Project Medical Wearables
Report

Epilepsy detection using image classification technique

Date:

29 July 2022

Author:

Raghu MS

Institute of Biomedical Engineering/CIT

I. Introduction

Epilepsy: - Epilepsy is a neurological disorder in which brain activity becomes abnormal thus causing seizures during which the person behaves abnormally and sometimes losses awareness.

Symptoms: - Some of the symptoms of epilepsy include:

- Confusion.
- Stiff muscles.
- Movements of the arms and legs.
- Loss of consciousness.

Usually whenever a person starts fitting, then it is necessary to seek immediate medical help, primarily when the following occurs.

- seizure lasts more than five minutes.
- Breathing / Consciousness doesn't return even after the seizure stops.
- When there are multiple seizures continuously.
- Injury occurred during the seizure.

https://brainlaw.com/epilepsy/

A person who is suffering from one of the above conditions will not be able to contact the doctor by themselves and would need other people around to contact the medics. What if the person is in a remote location alone and then a seizure happens it can be severe or many times fatal because of the dependency on other people to be able to identify the person and then to call the medics meaning to say a lot of time is wasted.

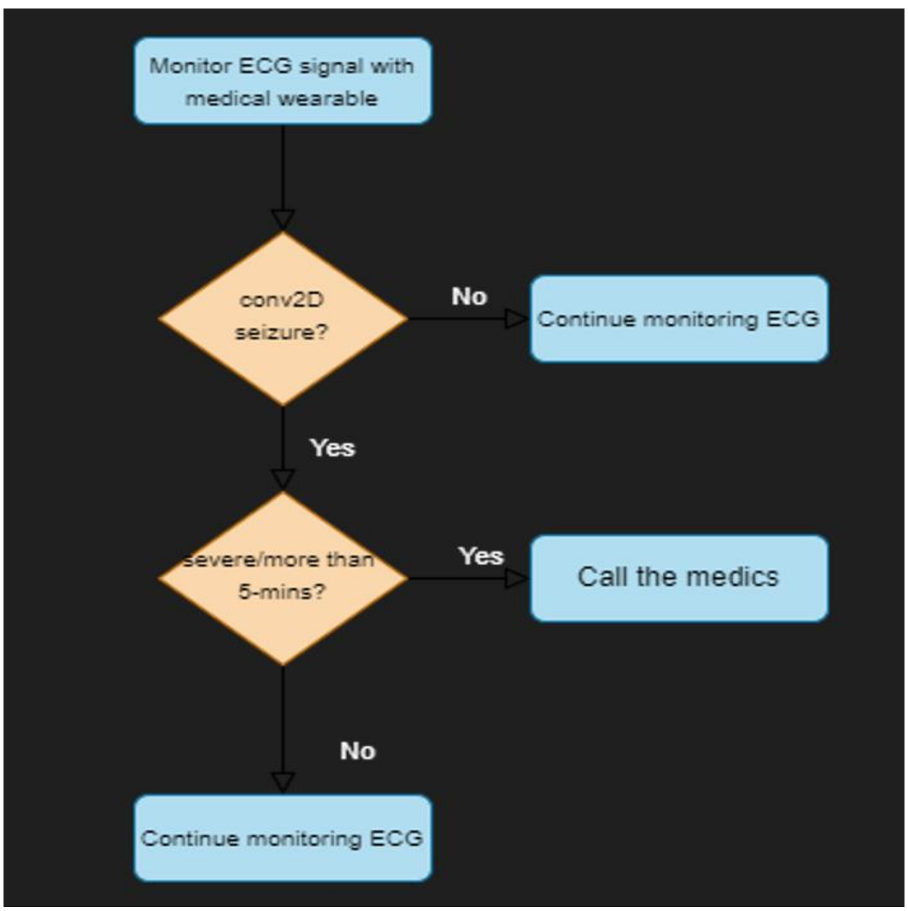
What if we have an automated approach, like a medical wearable that an epileptic person can wear that can identify epilepsy whenever there is a seizure and be able to notify the medics automatically without anyone's intervention that would save a lot of time.

https://news.mit.edu/2016/empatica-wristband-detects-alerts-seizures-monitors-stress-0309

There are many such medical wearables readily available in the market, but most of them are expensive and would cost nothing less than 299$ and is not affordable to everyone. The overall idea of this project is to implement such a medical wearable for a cheaper, more affordable price.

