Escola Tècnica Superior d'Enginyeria de Telecomunicació de Barcelona

UNIVERSITAT POLITÈCNICA DE CATALUNYA

Personalized Image Classification from EEG Signals using Deep Learning

A Degree Thesis
Submitted to the Faculty of the
Escola Tècnica d'Enginyeria de Telecomunicació de Barcelona
Universitat Politècnica de Catalunya

In partial fulfillment
of the requirements for the degree in
TELECOMMUNICATIONS SYSTEMS ENGINEERING

Author: Alberto Bozal
Advisor: Xavier Giró-i-Nieto

Universitat Politècnica de Catalunya (UPC)
2016 - 2017

Abstract

This thesis explores the semantic classification of images based processing of electroencephalogram (EEG) signals generated by the viewer's brain. The work extends an existing solution by exploring the gains obtained when the parameters of the classifier are adapted to the user. Firstly, we developed an universal end-to-end model based on deep learning that extracts features from the *EEG* raw signals predicts the semantic content of the image between 40 possible classes from the ImageNet dataset. Our main contribution aims at adapting this universal model to new users, in order to build a personalized model based on the minimum feedback from the new user. We explored different deep learning architectures and hyperparameters to obtain a better accuracy than the baseline by Spampinato et al (CVPR 2017). We achieve a result of 89.03 % and 90.34 % of the universal and personalized model respectively.The developed software and models are publicly available at `https://github.com/Albocal/EEG-Signals-using-LSTM`.

Resum

Aquesta tesi explora la classificació semàntica d'imatges basat en el processament de senyals de electroncefalograma (EEG) generades pel cervell de l'espectador. El treball estén una solució existent explorant la ganacia obtinguda quan els paràmetres del classificador són adaptats a l'usuari. En primer lloc, desenvolupem un model universal d'extrem a extrem basat en el *Deep Learning* que extreu característiques dels senyals raw *EEG* i prediu el contingut semàntic de la imatge entre 40 classes possibles del conjunt de dades ImageNet. La nostra principal contribució té com a objectiu l'adaptació d'aquest model universal als nous usuaris, amb la finalitat de construir un model personalitzat basat en la retroalimentació mínima del nou usuari. Explorem diferents arquitectures d'aprenentatge profund i hiperparámetros per obtenir una major precisió que *baseline* de Spampinato (CVPR 2017). Aconseguim un resultat de 89.03 % i 90.34 % del model universal i personalitzat respectivament. El programa i els models desenvolupats estan disponibles públicament en `https://github.com/albocal/eeg-signals-using-lstm`.

Resumen

Esta tesis explora la clasificación semántica de imágenes basado en el procesamiento de señales de electroncefalograma (EEG) generadas por el cerebro del espectador. El trabajo extiende una solución existente explorando la ganacia obtenida cuando los parámetros del clasificador son adaptados al usuario. En primer lugar, desarrollamos un modelo universal de extremo a extremo basado en el *Deep Learning* que extrae características de las señales raw *EEG* i predice el contenido semántico de la imagen entre 40 clases posibles del conjunto de datos ImageNet. Nuestra principal contribución tiene como objetivo la adaptación de este modelo universal a los nuevos usuarios, con el fin de construir un modelo personalizado basado en la retroalimentación mínima del nuevo usuario. Exploramos diferentes arquitecturas de aprendizaje profundo e hiperparámetros para obtener una mayor precisión que el *baseline* de Spampinato (CVPR 2017). Alcanzamos un resultado de 89.03 % y 90.34 % del modelo universal y personalizado respectivamente. El software y los modelos desarrollados están disponibles públicamente en `https://github.com/Albocal/EEG-Signals-using-LSTM`.

Acknowledgements

First of all, I would like to thank to my tutor Xavier Giro-I-Nieto for helping me during this project as well as giving me the possibility to join to the X-Theses group.

I also want to thank all mates in X-theses group. The weekly meetings were greatly useful to the progress of the thesis and learning new approaches of Deep Learning.

Concetto Spampinato deserve also a mention due to without the cession of the dataset this thesis could not be possible.

And finally, my family and friends for supporting me in the weak moments.

Revision history and approval record

Revision	Date	Purpose
0	05/06/2017	Document creation
1	27/06/2017	Document revision
2	28/06/2015	Document revision

DOCUMENT DISTRIBUTION LIST

Name	e-mail
Alberto Bozal	albertobozal@gmail.com
Xavier Giró i Nieto	xavier.giro@upc.edu

Written by:		Reviewed and approved by:	
Date	05/06/2017	**Date**	28/06/2017
Name	Alberto Bozal	**Name**	Xavier Giró i Nieto
Position	Project Author	**Position**	Project Supervisor

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Statement of purpose

This thesis aims at improving the communications between humans and machines through the brain. Brain Computer Interfaces (BCI) read the activity generated by the brain and this is interpreted by a machine [13]. There are different technologies to read brain signals, but the most common is using the electroencephalograms (EEG).

To explore the EEG signals, we are going to use machine learning techniques, deep learning specifically. Deep learning is a discipline which has become extremely popular in the last years. It consist of using artificial neural networks (NN) to learned feature representations optimized for a certain task.

Brain Computer Interfaces seem futuristic, however they have started being explored by the industry for mass consumption. One of the most important companies based on the Internet, Facebook, has announced research lines on EEG reading. In the *2017 Facebook Developer Conference(F8)*, Mark Zuckerberg was talking about numerous applications of the understanding EEG signals: "We're talking about decoding those words. A silent speech interface - one with all the speed and flexibility of voice." or "We're working on a system that will let you type straight from your brain about five times faster than you can type on your phone today." Elon Musk who known by many as an innovator and visionary of the 21st century, is a South African-born Canadian-American business magnate, investor, engineer, and inventor. He is another admirer of BCI, wanting to create a brain implant to improve his efficiency doing his own life, it is similar to a cyborg[1].

BCI have already nowadays a large extension of applications in helping handicapped people to better communicate or control some damaged organs [17].

Next steps may address virtual reality, with video games where you could play directly with your brain instead of your hands [3]. Medicine could be improved as well if we would understand better how our brain works, so mental diseases could have better treatments.

In our work, we will focus on the specific task of analysing the EEG signals generated in the brain when an image is seen, so that the type of object represented in the image can be distinguished. In particular, we extend the work in [14] by improving the performance and exploring how to quickly personalize a model to a new user.

[1] `https://www.theguardian.com/technology/2017/mar/28/elon-musk-merge-brains-computers-neuralink`

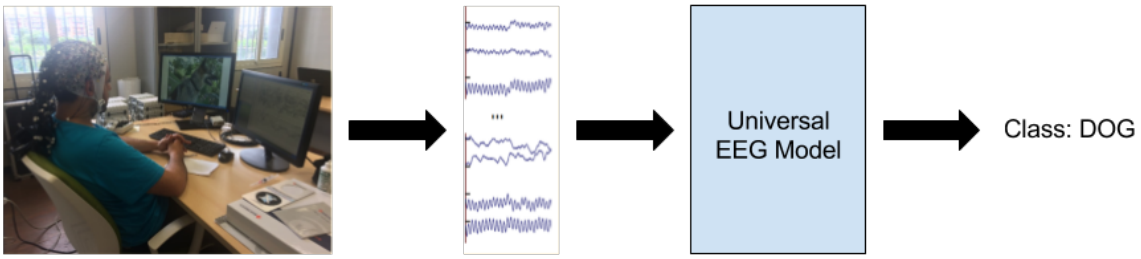

Figure 1.1: Universal model Diagram

Deep learning provides different approaches to solve this classification problem, as simple as a Multi Layer Neural Network (MLNN) or more complex as Convolutional Neural Networks (CNN). We will use a Long Short Term Memory networks (LSTM), that is one of the best models for temporal sequences.

