


APPROVAL SHEET

Title of Thesis: Efficient Artifact Identification in multi-channel EEG Data

Name of Candidate: Mohit Khatwani
Master of Science, 2019

Thesis and Abstract Approved: _____


Tinoosh Mohsenin
Assistant Professor
Department of Computer Science and
Electrical Engineering

Date Approved: 07/30/2019

ABSTRACT

Title of dissertation: Efficient Artifact Identification in
Multi-Channel EEG Data

Name of Student: Mohit Khatwani, Computer Science, 2019

Dissertation directed by: Professor Tinoosh Mohsenin
Department of Computer Science and
Electrical Engineering

Neurophysiological signals such as Electroencephalogram (EEG) can be used in a variety of purposes including detecting fatigue, stress, brain disorders, brain-computer interfaces (BCIs), or building better models of human variability and human brain. However, EEG signal is frequently contaminated with other sources not related to brain activity. These artifacts may emerge from external sources such as eye blinks, muscle head movement. In this work, we examine the problem of identifying multiple artifacts on continuous multi-channel EEG data. We first propose a Convolution Neural Networks (CNN) architecture for binary detection of EEG artifact, then further modify the architecture for classifying multiple types of artifacts. The proposed models do not need expert knowledge for feature extraction or pre-processing of EEG data and have a very efficient architecture implementable on mobile devices. We further enhance the architecture to reduce the computation and parameter size through hyperparameter optimization and the use of Depthwise and Separable convolution layers. We propose five different CNN models and evaluate

and compare against each other in terms of accuracy, weight parameters, and computation requirements. Our optimized network achieves 94.17% classification accuracy averaged across 17 patients and 9 artifact classes. Compared to the original CNN based architecture, the optimized architecture provides 4.2x and 2.7x less parameters and computation, respectively and has 17.5% higher accuracy. The proposed model was also evaluated on an EEG dataset collected in our lab using a 14-channel Emotiv EPOC headset, and achieves 93.5% accuracy in detecting eye blink artifact.

Efficient Artifact Identification in Multi-Channel EEG Data.

by

Mohit Khatwani

MS Thesis, 2019

Advisory Committee:

Dr. Tinoosh Mohsenin Chair/Advisor

Dr. Tim Oates

Dr. Hamed Pirsiavash

Dr. Nick Waytowich

To my parents Sunil Khatwani and Kajal Khatwani for their constant support and unconditional love.

ACKNOWLEDGMENTS

I sincerely like to thank everyone behind the successful completion of my thesis. I would like to express sincere gratitude to my advisor Dr. Tinoosh Mohsenin. I thank you from the bottom of my heart for the foresight of my research, immediate response whenever approached for help and constantly belief that i could achieve it. Without her guidance, support and foresight my thesis would not have been possible. I would also like to thank my committee members, Dr. Tim Oates, Dr. Hamed Pirsiavash and Dr. Nick Waytowich for all their guidance and feedback. I thank my fellow EEHPC lab members particularly Morteza Hosseini, Bharat Prakash, Hiren Panhelia and Mark Horton for their contributions and help with this thesis. Finally, I like to thank my family and friends for their constant support and encouragement. It is because of all of you I am able to achieve this.

TABLE OF CONTENTS

DEDICATION	ii
ACKNOWLEDGMENTS	iii
LIST OF TABLES	vi
LIST OF FIGURES	vii
Chapter 1 INTRODUCTION	1
1.1 Introduction	1
1.2 Contributions	3
1.3 Organization of Sections	3
Chapter 2 BACKGROUND AND RELATED WORK	4
2.1 Deep Learning	4
2.1.1 Convolutional Neural Network	4
2.2 EEG Signal Classification	11
2.2.1 Traditional Approaches	12
Chapter 3 ARTIFACT DETECTION IN EEG SIGNAL USING CNN	15
3.1 Problem Definition	15
3.2 High Variability in Data	16
3.3 Collection of Dataset	17
3.4 Preprocessing of EEG Data	18
3.5 Proposed DNN Architecture	19
3.6 Classification Analysis and Results	20
Chapter 4 EFFICIENT CNN FOR ARTIFACT IDENTIFICATION	24
4.1 Visualization of EEG signals	24
4.1.1 EEG signal plot	24

4.1.2	Topographical Plot	29
4.2	Implementation and Results	31
4.2.1	Architectures	31
4.2.2	Optimized Separable + DepthWise	35
4.2.3	Optimization of Architectures	36
4.2.4	Classification Accuracy	37
4.3	Experimental Study	39
Chapter 5	CONCLUSION AND FUTURE WORK	41
5.1	Conclusion	41
5.2	Future Work	42
5.2.1	More Datasets	42
5.2.2	More Model Search	42
5.2.3	Transfer Learning	43
Appendix A	APPENDIX	45
A.1	Code for data extraction of every patient	45
A.2	Code for plotting Topographical Maps	47
A.3	Model Summary	49
A.3.1	Traditional Convolutional Neural Network	49
A.3.2	DepthWise Convolution Neural Network	50
A.3.3	Separable Convolution Neural Network	51
A.3.4	Depthwise + Separable Convolution Neural Network	52
REFERENCES	55

LIST OF TABLES

2.1	Parameter and Operation equations for different types of convolution layers.	11
3.1	Description on nine artifacts performed by every patient.	18
3.2	Detection accuracy and precision for different artifacts averaged for all patients using leave one subject out cross-validation technique among 9 patients. Results are compared against Auto-Regressive baseline technique [Lawhern <i>et al.</i> 2012]. Values in the parentheses indicate the standard deviation. Asterisks (*) indicate significant accuracy improvement over the AR technique.	21
4.1	Comparison of computation in each layer for EEGNet and its optimized architecture.	37
4.2	Comparison of parameters, computations and average accuracy (17 patients and 10 classes which includes 9 artifacts and 1 plain signal) of different model configurations. All the models classify 9 different artifacts with test data and training data for the same patient	38

LIST OF FIGURES

2.1	Convolution operation over the input.	6
2.2	Traditional convolution layer with input shape of $D_f \times D_f \times M$ and output shape of $D_p \times D_p \times N$	7
2.3	Depthwise convolution operation.	8
2.4	Separable convolution layer which is a combination of depthwise convolution and pointwise convolution.	10
3.1	Proposed CNN Architecture which consists of 5 layers. 2 convolutional layers, 2 max pool layers and 1 softmax layer.	20
3.2	Flow chart of the heuristic hyperparameter search.	23
4.1	Visualization of jaw movement related artifacts performed by patients. Instructions were given to patients every 2 secs and it was advisable to perform the task in the first second. Vertical line indicates the start of experiment. Artifact names for given artifact codes are explained in Table 3.1.	25
4.2	Visualization of ocular related artifacts performed by patients. Vertical line indicates the start of experiment. Artifact names for given artifact codes are explained in Table 3.1.	26
4.3	Visualization of muscular movement related artifacts performed by patients. Vertical line indicates the start of experiment. Artifact names for given artifact codes are explained in Table 3.1.	27
4.4	64 EEG signal channel locations.	28
4.5	TopoMap figures of jaw related artifacts	29
4.6	Topomap graph for ocular artifacts	30
4.7	Topomap graph for muscular artifacts	30
4.8	Traditional CNN Architecture for artifact identification which consists of 5 layers. 2 convolutional layers, 2 max pool layers and 1 softmax layer. Total parameters required for this architecture is 24,842.	32
4.9	33
4.10	EEGNet architecture which uses combination of depthwise and separable convolution layers. Total parameters required for this architecture is 5,002.	35
4.11	Optimized EEGNet architecture with filter size of 64×16 in first convolution layer which reduces the computation by 3.64x . Total parameters required for this architecture is 9,194.	36
4.12	Class-wise and average accuracy for proposed models. Using only separable convolution layers provides least average accuracy. Using combination of depthwise and separable convolution layers proves to be most beneficial in artifact identification.	39

4.13 Emotiv EPOC 14 channel headset capturing EEG data, Eye blink is performed once every two seconds.	40
--	----

List of Abbreviations

CNN	Convolution Neural Network
EEG	Electroencephalogram
DNN	Deep Neural Network
ICA	Independent Component Analysis
MLP	Multi Layer Perceptron
MNIST	Modified National Institute of Standards and Technology
ReLU	Rectified Linear Unit
CUDA	Compute Unified Device Architecture

Chapter 1

INTRODUCTION

1.1 Introduction

Electroencephalography is a method of recording non invasive electrical signals of brain through electrodes. EEG signals can be easily contaminated through noise originating from line electrical noise, muscle movement or ocular movements. These distortions in the EEG signals can be referred to as artifacts. These artifacts can lead to difficulties in extracting underlying neuro information. Removal of artifacts from EEG signals can be beneficial in various applications [Delorme, Sejnowski, & Makeig2007] [Iriarte *et al.*2003] [Nuwer1988]. The classification of brain signals recorded by imaging devices using machine learning approaches is a very powerful tool in many of these areas of research. For example, machine learning techniques show promise in the early detection of Alzheimers or giving warning before an epileptic seizure. These techniques are already being used in devices such as the P300 speller to provide a communication device for the severely handicapped.

Artifacts can overlap the EEG signal in spectral as well as temporal domain which

turns out to be difficult for simple signal processing to identify artifacts [Islam, Rastegarnia, & Yang2016]. A method involving regression which subtracts the portion of signal with reference signal was widely used. Problem with this method was that it needs one or more reference channels. As of now, the independent component analysis technique (ICA) is one of the most frequently used method for EEG artifact detection [Jafari *et al.*2017]. The ICA is a denoising technique that involves the whitening of data and separation of linearly mixed sources [Winkler, Haufe, & Tangermann2011] [Jung *et al.*1998] [Delorme, Sejnowski, & Makeig2007]. A major drawback of this method is that it is not fully automated and still requires an expert person to label and tag the EEG artifacts. ICA is computationally intensive [Jafari *et al.*2017] which makes it unsuitable for use in embedded hardware applications.

Convolution Neural Networks (CNNs) have been successfully used in computer vision tasks such as image classification. Advantage of using CNNs in these tasks is that it doesn't need hand crafted features from people, it learns them automatically using raw data. Time series signals from all EEG channels are combined (2D image) and passed to these convolution layers [Jafari *et al.*2018]. These features learnt using convolution layers are then passed on to Multi Layer Perceptron (MLP) to perform final classification. One major disadvantage of using CNNs is its high memory and computation requirements [Zheng *et al.*2014]. Depthwise and separable convolution layers to create memory and computationally efficient CNNs which are used for multiple artifact identification from continuous multi-channel EEG signal.

1.2 Contributions

Main contributions of this work are listed below:

- Propose an optimized architecture with reduced number of computations for multiple artifact identification.
- Evaluate and compare optimized CNN with various other architectures in terms of identification accuracy (multi-class), number of parameters, and total number of computations.
- Compared the proposed method for accuracy, f-1 score to show improvements over existing work.

1.3 Organization of Sections

The rest of the thesis is organized as follows: Chapter 2 will review the related works that forms the basis for this research. Chapter 3 will define the specific problems and approaches used in this thesis. Chapter 4 will include a summary of the datasets used, the implementation of the techniques used, and the results of the experiments. Chapter 5 will present a summary of the findings and a discussion on future work.

Chapter 2

BACKGROUND AND RELATED WORK

2.1 Deep Learning

2.1.1 Convolutional Neural Network

Deep learning is a subfield of machine learning that has evolved out of the traditional approaches to artificial neural networks. Artificial neural networks are computational systems originally inspired by the human brain. They consist of many computational units, called neurons, which perform a basic operation and pass the information of that operation to further neurons. The operation is generally a summation of the information received by the neuron followed by the application of a simple, non-linear function.

The ability to train a network with multiple layers was crucial for the continued development of neural networks. Backpropagation is a method which is used to train neural network with multiple layers. Backpropagation allows multiple layer networks to be trained by iteratively using the gradient of a loss function with respect to the weights of the network to assign gradients to previous neurons in the network. Due to this new forms

of neural network became significant. In [Hinton, Osindero, & Teh2006], authors showed promising results in classification of the Modified National Institute of Standards and Technology handwritten dataset (MNIST).