II. Methods

Below is a basic call flow that shows the big picture of the end-to-end project that was in the plan.



The medical wearable (wrist band) would do the following functionalities,

- Monitor the ECG continuously.
- The medical wearable has an inbuilt CPU/GPU on which we have the deep learning algorithm running.
- Deep learning classifies based on the ECG whether the person is having a seizure or not, if there is a seizure activity then there would be a notification that there is a seizure with intelligence to call the nearest medics by itself.

However, considering the time limit implementing the whole idea is not feasible. And hence as part of the research question, we mainly focused on the deep learning algorithm, and we have implemented a basic model wherein the algorithm can classify between a normal and epileptic scenario and display the accuracy.

Also, EEG data was used as it had more samples as part of the data set than the ECG and hence the neural network could be trained better, however, the same procedure can also be used with ECG.

<u>EEG dataset</u>

The original dataset from the reference consists of 5 different folders, each with 100 files, with each file representing a single person. Each file is a recording of brain activity for 23.6 seconds. The corresponding time series is sampled into 4097 data points. Each data point is the value of the EEG recording at a different point in time. So, we have a total of 500 individuals with each having 4097 data points for 23.5 seconds.

These data points are divided and shuffled into 23 chunks, each chunk containing 178 data points for 1 second, and each data point is the value of the EEG recording at a different point in time. So now we have 23 x 500 = 11500 samples of information in the row, each information contains 178 data points for 1 second in the column, and the last column represents the label y {1, 2, 3, 4, 5}.

A pictorial representation of the above-described data set is as below.

	X1	X2	X3	X4	X5	X6	X7	X8	X9	X10	...	X170	X171	X172	X173	X174	X175	X176	X177	X178	y
Unnamed																					
X21.V1.791	135	190	229	223	192	125	55	-9	-33	-38	...	-17	-15	-31	-77	-103	-127	-116	-83	-51	4
X15.V1.924	386	382	356	331	320	315	307	272	244	232	...	164	150	146	152	157	156	154	143	129	1
X8.V1.1	-32	-39	-47	-37	-32	-36	-57	-73	-85	-94	...	57	64	48	19	-12	-30	-35	-35	-36	5
X16.V1.60	-105	-101	-96	-92	-89	-95	-102	-100	-87	-79	...	-82	-81	-80	-77	-85	-77	-72	-69	-65	5
X20.V1.54	-9	-65	-98	-102	-78	-48	-16	0	-21	-59	...	4	2	-12	-32	-41	-65	-83	-89	-73	5

Label 1, here indicates seizure activity and all the other cases indicate nonseizure. We use a simple binary classification technique here, hence we categorize the samples as belonging to either label 1 (seizure) or label 0 (non-seizure). It means that a data point or sample with 1 is said to be epileptic and non-epileptic otherwise.

Using the above dataset, a deep neural network can be trained to classify epileptic and non-epileptic cases in one of the following ways

1) In the time domain
2) By image conversion

As part of the research question in this paper, the image conversion technique was used for the classification.

Working: -

- The data set shown above is initially in the time domain and is available in an excel sheet with a .csv extension, it is first loaded using python.
- The data samples are converted into images with a .png extension. These images consist of epileptic and non-epileptic waveforms that a human cannot classify it is epileptic or not, and this is where the intelligence of the neural network lies where-in it is just not simple image classification like cats and dogs but also some complex images could be classified, the goal of this research question was to find if that was possible or not.
- A conv2d neural network was then used that was trained using the labels extracted from the data set and the converted images and then binary image classification technique was used to classify between epileptic and non-epileptic cases.
- Accuracy and ROC curve was then displayed.

Code walkthrough

Here I have used TensorFlow rather than PyTorch to design the network since TensorFlow provides better visualization, is simpler to use, and has more options with easier debug options.

The code is written on a google Colab notebook.

Below is a brief overview of the code.

1) Libraries

```
import imageio
import glob
import numpy as np
```

These are the libraries needed for the np array and image conversion.

2) Read the data

```
data = "/content/drive/My Drive/medical_project/Epileptic_Seizure_Recognition.csv"
import pandas as pd
df = pd.read_csv(data, header=0, index_col=0)
```

As described above the data set is present in an excel sheet in a .csv format which we load and save in a variable df.

