EEG binary classification using convolutional neural networks

Leonardo Clemente and Leonardo Garrido
Tecnológico de Monterrey, Department of Chemistry and Nanotechnology, Campus Monterrey

A series of experiments were done with the objective to perform binary classification of EEG data related to physical reaction tasks. Hyperparameters such as ConvNet's number, Kernel Size, convolutional layer depth, batch size were varied. Pre-processing techniques of data to complement classification performance was included. Experiments show that this type of architecture combinations did not perform well, obtaining accuracy rates that did not go past .70 .

I. OBJECTIVES

Electroencephalography is a technique used to record electrical activity coming from the brain. The way EEG data is obtained is via a set of electrodes scattered around the individual's scalp (it can be either inside or over the scalp surface). These electrodes record electric potential variations. It is of high interest from the scientific community to understand statistical nature of this phenomena in order to support research or generate brain-assisted applications applications (so called BCIs). Interest in BCI is strong due to better recording techniques and novel data analysis techniques being developed. Analyzing EEGs is a big challenge given their non-linear and non-stationary nature.

EEGs are usually noisy. noise can come either from the inside (regarding electric activity we're not interested in but still record) or the outside (corresponding to room noise which origins on sources such as nearby electronic devices).

Eeg data is multi-dimensional which might (or not) be highly redundant. The fact that data is highly dimensional means that we also need much more information to optimize our model. This is a problem because data measurements regarding the same experiment coming from different individuals most likely are considered statiscally different datasets (even when data comes from the individual but gets recorded on different sessions this might hold true as well).

For EEG classification, vast majority of classifying techniques related to neural networks involve the use of multilayer perceptrons. This network architecture was described as not being able to account temporal information and 'unstable', given the fact that changes in the triaining set may highly affect performance. [1]

Recent changes and breakthroughs in neural network classiffication (see work from Andrew ng, Andrej Karpathy, Susan sie, etc) turn the attention of researchers back to these models in hopes of producing better models for their BCI applications. New neural network architectures (deep learning) become good prospects in EEG classification given their differences compared to the MLP. Convolutional networks specifically become the aim of this activity. CNNs are neural network models which, in comparison to the MLP, have analyze data in a different fashion. CNNs are suitable for analyzing data that can be input as grid-like topologies [2]. Operationally, CNNs work based on the convolution operation

$$s(t) = (x * w)(t) = \sum_{a=-\infty}^{\infty} x(a)w(t-a)$$

Convolution operates in two arbitrary functions x and w. x and w are discrete sequential functions and can be multidimensional. In practice, x is defined as the input function and w as the 'kernel'. Both input and kernel are described as matrices with different sizes (size(x) ¿ size(w) in every matrix dimension). the convolution operation provides the model with one important characteristic called **Parameter sharing**. this means that models use more efficiently their paremeters in the sense that if they learn something that is useful (for example, lets say we know that one important feature from our EEG has an specific voltage variation shape), the parameters that learnt this abstraction are not restricted one specific area (remember data input to CNNs is in grid-like topology, one good way to see data is as a simple pixel image with dimensions nxm) of the data input. This might be useful based on the fact that EEG signals we're interested in do not occur synchronously and sometimes they're more active in specific head regions, which is mapped to activity only in specific subsets of electrodes.

The aim of this project is the development of a brain-computer interface using deep learning architectures. Motivation comes from australia's IBM research group publication[2] Free EEG databases were used for the experiment. The first corresponds to IBMs research laboratory[1] and the second corresponds to Stefan Haufe research group.

IBMs EEG dataset corresponds to a hand-squeezing experiment. It provides a total of 3 classes (left-squeeze, right-squeeze and no-squeeze) with 1868 experiment samples. 1305 samples are used (corresponding to the no-squeeze and left-hand squeeze classes). Data was recorded using 46 EEG electrodes. 100 samples of eeg recording are provided per sample (a total of 4600 numbers) Each data sample is structured as a 46*100 matrix. A more in depth explanation about the dataset and its pre-processing can be found in [2]

Haufe's dataset corresponds to an emergency braking test. Dataset comes in a long-record format (around 2:30 hours of recording per patient) with time cues for each of the classifications. Classifications are related to normal car driving, car braking, car holding, car collision, and

emg reaction data. Data is sampled at 200 hz. 'normal car driving' and 'emg reaction' were chosen for classification. a total of 1239 samples with 1 second of data each were created. data was decimated from 200 values down to 100 values using MATLAB.

II. DATA PRE-PROCESSING

main pre-processing techniques were separately applied to both datasets.

1. Mean substraction: Transform all data to make it zero-mean channel-wise.
2. Data normalization: scaling all data channels into the same magnitude order and prevent prioritazion during the learning process.
3. Median filtering: Technique to remove outlier spikes and noise. Basically consists on dividing a vector of data V with n number of samples in n subsets of data called windows.The ith window has the ith sample in the middle of the subset and all other values (window size) correspond to its sequential neighbors. for each of this window we choose the median value and replace the ith sample from V with that value.
4. Moving average filtering: Another smoothing technique that consists in separating a vector V of data of n samples in n different windows. in this technique we replace the ith sample from V with the average value of each window. Other way of looking at if is a convolution operation of the vector V with a n sized window that mo

A graphic representation of filtering effects is shown in the next set of figures (1,2 and 3)

III. NEURAL NETWORK ARCHITECTURE

Convolutional network forward propagation is described as follows.