2.1.1.1 Stochastic Gradient Descent Gradient descent is a first-order iterative optimization algorithm used for finding the minimum of a function. In neural networks, it is used in conjunction with backpropagation to update the weights in the network. It is formally defined with the update rule:

$$(2.1) \quad x_k = x_{k-1} - \nabla f(x_{k-1})$$

Here, x_k is the current point and x_{k-1} is the previous point and $\nabla f(x_{k-1})$ is the gradient to optimize the function. This algorithm is much more easier for training big datasets efficiently. Other SGD derived algorithms, such as Adaptive Moment Estimation (ADAM), calculate per parameter adaptive learning rates, enabling even more efficient training, at the cost of memory.

2.1.1.2 Activation Function When a gradient is propagated back through the network the errors are multiplied by values which are between 0 and 1. This leads to problem where values trend towards 0 exponentially, which further makes the gradients close to 0 in early layers of multi layer perceptron.

Sigmoid function is most prominently used activation function. Sigmoid function is

defined as : $\frac{1}{1 + e^{-x}}$. Another very famous activation function is ReLU. ReLU pushes the input which are less than zero to zero. This makes the large value gradients vanish less quickly which benefits in vanishing gradient problem.

2.1.1.3 Types of Layers With advancements in deep learning, new types are layers are being introduced with various advantages. In traditional neural networks, focus is mainly on fully connected layers. In fully connected layers every neuron is connected to every other neuron in the next layer.

With introduction of convolutional neural network, the filters are convolved over the input. This weights sharing structure of convolution layer allows to learn various pattern in the input data. Figure 3.2 shows the convolution of filter, where each version of filter can create a different processed version of the input.

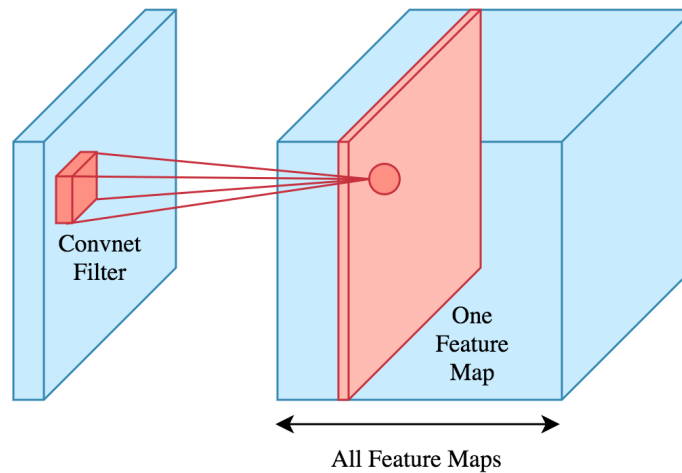


FIG. 2.1. Convolution operation over the input.

- Traditional Convolution layer:** In traditional convolution layer if the input is of size $D_f \times D_f \times M$ and filter applied to this input is of size $D_k \times D_k \times M$ then output of this layer without zero padding applied is of size $D_p \times D_p \times M$. In this layer the filter convolves over the input by performing element wise multiplication and summing all the values. A very important note is that depth of the filter is always same as depth of the input given to this layer. Total number of multiplication operations required for this layer is $M \times D_k^2 \times D_p^2 \times N$.

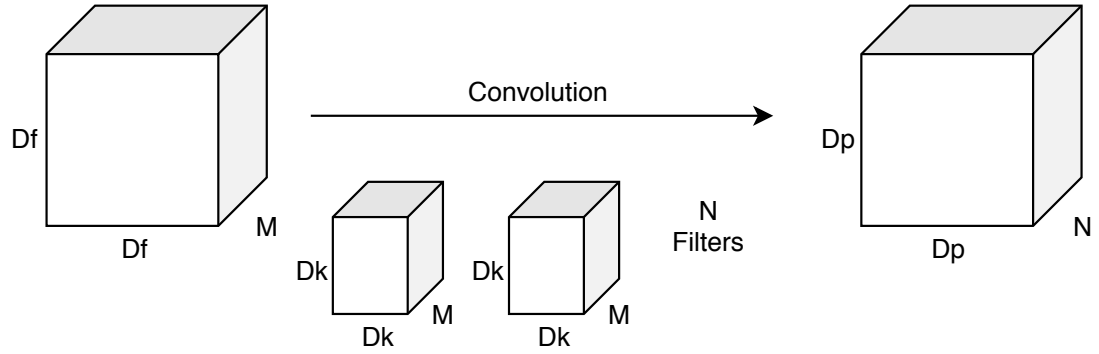


FIG. 2.2. Traditional convolution layer with input shape of $D_f \times D_f \times M$ and output shape of $D_p \times D_p \times N$.

- DepthWise Convolution layer:** For every input of size $D_f \times D_f \times M$ we have M filters of shape $D_k \times D_k$ and depth 1. $D \times M$ filters are used in depthwise convolution where D is the depth multiplier.

As every input channel in depthwise convolution has a separate filter. This leads to number of multiplication equal to $N \times D_k^2 \times D_p^2$ which is $N \times$ less than traditional convolution. [Howard *et al.* 2017] [Chollet 2017].

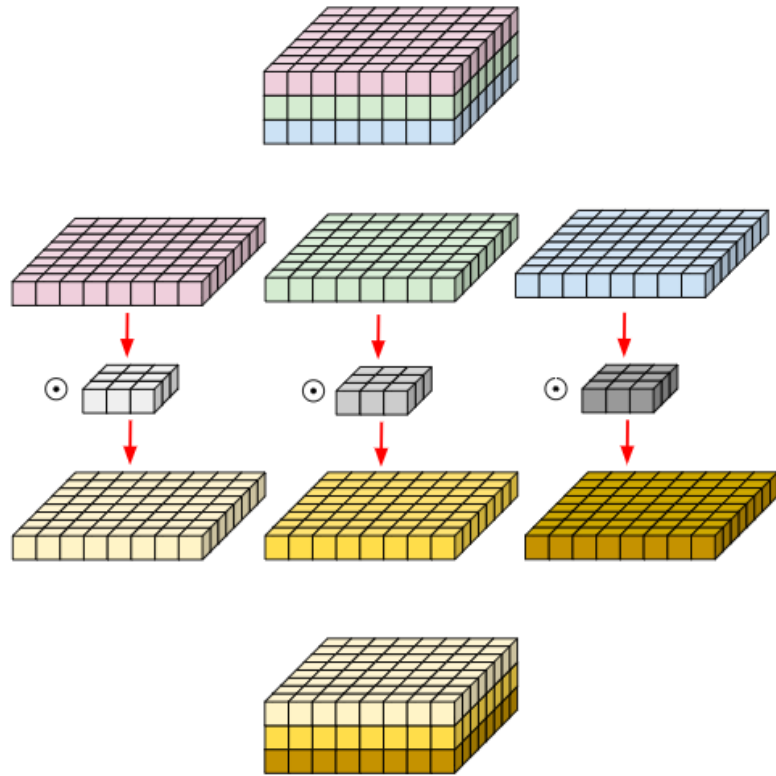


FIG. 2.3. Depthwise convolution operation.

- **Separable Convolution layer:**

Separable convolution is a combination of depthwise and pointwise convolution [Kaiser, Gomez, & Chollet2017] [Podlozhnyuk2007]. In depthwise operation, convolution is applied to a single channel at a time unlike standard CNNs in which it is done for all the M channels. So here the filters/kernels will be of size $D_k \times D_k \times 1$. Given there are M channels in the input data, then M such filters are required. Output will be of size $D_p \times D_p \times M$.

A single convolution operation require $D_k \times D_k$ multiplications. Since the filter are slided by $D_p \times D_p$ times across all the M channels. The total number of multiplications for one depthwise convolution operation comes to be $M \times D_p^2 \times D_k^2$.

In point-wise operation, a 1×1 convolution operation is applied on the M channels. So the filter size for this operation will be $1 \times 1 \times M$. If we use N such filters, the output size becomes $D_p \times D_p \times N$.

A single convolution operation in this requires $1 \times M$ multiplications. The total number of multiplication for one pointwise convolution operation is $M \times D_p^2 \times N$. Therefore, total number of multiplications required for one depthwise separable convolution operations is $M \times D_p^2 \times D_k^2 + M \times D_p^2 \times N$.

2.1.1.4 Comparison of Layers In Table 2.1, we compare the parameter equation of different convolution layers used in our networks. Here $D_k \times D_k$ is the size of filter used, M is number if input channels and N is number of output channels.

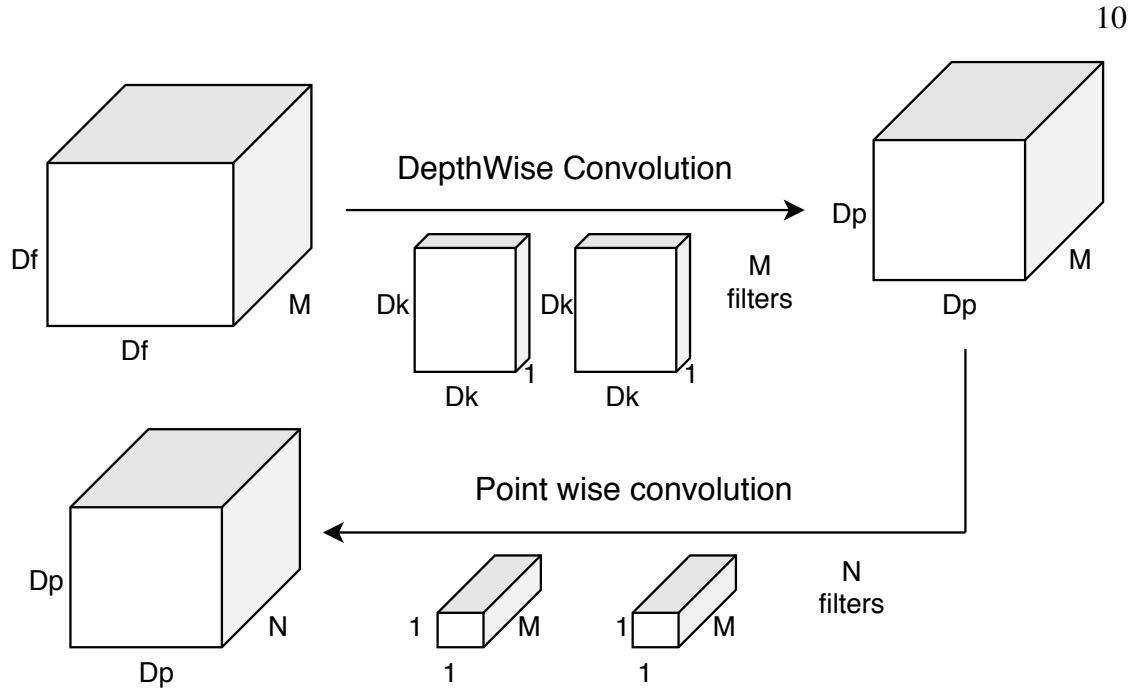


FIG. 2.4. Separable convolution layer which is a combination of depthwise convolution and pointwise convolution.

2.1.1.5 Regularizer and Dropout Overfitting has been a constant threat in training big datasets using deep learning. Regularization techniques prevent the model from becoming too complex and specific to the training data, thus reducing the tendency to overfit. L2 regularization is a standard technique used in many forms of optimization, in which the squared sum of the weights is applied as a penalty to the optimization function. This makes the models more generalizable.

Another technique used is Dropout layer, random set of neurons are remove from the layer where it is applied. This allows a single model to act as as ensemble method for training on the data. It prevents memoization as different set of neurons are active for every training phase.

Table 2.1. Parameter and Operation equations for different types of convolution layers.

Convolution Layers	Parameter	Operation
Traditional	$M \times D_k^2 \times N$	$M \times D_k^2 \times D_p^2 \times N$
DepthWise	$D_k^2 \times M$	$M \times D_k^2 \times D_p^2$
Separable	$D_k^2 \times M + M \times N$	$D_p^2 \times D_k^2 \times M + M \times D_p^2 \times N$

2.1.1.6 Powerful graphics cards and training on GPU NVIDIA debuted the Compute Unified Device Architecture (CUDA) API to that enables direct and easy programming for their graphical processing units (GPUs). CUDA gained popularity for use in highly parallelizable tasks able to take advantage of the large number of cores in GPUs. The hardware and software advances in GPU programming were critical and have made GPU implementations tractable. Various networks in deep learning require large matrix operations which are implemented efficiently on GPUs.

2.2 EEG Signal Classification

The classification of EEG signals presents several challenges that make it a uniquely difficult problem in machine learning. The EEG signal is high dimensional, with both spatial and temporal covariance. However, many time series approaches for feature extraction face issues with the non-stationary nature of the EEG signal. A stationary signal is one in which the probability distribution does not change over the course of time, features like mean and variance will not change.