3) Fetch the labels

```
df["seizure"] = df.y == 1
df["seizure"] = df["seizure"].astype(int)
target = df["seizure"].values
df1 = df.drop(["seizure", "y"], axis=1)
```

We have labels y {1,2,3,4,5}. 1 indicates epilepsy, all the other categories are non-epileptic and we categorize them as 0.
So, 1 means epilepsy, and 0 is non-epilepsy, we thus have two labels.

4) Image conversion

```
import matplotlib.pyplot as plt

index = 0
for index in range(200):
  image =  df.iloc[index,0:178]
  plt.plot(range(178), image)
  plt.xlabel("X")
  plt.ylabel("Y")
  plt.title('Epileptic')
  savedir = f"/content/drive/My Drive/medical_project/epileptic{index}.png"
  plt.savefig(savedir)
  plt.close()
```

The above code converts the data samples saved in variable “df” into images with a .png extension and saves it in a newly created directory using the “plt. savefig” function.

5) Load saved images

```
#Load the saved images

imstack = []
index = 0
for im_path in glob.glob("/content/drive/My Drive/medical_project/*.png"):
  #print(im_path)
  if index == 0:
    imstack = imageio.imread(im_path)
    index += 1
  elif index == 1:
     im = (imageio.imread(im_path))
     imstack = np.stack([imstack, im], axis=0)
     index += 1
  else:
     im = (imageio.imread(im_path))
     im = np.expand_dims(im,axis=0)
     imstack = np.concatenate([imstack, im], axis=0)
```

The converted images saved in the directory are fetched and converted into np format to be fed into the neural network.

6) Define, and train the neural network

```
model = Sequential()
model.add(layers.Conv2D(32, (3, 3), activation = 'relu', input_shape = (288 ,432, 4)))
model.add(layers.MaxPooling2D((2, 2)))
#model.add(layers.Conv2D(32, (3, 3), activation = 'relu'))
#model.add(layers.MaxPooling2D((2, 2)))
model.add(layers.Conv2D(64, (3, 3), activation = 'relu'))
model.add(layers.MaxPooling2D((2, 2)))
model.add(layers.Conv2D(128, (3, 3), activation = 'relu'))
model.add(layers.MaxPooling2D((2, 2)))

model.add(layers.Flatten())
model.add(layers.Dense(units = 512, activation = 'relu'))
model.add(layers.Dense(units = 1, activation = 'sigmoid'))
model.summary()
```

```
x_train, x_test, y_train, y_test = train_test_split(imstack, target[:200], test_size=0.2, random_state=42)
```

```
model.compile(loss = 'binary_crossentropy',
              optimizer = 'adam',
              metrics = ['acc'])
```

```
history = model.fit(x_train, y_train, epochs = 10, batch_size=16, validation_data=(x_test, y_test), verbose=2)
```

I have used a Keras library since it provides more features needed to build a network. Since we are working on image classification a conv2D network is made use which also has MaxPooling along with relu activation and a sigmoid activation at the last layer.
Test and training set is created with train_test_split after that the network is compiled and fitted.