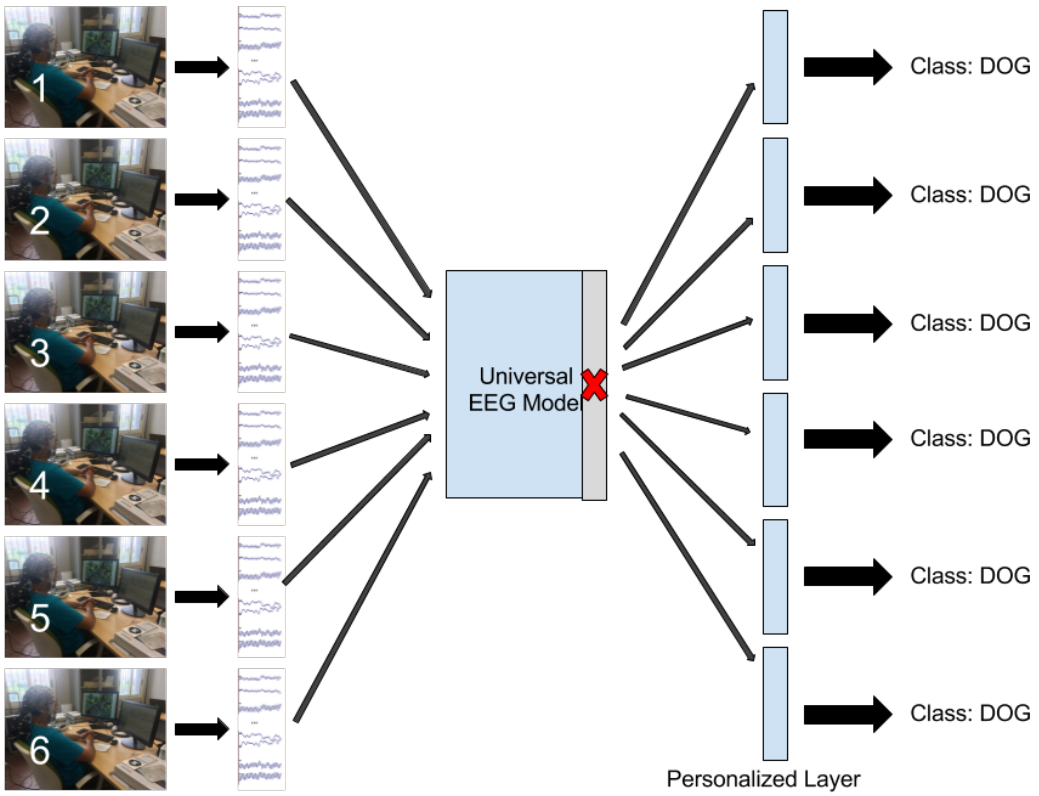

Figure 1.2: Personalized model Diagram

The main objectives of this project are:

- Create an end-to-end model with an accuracy similar to [14].

- Try different solutions and parameters to increase model's accuracy.
- Contribute a solution of the diversity, using a personalized solution.
- Prove the scalability of the personalized model.

1.2 Requirements and specifications

This thesis is used to get closer to state of art of Deep learning processing EEG signals. It is a research project.

Exploring EEG signals, the requirements of this project are the following:

- Create generic model, obtaining a high accuracy
- Create a personalized architecture versatile, simple and faster to fight the diversity of different persons.
- Evaluate the personalized model, finding its benefits and problems.

The specifications are the following:

- Build the software using a Python[2] language
- Use a deep learning framework for project, Keras[3]. It can run upon Theano[4] or TensorFlow[5] backends.
- Create a Github repository to share the project Open Source

1.3 Methods and procedures

This thesis analyses EEG signals using deep learning. For this reason, the Keras software framework is adopted, which is run in the computational servers of the Image Processing Group (GPI) at the Universitat Politecnica de Catalunya (UPC). These servers were run by a Slurm workload manager[6] whose functionality was to distribute the available computational resources between GPUS users.

1.4 Work Plan

This project has followed the established work plan, with a few exceptions and modifications explained in the section 1.5.

[2] https://www.python.org/
[3] https://keras.io/
[4] http://deeplearning.net/software/theano/
[5] https://www.tensorflow.org/
[6] https://slurm.schedmd.com/

1.4.1 Work Packages

- WP 1: Documentation
- WP 2: State of the art
- WP 3: Software
- WP 4: Datasets
- WP 5: Experiments
- WP 6: Oral communication

1.4.2 Gantt Diagram

Name	Duration	IStart	Finish
Documentation	**86d?**	**03/03/2017**	**30/06/2017**
Project Proposal redaction	2d	03/03/2017	06/03/2017
Project Proposal revision	1d	07/03/2017	07/03/2017
Project Proposal revision	1d	08/03/2017	08/03/2017
Project Proposal approval	4d	03/05/2017	08/05/2017
Critical Review redaction	1d	09/05/2017	09/05/2017
Critical Review approval	1d?	10/05/2017	10/05/2017
Final Report redaction	2d	23/06/2017	26/06/2017
Final Report revision	4d	26/06/2017	29/06/2017
Final Report approval	5d	26/06/2017	30/06/2017
State of the art	**65d?**	**03/03/2017**	**01/06/2017**
Learn general concepts of Deep Learning	29d?	03/03/2017	12/04/2017
Learn how works actually state of art for EEG analysis techniqu	33d	15/03/2017	28/04/2017
Read papers about EEG	57d	15/03/2017	01/06/2017
Software	**78d?**	**03/03/2017**	**20/06/2017**
Learn Python, Keras and Tensorflow	41d?	03/03/2017	28/04/2017
Test some End-to-end structures	25d?	13/04/2017	17/05/2017
Develop the project	44d?	20/04/2017	20/06/2017
Datasets	**36d?**	**03/03/2017**	**21/04/2017**
Research useful dataset of EEG signals	36d?	03/03/2017	21/04/2017
Experiments	**27d?**	**11/05/2017**	**16/06/2017**
Train and test the solution	8d?	11/05/2017	22/05/2017
Improve the solution	5d?	22/05/2017	26/05/2017
Evaluate the results	12d?	01/06/2017	16/06/2017
Oral presentation	**11d?**	**30/06/2017**	**14/07/2017**
Rehearsal of the presentation	7d?	30/06/2017	10/07/2017
Oral defense	5d?	10/07/2017	14/07/2017

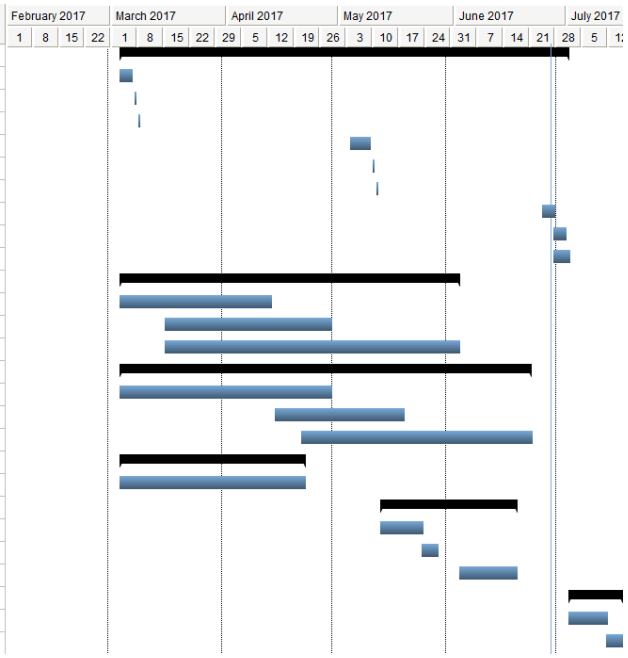

Figure 1.3: Gantt Diagram of the Degree Thesis

1.5 Incidents and Modification

This research thesis was constantly growing and adapting to the new results. In a first phase, the development of a universal model that obtains a high accuracy was the main goal. However, as the project advanced, we identified the practical limitation of such approach and focus on the specific problem of adapting the universal to a new user, which better captures the real world application of these technologies.