Major issue faced in the field is a lack of comparability between experiments. Unlike

in image classification, there are no standard datasets used as performance benchmarks. Data collected for every experiments is with different setup and different configuration of EEG cap. This leads to problems in making a dataset for benchmarking. Individuals with differences in physiology, but entirely different tasks are also performed during data collection leading to different target brain activities. Furthermore, some approaches use models for individuals, whereas others attempt to make a universal model, training and testing with samples from all individuals at one time.

2.2.1 Traditional Approaches

In [Islam *et al.*2017], authors have used feature extraction and traditional machine learning classifiers such as K-nearest neighbors (KNN) and Support Vector Machine (SVM) to build a fully automated EEG artifact classifier. This method outperforms the ICA based methods by having lower computation and memory requirements. Proposed architecture is also implemented on embedded ARM Cortex CPU. On average, it consumes 1.5 W power at 1.2 GHz frequency.

In [Barachant *et al.*2010], Riemannian geometry is used to propose a framework for classification in BCI applications. In this approach classification task is performed by calculating the covariance matrices of the given input epoch without performing any pre-processing. This algorithm relies on mean computation of covariance matrices which is obtained by mapping the dataset into tangential space which makes it difficult for computation in small embedded systems for real time applications. [Majidov & Whangbo2019]

overcome this problem by using CNN for their classification task. As deep neural networks require a lot of data they solve it by performing data augmentation.

Use of deep neural networks have grown due to their success in image classification problems [Krizhevsky, Sutskever, & Hinton2012]. In [Schetinin & Schult2004] authors have used feed forward neural network combined with decision tree to detect ocular artifacts in EEG signal. Another method which accomplishes desired results is that of, recurrent neural networks [Petrosian *et al.*2001] [Güler, Übeyli, & Güler2005]. Research involving the former has constantly delivered precise analysis and detection of contamination in handwriting and speech recognition. Convolution neural networks have been used in [Khatwani *et al.*2018] for detecting ocular and muscular related artifacts. One disadvantage of using CNN is its high memory and computation requirements. Depthwise and separable convolution layers can be used to reduce the weight parameters. This can lead to increase in efficiency without decreasing performance. Use of depthwise separable convolution was also present in the first layer of Inception V1 [Szegedy *et al.*2017]. Use of Xception model on ImageNet dataset led to small change in classification performance with large improvement in computational requirements [Chollet2017].

The Fourier transform is likely the most common feature extraction method. It provides information on the frequency spectrum of the data, at the cost of losing temporal information. The Fourier transform is used to calculate features, such as power spectra, which match nicely with EEG literature on brain activities that occur within different frequency bands used power spectral density values in 1 Hz bins from 2 Hz to 40 Hz to classify

affect data from self-elicited emotions. There were 10 emotions used, with 180 trials each for 1800 samples per subject across three subjects.

Further more attempts have been made to apply deep learning on EEG signal classification. They use a mixture of convolution and recurrent neural network on the EEG epoch with channels projected in two dimensional space. After the projection, the Fourier analysis was applied across several time bins to reduce the data. A hybrid neural network containing a convolutional layer fed into a mixed long short-term memory (LSTM) and 1-D convolutional layer, which was fed into a fully connected layer for classification.

Chapter 3

ARTIFACT DETECTION IN EEG SIGNAL USING CNN

3.1 Problem Definition

The problem explored in this thesis is the classification of EEG data using deep learning techniques. While the classification of EEG signals can be useful in many areas, such as the detection of disease state or brain computer interfaces, first part of this thesis focuses on binary classification problem of artifact detection. Second part of the thesis focuses on multi-class classification problem of actually identifying artifact. The problem of EEG classification in these types of experiments has a number of challenges that must be considered, including:

- There is high variability between subjects and within subjects
- There is limited availability of data
- The data is composed of multiple channels of time series information

This thesis will address these challenges by exploring several variations of architecture

selection, model search, regularization, and training paradigms within a deep learning context, with the aim of harnessing the expressive power of deep learning while avoiding overfitting.

To state the problem formally: Given a training set of EEG signals, $E = \{s_1, s_2, s_3 \dots\}$ are composed of 64 with set of labels $C = \{c_1, c_2 \dots c_{10}\}$ with 10 classes. The goal is to create a classifier K , with parameters Θ and hyperparameters which are used with evaluation metric like accuracy, f-1 score. In other words we have to find a function f which has parameters with highest accuracy on training dataset T .

3.2 High Variability in Data

The highly variable nature of EEG data leads to difficulty in classification. Samples in the same class may be very different in nature from one another. There are various sources of variability which are summarized briefly below:

- A constant issue in EEG data analysis is that the electrical activity created from muscle movement has a far higher magnitude than that produced by brain activity. This is particularly noticeable and problematic when the subject blinks.
- There is a certain amount of uncertainty inherent from the machine itself. Slow drifts in the data are common and are caused by either slight movements of the electrode or sweat interfering with the sensor. Movement in wires connecting the electrodes can cause similar issues. These issues can often be mitigated through the use of band

pass filters.

- Some subjects will be more focused on the task than others. Some will perform better than others on the experimental task. Differing behavior is associated with differences in brain activity patterns.

3.3 Collection of Dataset

The EEG dataset was recorded where patients were instructed to perform a series of different ocular and muscular tasks. EEG dataset was recorded using a 64 channel BioSemi ActiveTwo system with a sampling rate of 512Hz. Participants were required to perform a series of noise-inducing body, facial, head or eye movements, which were gathered as part of a larger study [Lawhern *et al.*2012]. The list of movements were reviewed before starting the experiment so that every patient is familiar with it.

It was up to the participants to determine the precise choreography of each movement and to perform movements which felt more natural to them. Each movement was performed as a separate set of 20 repetitions. A screen was put in place in order to remind the participants of the movement they should make. A male voice initially counted down from 3 at a rate of every 2 Sec followed by a tone every 2 Sec. This procedure was done for each set. The participants would make the movements in time with the vocal commands. They were advised to perform the tasks in the first second of the 2 Sec period and to relax in the remaining 1 Sec. Additionally, each participant performed a baseline recording session

where they were instructed to keep still and look straight at a blank computer screen for around 8 seconds at the start of every run. EEG data from this baseline session was used as "clean" (or artifact-free) data. Artifacts considered are clenching jaw (CJ), move jaw (MJ), blink eyes (BE), move eyes leftwards (EL), move eyes rightwards (ER), raise eyebrows (RE), rotate head (RH), Shrugging shoulders (SS) and Rotate torso (RT). Table 3.1 gives a brief description of nine artifacts which were performed by every patient.

Table 3.1. Description on nine artifacts performed by every patient.

Artifact Code	Artifact Description
101	Clench Jaw
102	Move Jaw Vertically
103	Blink Eyes
104	Move Eyes Leftward
105	Move Eyes Rightward
106	Raise Eyebrows
107	Rotate head
108	Shrug Shoulders
109	Rotate Torso

3.4 Preprocessing of EEG Data

Our CNN operates on the raw EEG data so that it can learn to extract the relevant features required for artifact detection. As such, minimal pre-processing of the raw EEG

data was performed aside from removing the DC offset such that the EEG signals are centered around zero. EEG epochs of size 64×512 were extracted from the artifact and baseline data creating a total of around 250 trials per subject for each artifact type.

3.5 Proposed DNN Architecture

The input to this CNN is a two dimensional EEG epoch ($64 \text{ channels} \times 512 \text{ time points}$). The full network architecture, shown in Figure 3.1, consists of two convolutional layers, two max-pool layers and ends with a softmax layer for classification. The first 2-d convolution layer consists of 128 kernels of size 64×3 . this ensures that the an adequate spatial filter is learned in the first layer. The second 2-d convolution layer learns 64 kernels of size 1×3 . Each convolution layer is followed by a max-pooling layer with a pool size of 1×2 . All layers are followed by a rectified linear unit (ReLU) activation function. The output of the second max pooling layer is then flattened into a single vector and passed to a softmax layer consisting of two outputs for binary classification to detecting whether or not an artifact is present in the EEG data.

All the layers of the network have their weights initialized from a normal distribution. The network was trained using the RMSprop optimization method and a learning rate of 0.0001. Categorical cross-entropy was used as the loss function. In total, the network has 65280 parameters, and requires 35.4 million operations (either multiplication or addition) in order to process one input frame. Note, that this is much lower than CNNs for image classification which usually contain millions of parameters [Simonyan & Zisserman2014].

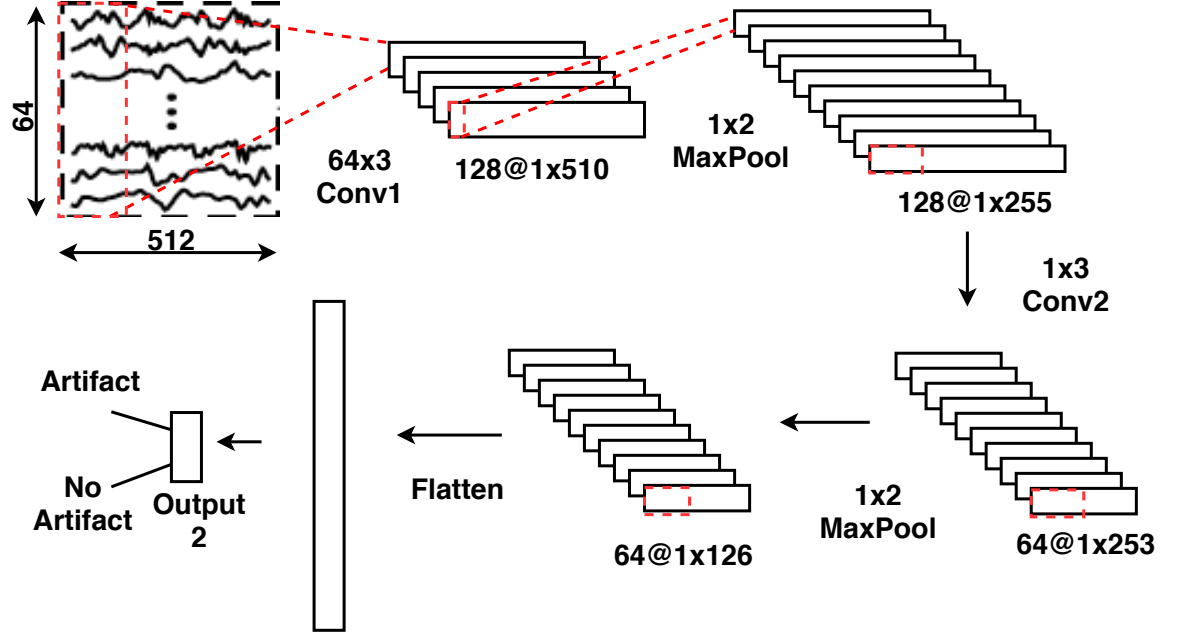


FIG. 3.1. Proposed CNN Architecture which consists of 5 layers. 2 convolutional layers, 2 max pool layers and 1 softmax layer.

3.6 Classification Analysis and Results

Our architecture is evaluated for 9 patients for 7 different artifacts using a transfer learning setting where models are trained and tested on EEG data from different subjects. This allows us to test the ability of our CNN model to generalize across subjects when detecting EEG artifacts, a task that is difficult for traditional methods. We trained a cross-subject model using a leave-one-subject-out cross validation procedure where data from one subject was held out for testing and data from the remaining subjects were used for training. On average 3790 samples were used for training and 490 samples were used for testing. We compare our results with previous work that uses an autoregressive model +

SVM classifier for artifact detection on the same dataset [Lawhern *et al.*2012].

Results in Table 3.2 show that our model detects all 7 artifacts by average accuracy of 73.6% which ranges between 64.3% and 84.8%. The raise and lower eyebrows artifact was easiest to detect with average accuracy of 84.8%, while the moving jaw had lowest detection accuracy of 64.3%.

In Table 3.2 the autoregressive (AR) [Lawhern *et al.*2012] baseline technique achieved 68.42% of average accuracy using the leave-one-subject-out cross-validation technique which ranges between 52% and 95%. Our CNN model achieves a statistically significant improvement in detection accuracy on 5 of the 7 EEG artifacts using a one sample, two-sided t-test with a significance threshold of $p < 0.05$.

Table 3.2. Detection accuracy and precision for different artifacts averaged for all patients using leave one subject out cross-validation technique among 9 patients. Results are compared against Auto-Regressive baseline technique [Lawhern *et al.*2012]. Values in the parentheses indicate the standard deviation. Asterisks (*) indicate significant accuracy improvement over the AR technique.

