Seizure Classification of EEG Data using the Continuous Wavelet Transform in Conjunction with a Neural Network

Lilly Roelofs – 30 November 2022

I. Introduction

Personal health devices which provide automatic detection of life-threatening events, i.e. heart attack or stroke, are highly sought out to encourage patient well-being and safety. An important application of this is seizure identification since the severeness and duration of a seizure may require immediate medical attention. One way to quantify or detect whether a patient is undergoing a seizure is through recording and analyzing the patient's electroencephalogram (EEG) signal. These signals are commonly collected through a head cap surrounding the scalp, which makes EEG signal acquisition safe, portable, and cost efficient. However, EEG signals commonly face issues with low temporal resolution, and since the electrodes are located outside of the skull, the signals are not well localized. Regardless of the disadvantages of using EEG, this is still the most widely used technique for measuring seizure activity. My project focus is to contribute to efforts in effectively classifying seizure occurrence through conducting feature extraction, neural network modeling, and exploratory data analysis on a subset of the well-known CHB-MIT Scalp EEG Database.

II. Data

The dataset I am using is found on the Kaggle website titled EEG Seizure Analysis Dataset (Adrian-Catalin, & Vlad Adam, 2021). This dataset is a semi-processed subset of the original CHB-MIT Scalp EEG Database (Shoeb, 2010).

A. CHB-MIT Scalp EEG Database

The CHB-MIT database was collected at the Children's Hospital of Boston and includes 23 cases (22 unique patients with intractable seizures, one of which participated a second time). The EEG signals were all sampled at 256 Hz with 16-bit resolution. For each case, the data is saved as a collection of .edf files (European data format), and these files contain time series EEG signals for each channel. The number and configuration of EEG channels varies between cases. For example, the channel configuration for cases 9 and 20 and can be seen in Figure 1a and 1b, respectively (Goldberger et al., 2000). From this, it is clear that the channels cannot be compared directly unless the data is reorganized/adjusted. Also, in certain cases some of the channels record other electrophysiological signals such as the vagus nerve or EKG, adding to the heterogeneity of the database. Lastly, a seizure annotation file is included in each case folder which states the beginning and end time points of seizure as observed by the physician.

B. Kaggle EEG Seizure Analysis Dataset

The EEG Seizure Analysis Dataset is a subset of the CHB-MIT Database, in which the authors created two datasets with different goals: seizure analysis (for classifying seizure vs non-seizure instances), and predictive seizure analysis (for predicting the onset of a seizure within 30 seconds). I chose to focus on the seizure classification task as this seemed like a good starting point that would be reasonable to achieve. The seizure analysis dataset consists four .npz (numpy zip) files: training, validation, class-balanced validation, and testing. For my purposes, I focused on the training and class-balanced validation sets ('eeg-seizure_train.npz' and 'eeg-seizure_val_balanced.npz', respectively). The signals are saved as three dimensional arrays of shape (i, 23, 256) where i represents the number of instances in that dataset. The second axis represents each EEG channel (with 23 total channels recorded per instance) and the third axis represents the signal within that channel (with 256 points corresponding to the sampling frequency). Figure 1c, 1d, and 1e depict further information regarding the seizure analysis training, validation, and

testing sets, including their size, percentage of seizure occurrences, and percentage of null channels. Further discussion of the data will be detailed in the next section.

The main purpose of the EEG Seizure analysis data subset is to tackle the class imbalance found in the original CHB-MIT Database. According to the authors, non-seizure samples outnumbered seizure-present samples by a ratio of 300:1 (Adrian-Catalin, & Vlad Adam, 2021). It is known that severely imbalanced data hinders model performance, so the authors created this dataset to provide an easy-to-adapt version with an appropriate class ratio.

Figure 1. Figures (a) and (b) contain the channel montage lists for two separate cases from the CHB-MIT Database, cases 9 and 20. Figures (c), (d), and (e) describe several features of the Kaggle seizure analysis datasets, including information of the versions post-cleaning (after the null channels were removed).