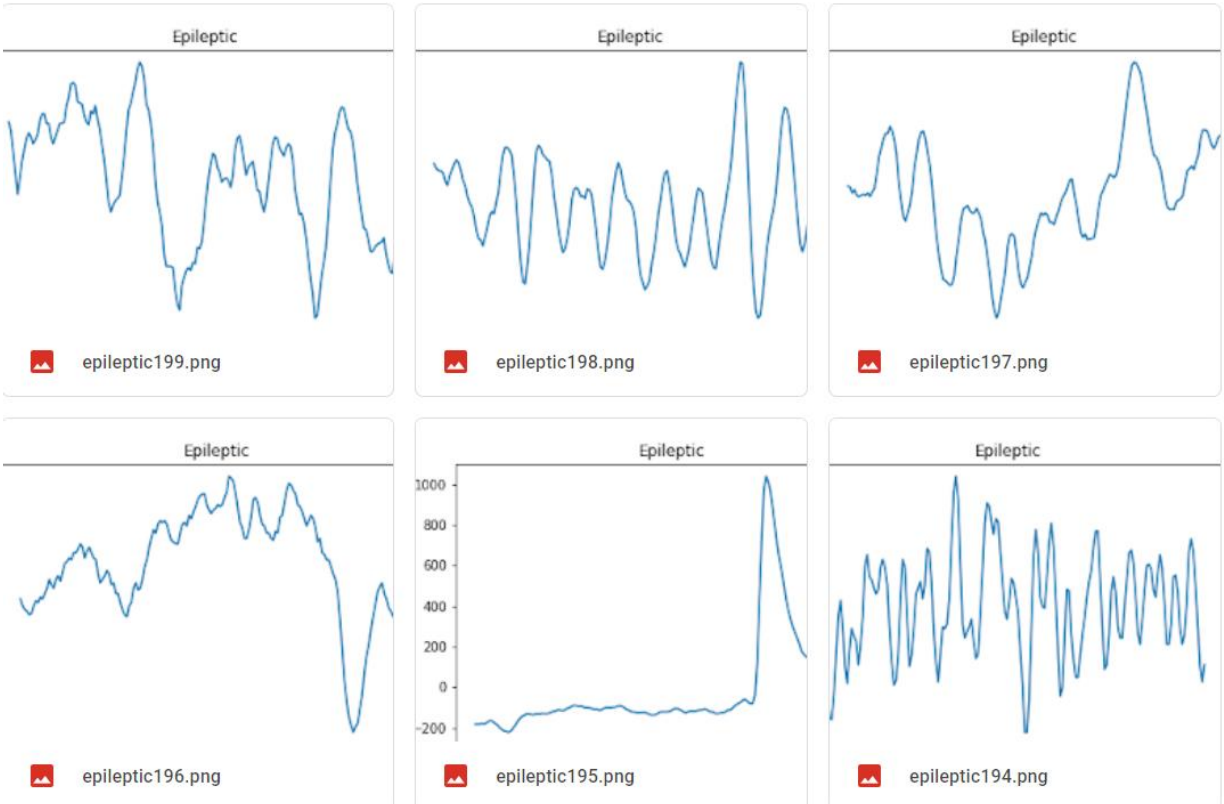
7) Print ROC

```
from sklearn.metrics import roc_curve, auc

y_pred = model.predict(x_test).ravel()
fpr_keras, tpr_keras, thresholds_keras = roc_curve(y_test, y_pred)
AUC = auc(fpr_keras, tpr_keras)

plt.plot(fpr_keras, tpr_keras, label='Keras Model(area = {:.3f})'.format(AUC))
plt.xlabel('False positive Rate')
plt.ylabel('True positive Rate')
plt.title('ROC curve')
plt.legend(loc='best')
plt.show()
```

The last part of the code plots the ROC curve.