Artifact Code	This work: Accuracy	AR: Accuracy [Lawhern <i>et al.</i> 2012]	This work: Precision
CJ	74.1(9.2)	95*	94.7(5.1)
MJ	64.3(11.8)	76	91.4(9.8)
BE	76.1(10.9)*	74	94.3(5.6)
EL	77.5(9.6)*	49	94.0(5.6)
ER	72.6(9.1)*	52	92.6(7.8)
RE	84.8(7.9)*	75	99.1(1.1)
RH	73.3(14.2)*	58	92.7(6.1)
Average	73.6(11.8)*	68.42	92.0(7.3)

If training accuracy is high, then the validation accuracy and test accuracies are con-

sidered. If the validation and test accuracies are also high, then the selection of hyperparameters can be accepted, or more tweaks can be made to the hyperparameters to determine if the working definition of high accuracy is sufficient. If the test and validation accuracies are low when the training accuracy is high, the model is most likely overfitting. There are two main ways of handling overfitting: increasing the amount of regularization in the model and decreasing the number of parameters in the model. Increasing the amount of regularization includes changes such as adding or increasing L2 penalties on weights, adding or increasing dropout, or adding batch normalization to layers. Reducing the number of parameters is achieved by either reducing the number of neurons in the layers or reducing the number of layers. Generally, reducing the parameters of the model is reserved for when increasing regularization fails to improve validation accuracy to avoid unnecessarily reducing the expressive power of the model. However, if the model performs nearly perfectly on the training data while performing very poorly on the validation data, reducing the number of parameters can be the most effective change to the hyperparameters.

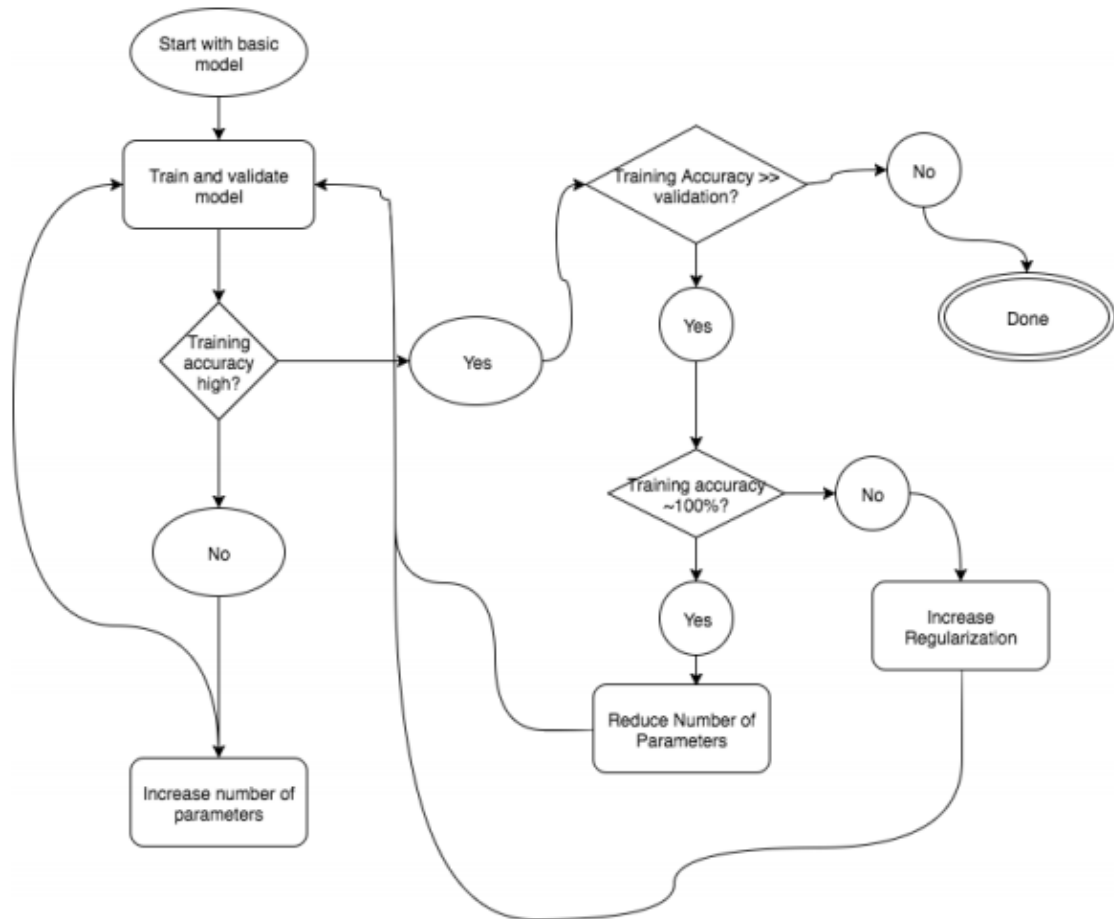


FIG. 3.2. Flow chart of the heuristic hyperparameter search.

Chapter 4

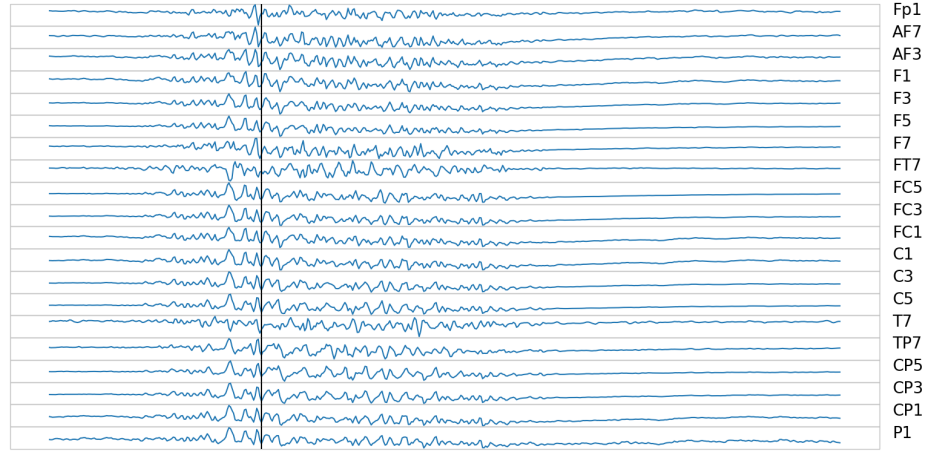
EFFICIENT CNN FOR ARTIFACT IDENTIFICATION

As seen in Chapter 3 use of CNN can be more beneficial in classification tasks of EEG signal. Though implementing deep learning models is proved to be more beneficial in terms of accuracy and f-1 scores. The model developed consists of large number of parameters which makes it difficult to train for small amount of data. In this chapter we overcome this problem by implementing new types of convolution layers which have benefits in reducing number of parameters and train the model efficiently.

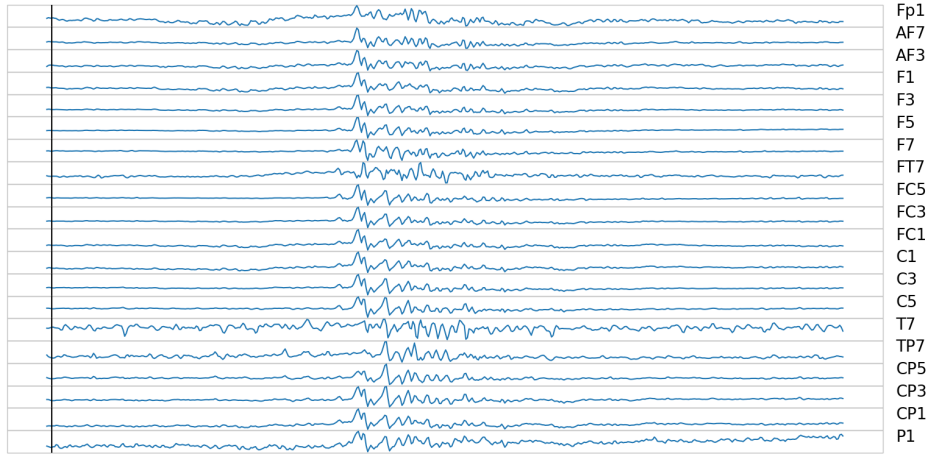
4.1 Visualization of EEG signals

4.1.1 EEG signal plot

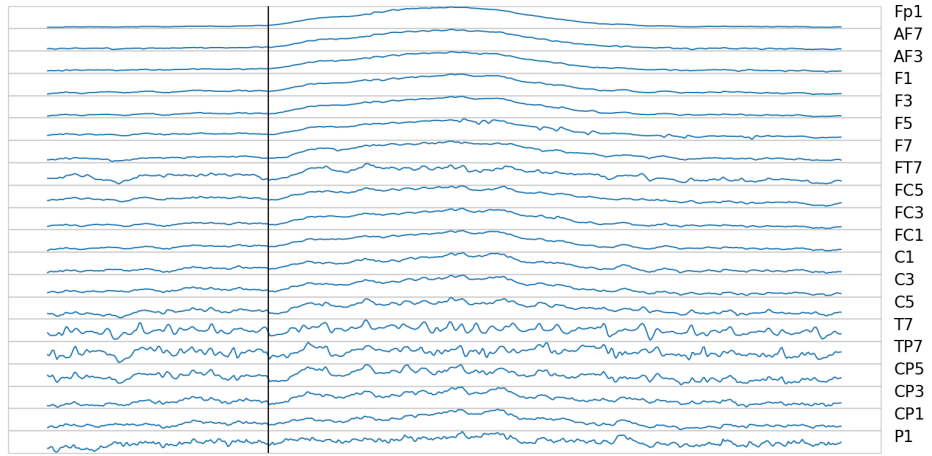
Figure 4.1 shows plot for first 20 of 64 electrodes placed on the scalp to capture the EEG signals. This can be useful to inspect which electrodes are significant in capturing the specified artifacts. This plot shows 9 artifacts for single patient. Every artifact generates different pattern which helps in identifying the specified artifact.



