.
5	Must accept Neurodata Without Borders (NWB) files as input [1]	Constraint	Same as Requirement #1, but NWB files from the Native Data dataset will be used instead of the CHB-MIT EEG dataset.

3.0 Description of Testing

This section lists the types of tests in the order of their corresponding requirement ID, as listed in Table 1, as well as the parameters with which they will be conducted. All tests for the requirements in Table 1 will be performed separately for the three ML approaches. Sections 3 to 5 of this document outline the same testing approaches for the CNN and stacked autoencoder models, but the clustering models will follow their own procedure, i.e., Sections x.1 will describe approaches for CNN and stacked autoencoder and Sections x.2 will describe approaches for clustering.

3.1 CNN and Stacked Autoencoder

1. Generate correct labels for classifying input data between seizure and non-seizure

    a. Used a single European Data Format (EDF) file from the CHB-MIT dataset
    b. Retrieved the signals from each EEG channel in the file, and split them into 5-second segments
    c. Transformed EEG segments into scalograms using the Continuous Wavelet Transform (CWT) function from the PyWavelets Python library
2. The model should be accurate
    a. Same points as Test #1
3. Label data faster than signal sampling rate
    a. Used the same signal processing methods from points a-c under Test #1
    b. Timestamps are taken before the file is inputted and after labels are generated using the Time Python library
4. Provide clinical insights into how labels are determined
    a. Used the signal processing methods as described in points a-c under Test #1
    b. Saliency heat maps [2] for all segments are displayed
5. Must accept NWB files as input
    a. Used a single NWB file from the Native Data dataset
    b. Used the signal processing methods as described in points b and c under Test #1

3.2 Clustering

1. Generate correct labels for classifying input data between seizure and non-seizure
    a. Used a single EDF file from the CHB-MIT dataset
    b. EEG signals split into 5-second segments and transformed using Discrete Wavelet Transform (DWT) function from the PyWavelets Python library
    c. Mean, entropy, standard deviation, and average power of segments calculated from DWT coefficients
    d. EEG signal segments are organized into clusters using clustering algorithms and are assigned labels
    e. Rand index (RI) of clusters calculated using scikit-learn metrics library
2. The model should be accurate
    a. Same points as Test #1
3. Label data faster than signal sampling rate
    a. Used the signal processing methods as described in points a-d under Test #1
    b. Timestamps are taken before the file is inputted and after labels are generated using the Time Python library

4. Provide clinical insights into how labels are determined
    a. Used the signal processing methods as described in points a-d under Test #1
    b. Signal features for all segments are displayed
5. Must accept NWB files as input
    a. Used a single NWB file from the Native Data dataset
    b. Used the signal processing methods as described in points b-d under Test #1

4.0 Test Plan

This section details the setup of the environment in which tests will be conducted. The testing of all three ML approaches will be done in a pure software environment, with all tests running in parallel. Figure 1 below shows the overall infrastructure of the testing for all three ML approaches.

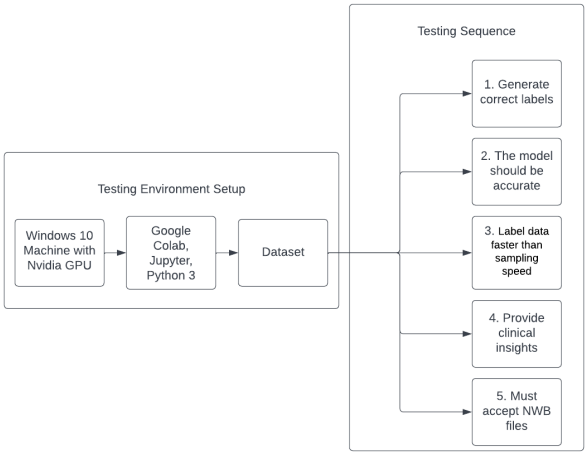

Figure 1. Diagram of the overall testing environment and the sequence of tests to be conducted

4.1 CNN and Stacked Autoencoder

The machine that will be used is a DELL laptop with the following specifications: Windows 10, AMD Ryzen 7 5800H 3.20 GHz processor, 16 GB RAM, NVIDIA GeForce RTX 3050 TI GPU 20 compute units (CU), 128 CUDA cores per CU, 4GB VRAM. The application running the test will be Google Colab. The test will be contained in a Google Colab notebook and will be written in Python 3 using the Pytorch, PyWavelets, and PyEDFlib libraries. Command Prompt will be used to launch a Jupyter Notebook local runtime, which Google Colab will be connected to. The CHB-MIT EEG and the Native Data datasets will be used, containing the EDF and NWB files used for testing, respectively. Only the latest versions of the CNN and stacked autoencoder models will be tested.