Chapter 2

State of the art

2.1 Image analysis from EEG signals

Analysing images from EEG readings with classic machine learning has been previously explored by different works, but using deep learning [5] is still in its early days. The work in [6] tries to detect objects but with a black background, a similar problem to ours but with a more simple problem. The work in [2] is about image classification but only with airplanes with the difficulty of these images being shown at a high frame rate. The work in [18] adapts a general model to a individual users. The goal in his work is create a personalized of a SVM model for different user. Our thesis has the same goal, but addresses the same problem using a deep learning approache.

Other works, less related to ours, do not analyse the raw EEG signal but try to detect a P300 signal [4, 7]. P300 is a 300 ms latency way which is related with the process of decision making. ERP (event related potentials) includes the search of p300 signal.

The system in [11] that allows the possibility of move a robotic arm with the brain. In it, 13 users learn how to use the robotic arm only from their brain activity.

Mentioned in the introduction, there are some studies related with mental issues as detection of epilepsy or compression better the diseases in [17].

The study of airplanes [2] works exploring ways to improves the extraction of features using frequency-bands. In this work [9] is based on Deep Learning applications for EEG signals works too on this, using frequency bands features extraction. However raw data are a new possibility for processing EEG signals.

2.2 Processing EEG signals with Deep Learning techniques

Exploring EEG signals using deep learning we can find [10, 19] related with emotion recognition. So, they use different techniques, one of them is [10] shows a video which it is labeled with the emotion felt in each time lapse by the users. The other study in [19] is about the emotion when you are listening music.

The work in [12] the extraction features using deep learning for uses on more traditional machine learning techniques. This is another way that is exploring due to the old systems works pretty well to do decision and the problems are in process the data.

All studies appeal to different Deep Learning techniques depending on the dataset and their finality. Simple architectures as stack of dense layers or Deep belief networks in [19]. When it uses the dataset in frequency band is common the Convolutional Neural Network (CNN). The work in [2] plots the frequency responses and processes the resulting information as an image. Others works incorporate the spatial distribution of the sensors over the skull, creating a image

of the electrode values [1].

The work in [15] explores the use of EEG signals directly with raw data. Using Deep Learning they can learn how to extract features without frequency transformations. The most useful Deep Learning architecture is LSTM for time lapses as this dataset.

Our work is greatly influenced by a recent work by Concetto Spampinato et al [14], which will be presented in July 2017 in the IEEE Conference on Computer Vision and Pattern Recognition (h5-index=140). The dataset collected in that work has been used in this thesis for all experiments. The paper [14] presents two approaches, one is an image classification task between 40 classes from ImageNet, while the second task is transfer learning experiment to classify images using features from an EEG manifold. They use the raw data to classify the EEG signals using LSTM. They test some models, and the best is a stack of LSTMs with 128 neurons.

Chapter 3

Methodology

3.1 Development Framework

3.1.1 Deep Learning Libraries

Deep learning development is highly dependent on the software framework chosen to develop the project. There exist different frameworks available with open source license: TensorFlow, caffe/caffe2, Theano, Pytorch and Keras. The first option was TensorFlow, an open-source software library for deep learning created and maintained by Google. However, its programming require a high level of detail that was not necessary for this thesis. Caffe/Caffe2 are focus on production and industry, but not as much for researche. Caffe2 was very new and the related information still short. Another two interesting frameworks were Pytorch and Theano. PyTorch is growing up and becoming very popular among researchers. PyTorch is an interface of Torch, which is programmed with lua, also PyTorch is a relationship with C/C++ code that is used on most libraries for scientific computations. As TensorFlow, PyTorch has a slow learning curve. Theano was used by me in another project and it did make me not feel comfortable working with this.

The final decision was use Keras. It is high level language, getting an advantage to create a precompiled models. Usual functions like padding or dropping are included. Using Keras, we had to choose the backend between Theano and TensorFlow, and we opted for the later.

3.1.2 Synchronized environment with server

To develop the project correctly, we had to choose the Integrated Development Environment (IDE). Pycharm with student license was the selected option. The settings presents many possibilities, we had to configure all to run and debug the project. Configuring the Project Interpreter remotely we were saving some compatibility problems of libraries. Deployment also has been configured sending the files before of run.

3.1.3 Github

In order to control the changes, the issues or restore any version, we decided to use Version Control System (VCS). The most common VCS is Git, it allows to have two branches one for each architecture, the general and the personalized model.

For our Git repository we decide use Github. Github allows free repositories in exchange of offer the code open source taking the possibility of whoever can use the code. We use Github repository in this project.

3.1.4 GPI Computation Server

The Image Processing Group (GPI) at UPC provided the necessary CPUs and GPUs to train our models. Training deep learning models typically require high amounts of time and resources. We used Tmux to maintain an open session, while it was being trained. Tmux is a terminal multiplexer, giving the possibility of has more than one terminals open at the same time remotely on the server. Even using the GPI servers, we had a big bottleneck in the computations. One simulation could be delay 1 day or even 4 days during periods of high demands of GPUs.

3.2 Dataset

Dataset is a very important part for machine learning projects as they model the problem to be solved. In this thesis one of the big challenges was finding a good dataset. During the search period, we considered the different datasets shown in Table 5.1.

First author	Year	Publication	Application	Dataset	Subjects/ Samples	Layers used	URL
Bashiva.	2016	ICLR	Repeated char	EEGLearn[1]	13 / 2670	CONV LSTM	Slides GPI[2]
Lawhern	2016	IEEE Transactions	P300	Kaggle[3]	16 / 80	CNN	Preprint[4]
Spampi.	2016	CVPR 2017	Image classification	PRIVATE	6 / 12000	LSTM	Preprint[5]
Stober	2016	-	Emotional Music	OpenMIIR[6]	10 / 120	CNN	Preprint[7]

Table 3.1: Collection of Datasets related paper. More Datasets can be found in BNCI Horitzons2020[8]

As previously state, the selected dataset was the one provided by Dr Spampinato [14]. During its acquisition 10-20 electrode were employed using a 128 channels brain cap. The subjects were stimulated with images while EEG signals were recording. The dataset consist in EEG signals of 6 subjects, 40 classes of ImageNet(dog, cat, etc.[9]). Every class has 50 samples images. One

[-1] http://bnci-horizon-2020.eu/database/data-sets\unskip\penalty\@M\vrulewidth\z@height\z@depth\dpff
[0] https://github.com/pbashivan/EEGLearn
[1] https://www.slideshare.net/xavigiro/learning-representations-from-eeg-with-deep-recurrent-convolutional-n
[2] https://www.kaggle.com/c/inria-bci-challenge
[3] https://arxiv.org/abs/1611.08024
[4] https://arxiv.org/abs/1609.00344
[5] https://github.com/sstober/openmiir
[6] https://arxiv.org/abs/1511.04306
[7] http://bnci-horizon-2020.eu/database/data-sets\unskip\penalty\@M\vrulewidth\z@height\z@depth\dpff
[8] http://bnci-horizon-2020.eu/database/data-sets\unskip\penalty\@M\vrulewidth\z@height\z@depth\dpff
[9] ImageNet classes used: dog, cat, butterfly, sorrel, capuchin, elephant, panda, fish, airliner, broom, canoe, phone, mug, convertible, computer, watch, guitar, locomotive, espresso, chair, golf, piano, iron, jack, mailbag,

sample EEG signal of one image has 128 raw channels. The time showing every images was half second. Due to some issues subjects 1 and 2 have only 30 image classes of ImageNet.

Dataset partition was 80% Train Dataset, 10% validation Dataset and 10 Test Dataset. This partition of the dataset follows the same proportion as the ones used in [14] to facilitate the comparison of results.

Number of classes	40
Number of images per class	50
Number of subjects	4
Time for each image	500ms
Total of samples	8000

Table 3.2: Information Dataset 1.

Number of classes	30
Number of images per class	50
Number of subjects	2
Time for each image	500ms
Total of samples	3000

Table 3.3: Information Dataset 2.

The dataset was delivered as a Matlab file, which is not the best option when working in Python. In order to organize one-file-sample and improving the optimization, we decide to created a HDF5 dictionary. HDF5 is a data model, library, and file format for storing and managing data with a high compatibility with python and it is commonly used in research. Inside this HDF5 file, the classes were the top selection followed by each unique image ID and finish with the subjects.