(a) 101: Clench Jaw

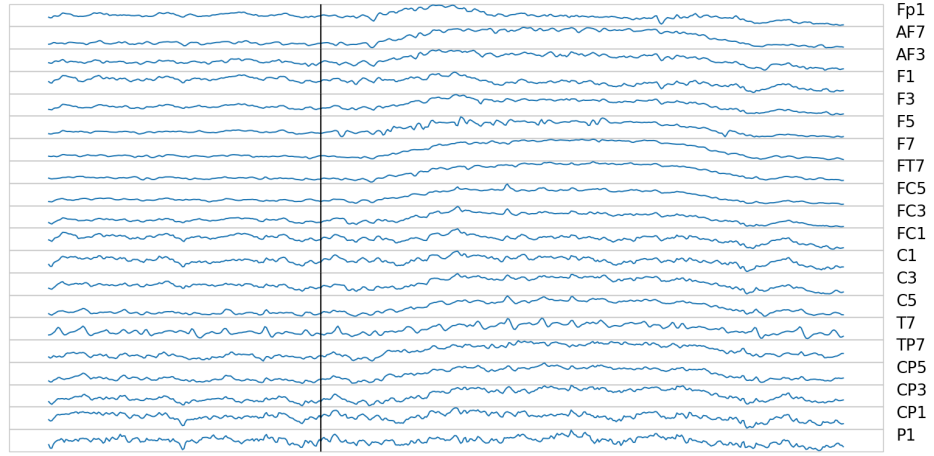


(b) 102: Move Jaw

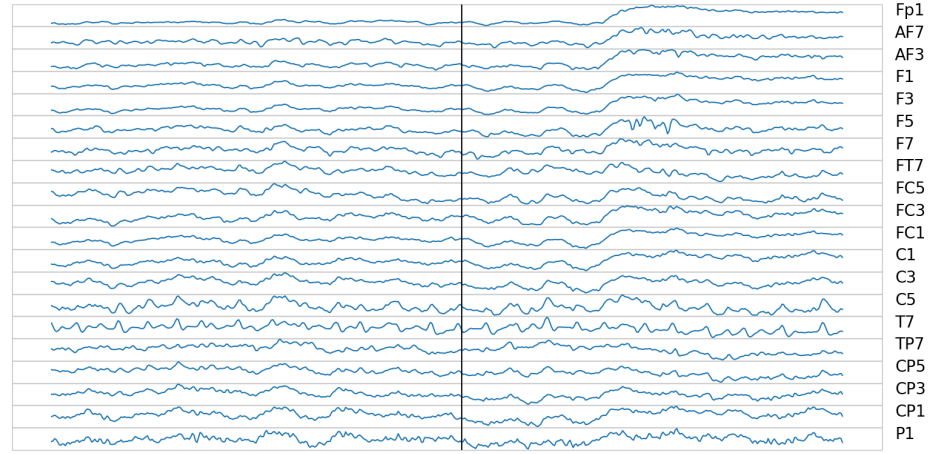


(c) 103: Blink Eyes

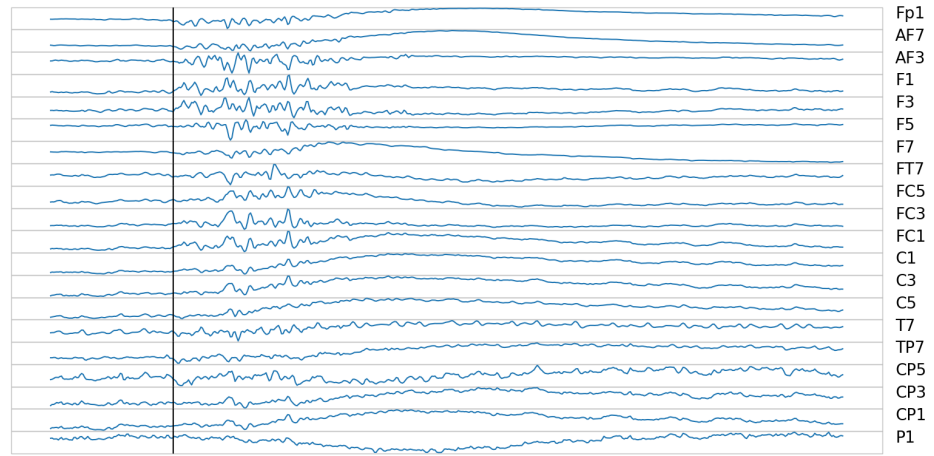
FIG. 4.1. Visualization of jaw movement related artifacts performed by patients. Instructions were given to patients every 2 secs and it was advisable to perform the task in the first second. Vertical line indicates the start of experiment. Artifact names for given artifact codes are explained in Table 3.1.



(a) 104: Eyes Leftward

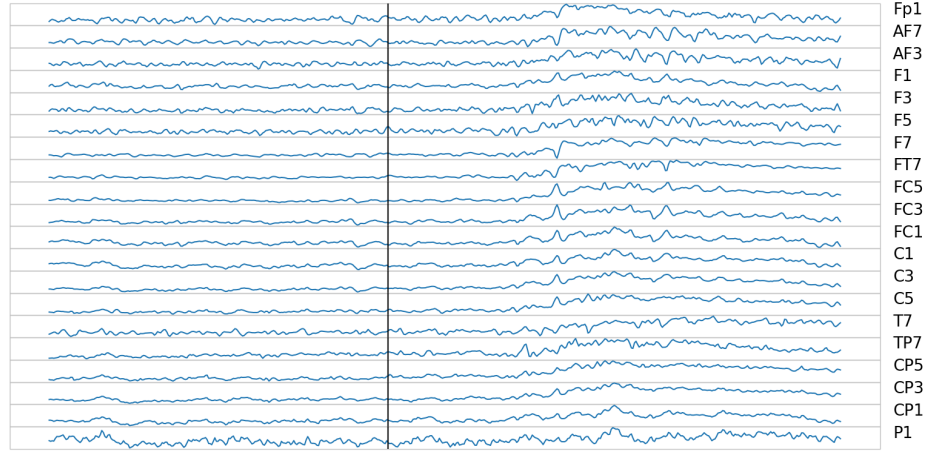


(b) 105: Eyes Rightward

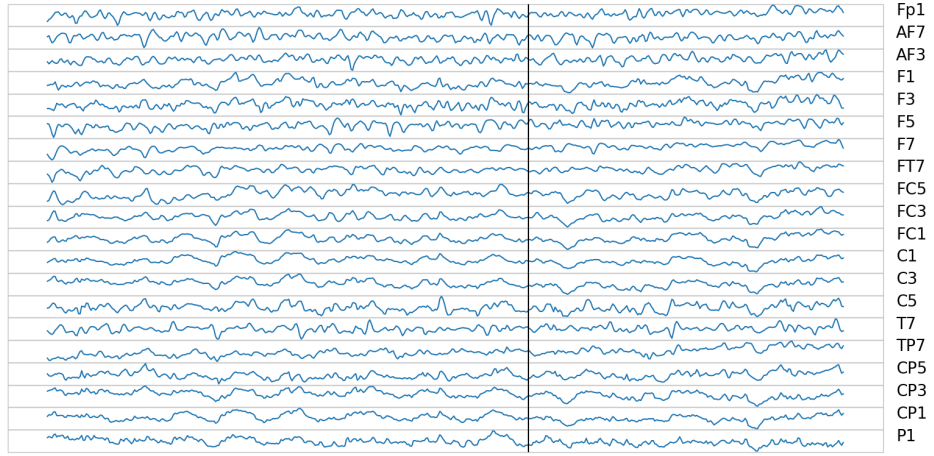


(c) 106: Raise Eyebrows

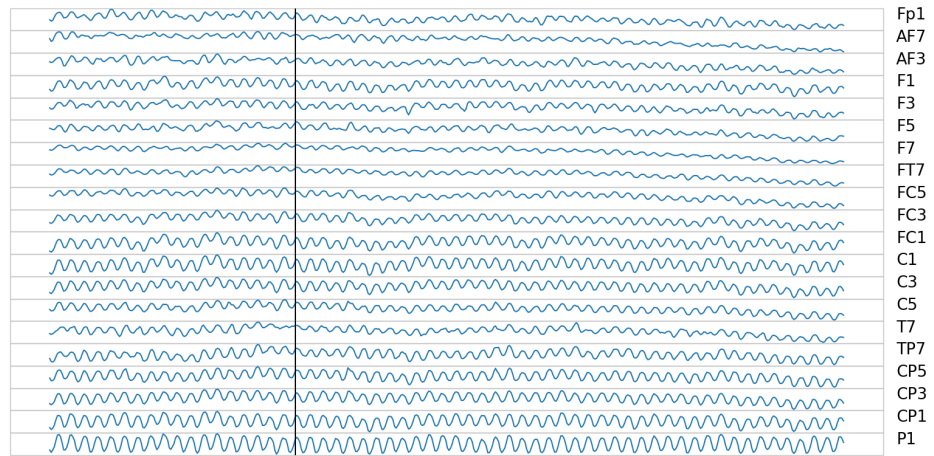
FIG. 4.2. Visualization of ocular related artifacts performed by patients. Vertical line indicates the start of experiment. Artifact names for given artifact codes are explained in Table 3.1.



(a) 107: Rotate Head



(b) 108: Shrug Shoulders



(c) 109: Rotate Torso

FIG. 4.3. Visualization of muscular movement related artifacts performed by patients. Vertical line indicates the start of experiment. Artifact names for given artifact codes are explained in Table 3.1.

Figure 4.3 shows muscular related artifacts. Muscle related artifacts are difficult to detect as they are more prone to noise. Specifically Artifact 109: Rotating Torso is difficult as most patients are not able to perform that task correctly. This leads to a lot of noise captured while performing this task. Same goes with Artifact 103: Eye blinking. Blinking eyes is an involuntary movement which makes it difficult to detect this artifact. While patients are instructed to stay still for first two seconds for collecting plain: artifact free signal, most patients tend to blink their eyes making it difficult to capture noise free and artifact free plain (artifact free) signal.

Figure 4.4 shows position of 64 electrodes used for capturing the EEG data.

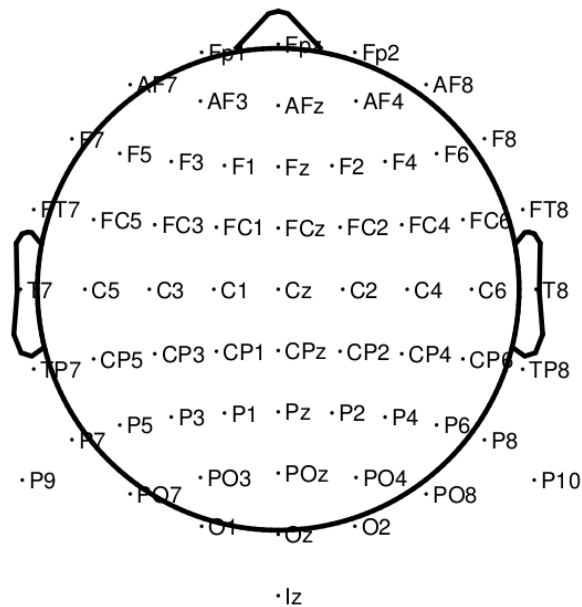


FIG. 4.4. 64 EEG signal channel locations.

FIG. 4.5. TopoMap figures of jaw related artifacts

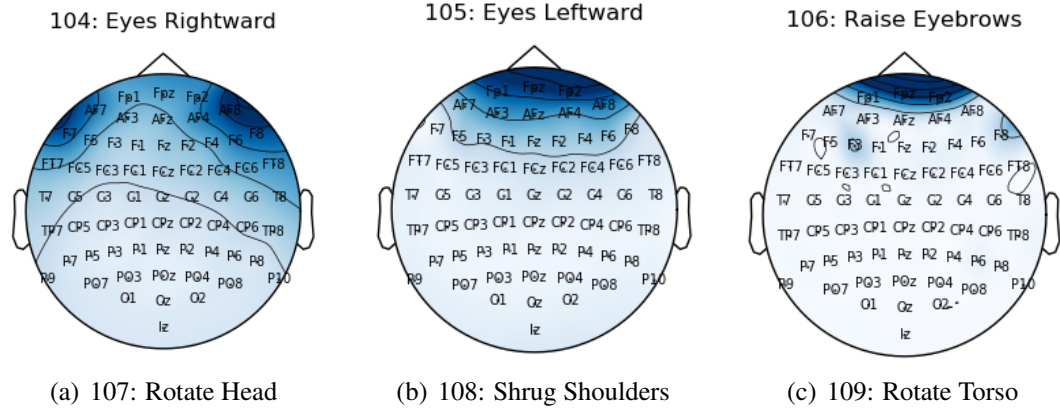


FIG. 4.6. Topomap graph for ocular artifacts

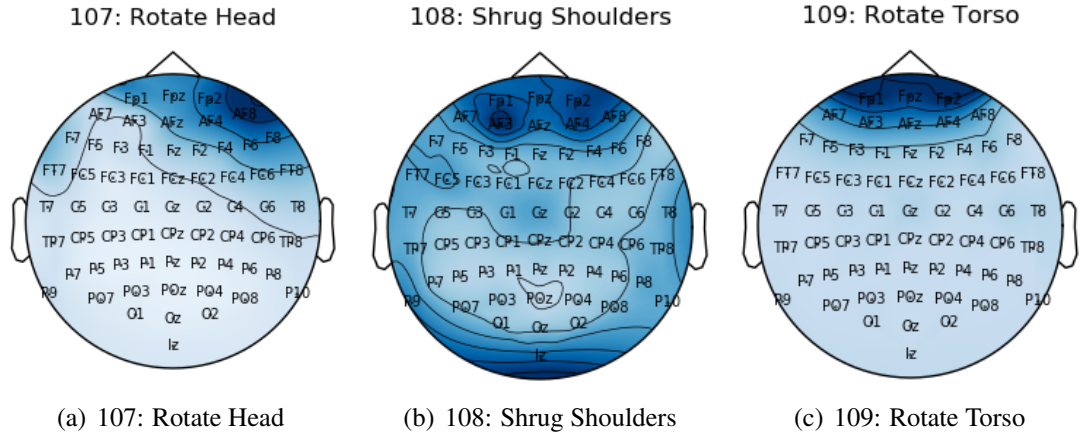


FIG. 4.7. Topomap graph for muscular artifacts

4.2 Implementation and Results

4.2.1 Architectures

In this section we first present the models determined by the model search and then the final classification results of these models in each task performed. We propose using various combinations of convolution layers discussed in Chapter 2 for EEG based artifact identification.

The input to our models is a two dimensional EEG epoch($64 \text{ channels} \times 512 \text{ time points}$). The final output layer is 10 neuron dense layer which is passed to a softmax layer consisting of output of artifact identified. ReLU activation function is used after all the convolution layers.