4.2 Clustering

The machine that will be used is a PC running Windows 10 on an NVIDIA GeForce RTX 2060 SUPER GPU with 2176 CUDA cores, and 32 GB of RAM. The tests will be conducted in Google Colab using a local Jupyter Notebook runtime, running Python 3. The scikit-learn, PyEDFlib, and PyWavelets Python libraries will be used for signal processing and feature extraction. The CHB-MIT EEG and the Native Data datasets will be used, containing the EDF and NWB files used for testing, respectively. There will be three versions of clustering being tested in parallel: K-Means, Fuzzy C-Means, and BIRCH.

5.0 Test Method

This section describes the step-by-step testing procedure for each requirement. The tests are ordered by their corresponding requirement ID as in Section 3.0.

5.1 CNN and Stacked Autoencoder

1. Generate correct labels for classifying input data between seizure and non-seizure

   The following procedure applies for the testing of each EDF file. 20% of the EDF files from each patient folder in the CHB-MIT EEG dataset are used for testing.

   a. Start Google Chrome and open the "CNN_Autoencoder_Testing.ipynb" Google Colab notebook
   b. Start a Jupyter Notebook runtime in Command Prompt
   c. Connect to the local runtime from Google Colab
   d. Open the file and read all the signals from each EEG channel
   e. Split the signals into 5-second segments and apply a CWT transform on them, appending them to a list
   f. Assign the corresponding truth labels to each segment in another list and close the file
   g. Loop through each segment and pass them through the model, running it on the GPU using CUDA
   h. Compare the output predictions (seizure or no seizure) from the model with the ground truth labels and store the errors for all segments
   i. Return the accuracy based on the number of correct predictions made

   Once this procedure is completed for all the EDF files being tested, a final mean average will be taken based on all the collected accuracies.

2. The model should be accurate

Same procedure as Test #1

3. Label data faster than signal sampling rate

The following procedure applies for the testing of each EDF file. 20% of the EDF files from each patient folder in the CHB-MIT EEG dataset are used for testing.

a. Start up the machine, or reboot it if it is already turned on to clear RAM and close unneeded processes
b. Perform steps a-c under Test #1
c. Call the time() function to return an initial timestamp
d. Perform steps d-g under Test #1
e. Call the time() function again to return the final timestamp and acquire the total elapsed time duration
f. Get the labelling rate as follows:

$$\frac{(256\ samples/s * 5s) * total\ \#segments}{time\ duration} \quad \text{(eq. 1)}$$

The numerator obtains the total number of samples processed by the algorithm in the EDF file by multiplying the sampling rate used in the dataset by the 5 second duration of each segment, multiplied by the total number of labelled segments. The denominator is the total time duration in seconds from when the file is first inputted to when all the labels have been generated. The final value is the labelling speed in terms of samples labelled per second. Note, however, that it is the segments that each receive a label, not the individual samples contained within the segments.

Once this procedure is completed for all the EDF files being tested, a final mean average will be taken based on all the collected labelling speeds.

4. Provide clinical insights into how labels are determined

The following procedure applies for the testing of each EDF file. 20% of the EDF files from each patient folder in the CHB-MIT EEG dataset are used for testing.

a. Perform steps a-g under Test #1
b. Display the saliency heat map of each EEG segment

If a saliency heat map can be displayed for each individual segment across all EDF files, then the requirement is satisfied.