Level 0	**Level 1**	**Level 2**	**Length**
ID CLASS	-	-	40
$\longrightarrow$	ID IMAGE	-	50
$\longrightarrow$	$\longrightarrow$	ID SUBJECT	4 / 6

Table 3.4: Information Dictionary Dataset DHF5.

missile, mitten, bike, tent, pajama, parachute, pool, radio, camera, gun, shoe, banana, pizza, daisy and bolete (fungus)

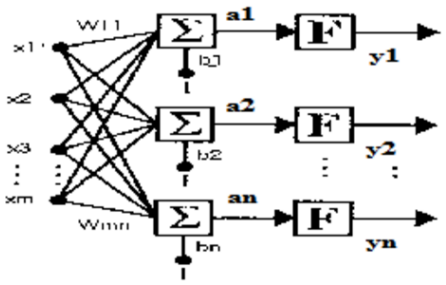

Figure 3.1: Dense layer architecture

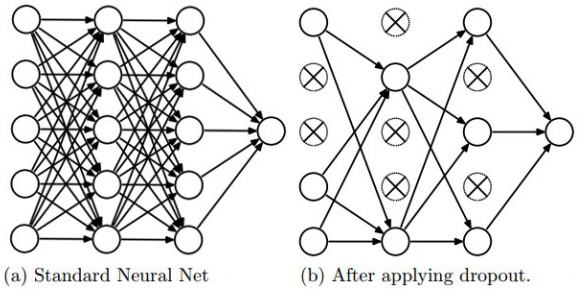

Figure 3.2: Dropout example

3.3 Deep Neural Networks

3.3.1 DENSE layer

A Dense layer, Fully-Connected neural network layer, is the most simple model in deep Learning. All architectures are based on this idea.

This architecture consist in a only one layer of neurons. As Figure 3.1 each neuron has a bias, a constant number that initialized randomly. The connections between neurons are the weights, each connection has a value that it will be changing when the network is fitting. A DENSE layer is a layer of neurons which each neuron is connected with all other neurons of the previous layer.

$$\mathbf{H}_n = \mathbf{H}_{n-1} * \mathbf{W}^t + \mathbf{B} \tag{3.1}$$

The operation is based on all weights of connections have to be multiplied with all previous values of previous layer of neurons and after that it has to be added the bias.

3.3.2 Long Short Term Memory (LSTM)

Given the nature of a temporal sequence of the EEG data, we adopted as a deep neural network the Keras implementation of the Long Short Term Memory network (LSTM) [8]. LSTMs are a type of Recurrent Neural Network (RNN) with some gating mechanisms that reduce the problem of exploding and vanishing gradients common in the vanilla RNNs.

The recurrency states are described by:

$$
\begin{aligned}
\mathbf{f}_t &= \sigma(\mathbf{W}_f \cdot [h_{t-1}, \mathbf{x}_t] + \mathbf{b}_f) \\
\mathbf{i}_t &= \sigma(\mathbf{W}_i \cdot [h_{t-1}, \mathbf{x}_t] + \mathbf{b}_i) \\
\widetilde{\mathbf{C_t}} &= \tanh(\mathbf{W}_C \cdot [h_{t-1}, \mathbf{x}_t] + \mathbf{b}_C) \\
\mathbf{C}_t &= \mathbf{f}_t * \mathbf{C}_{t-1} + \mathbf{i}_t * \widetilde{\mathbf{C_t}} \\
\mathbf{o}_t &= \sigma(\mathbf{W}_o \cdot [h_{t-1}, \mathbf{x}_t] + \mathbf{b}_o) \\
\mathbf{h}_t &= \mathbf{o}_t * \tanh(\mathbf{C_t})
\end{aligned}
\tag{3.2}
$$

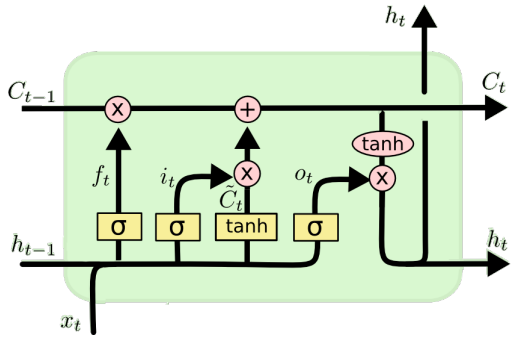

Figure 3.3: LSTM behavior example

In Figure 3.3, the top line $\mathbf{C_t}$ corresponds to the memory of the model. Firstly, a sigmoid decide if it is necessary to forget the previous step. Continue calculating the update of the cell adding a value. This value is the result of a sigmoid which decides if has to be updated and the value of tanh of a neural network. Finally the last sigmoid decide the output of the LSTM.

3.3.3 Training the model

Training a deep learning model is computationally complicated. There are many internal hyper parameters: number epoch, number batches, loss, optimizers, etc. We try different epoch and batches, doing different trials. The loss function in our case was categorical cross entropy, a common choice when classifying between different classes.

Dropout is a technique to do no over-fit. It is used in our model so we explain the main idea using the Figure 3.2. Imagine that you have the output layer, this layer can be 40 neurons, it is an example. When you put dropout, you are deactivating the ratio number, dropout, between 0 to 1. Dropout 0.5 of 40 neurons means that you are receiving in the next layer only 20 neurons. However, you are not disabling always the same neurons, it is random. Doing Dropout you are creating a more robust model. Dropout is commonly used in architectures with many parameters to be learned.

The optimizer is the method to use the loss and calculate the new possible value. There are a lot of optimizers, but testing all the possible optimizers requires too much time. We follow the results in [16] for LSTMS to limit the search to two optimizers: *adam* and *rmsprop*.

3.4 Universal model

The universal model consists in a similar model to the one in [14]. We stack two LSTMs and at the end, we added a Dense layer using the *Softmax* function to activate the final result. The final predictions are encoded with one-hot which is used for the classifications outputs. It is based on the idea which if you are classifying between dogs, cats or horses, you can only choose one result; the result can not be a dog and horse at the same time.

The dataset uses a 200 samples window instead of 500ms (time of the full sample). Each EEG signal recording has a 128 channels electrodes for each millisecond. Our final input dataset is a

vector of [128,200]. We take this idea from [14], where using all dataset of one sample obtains worse accuracy than using evolved.

We started working in a first model whose layers were decreasing in number of output neurons (Figure 3.4).

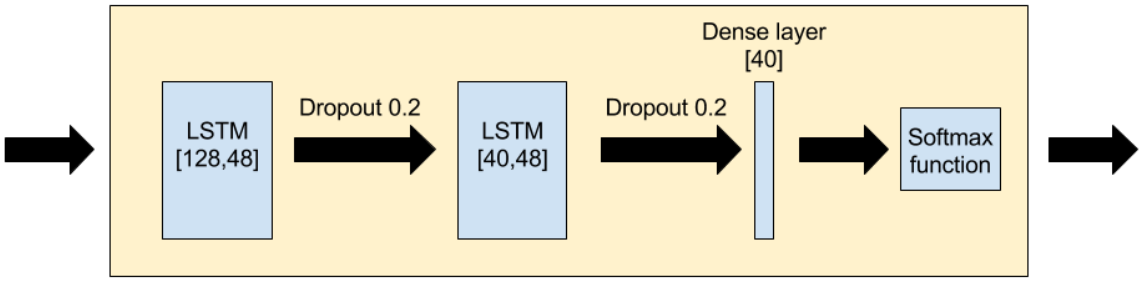

Figure 3.4: Universal model initial architecture

We decided to change the architecture due to poor results. We gave the important function to the Dense layer. Creating a model that increase the number parameters, due to inside LSTMs contains a vector [128,200] and the output of LSTM is only one layer (Figure 3.5).