All the layers of the network have their weights initialized from a normal distribution. The networks were trained using the RMSprop optimization method and a learning rate of 0.0001. Categorical crossentropy was used as the loss function. CNN operates on the raw EEG data so that it can learn to extract the relevant features required for artifact detection. As such, minimal pre-processing of the raw EEG data was performed aside from removing the DC offset such that the EEG signals are centered around zero. EEG epochs of size 64×512 were extracted from the artifact and baseline data creating a total of around 250 trials per subject for each artifact type.

4.2.1.1 Traditional CNN

Figure 4.8 shows the use of traditional convolution layer. This network starts with a convolution layer with 128 filters of size 64×3 . The second convolution performed consists of 64 filters of size 1×3 . The convolution layers are followed by average pooling layers with pool size of 1×2 .

In total, the network has 24,842 parameters, and requires 35.4 million operations (either multiplication or addition) in order to process one input frame. Note, that this is much lower than CNNs for image classification which usually contain millions of parameters.

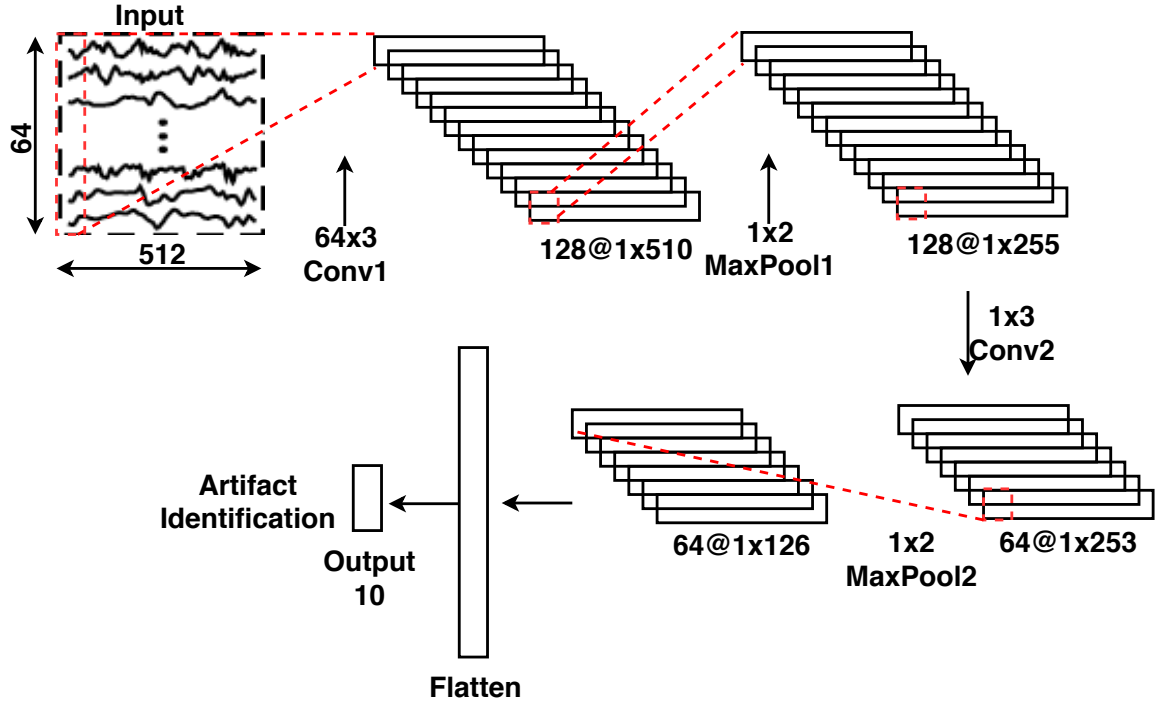


FIG. 4.8. Traditional CNN Architecture for artifact identification which consists of 5 layers. 2 convolutional layers, 2 max pool layers and 1 softmax layer. Total parameters required for this architecture is 24,842.

4.2.1.2 Separable CNN In this network we have replaced the traditional convolution layer with depthwise separable convolution layer. All the filter sizes and number of filters are kept same. The network size is reduced to 8,906 parameters. It requires 6.53 million operations for processing the input.

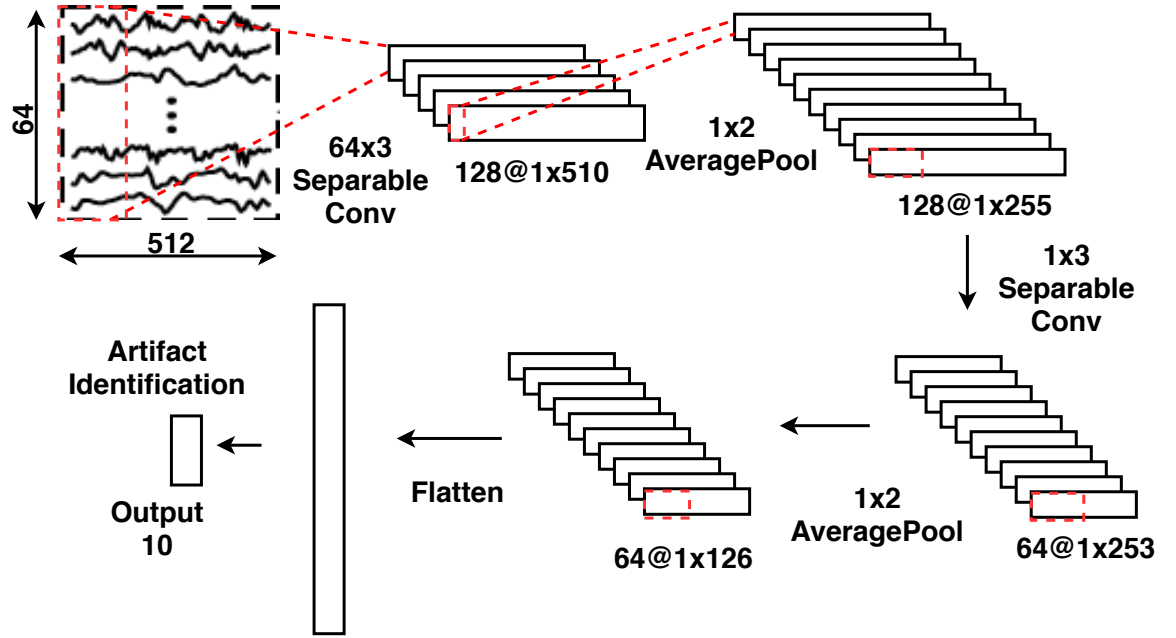


FIG. 4.9. .

4.2.1.3 DepthWise CNN In this network a depthwise convolution is applied at the start. There is a slight change in input format of data for this network. 64 EEG signals which were aligned one below each other are now aligned along the depth which makes the shape of input of data to be $1 \times 512 \times 64$. This depthwise convolution is applied with filter size of 1×16 and depth multiplier of 1 i.e. one filter for each depth input. This is followed by a separable convolution layer with filter size of 1×16 and 16 filters. Same as all other

networks every convolution layer is followed by an average pooling with pool size of 1×5 .

The network size here is 3,562 parameters. It requires 0.6 million operations for processing the input.

4.2.1.4 Separable + DepthWise CNN Figure 4.10 shows the architecture which is similar to the one used in [Lawhern *et al.*2018]. This networks has filter shape of 1×64 in first convolution layer with zero padding added. This is followed by a depthwise convolution layer with filter shape of 64×1 . A separable convolution layer is used with filter shape of 1×16 . Depthwise and separable convolution layers are followed by average pooling layer with pool size of 1×8 and 1×16 respectively [Lawhern *et al.*2018].

Total number of parameters for this architecture is 5,002 with 30.4 million number of operations.

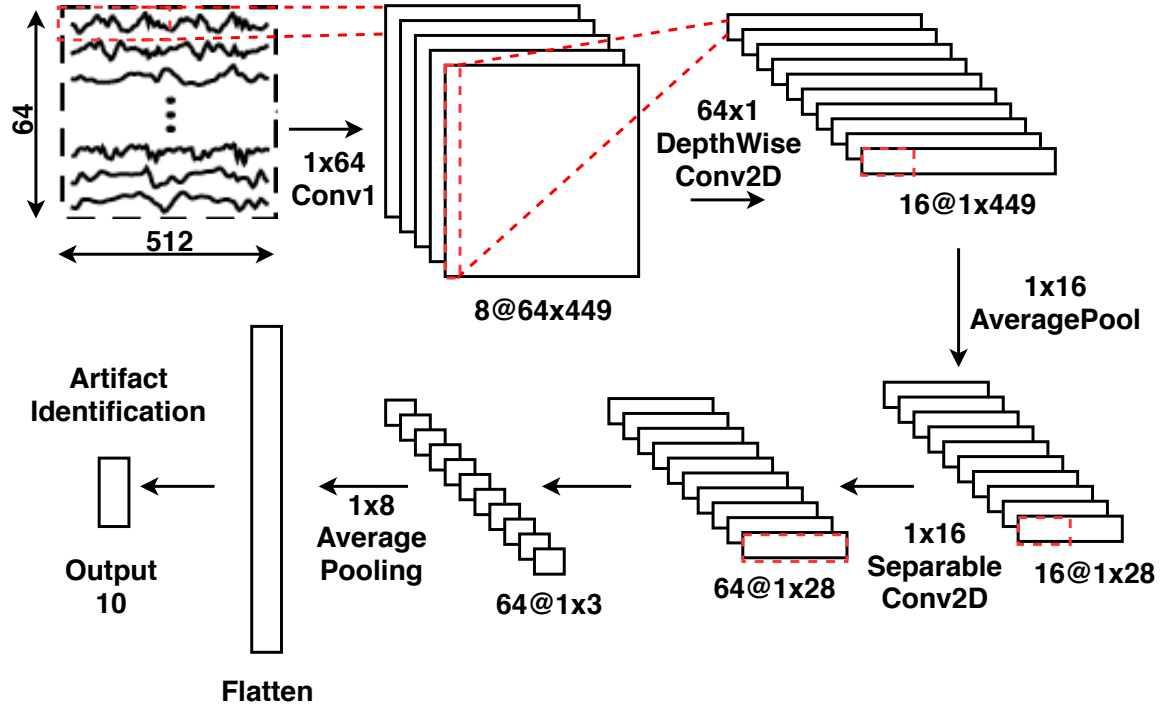


FIG. 4.10. EEGNet architecture which uses combination of depthwise and separable convolution layers. Total parameters required for this architecture is 5,002.

4.2.2 Optimized Separable + DepthWise

Figure 4.11 shows the network starts with a convolution layer with 8 filters of size 64×16 . Zero padding is avoided in these type of architectures to avoid large computations. A depthwise convolution is used with filter of size 1×8 and depth multiplier of 2 which means there will be 2 filters associated with each depth. This is followed by an average pooling layer with pool size of 1×16 . A separable convolution is further used with 1×8 filter size. Here output feature map of first layer has reduced height as compared to original version of EEGNet. This saves computation required in first two layers. Output of feature

map is reduced from $8@64 \times 449$ to $1@1 \times 497$.

The number of parameters of this network increase from EEGNet due to the filter shape of first convolution layer but this leads to a very significant decrease in number of computations. Parameter size is 9,194 with 8.3 million operations for processing the input.