III. Results

<u>Converted images</u>

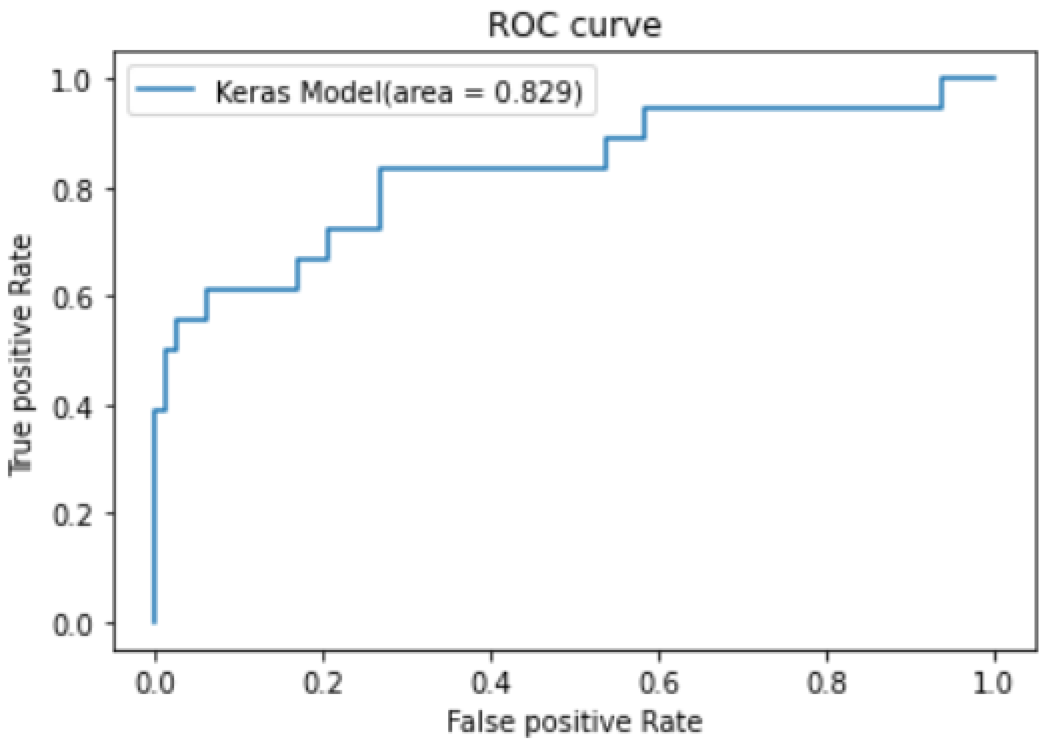


Since we have 11500 samples of data, we would be converting them into 11500 waveform images as shown above and will be saved in a directory, as can be seen, one cannot classify by seeing the images which is epileptic and which are not hence we use a conv2D neural network for classification.

```
Epoch 1/10
10/10 - 55s - loss: 1091.3911 - acc: 0.6812 - val_loss: 15.8133 - val_acc: 0.2000 - 55s/epoch - 5s/step
Epoch 2/10
10/10 - 55s - loss: 32.9791 - acc: 0.6687 - val_loss: 4.6270 - val_acc: 0.2000 - 55s/epoch - 5s/step
Epoch 3/10
10/10 - 50s - loss: 3.3378 - acc: 0.6687 - val_loss: 10.8128 - val_acc: 0.8000 - 50s/epoch - 5s/step
Epoch 4/10
10/10 - 47s - loss: 4.0384 - acc: 0.8125 - val_loss: 0.5690 - val_acc: 0.7250 - 47s/epoch - 5s/step
Epoch 5/10
10/10 - 49s - loss: 0.4134 - acc: 0.8188 - val_loss: 0.4826 - val_acc: 0.8000 - 49s/epoch - 5s/step
Epoch 6/10
10/10 - 47s - loss: 0.3253 - acc: 0.8125 - val_loss: 0.4762 - val_acc: 0.8000 - 47s/epoch - 5s/step
Epoch 7/10
10/10 - 49s - loss: 0.2321 - acc: 0.9312 - val_loss: 0.5190 - val_acc: 0.8000 - 49s/epoch - 5s/step
Epoch 8/10
10/10 - 47s - loss: 0.1361 - acc: 0.9563 - val_loss: 0.6218 - val_acc: 0.8000 - 47s/epoch - 5s/step
Epoch 9/10
10/10 - 49s - loss: 0.0649 - acc: 0.9937 - val_loss: 0.5846 - val_acc: 0.7500 - 49s/epoch - 5s/step
Epoch 10/10
10/10 - 47s - loss: 0.0335 - acc: 1.0000 - val_loss: 0.6458 - val_acc: 0.7750 - 47s/epoch - 5s/step
```

This snippet shows that the network can train for a specified value of 10 epochs successfully.

<u>ROC plot</u>

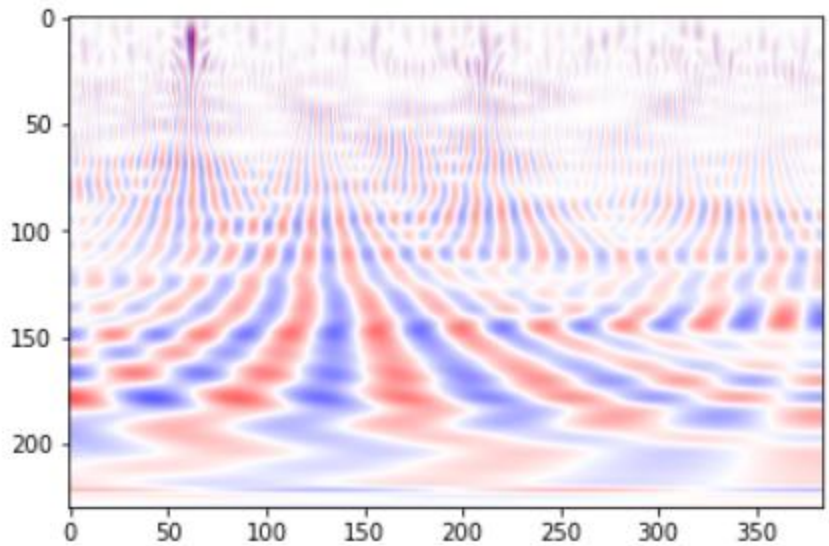


The above figure shows the plot of ROC once the training is completed.