(a)

Case 9

```
Channels in EDF Files:
**********************
Channel 1: FP1-F7
Channel 2: F7-T7
Channel 3: T7-P7
Channel 4: P7-O1
Channel 5: FP1-F3
Channel 6: F3-C3
Channel 7: C3-P3
Channel 8: P3-O1
Channel 9: FP2-F4
Channel 10: F4-C4
Channel 11: C4-P4
Channel 12: P4-O2
Channel 13: FP2-F8
Channel 14: F8-T8
Channel 15: T8-P8
Channel 16: P8-O2
Channel 17: FZ-CZ
Channel 18: CZ-PZ
Channel 19: P7-T7
Channel 20: T7-FT9
Channel 21: FT9-FT10
Channel 22: FT10-T8
Channel 23: T8-P8
```

(b)

Case 20

```
Channels in EDF Files:
**********************
Channel 1: FP1-F7
Channel 2: F7-T7
Channel 3: T7-P7
Channel 4: P7-O1
Channel 5: .
Channel 6: FP1-F3
Channel 7: F3-C3
Channel 8: C3-P3
Channel 9: P3-O1
Channel 10: .
Channel 11: FZ-CZ
Channel 12: CZ-PZ
Channel 13: .
Channel 14: FP2-F4
Channel 15: F4-C4
Channel 16: C4-P4
Channel 17: P4-O2
Channel 18: .
Channel 19: FP2-F8
Channel 20: F8-T8
Channel 21: T8-P8
Channel 22: P8-O2
Channel 23: .
Channel 24: P7-T7
Channel 25: T7-FT9
Channel 26: FT9-FT10
Channel 27: FT10-T8
Channel 28: T8-P8
```

(c)

```
Parameters of eeg-seizure_train.npz
Files stored in the numpy zip: ['train_signals', 'train_labels']
Size of train_signals: (37666, 23, 256)
Size of train_labels: (37666,)
Percentage of seizure occurences: 21.436%
Percentage of instances that contain at least 1 null channel: 26.727%
-------- AFTER ZERO ENTRY REMOVAL: --------
New signal data size: (27599, 23, 256)
New label vector size: (27599,)
Percentage of seizure occurences in cleaned dataset: 16.258%
```

(d)

```
Parameters of eeg-seizure_val_balanced.npz
Files stored in the numpy zip: ['val_signals', 'val_labels']
Size of val_signals: (3546, 23, 256)
Size of val_labels: (3546,)
Percentage of seizure occurences: 50.0%
Percentage of instances that contain at least 1 null channel: 34.123%
-------- AFTER ZERO ENTRY REMOVAL: --------
New signal data size: (2336, 23, 256)
New label vector size: (2336,)
Percentage of seizure occurences in cleaned dataset: 40.882%
```

(e)

```
Parameters of eeg-seizure_test.npz
Files stored in the numpy zip: ['test_signals']
Size of test_signals: (8072, 23, 256)
Percentage of instances that contain at least 1 null channel: 26.4739999
-------- AFTER ZERO ENTRY REMOVAL: --------
New signal data size: (5935, 23, 256)
```

III. Methodology

This section closely follows the python script I created to retrieve and model the data (called ‘project2_seizure_classification.py’).

A. Data Retrieval & Cleaning

Figure 1c, 1d, and 1e depict the basic features of the three datasets. The split between training, validation, and testing is appropriate, with the following proportions of the total data: 76%, 7%, and 16%, respectively. While this data subset is greatly reduced from the original data, it is still very large, with the