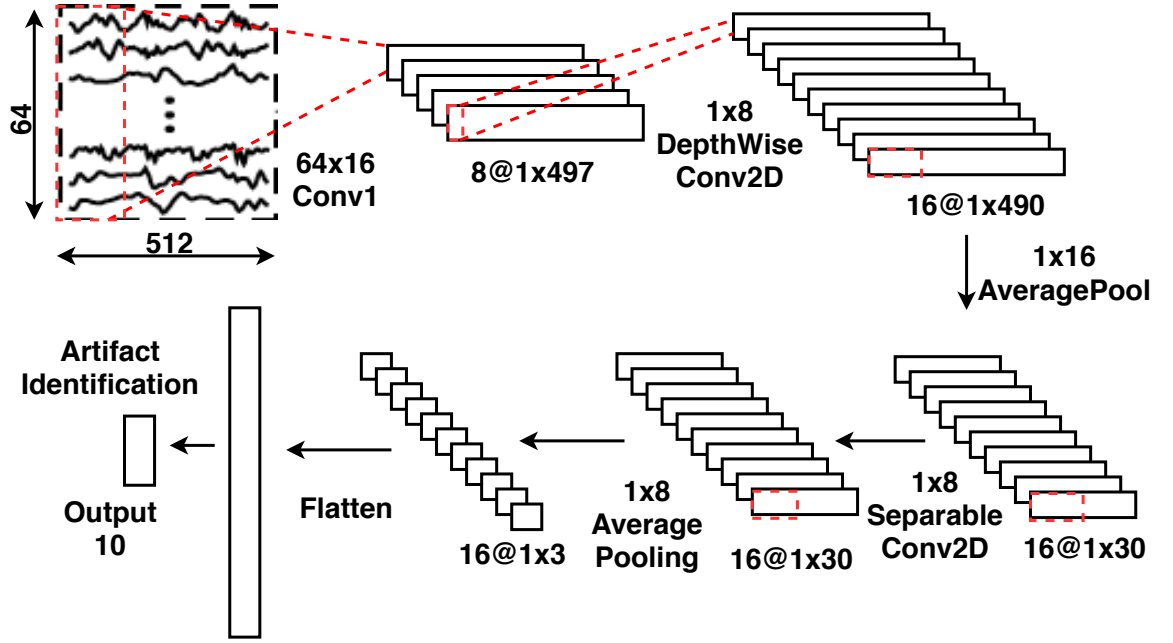


FIG. 4.11. Optimized EEGNet architecture with filter size of 64×16 in first convolution layer which reduces the computation by **3.64x**. Total parameters required for this architecture is 9,194.

4.2.3 Optimization of Architectures

EEGNet utilizes one traditional convolution layer, followed by a depthwise convolution layer, followed by one average-pooling layer, after that one depthwise-separable

convolution layer, then one more average-pooling layer and at last fully-connected layer. First convolution layer has 8 filter sets and each filter size is $M \times 5$ where M is the number of input channels. In this section, we explain the reason behind of choosing EEGNet architecture and parameters.

Table 4.1. Comparison of computation in each layer for EEGNet and its optimized architecture.

Layers	Computation in EEGNet	Computation in Optimized EEGNet	Improvement
Conv1	29,425,664	8,142,848	3.6x
Depthwise	919,552	125,440	7.3x
AveragePool1	16,384	15,360	1.06x
Depthwise Separable	32,768	61,440	0.53x
Average2	2,048	3,072	0.66x
SoftMax	320	960	0.33x
Total	30,396,736	8,349,120	3.64x

Table 4.1 shows the improvement in computation in each layer between EEGNet and optimized EEGNet. In conv1 layer, EEGNet has 29.4M computation while optimized EEGNet has 8.1M computation which is 3.6x of EEGNet. Optimized EEGNet has total 3.6x reduced computation of EEGNet.

4.2.4 Classification Accuracy

Our architectures are evaluated for 17 patients for 9 different artifacts. Models are trained and tested using intra patient setting where models are trained using 70% of the data,

10% is used for validation and remaining 20% is used for testing. All the ten classes are balanced for classification task. Variations of convolution layer is used for different models proposed which leads to variations in parameters as well as artifact identification accuracy.

Results in Table 4.2 show average accuracy of proposed models which are averaged across all 17 patients and 9 artifacts. Model which uses combination of both depthwise and separable convolution layer provides the best identification accuracy of 95.30%.

Figure 4.12 shows classwise accuracy of the four models proposed. It can be concluded that muscle related artifacts such as Shrugging shoulders (108) and rotating torso (109) are difficult to identify as compared to other artifacts. The artifact with best in-class accuracy for all the models is raising eyebrows (106).

Table 4.2. Comparison of parameters, computations and average accuracy (17 patients and 10 classes which includes 9 artifacts and 1 plain signal) of different model configurations. All the models classify 9 different artifacts with test data and training data for the same patient

Models	Weight Parameters	No. of Computations (millions)	Accuracy (%)
CNN	24,842	35.4	80.37
DepthWise	3,562	0.6	76.97
Separable	8,906	6.53	68.1.3
EEGNet	5,002	30.4	95.30
Optimized EEGNet	9,194	8.3	94.17

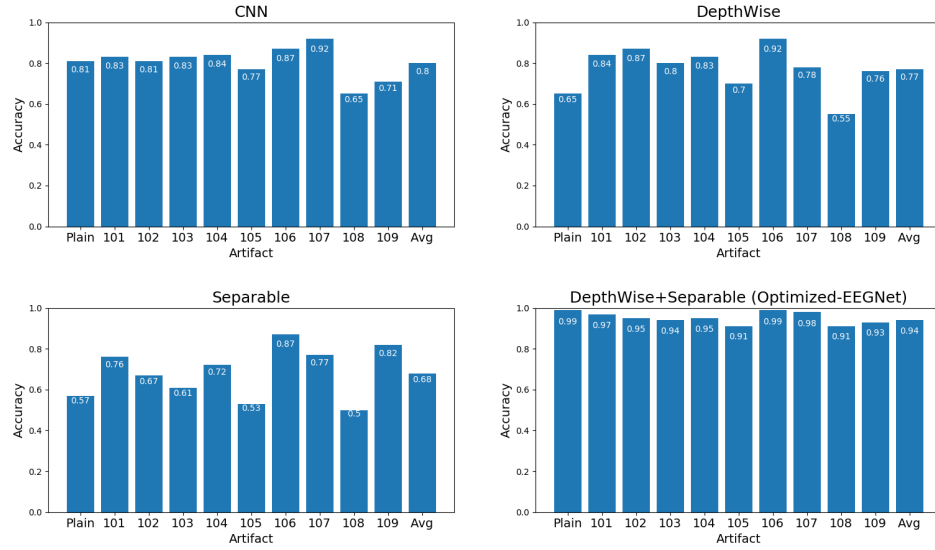


FIG. 4.12. Class-wise and average accuracy for proposed models. Using only separable convolution layers provides least average accuracy. Using combination of depthwise and separable convolution layers proves to be most beneficial in artifact identification.

4.3 Experimental Study

This experiment was performed using Emotiv EPOC 14 channel headset. The user was instructed to blink once every two second. First part of this 2 second windows was extracted and labeled as artifact. Second part of this window is labeled as artifact-free data.

Data is collected with sampling rate of 128Hz. Input given to the network is of shape 14×128 . The data was collected in 10 different sessions. Each session consisted of 10 eye blinks. Figure 4.13 shows two second window of EEG data captured. Our network was trained on 70% of all the data captured, 10% was used for validation and remaining 20% was used for testing. In this experiment we achieved 93.5% accuracy for detecting

eye blink artifact.

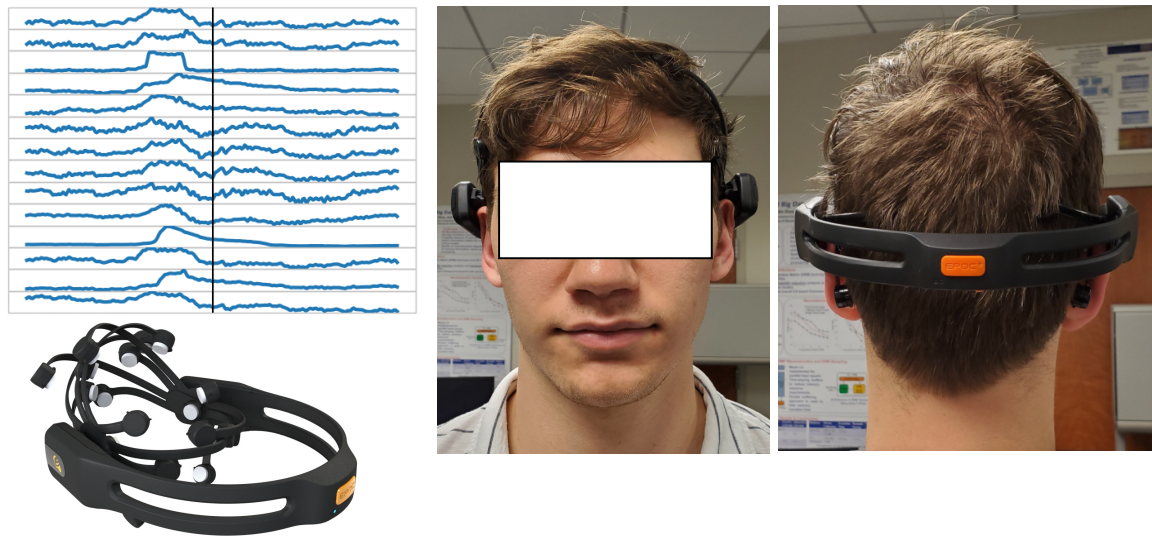


FIG. 4.13. Emotiv EPOC 14 channel headset capturing EEG data, Eye blink is performed once every two seconds.

Chapter 5

CONCLUSION AND FUTURE WORK

5.1 Conclusion

In this thesis, we propose an energy efficient CNN-based architecture for detecting artifacts in EEG that achieved an average accuracy of 74% across all artifacts and subjects. On average, our CNN architecture significantly outperformed the baseline auto-regressive method [Lawhern *et al.*2012] on the majority of the tested artifact types. Additionally, our CNN method is a fully automated technique which doesn't require manual labeling of the EEG trials which was required in [Lawhern *et al.*2012]. An optimized CNN architecture for artifact identification was also proposed with identification accuracy of 94.17% over all the patients and artifacts. Use of depthwise and separable convolution layers which leads to $2.7\times$ reduction in total number of parameters and $4.26\times$ improvement in computations required as compared to a network which uses traditional convolution layers.

5.2 Future Work

There are three main avenues for the extension of this research: application to more datasets, expanded model search, and transfer learning.

5.2.1 More Datasets

The dataset in this thesis involved classification of artifacts. There are several other common tasks that should also be explored. Motor imagery is perhaps the most common task in EEG classification, and is used in several forms of brain computer interface. Similarly, experiments where the subject thinks about words from a set vocabulary have immediate potential use in communication based BCIs, and would also show a valuable extension of this research.

The dataset explored in this thesis was collected on a 64-channel EEG. Many modern EEGs have 128 or even 256 channels, leading to several times the number of features. The negative impact of feature reduction on deep learning found in our results suggests that having a higher number of recorded features may improve the classification of the signal, despite the issues with the curse of dimensionality.

5.2.2 More Model Search

Work presented in this thesis covered many versions of CNNs explored, though there are other architectures of deep learning that look promising.

The first category of models that may show promise are those based on a different

representation of the data. While strictly recurrent architectures can be implemented to explore the accuracy. Another extension of Convolution Neural Network can be to use 3D projection of the data in X by Y by time format, that spatial information may be largely retained. Many architectures used in the classification of video may be valuable in these situations. This includes 3D convolutional models, which aggregate information of a localized area over time. 2D convolutional models over space fed into recurrent architectures over time are also promising.

There are also other layer types that should be considered. For example, Gated Recurrent Units (GRUs) have been gaining favor over LSTM models in other time series data, such as natural language processing.

5.2.3 Transfer Learning

Transfer learning is a family of techniques involving the use of one set of data to create an initialization for the classification of another set of data. Transfer learning for our dataset can be performed in two ways.

- To train the network on all but one patient and then fine tune the network on the patient left.
- To train the network on the all but one artifact and then fine tune it on the artifact left.

The most straightforward change that could be made in the transfer learning paradigm is simply freezing the bottom layers of the network during fine-tuning. That is to say, only

allow the classification layer and perhaps the last fully connected layer to update during the fine-tuning phase. The benefit of this approach is that it may reduce overfitting during the fine-tuning phase.

Another approach to transfer learning that is worth exploring is transferring between datasets. There are numerous known common features in EEG data. Thus, it is reasonable to believe that applying transfer learning over several datasets to learn universal basic features for the early filters in the network could be beneficial. There are some barriers that would need to be considered, however, especially if the data is collected on different machines or has a different time span.