IV. Discussion

- The AUC achieved is 0.829 however, the intention of the research question was not to achieve a very high accuracy rather it was more to try out if this unique idea of conversion of data samples in the time domain to images and then classify epileptic cases using the converted images is possible or not, and as we can see that the idea has indeed worked and the neural network can train and differentiate between epileptic and non-epileptic cases based on images.
- We have used not used all the 11,500 samples for conversion but rather only 500 samples and then converted them into 500 images and then used them for classification, the same technique is also applicable for all the 11,500 samples and we would have to be generating 11,500 images instead.
- The number of epochs used is 20 so that the execution is faster however it can also be made to 100.
- To improve the AUC, further improvements can be done like hyperparameter tuning, changing the number of layers in the neural network, or introducing early stopping.
- When the samples are not converted into images and directly operated in their time domain, the accuracy was found to be 0.994.

<u>Scaleogram approach</u>



- Apart from converting the samples to images with the .png extension, we also tried converting samples into a scaleogram as shown above using the available "cwt" function, the sample code is as shown below

```
from glob import glob
import scipy.io
import torch.nn as nn
import torch
import numpy as np
import mne
from ssqueezepy import cwt
from ssqueezepy.visuals import plot, imshow
import os
import re
import pandas as pd
```

```
test[0][0].shape
```

```
(384,)
```

```
Wx, scales = cwt(test[0], 'morlet')
Wx.shape
```

```
(14, 230, 384)
```

```
imshow(Wx[0])
```

- After converting into a scaleogram the conv2D neural network is used to classify between epileptic and non-epileptic cases, this approach would also result in better accuracy but we could not complete it as it was more complex and time-consuming.

<u>Challenges Faced</u>

Here are some of the challenges that were encountered when implementing this project.

- Uncertainty: - First of all we were not sure if this approach even works and was only a proposal and we only had to implement and find out if it works or not.
- Coding challenge: - Since the idea is novel, the code was not available on the internet for reference and coding had to be done from scratch which was time-consuming and challenging.
- Network wasn't training: - The accuracy was initially very low as the network was not training and was hung, this was due to the wrong activation used at the last layer of the conv2D

network, after making it sigmoid and reducing the number of layers of the conv2D network the network started to train.

- Scaleogram approach: - scaleogram approach did not work straight forward as expected, we were converting the 11,500 samples into “mne” format and then “cwt” was used to generate the scaleogram, however conversion into “mne” format was not needed and this was causing a problem where the samples were cut down from 11,500 to only 800. Later it was figured out that conversion to “mne” is not needed and the samples can be directly converted to a scaleogram.

Drawback

The major drawback of the project is that we convert the samples into images. As can be seen here there are 11,500 samples hence, we would be creating 11,500 images likewise, the number of images that has to be created grows depending on the number of samples, which may not be the best approach when there are memory limitations.

Future work

- The same approach can be tried on ECG data as well, provided there are more data samples available in the dataset as the neural network needs more data to train efficiently.
- Improve the scaleogram technique, since the scaleogram captures more information, it is a good technique to improve accuracy hence it can be taken for completion.
- Implement the code on an actual medical wearable.

V. Links to resources

https://gitlab.uni-ulm.de/project-medical-wearables/ss2022/raghu-mysore-shantharam/final_version_seizure_detection_with_images

(Above link is the git repo link where my code is submitted.

Note :- I’m not submitting the code of scaleogram approach as it was not completely done to avoid confusion)

https://archive.ics.uci.edu/ml/datasets/Epileptic+Seizure+Recognition

https://github.com/christianversloot/machine-learning-articles/blob/main/how-to-use-conv2d-with-keras.md

https://docs.scipy.org/doc/scipy/reference/generated/scipy.signal.cwt.html

https://www.cdc.gov/epilepsy/about/types-of-seizures.htm

https://www.geeksforgeeks.org/keras-conv2d-class/

https://mne.tools/stable/auto_examples/visualization/channel_epochs_image.html