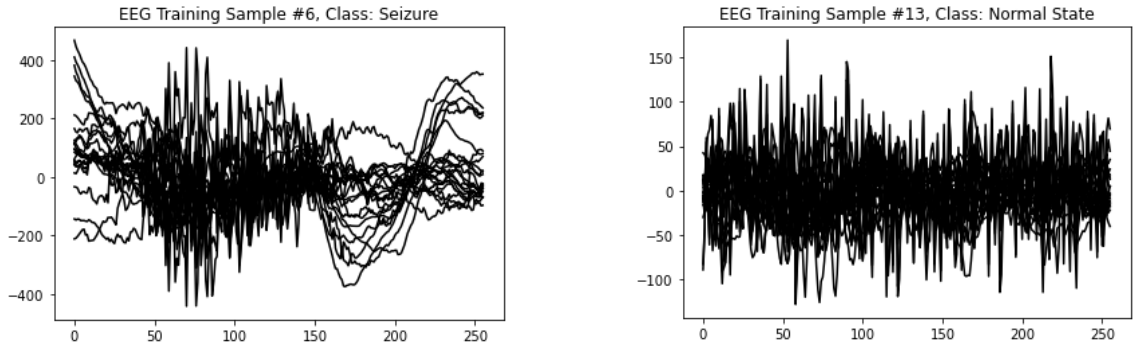
training data alone occupying over 220 million data points. I also made sure that the label vector and signal data had the same number of entries per dataset, as well as confirmed that the 23x256 signal shape was conserved. After checking that the data was shaped appropriately, I calculated the percentage of seizure cases for each dataset to gain a better understanding of the class balance. As shown in Figure 1c and 1d, the class balance was 21.436% for training and 50% for validation (as intended by the authors). To ensure the integrity of the data, I looked at individual signals to see if there were any odd phenomena occurring. I quickly found that there were many cases in which several of the channels were zero vectors. It seems that the author of the EEG Seizure Analysis subset did not account for the mismatched montage between cases (shown in Figure 1a and 1b), and instead cut the first 23 channels off of each signal. However, this would not be appropriate to train on as 1) the data is not structured synchronously and 2) we would be training the model off of zero vectors, which is not representative of actual EEG signals. Because of this, I decided to remove all instances that contained at least one zero-vector channel, so that the only data left followed the same 23-channel montage. This was achieved through the `find_zero` user defined function. The data shape and number of seizure occurrences after the "null" channel removal are also shown in figure 1c, 1d, and 1e. Because of the substantial amount of data provided initially, I was not concerned about the loss of entries. However, the class imbalance worsened after the zero-channel removal, but not to a critical point. Lastly, it is important to note that I did not use the entire cleaned datasets for my model – I ran into several memory errors and cut down the number of entries by a large amount (the exact numbers are detailed in the code).

B. Exploratory Data Analysis (EDA)

I performed minimal EDA on the datasets and mainly focused on visualizing the signals to ensure there weren't any obvious issues. Because I am not very familiar with EEG signals, I couldn't find any characteristics (other than the all-zero channels) that were particularly worrisome, so I kept the data as is for the rest of the model. Instead, I tried to compare instances of seizures vs. non-seizures to get a lead for what technique would be most useful to distinguish them. Figures 2a and 2b show an example of from each class. While there are some clear differences here (i.e. the burst in the seizure class and low amplitude signal in the normal state), not all samples follow these same characteristics, and I found it to be unclear how to categorize the data based solely on raw signals. Because of this, I wanted to implement a technique that would emphasize other features, i.e. in the frequency domain, to provide more relevant information to the model.

Figure 2. EEG signal visualization for instances from each class: seizure vs. non-seizure.

(a) **(b)**



C. Continuous Wavelet Transform

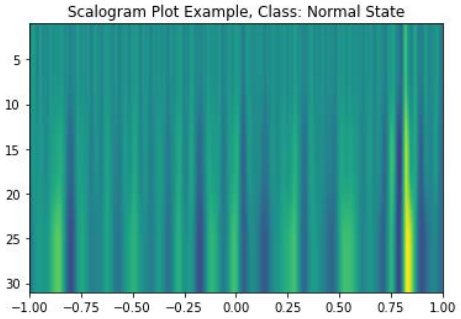
I did some research on different transforms and feature extraction techniques for time signal analysis, and found a few common methods such as Fourier transforms, wavelet transforms, and eigenvector-based methods. One particular paper compared several of these methods and addressed the advantages and

limitations of each. The wavelet transform specifically had multiple desired assets for my application, including that it is well equipped at detecting sudden, transient changes in the signal as well as recognizing irregular data patterns (Al-Fahoum, 2014). Because my data is set up as independent, 1-second signals, being able to detect instantaneous changes is vital. Along with this, detecting irregularities in signals would be immensely helpful in characterizing seizure states (which are inherently irregular). For these reasons, along with the indication that one of the opposing techniques (fast Fourier transform) does poor characterization of epileptic states, I decided to apply the wavelet transform to my data.