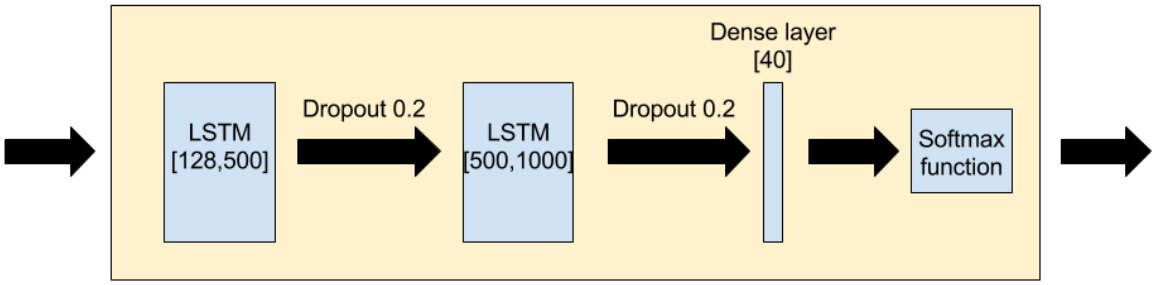

Figure 3.5: Universal model final architecture

This architecture has some hyper-parameters. We decided by trial an error to use 20-30 epoch and 200 samples per batch. We added also a early stopping: if we were training a model and the validation accuracy did not increase in 3 epochs, it stooped the training.

Previously, we explained the windows size. So, if we have 500ms and we only choose 200 ms (200 samples). What samples we have to choose? Based on [14], the first 50 samples were ignored due to the possibility of interferences with previous displayed image. We select two possibilities, the first possibility is taking the 200 samples with an offset of 50 ms; and the second also based on [14], taking the 200 last samples. For the simplicity, we prefer use a 250ms offset. To sum up, we have two options: 50-250ms and 250-450ms.

3.5 Personalized model

After some experiments we realized that, depending of the subject, the accuracy of the model could present a high variance.

In this thesis we want contribute a new approach, a personalized model. This solution focuses on the topic of diversity of users. Our dataset depends of who is watching the images. This new approach consist of personalizing the universal model for different subjects, obtaining one model optimized for each new subject.

In machine learning, transfer learning refers to using a model for a different but related problem. In deep learning the transfer learning is commonly used for save time training an architecture that has already been trained to a similar problem. Using the fine-tune technique is possible to reuse a neural network without retraining all its parameters. For example, by changing the last layers and adapting them to your problem. In our case, we tested the accuracy of how many layers we have to freeze.

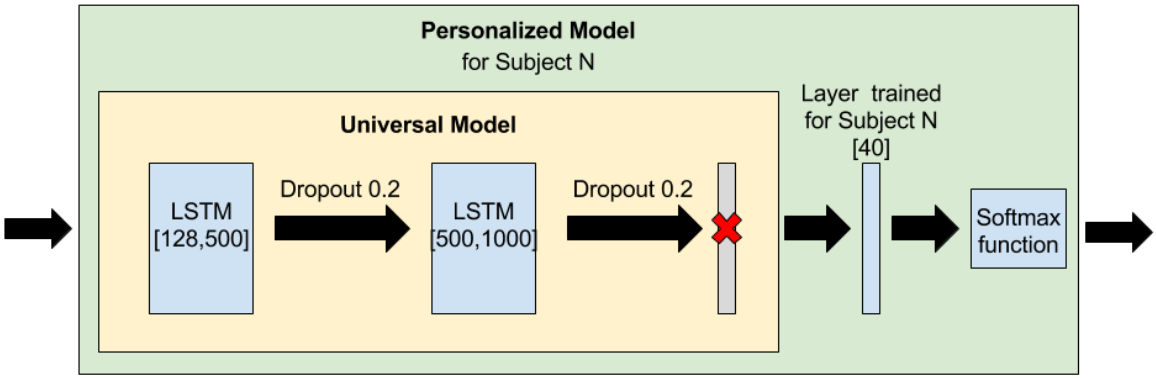

Figure 3.6: Personalized model architecture

An advantage of using fine-tuning is that the training part is much faster than training from scratch. In this case, we are only training one dense layer, adapting this layer for the specific subject.

As the dataset is too small to create a generalized model and test it with unknown subjects. So, we had created 6 models, each model without one. We will evaluate this models with the held-out subject.

Chapter 4

Results

In this chapter, the proposed models are evaluated and compared.

4.1 Training results: losses and confusion matrix

The accuracy and loss plots related with the final model are in Figures 4.1 and 4.2. Observing the two figures, there are multiple peaks of accuracy, but we obtain the model by saving the model weights at the largest peak.

Figure 4.3 contains the probability distribution among classes. It was a simple method to know if all errors were in the same classes and the probability on each class was distributed uniform or there are classes less probable and others more.

Observing the Figure 4.3 the classification is good, all predicted and real classes are near. However the errors are intensive at two classes, for these situations the graph was painted.

Another method used to study the behavior of the model is through the confusion matrixs in Figures 4.4 and 4.5. We also added an average of the probability matrix of each class obtained in Softmax output in Figure 4.6.

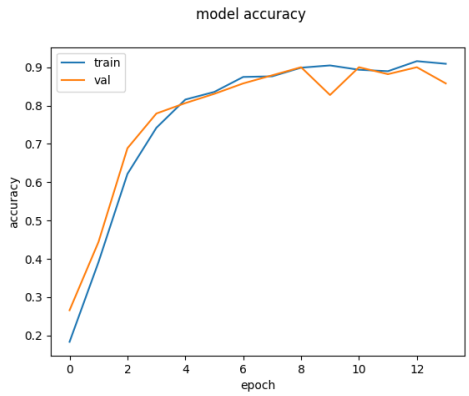

Figure 4.1: Accuracy curve in final model

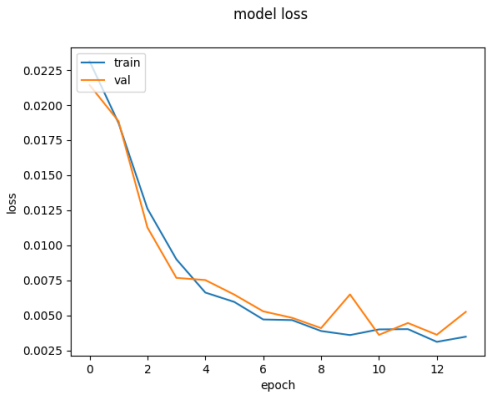

Figure 4.2: Loss curve in final model

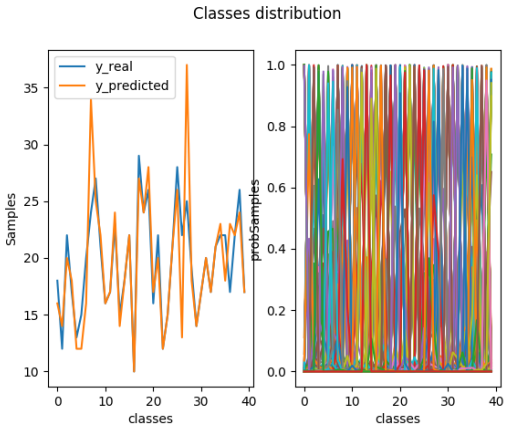

Figure 4.3: Classes distribution: Real vs predicted Samples at left and Probability distribution at right

Observing this last Figures 4.4, 4.5 and 4.6. We can understand better the errors among the classes. The results of the three figures are spectacular. Using the two first figures you can know the accuracy and the distribution of the classes clearly. Adding the last figure, you can understand the real behavior of the model, the real possibilities before of the model choose the output result. With the model we expect a good results but we never expected this results, the average probability of correct output is 0.9.