Appendix A

APPENDIX

A.1 Code for data extraction of every patient

```
data_segments = []
onehot_labels = []

for patient in patients:

    data_mat = sio.loadmat('./B_' + str(patient) + '_ART.mat')['EEG']

    data = data_mat['data'][0][0][:64]

    event = data_mat['orig_event'][0][0][0]

    latency = numpy.zeros(shape=(event.shape[0]))

    for i in range(event.shape[0]):

        latency[i] = event[i][1][0][0]

    # print(latency)
```

```

labels = numpy.zeros(shape=(event.shape[0]))

for i in range(event.shape[0]):

    labels[i] = event[i][0][0][0]

# print(labels)

plain_ind = numpy.where(labels == 17)

plain_ind = plain_ind[0]

for i in range(int(latency[plain_ind[0]]), int(latency[plain_ind[1]])):

    temp = []

    for j in range(64):

        d = data[j][i: i + 512]

        median_d = numpy.mean(d)

        d = numpy.subtract(d, median_d)

        temp.append(d)

    data_segments.append(temp)

    onehot_labels.append(to_onehot(0, 10))

for i in range(len(plain_ind), len(labels)):

    for k in range(int(latency[i] - 200), int(latency[i] + 300), 100):

        temp = []

        for j in range(64):

            d = data[j][int(k): int(k) + 512]

```

```

        median_d = numpy.mean(d)

        d = numpy.subtract(d, median_d)

        temp.append(d)

    data_segments.append(temp)

    if int(labels[i]) % 100 == 17:

        label = 0

    else:

        label = int(labels[i]) % 100

    onehot_labels.append(to_onehot(label, 10))

data_segments = numpy.asarray(data_segments)

onehot_labels = numpy.asarray(onehot_labels)


data_segments, onehot_labels = shuffle(data_segments, onehot_labels, 1)

return data_segments, onehot_labels

```

A.2 Code for plotting Topographical Maps

```

pos = mne.find_layout(raw.info).pos

artifact_name = ['101: Clench Jaw', '102: Move Jaw', '103: Blink eyes']

times_picker = [[53.6, 53.9], [102.4, 102.6], [151.8, 152.1], [205.2, 205.5]]

fig, ax = plt.subplots(3, 3)

```

```

for ii in range(9):

    start, stop = raw.time_as_index(times_picker_signal[ii])

    slice = raw.get_data( start = start, stop = stop)

    print(slice.shape)

    mean_raw = numpy.median(slice, axis = 1)

    for i in range(len(mean_raw)):

        slice[i][:] = slice[i][:] - mean_raw[i]

    r_mean_slices = []

    for s in slice:

        r_mean_slices.append(numpy.sqrt(numpy.mean(numpy.square(s))))

    r_mean_slices = numpy.array(r_mean_slices)

    # mean_slices = numpy.mean(slices_mean, axis = 1)

    im,_ = plot_topomap(r_mean_slices, pos, names = ['Fp1', 'AF7',
    'AF3', 'F1', 'F3', 'F5', 'F7',
    'FT7', 'FC5', 'FC3', 'FC1', 'C1', 'C3',
    'C5', 'T7', 'TP7', 'CP5', 'CP3', 'CP1',
    'P1', 'P3', 'P5',], cmap = 'Blues' ,show_names = True,
    show = False, axes = ax[ii//3][ii%3])

    ax[ii//3][ii%3].set_title(artifact_name[ii])

plt.show()

```

A.3 Model Summary

A.3.1 Traditional Convolutional Neural Network

Layer (type)	Output Shape	Param #
=====		
conv1 (Conv2D)	(None, 1, 510, 64)	12288
=====		
average_pooling2d_1 (Average	(None, 1, 102, 64)	0
=====		
conv2d_1 (Conv2D)	(None, 1, 100, 32)	6144
=====		
average_pooling2d_2 (Average	(None, 1, 20, 32)	0
=====		
flatten_1 (Flatten)	(None, 640)	0
=====		
dense_1 (Dense)	(None, 10)	6410
=====		
Total params: 24,842		
Trainable params: 24,842		
Non-trainable params: 0		

A.3.2 DepthWise Convolution Neural Network

Layer (type)	Output Shape	Param #
=====		
input_1 (InputLayer)	(None, 1, 512, 64)	0
depthwise_conv2d_1 (Depthwis	(None, 1, 497, 64)	1024
activation_1 (Activation)	(None, 1, 497, 64)	0
average_pooling2d_1 (Average	(None, 1, 31, 64)	0
dropout_1 (Dropout)	(None, 1, 31, 64)	0
separable_conv2d_1 (Separabl	(None, 1, 31, 16)	2048
activation_2 (Activation)	(None, 1, 31, 16)	0
average_pooling2d_2 (Average	(None, 1, 3, 16)	0

dropout_2 (Dropout)	(None, 1, 3, 16)	0
flatten (Flatten)	(None, 48)	0
dense (Dense)	(None, 10)	490
softmax (Activation)	(None, 10)	0
=====		
Total params: 3,562		
Trainable params: 3,562		
Non-trainable params: 0		

A.3.3 Separable Convolution Neural Network

Layer (type)	Output Shape	Param #
=====		
conv1 (SeparableConv2D)	(None, 1, 510, 64)	256

average_pooling2d_1	(Average (None, 1, 102, 64))	0
<hr/>		
separable_conv2d_1	(Separabl (None, 1, 100, 32))	2240
<hr/>		
average_pooling2d_2	(Average (None, 1, 20, 32))	0
<hr/>		
flatten_1	(Flatten) (None, 640)	0
<hr/>		
dense_1	(Dense) (None, 10)	6410

=====
Total params: 8,906

Trainable params: 8,906

Non-trainable params: 0

A.3.4 Depthwise + Separable Convolution Neural Network

Layer (type)	Output Shape	Param #
--------------	--------------	---------

input_1 (InputLayer)	(None, 64, 512, 1)	0
conv2d_1 (Conv2D)	(None, 64, 512, 8)	256
activation_1 (Activation)	(None, 64, 512, 8)	0
depthwise_conv2d_1 (Depthwis	(None, 1, 510, 24)	4608
activation_2 (Activation)	(None, 1, 510, 24)	0
average_pooling2d_1 (Average	(None, 1, 63, 24)	0
dropout_1 (Dropout)	(None, 1, 63, 24)	0
separable_conv2d_1 (Separabl	(None, 1, 63, 24)	960
activation_3 (Activation)	(None, 1, 63, 24)	0
average_pooling2d_2 (Average	(None, 1, 15, 24)	0
dropout_2 (Dropout)	(None, 1, 15, 24)	0

flatten (Flatten)	(None, 360)	0
dense (Dense)	(None, 10)	3610
softmax (Activation)	(None, 10)	0
=====		
Total params: 9,434		
Trainable params: 9,434		
Non-trainable params: 0		

REFERENCES

- [Barachant *et al.*2010] Barachant, A.; Bonnet, S.; Congedo, M.; and Jutten, C. 2010. Riemannian geometry applied to bci classification. In *International Conference on Latent Variable Analysis and Signal Separation*, 629–636. Springer.
- [Chollet2017] Chollet, F. 2017. Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 1251–1258.
- [Delorme, Sejnowski, & Makeig2007] Delorme, A.; Sejnowski, T.; and Makeig, S. 2007. Enhanced detection of artifacts in eeg data using higher-order statistics and independent component analysis. *Neuroimage* 34(4):1443–1449.
- [Güler, Übeyli, & Güler2005] Güler, N. F.; Übeyli, E. D.; and Güler, I. 2005. Recurrent neural networks employing lyapunov exponents for eeg signals classification. *Expert systems with applications* 29(3):506–514.
- [Hinton, Osindero, & Teh2006] Hinton, G. E.; Osindero, S.; and Teh, Y.-W. 2006. A fast learning algorithm for deep belief nets. *Neural computation* 18(7):1527–1554.
- [Howard *et al.*2017] Howard, A. G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Andreetto, M.; and Adam, H. 2017. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*.

- [Iriarte *et al.*2003] Iriarte, J.; Urrestarazu, E.; Valencia, M.; Alegre, M.; Malanda, A.; Viteri, C.; and Artieda, J. 2003. Independent component analysis as a tool to eliminate artifacts in eeg: a quantitative study. *Journal of clinical neurophysiology* 20(4):249–257.
- [Islam *et al.*2017] Islam, R.; Hairston, W. D.; Oates, T.; and Mohsenin, T. 2017. An eeg artifact detection and removal technique for embedded processors. In *2017 IEEE Signal Processing in Medicine and Biology Symposium (SPMB)*, 1–3. IEEE.
- [Islam, Rastegarnia, & Yang2016] Islam, M. K.; Rastegarnia, A.; and Yang, Z. 2016. Methods for artifact detection and removal from scalp eeg: a review. *Neurophysiologie Clinique/Clinical Neurophysiology* 46(4):287–305.
- [Jafari *et al.*2017] Jafari, A.; Gandhi, S.; Konuru, S. H.; Hairston, W. D.; Oates, T.; and Mohsenin, T. 2017. An eeg artifact identification embedded system using ica and multi-instance learning. In *2017 IEEE International Symposium on Circuits and Systems (IS-CAS)*, 1–4.
- [Jafari *et al.*2018] Jafari, A.; Ganesan, A.; Thalisetty, C. S. K.; Sivasubramanian, V.; Oates, T.; and Mohsenin, T. 2018. Sensornet: A scalable and low-power deep convolutional neural network for multimodal data classification. *IEEE Transactions on Circuits and Systems I: Regular Papers* 1–14.
- [Jung *et al.*1998] Jung, T.-P.; Humphries, C.; Lee, T.-W.; Makeig, S.; McKeown, M. J.; Iragui, V.; and Sejnowski, T. J. 1998. Extended ica removes artifacts from electroen-

cephalographic recordings. In *Advances in neural information processing systems*, 894–900.

[Kaiser, Gomez, & Chollet2017] Kaiser, L.; Gomez, A. N.; and Chollet, F. 2017. Depthwise separable convolutions for neural machine translation. *arXiv preprint arXiv:1706.03059*.

[Khatwani *et al.*2018] Khatwani, M.; Hosseini, M.; Paneliya, H.; Mohsenin, T.; Hairston, W. D.; and Waytowich, N. 2018. Energy efficient convolutional neural networks for eeg artifact detection. In *2018 IEEE Biomedical Circuits and Systems Conference (BioCAS)*, 1–4. IEEE.

[Krizhevsky, Sutskever, & Hinton2012] Krizhevsky, A.; Sutskever, I.; and Hinton, G. E. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, 1097–1105.

[Lawhern *et al.*2012] Lawhern, V.; Hairston, W. D.; McDowell, K.; Westerfield, M.; and Robbins, K. 2012. Detection and classification of subject-generated artifacts in eeg signals using autoregressive models. *Journal of neuroscience methods* 208(2):181–189.

[Lawhern *et al.*2018] Lawhern, V. J.; Solon, A. J.; Waytowich, N. R.; Gordon, S. M.; Hung, C. P.; and Lance, B. J. 2018. EEGNet: a compact convolutional neural network for EEG-based brain–computer interfaces. *Journal of Neural Engineering* 15(5):056013.

- [Majidov & Whangbo2019] Majidov, I., and Whangbo, T. 2019. Efficient classification of motor imagery electroencephalography signals using deep learning methods. *Sensors* 19(7):1736.
- [Nuwer1988] Nuwer, M. R. 1988. Quantitative eeg: I. techniques and problems of frequency analysis and topographic mapping. *Journal of clinical neurophysiology: official publication of the American Electroencephalographic Society* 5(1):1–43.
- [Petrosian *et al.*2001] Petrosian, A.; Prokhorov, D.; Lajara-Nanson, W.; and Schiffer, R. 2001. Recurrent neural network-based approach for early recognition of alzheimer’s disease in eeg. *Clinical Neurophysiology* 112(8):1378–1387.
- [Podlozhnyuk2007] Podlozhnyuk, V. 2007. Image convolution with cuda. *NVIDIA Corporation white paper, June 2007*(3).
- [Schetinin & Schult2004] Schetinin, V., and Schult, J. 2004. The combined technique for detection of artifacts in clinical electroencephalograms of sleeping newborns. *IEEE Transactions on Information Technology in Biomedicine* 8(1):28–35.
- [Simonyan & Zisserman2014] Simonyan, K., and Zisserman, A. 2014. Very deep convolutional networks for large-scale image recognition. *CoRR* abs/1409.1556.
- [Szegedy *et al.*2017] Szegedy, C.; Ioffe, S.; Vanhoucke, V.; and Alemi, A. A. 2017. Inception-v4, inception-resnet and the impact of residual connections on learning. In *Thirty-First AAAI Conference on Artificial Intelligence*.

- [Winkler, Haufe, & Tangermann2011] Winkler, I.; Haufe, S.; and Tangermann, M. 2011. Automatic classification of artifactual ica-components for artifact removal in eeg signals. *Behavioral and Brain Functions* 7(1):30.
- [Zheng *et al.*2014] Zheng, Y.; Liu, Q.; Chen, E.; Ge, Y.; and Zhao, J. L. 2014. Time series classification using multi-channels deep convolutional neural networks. In *International Conference on Web-Age Information Management*, 298–310. Springer.