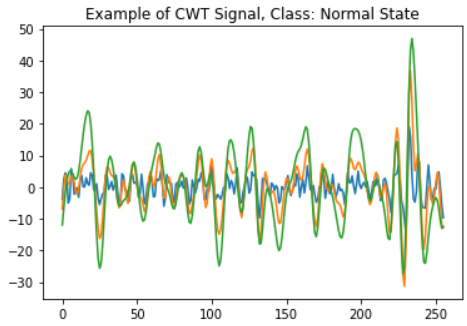
Because a function for the continuous wavelet transform (CWT) was available in SciPy (a well-known library in python that I have used before), I used this method to apply the transform. The function is called `scipy.signal.cwt`, and the user defined function I implemented to apply the CWT is called `cwt_convert`. An important factor I considered was the "width" of the transform, or the dimensionality of the transform output. I was under the assumption that the larger number of dimensions, the more informed the model would be, and therefore the better classification. However, when running tests on several different width values, I found that the lower dimensionality provided better accuracy of the neural network (described in the next section). This is likely because the additional information was capturing "irrelevant" features and therefore negatively impacted the model. For this reason, I chose to use 3 for the width value (which also helped the computational cost of running the model). This decision was also informed by a paper I read, which found that, when classifying seizure occurrences, increasing the number of EEG channels used in the model did not necessarily increase the accuracy in all cases (Moctezuma, 2020). While I did use all of the EEG channels for my model, what I took from this was that other research in this specific area has found similar findings: that less information can provide similar accuracies (or better accuracies) than more information. Lastly, figures 3a and 3b depict the scalogram and plotted output of the CWT for one of the entries. The scalogram is commonly used to visualize wavelet transforms, and some sources have used this image as the input of the neural network. The CWT plot shows the three different output signals (according to the width chosen) of the transform – it can be noted that, as you move into a larger number of outputs, the signals becoming increasingly less smooth (as we can see between the blue and green signals).

Figure 3. Visualization of the CWT output – scalogram vs plotted signal.

(a)



(b)

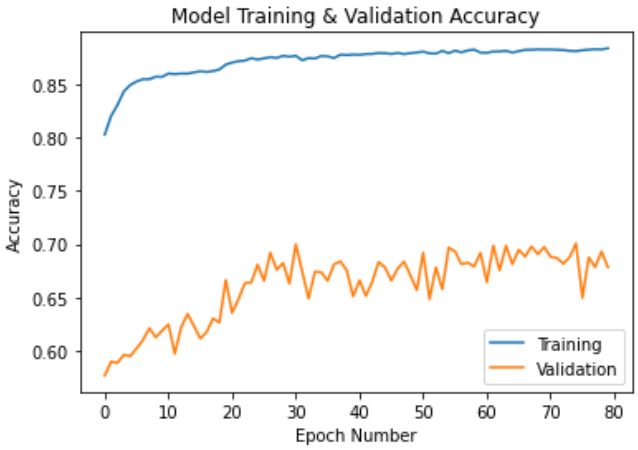


D. Neural Network Model

Lastly, the model I implemented was a very basic neural network using the keras library. This model was heavily adapted from a tutorial on MachineLearningMastery's website (Brownlee, 2022). I decided to use a neural network after seeing several other papers combine it with the CWT scalogram. While I wanted to use the scalogram as the input (neural networks are known to perform well on image data), I was concerned about the amount of storage I would need to save each image. Instead, I flattened the signals down to three dimensions and used that as the input of the network. The exact neural network is detailed in the code, but to summarize, the model used three fully connected layers (of 8 nodes, 4 nodes, and 1 output node) with ReLU and sigmoid activation functions as well as the binary cross entropy loss

function. I also used 80 epochs with a batch size of 10. These values/options were chosen solely on what is standard in the field, and I conducted some limited testing to see if other values would affect the accuracy. Plots of the training and validation accuracies through each epoch are shown in figure 4. The final accuracy for training is 88.65% and for validation is 67.84%. It is important to note that the training accuracy is much greater than the validation, which is an indication of overfitting. Along with this, the validation accuracy seems to oscillate quite a bit, which may indicate some undesirable effect occurring.

Figure 4. Plot of the training and validation accuracy of the neural network.



V. Validation

I implemented limited validation to the project results, mainly due to time constraints. However, I tested the model on the validation set to ensure that the accuracy was sufficient on unseen data. While the validation accuracy was ~20% lower than the training accuracy, it still achieved a value of 67.84%, which shows that the neural network learned to some degree. However, there is definitely still room for improvement. Further validation could be implemented through running the testing data through the model and visualizing where the seizure vs non-seizure predictions are located in the feature space (requiring feature reduction of the signals). While this may not give a concrete answer on the validity of the model, it could provide insights on how well separated the two classes are and whether that is reflected by the model predictions. Otherwise, I think applying the model to a completely different EEG dataset would be helpful in determining it's validity (because this will indicate the generalizability of the model outside of these specific cases).