1. Convolution with RELU, different kernel sizes
2. Pooling (Max) size 2x2
3. Convolution with RELU, kernel size 5x5
4. Pooling (Max operation) size 2x2
5. Fully connected layer, 300 neurons
6. Output classification, 2 neurons

First number of layers and kernel size was decided based on Nurse's architecture. Slight modifications are intorudced during experiments to analyze other options. A graphic representation of the net is described on figure 12

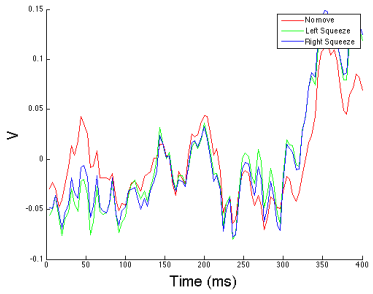

FIG. 1: EEG data prior to pre-processing

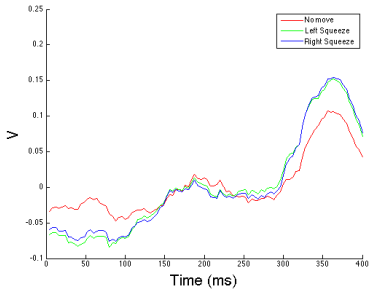

FIG. 2: Median filtering effect on EEG data for one channel.

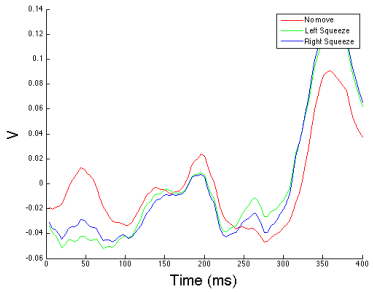

FIG. 3: Moving average effect on EEG data for one channel.

TABLE I: My caption

Hyper parameter	Values
ConvNets number	2,3,4
Kernel size*	3x3,5x5,10x5,10x10
Convolutional layer depth	40,50,80,100,10
Fully connected layer	100,300,500 neurons
output classification	1 or 2 neurons
Pooling type	Max, average
Batch Size	20,40,80,160

IV. EXPERIMENTS

Different convNet architectures were used to classify both datasets. Hyperparameters varied within the experiments are

This amounts to a total o 2880 experiments. An extra

experiment using an artificial signal was designed as a toy exercise to observe's architecture's performance on fully characterized data. artificial signal consisted on a sinusoidal pulse inserted within raw EEG data. Architecture found a good representacion and obtained almost .99 accuracy results. An image from one sample's channel and its activations can be found in figures 5 and 6.

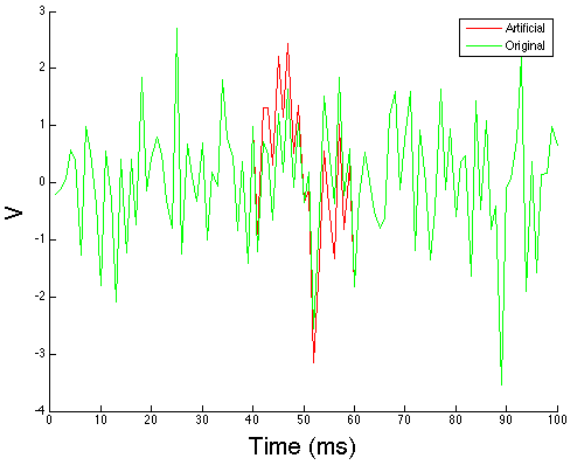

FIG. 4: Artificial signal inserted within the EEG data. Signal's amplitude was scaled to the point it fit channel's amplite

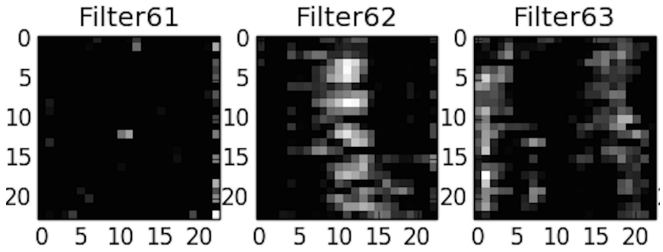

FIG. 5: Three map activations from the first layer of the convNet trained with the artificial dataset.

V. RESULTS

In general, the convnet architecture was unable to obtain a good representation in both datasets. Data regularly stopped at a plateau at recent steps Exceptions are described separately

datasets showed little to no improvement from preprocessing. In average, runs amounting for all types of pre-processing didn't go past .60 accuracy rate. This behavior may be attributed to different causes such as incorrect pre-processing application (techniques were used channel to channel for all the samples. 2D filtering is yet to be investigated) Another reason might be attributed to the fact that dataset is small and might not contain enough data for abstraction.