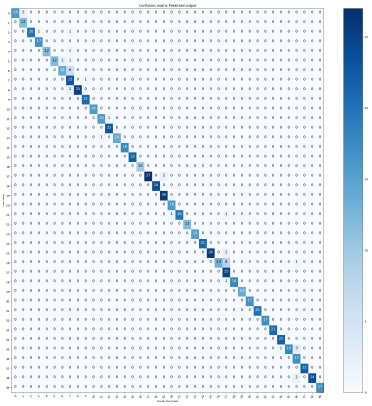

Figure 4.4: Confusion Matrix of samples

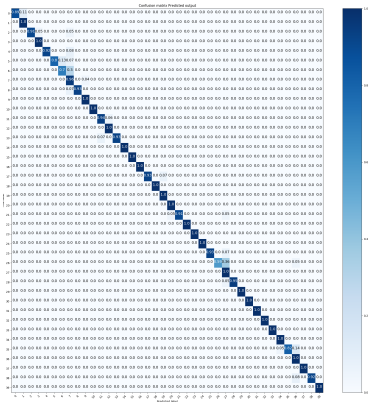

Figure 4.5: Normalized Confusion Matrix

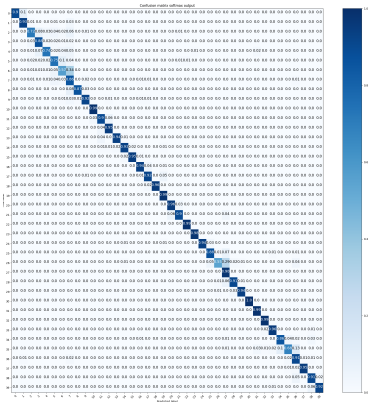

Figure 4.6: Probability Matrix at Softmax

4.2 Quantitative results: Universal Model

We evaluated the universal model classifying among 40 classes. We compared our four best results in Table 4.1 with the ones in [14].

Model	Classes	Train Acc	Val Acc	Test Acc
UNIVERSAL-Adam_o50	40	71.50 %	80.17 %	76.79 %
UNIVERSAL-Rmsprop_o50	40	89.28 %	89.51 %	87.63 %
UNIVERSAL-Adam_o250	40	74.49 %	79.32 %	75.38 %
UNIVERSAL-Rmsprop_o250	40	**91.06 %**	**91.50 %**	**89.03 %**
Spampinatto [14]	40	- %	85.4 %	82.9 %

Table 4.1: Results of Universal model 40 classes. Offset 50 vs 250 and Adam vs Rsmprop

The results in [14] used an adam optimizer, but we also tried with the rmsprop, obtaining a better accuracy. We can see that using and offset of 250 we obtain better accuracy than offset of 50. Furthermore, if we add the Rmsprop optimizer our accuracy increased considerably obtaining the best model at Table 4.1.

The Dataset's problems, our dataset dataset are divided in 2 parts: 4 subjects with 40 images classes and 2 subjects with 30 images classes, the personalized model was based on 30 classes having 6 subjects instead of 4 subjects. In order to compare the universal and the personalized model, we have to calculate the accuracy of universal for 30 classes, contained in Table 4.2.

Model	Classes	Train Acc	Val Acc	Test Acc
UNIVERSAL-Adam_o50	30	76.55 %	73.80 %	72.00 %
UNIVERSAL-Rmsprop_o50	30	87.6 %	88.54 %	87.98 %
UNIVERSAL-Adam_o250	30	77.72 %	77.46 %	74.04 %
UNIVERSAL-Rmsprop_o250	30	**89.50** %	**91.06** %	**89.80** %

Table 4.2: Results of Universal model 30 classes. Offset 50 vs 250 and Adam vs Rsmprop

4.3 Quantitative results: Personalized Model

The results for personalized model are evaluated using N-fold cross validation. We will use the 6 universal models with a held-out subject to test our model. In this case, we have to choose how many layers we want to fine-tune (Table 4.3).

Continue evaluating the personalized model, we have to train the different models missing one different user in each model in Table 4.4. We had chosen fine-tune only the last layer. It is a universal model but prepared to do the n-fold cross validation for testing in each subject.

Model	Frozen layers	Train Acc	Val Acc	Test Acc
Personalized Model 1	0/3	90.57 %	**92.13 %**	90.31 %
Personalized Model 2	1/3	86.45 %	89.89 %	89.80 %
Personalized Model 3	2/3	**92.00 %**	**92.13 %**	**91.33 %**

Table 4.3: Results of Fine-Tune. Freezing: None vs 1 LSTM vs 2 LTSMs

Model	T_Classes	Train Acc	Val Acc	Test Acc
MODEL-R_o250 Subjects-(N6)	1, 2, 3, 4, 5	91.07 %	91.24 %	91.7 %
MODEL-R_o250 Subjects-(N5)	1, 2, 3, 4, 6	91.68 %	92.45 %	86.12 %
MODEL-R_o250 Subjects-(N4)	1, 2, 3, 5, 6	90.13 %	93.35 %	88.57 %
MODEL-R_o250 Subjects-(N3)	1, 2, 4, 5, 6	91.45 %	90.33 %	87.89 %
MODEL-R_o250 Subjects-(N2)	1, 3, 4, 5, 6	89.13 %	88.22 %	85.71 %
MODEL-R_o250 Subjects-(N1)	2, 3, 4, 5, 6	94.16 %	96.98 %	95.24 %

Table 4.4: Results of the universal models for testing Personalized.

We need the results in Table 4.4 to test the personalized model. Something strange is that we can see without Subject 1 and 6 we have obtained better accuracy. Then, we are going to use this universal models to test the different Personalized Models.

The results in Figure 4.7 and Table 4.5 show the evolution when you add a new subject to the model. Firstly, we can see the generalization of the predecessors subjects, random choice is $100/30 = 3.33$ %, 3 of 6 models pass this accuracy, but Subjects 1 and 2 are different. The results of Personalized-FT1 and Personalized-FT2 do not follow the curve of the others. In [14], 10 classes among these subjects were rejected due to recording problems. It is possible that these two subjects had more noise that the others.

We are going to consider the averages in Table 4.6 respect Figure 4.7 as a results of our model. Observing the AVERAGE FT3-6, we can see clearly that all subjects are related a little bit. This can not be taking as true because we are ignoring two personalized models, subject 1 and 2.

Model	**Samples / class - TEST Accuracy %**						
-	**0**	**5**	**10**	**15**	**20**	**25**	**30**
Personalized-FT6	9.33	50.00	66.00	78.00	86.67	88.67	92.00
Personalized-FT5	2.67	61.33	80.67	89.33	92.67	93.33	92.67
Personalized-FT4	4.67	56.67	72.67	84.67	90.67	90.00	94.67
Personalized-FT3	6.67	45.33	58.67	84.00	89.33	89.33	91.33
Personalized-FT2	1.33	33.33	66.00	75.33	78.67	83.33	84.67
Personalized-FT1	2.67	50.00	73.33	77.33	75.33	82.67	86.67

Table 4.5: Results of Personalized models using Dataset with 30 classes. Test it with 150 samples

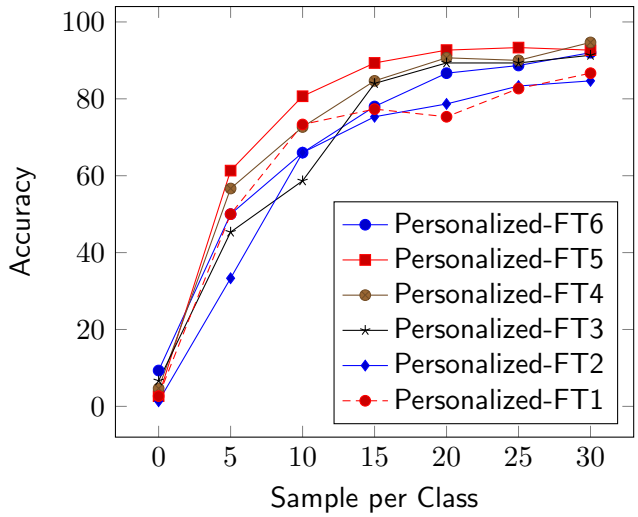

Figure 4.7: Plot of Personalized models using Dataset with 30 classes. Test it with 150 samples

Result	**Samples / class - Accuracy %**						
-	**0**	**5**	**10**	**15**	**20**	**25**	**30**
AVERAGE	**4.56**	**49.44**	**69.56**	**81.44**	**85.56**	**87.89**	**90.34**
AVERAGE FT3-6	5.84	53.33	69.50	84.00	89.84	90.33	92.67

Table 4.6: Average Results of Personalized models using Dataset with 30 classes.