5. Must accept NWB files as input

The following procedure applies for the testing of each NWB file in the Native Data dataset.

a. Perform steps a-g under Test #1, using NWB files instead of EDF files
b. Confirm no errors were encountered during the signal extraction and labelling process

If no errors are encountered for all NWB files, then the requirement is satisfied.

5.2 Clustering

1. Generate correct labels for classifying input data between seizure and non-seizure

The following procedure applies for the testing of each EDF file. 20% of the EDF files from each patient folder in the CHB-MIT EEG dataset are used for testing.

a. Start a local Jupyter Notebook session using the command prompt
b. Open the "Clustering_Testing.ipynb" notebook in Google Colab and connect to the local Jupyter Notebook runtime
c. Using the PyEDFlib library, open the EDF file and extract its EEG signals
d. Split the signals into 5-second segments
e. Using the PyWavelets library, perform DWT on the 5-second segments to obtain their wavelet coefficients
f. Compute the mean, entropy, standard deviation, and average power of each segment and save them into an array
g. Pass the array of computed values into the clustering algorithm to generate labels
h. To calculate accuracy, compute the RI using the ground truth labels

Once this procedure is completed for all the EDF files being tested, a final mean average will be taken based on all the collected accuracies.

2. The model should be accurate

Same procedure as Requirement #1

3. Label data faster than signal sampling rate

The following procedure applies for the testing of each EDF file. 20% of the EDF files from each patient folder in the CHB-MIT EEG dataset are used for testing.

a. Start up the machine, or reboot it if it is already turned on to clear RAM and close unneeded processes
b. Perform steps a and b under Test #1
c. Call the time() function to return an initial timestamp
d. Perform steps c-g under Test #1
e. Call the time() function again to return the final timestamp and acquire the total elapsed time duration
f. The labelling rate is obtained the same way as for the CNN and stacked autoencoder, using eq. 1.

Once this procedure is completed for all the EDF files being tested, a final mean average will be taken based on all the collected labelling speeds.

4. Provide clinical insights into how labels are determined

The following procedure applies for the testing of each EDF file. 20% of the EDF files from each patient folder in the CHB-MIT EEG dataset are used for testing.

a. Perform steps a-g under Test #1.
b. Display the extracted features of each EEG segment in the EDF file

If features can be displayed for each individual segment across all EDF files, then the requirement is satisfied.

5. Must accept NWB files as input

The following procedure applies for the testing of each NWB file in the Native Data dataset.

a. Perform steps a and b under Test #1
b. Using the PyNWB library, open the NWB file and extract its EEG signals
c. Perform steps d-g under Test #1
d. Confirm no errors were encountered during the signal extraction and labelling process

If no errors are encountered for all NWB files, then the requirement is satisfied.

6.0 Success Criteria

Table 2 outlines the requirements from Table 1 with their corresponding pass/fail criteria.

Table 2. Success Criteria

No. #	Requirement	Type	Pass/Fail Criteria
1	Generate correct labels for classifying input data between seizure and non-seizure	Function	Accuracy must be greater than 80% [3]
2	The model should be accurate	Objective	Accuracy should be 100%
3	Label data faster than signal sampling rate	Function	Must label more than 256 samples per second (each segment has 1280 samples) [4]
4	Provide clinical insights into how labels are determined	Function	Display signal features for all segments
5	Must accept NWB files as input	Constraint	Signal extraction and labelling run with zero errors

7.0 References

[1] (2022, July 15). *Re: Interest in Project for ECE496*.

[2] Rastogi, A. (2020, April 7). *Visualizing Neural Networks using saliency maps in pytorch*. Medium. Retrieved January 28, 2023, from https://medium.datadriveninvestor.com/visualizing-neural-networks-using-saliency-maps-in-pytorch-289d8e244ab4

[3] Jin Jing, P. D. (2020, January 1). Interrater reliability of experts in identifying interictal epileptiform discharges in eegs. JAMA Neurology. Retrieved September 23, 2022, from https://jamanetwork.com/journals/jamaneurology/fullarticle/2752670

[4] CHB-MIT Scalp Eeg Database. CHB-MIT Scalp EEG Database v1.0.0. (2010, June 9). Retrieved September 23, 2022, from https://physionet.org/content/chbmit/1.0.0/