Batch size and learning rate substantially affected architecture's way of learning. Experiments with training batch sizes smaller than 80 weren't unable to go past .55 accuracy rate.

Kernel size and convNets depth played no discernible role. Visual analysis of activation maps showed that bigger the number of layer maps accounted to redudancy (have two or more maps that show similar abstraction)

A. Haufe's dataset

Convnet (2 conv layers, conv kernels: 5x5, 5x5 , pooling layers: max 2x2, max 2x2, FC 300 neurons, output 1 neuron) Dataset contained information from EEG, EOG, EMG and pedal deflections.

Net reached almost perfect score with the samples. convNet activations show diagonal features (figures 10 through 12) learned over the first layer (figure 7 through 9). It is not clear the specific reason of this feature. It is not necessary. Second convolutional layer present more interesting activations. figure 5 and 6 show the net putting specific attention to the last input activations. According to haufe's data structure these belong directly to the input deflections (not EEG). It is suspected that the net only needed this type of information to perform the classification.

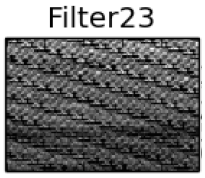

FIG. 6

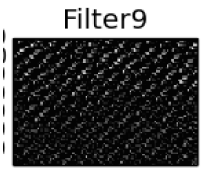

FIG. 7

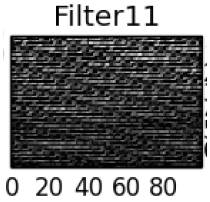

FIG. 8

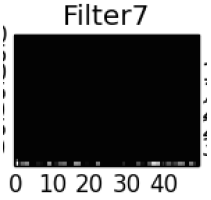

FIG. 9

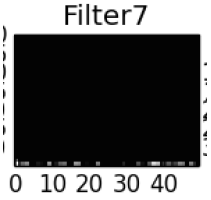

FIG. 10

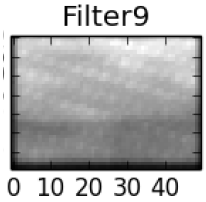

FIG. 11

VI. CONCLUSION

Convolutional net wasn't unable to find a good representation of data to classify the datasets. Causes relating this low performance are attributed to architecture's high capacity compared to dataset's limited information availability and high noise rates. Pre-processing techniques applied on data showed little to no performance improvement Hyper-parameter variations did not affect classification either. This type of behavior may suggest that more information on the dataset is needed. Classification of EEG data creates a bigger challenge than other type of classifications based on the fact that datasets may come with high noise and signals looked to be classified may not be recovered fully after preprocessing. Data availability is also strictly limited by the way EEG collecting experiments are done.

VII. APPENDIX

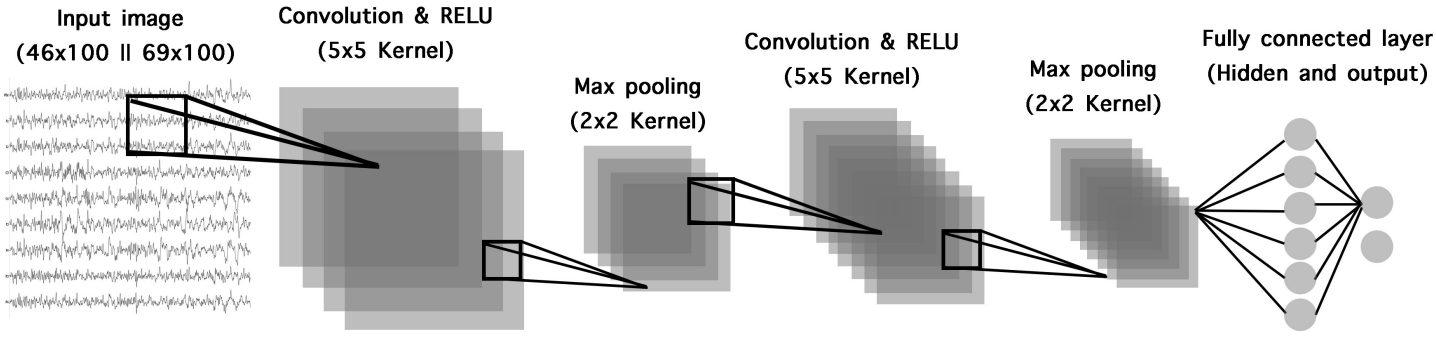

FIG. 12: Convolutional network architecture for the experiment.

VIII. REFERENCES

[1] Lotte et al.(2007).A review of classification algorithms for EEG-based brain-computer interfaces. Retrieved from https://hal.archives-ouvertes.fr/inria-00134950/document [2] Nurse et al.(2016) Decoding EEG and LFP Signals using Deep Learning:Heading TrueNorth. Retrieved from http://researcher.ibm.com/researcher/files/au1-sharrer/computing-frontiers-trends-paper-camera_ready%2002%2029%2016_copyright%20update.pdf [3] Karpathy. "ConvNets : Setting up the data and the network". Retrieved from http://cs231n.github.io/neural-networks-2/