Observing both average results in Table 4.6, AVERAGE and AVERAGE FT3-6, we can see the improve over the universal model of 30 classes. The maximum accuracy in universal model was 89.80 % in Table 4.2. We obtain 90.34 % or 92.67 % depending if we consider the subjects 1 and 2 or not, respectively.

4.3.1 Statistical significance on Personalized Model

We want to show accurate numbers about how our personalized works. Trying to be rigorous, the results of all personalized models will be tested. The personalized model results are going to test with statistical significance. The idea is know if the results are related with all trials or there are any kind of disagreement between them. We will use the p-value to know if exit the possibility of rejection the results due to null hypothesis. Rejecting the null hypothesis we have to obtain a p-value less than 0.05.

Calculating p-value we use a Tool from independent Treatment (ANOVA) and we obtain a result of p-value=4,79642E-13. According to this result, our results are statistically significant.

Chapter 5

Budget

This thesis has been a research. It can not offer any type of service or product to sell it. We can divide this budget in two important parts: hardware/computational usage and staffs.

5.1 Server Budget

The server used in this thesis was provided by GPI. The hardware will be mentioned to understand its magnitude. The server has some nodes but we used only one "c8" which consist in CPU: Intel Xeon 2.6GHz 16 cores (2600€), 120 GB RAM (1600€) and 1/8 GPUs: GTX Titan X(1200€we used only one GPU).

The computation time had be approximately 12h/day since we started the thesis. To calculate more approximately the real cost we can calculate the could instances servers. Amazon Web Services has a g2.8xlarge EC2 which is similar to our specifications, using 60GB of ram and 4 GPUs each with 4 RAM. The cost per hour is 2,808€. We used 12 h/day, total cost at day is 33,69 €. We spend 60 days approximately using the computing resources, thus giving an approximate cost of 2021,4 €.

5.2 Staff budget

In this thesis, the salary is the most important because it is a research. The team consist in a senior engineer as the advisor and myself as junior engineer. Since we started the thesis, it's been 24 weeks, mentioned in Gantt Diagram.

	Weeks	**Wage/hour**	**Dedication**	**Total**
Junior engineer	24	12 €/h	25 hour/week	**7200 €**
Senior engineer	24	25 €/h	2 hour/week	**1200 €**
				8400 €

Table 5.1: Salary Budget

Chapter 6

Conclusions

The main goal was process EEG signals with Deep Learning techniques. BCI is far to works perfectly, however there are some studies which partially works in its application.

We have presented two main contributions. In one hand we outperform the accuracy of [14]. We have been classified an EEG signals of displayed images among 40 classes of ImageNet with an accuracy of 89.90% using a end-to-end model.

In the other hand, we contribute with a new approach differentiating the subjects. We have take into account that each person works different and we created a model that it takes care the diversity of the subjects doing a modification of the first model for each subject, the personalized model.

Our personalized model has the befits of be versatile and it is faster to train than the generalized model. We also have obtained a better accuracy achieving 90.34 - 92.67 %(depending if we take into account the possibility of reject some users which it do not have the same behavior).

We analyze the results of personalized model against null hypothesis. According the analysis our result of personalized model are statistically significant.

Chapter 7

Appendices

7.1 Evolution of Personalized model at Fine-tunning

7.1.1 Training 0 Samples per class (Subject 6)

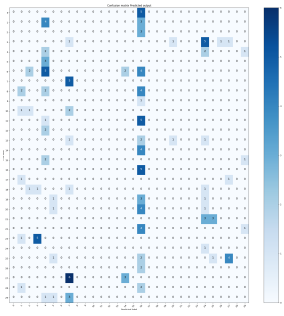

Figure 7.1: Confusion Matrix Samples (0 samples / class)

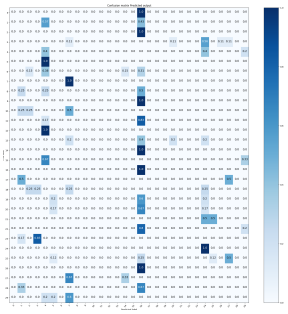

Figure 7.2: Confusion Matrix Normalized (0 samples / class)

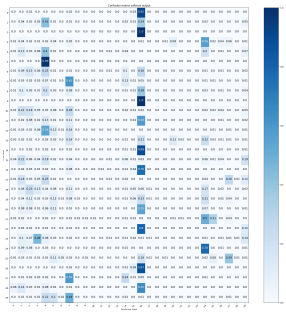

Figure 7.3: Probability Matrix of Classes (0 samples / class)

7.1.2 Training 5 Samples per class (Subject 6)

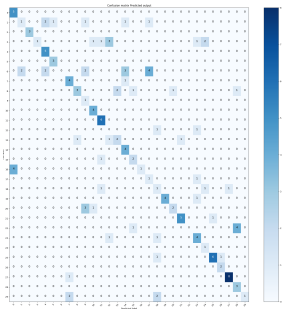

Figure 7.4: Confusion Matrix Samples (5 samples / class)

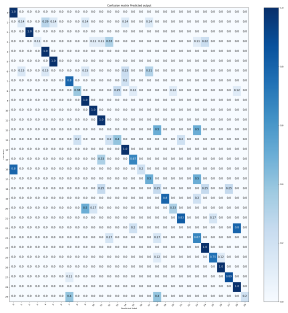

Figure 7.5: Confusion Matrix Normalized (5 samples / class)

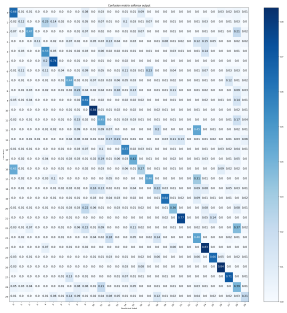

Figure 7.6: Probability Matrix of Classes (5 samples / class)

7.1.3 Training 10 Samples per class (Subject 6)

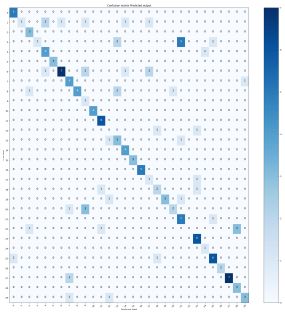

Figure 7.7: Confusion Matrix Samples (10 samples / class)

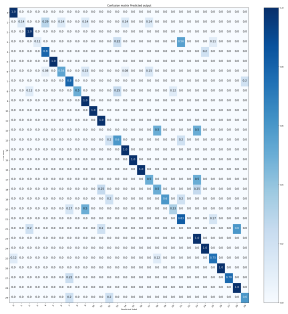

Figure 7.8: Confusion Matrix Normalized (10 samples / class)

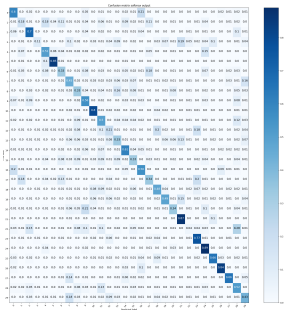

Figure 7.9: Probability Matrix of Classes (10 samples / class)

7.1.4 Training 15 Samples per class (Subject 6)

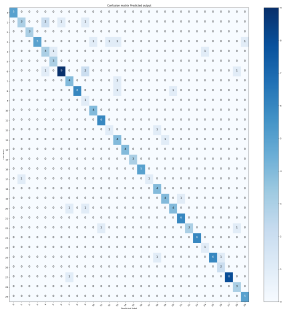

Figure 7.10: Confusion Matrix Samples (15 samples / class)

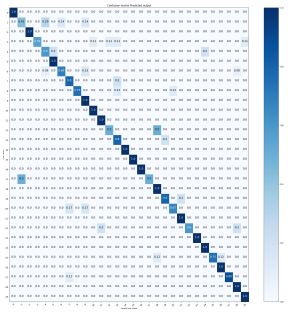

Figure 7.11: Confusion Matrix Normalized (15 samples / class)