VI. Justification

A. Programming language

I chose to use python as the programming language for my modeling project for several reasons. Firstly, the dataset I am using stored each of the files as .npz. The .npz file type is a zipped, uncompressed numpy data structure which can hold multiple numpy arrays. I've found that large, multi-dimensional numpy arrays are often difficult and time consuming to convert to other file types (i.e. csv or excel), so I chose to continue working in python to avoid the data conversion issue. Secondly, python has many open-source libraries that make it a strong candidate for this task. For example, tools such as scipy, keras, and matplotlib are well documented and have great functionality. While MATLAB may have libraries geared towards this specific topic (i.e. the signal processing toolbox), python offers similar resources as well (such as MNE or EEGLAB). Lastly, while I am very familiar with both MATLAB and python, I prefer python due to the flexibility in its syntax and plethora of online resources/tutorials.

B. Dataset

I chose to use the Kaggle subset instead of the original database for my model for several reasons. Firstly, I am not familiar with the .edf file format (I briefly looked into it and did not see very many tutorials), so I wanted to avoid the feat of converting these (and the possible errors that could bring). Secondly, the original database is very heterogeneous (i.e. different signal montages, length of recordings, number of recordings, inclusion of non-EEG signals, etc.) which would've required extensive data cleaning on my part. Thirdly, the original database is massive (~40 Gb) which my machine could not handle. While I could of course shorten it, that was one of the main advantages of the Kaggle subset (along with the class balance). Lastly, the original database did not provide an easy format of the class labels. I would've had to sort through the .seizure files, match it to the time points on the .edf files, and create labels for each time point. For these reasons, I found that the Kaggle dataset was the best option for me to use.

VII. Assumptions

Throughout this assignment I noted several assumptions I made regarding the integrity of the data as well as the model structure. Firstly, although I made sure to remove the instances with all-zero channels, I did not do any data preprocessing past this. In doing this I am inherently assuming that there is no extra noise (i.e. 50-60 Hz powerline interference) or other event-related artifacts in the signals. Also, the way the authors structured this data subset did not address which instances belonged to which patients, and it also did not follow the original time series of the signals. This could potentially cause an issue as some of the data points in the validation/testing sets could be chosen from the same patient at a nearby point in time. It can be argued that this would corrupt the integrity of the model accuracy as the model may have been trained on very similar data. Regarding the model, I made several assumptions regarding the parameters and hyperparameters input in the models. A few examples of this include the size of the training/ validation data, number of output dimensions for the CWT, numbers of layers for the neural network, types of layers, number of epochs & batches, loss function, etc. While I did some limited testing for different parameter values (informed by the validation accuracy), I believe looking into the field to justify some of these values and conducting more in-depth testing would be necessary.

References

Adrian-Catalin, B. & Vlad Adam, A. EEG Seizure Analysis Dataset. *Kaggle* (2021). Available at: https://www.kaggle.com/datasets/adibadea/chbmitseizuredataset.

Al-Fahoum, A. S. & Al-Fraihat, A. A. Methods of EEG signal features extraction using linear analysis in frequency and time-frequency domains. *ISRN Neuroscience* **2014,** (2014).

Brownlee, J. Your first deep learning project in python with keras step-by-step. *MachineLearningMastery.com* (2022). Available at: https://machinelearningmastery.com/tutorial-first-neural-network-python-keras/.

Goldberger, A. L. *et al.* Physiobank, PhysioToolkit, and PhysioNet: Components of a new research resource for complex physiologic signals. *Circulation* **101,** 215–220 (2000).

Moctezuma, L. A. & Molinas, M. EEG channel-selection method for epileptic-seizure classification based on multi-objective optimization. *Frontiers in Neuroscience* **14,** (2020).

Shoeb, A. H. Application of machine learning to epileptic seizure onset detection and treatment. *ResearchGate* (2010). Available at: https://www.researchgate.net/publication/43334909_Application_of_machine_learning_to_epileptic_seizure_onset_detection_and_treatment.