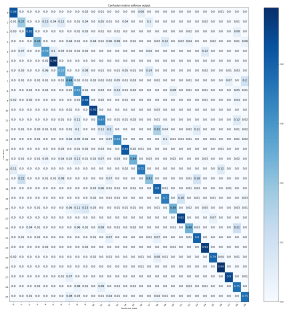

Figure 7.12: Probability Matrix of Classes (15 samples / class)

7.1.5 Training 20 Samples per class (Subject 6)

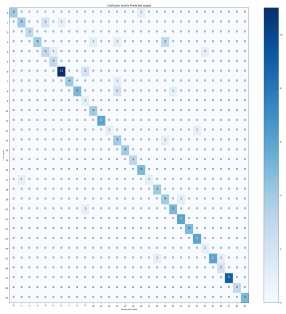

Figure 7.13: Confusion Matrix Samples (10 samples / class)

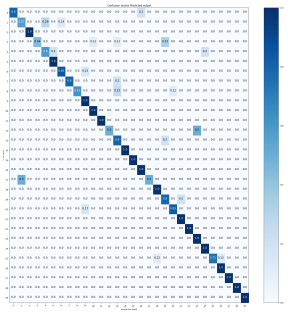

Figure 7.14: Confusion Matrix Normalized (20 samples / class)

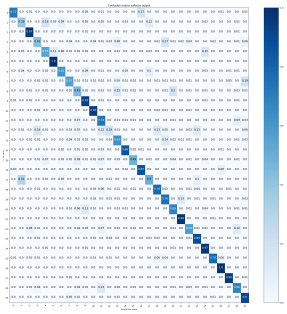

Figure 7.15: Probability Matrix of Classes (20 samples / class)

7.1.6 Training 25 Samples per class (Subject 6)

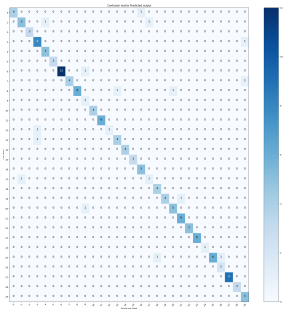

Figure 7.16: Confusion Matrix Samples (25 samples / class)

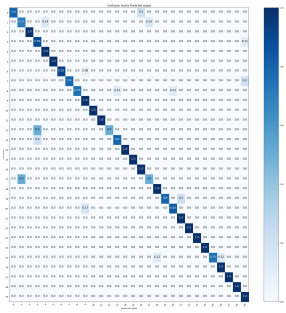

Figure 7.17: Confusion Matrix Normalized (25 samples / class)

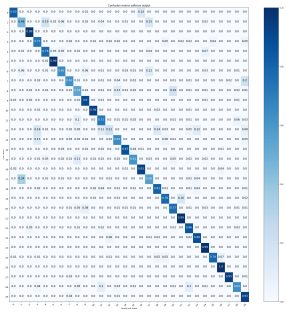

Figure 7.18: Probability Matrix of Classes (25 samples / class)

7.1.7 Training 30 Samples per class (Subject 6)

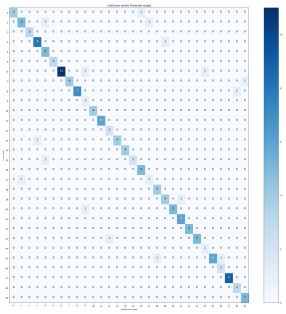

Figure 7.19: Confusion Matrix Samples (30 samples / class)

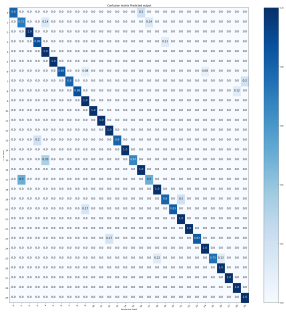

Figure 7.20: Confusion Matrix Normalized (30 samples / class)

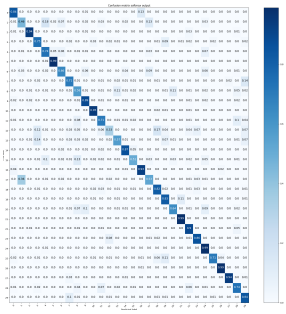

Figure 7.21: Probability Matrix of Classes (30 samples / class)

Bibliography

[1] P. Bashivan, I. Rish, and N. Yeasin, M. andCodella. Learning representations from eeg with deep recurrent-convolutional neural networks. *arXiv preprint arXiv:1511.06448*, 2016.

[2] N. Bigdely-Shamlo, A. Vankov, R. R. Ramirez, and S. Makeig. Brain activity-based image classification from rapid serial visual presentation. In *Neural Systems and Rehabilitation Engineering*, page 16(5):432–441. IEEE Transactions, 2008.

[3] B. Blankertz, M. Tangermann, C. Vidaurre, S. Fazli, C. Sannelli, S. Haufe, C. Maeder, L. E. Ramsey, I. Sturm, G. Curio, and K. R. Mueller. The berlin brain-computer interface: Non-medical uses of bci technology. In *Frontiers in Neuroscience, vol. 4, no. 198*, 2010.

[4] H. Cecotti and A. Graser. Convolutional neural networks for p300 detection with application to brain. In *Computer Interfaces. IEEE Trans. on Pattern Analysis and Machine Intelligence*, page 33(3):433–445, 2011.

[5] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[6] G. Healy and A. F. Smeaton. Optimising the number of channels in eeg-augmented image search. In *In Proceedings of the 25th BCS Conference on Human-Computer Interaction*, page 157–162, British Computer Society, 2011.

[7] G. Healy, Z. Wang, C. Gurrin, T. E. Ward, and A. F. Smeaton. An eeg image-search dataset: a first-of-its-kind in ir/iir. nails: neurally augmented image labelling strategies. 2017.

[8] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. In *Neural computation*, 1997.

[9] S. Jirayucharoensak, S. Pan-Ngum, and P. Israsena. Eeg-based emotion recognition using deep learning network with principal component based covariate shift adaptation. In *The Scientific World Journal, 2014*, 2014.

[10] S. Jirayucharoensak, S. Pan-Ngum, and P. Israsena. Eeg-based emotion recognition using deep learning network with principal component based covariate shift adaptation. 2014.

[11] J. Meng, S. Zhang, A. Bekyo, J. Olsoe, B. Baxter, and B. He. Noninvasive electroencephalogram based control of a robotic arm for reach and grasp tasks. In *Scientific Reports, 6*, 2016.

[12] E. Santana, A. J. Brockmeier, and J. C. Principe. Joint optimization of algorithmic suites for eeg analysis. In *Engineering in Medicine and Biology Society (EMBC)*, pages 2997–3000. 36th Annual International Conference of the IEEE, 2014.

[13] S. Saproo, J. Faller, V. Shih, P. Sajda, N. R. Waytowich, A. Bohannon, and D. Jangraw. Cortically coupled computing: A new paradigm for synergistic human-machine interaction. In *Computer, vol. 49, no. 9, pp.*, page 60–68, Sept 2016.

[14] C. Spampinato, S. Palazzo, I. Kavasidis, D. Giordano, M. Shah, and N. Souly. Deep learning human mind for automated visual classification. *arXiv preprint arXiv:1609.00344*, Sep, 2016.

[15] S. Stober, A. Sternin, A. M. Owen, and J. A. Grahn. Deep feature learning for eeg recordings. 2015.

[16] Y. C. Subakan and P. Smaragdis. Diagonal rnns in symbolic music modeling. *arXiv preprint arXiv:1704.05420*, 2017.

[17] J. van Erp, F. Lotte, and M. Tangermann. Brain-computer interfaces: Beyond medical applications. In *Computer, vol. 45, no. 4, pp.*, page 26–34, Apr. 2012.

[18] J. Yang. A general framework for classifier adaptation and its applications in multimedia. In *PhD thesis, Carnegie Mellon University*, 2009.

[19] W.-L. Zheng, J.-Y. Zhu, Y. Peng, and B.-L. Lu. Eeg-based emotion classification using deep belief networks. In *IEEE Int. Conf. on Multimedia and Expo (ICME)*, page 1–6, 2014.