



**DUBLIN CITY UNIVERSITY  
SCHOOL OF ELECTRONIC ENGINEERING**

**Detect Snap Points in Egocentric  
Images with Physiological Signals**  
Final report

**Marc Carné Herrera**  
April 2016

Undergraduate Project Exchange

Supervised by Dr. Cathal Gurrin  
and Xavier Giró-i-Nieto

## **Acknowledgements**

I would like to thank my supervisor Dr. Cathal Gurrin for allow me to use his egocentric images for this project and Xavier Giró, my supervisor from UPC (Polytechnic University of Barcelona). Thanks are also due to Alejandro Cartas, PhD candidate from University of Barcelona (UB) for his support on web application (that will be presented on this work) building. As there is an experiment as part of this work, and user interaction was needed I would like to thank all that volunteers that done the visual game, contributing to this research. Finally I would like to express my deep appreciation to Albert Gil and Josep Pujal, for helps me with the server set up and computing support.

## Declaration

I declare that this material, which I now submit for assessment, is entirely my own work and has not been taken from the work of others, save and to the extent that such work has been cited and acknowledged within the text of my work. I understand that plagiarism, collusion, and copying are grave and serious offences in the university and accept the penalties that would be imposed should I engage in plagiarism, collusion or copying. I have read and understood the Assignment Regulations set out in the module documentation. I have identified and included the source of all facts, ideas, opinions, and viewpoints of others in the assignment references. Direct quotations from books, journal articles, internet sources, module text, or any other source whatsoever are acknowledged and the source cited are identified in the assignment references. This assignment, or any part of it, has not been previously submitted by me or any other person for assessment on this or any other course of study.

I have read and understood the DCU Academic Integrity and Plagiarism at [https://www4.dcu.ie/sites/default/files/policy/1%20-%20integrity\\_and\\_plagiarism\\_ovpaa\\_v3.pdf](https://www4.dcu.ie/sites/default/files/policy/1%20-%20integrity_and_plagiarism_ovpaa_v3.pdf) and IEEE referencing guidelines found at <https://loop.dcu.ie/mod/url/view.php?id=448779>.

Name: Marc Carné Herrera

Date: 11/04/2016

## Abstract

This project addresses a novel problem that has appeared in the last years. The use of egocentric cameras, devices that take images of what we see, are growing and the main problem of these images is big data (because at the end of the day, we can have a thousand of images, some of them similar and sometimes with a bad quality or low information) and image retrieval (due to the big data, finding a certain moment is very difficult and if we don't avoid that problem, the properties of egocentric images become useless).

This work has two objectives: the first one explores images that have physiological signals associated in order to allow us to add some physiological features for the retrieval instead of only basing the retrieval on visual features as the actual state of the art. For this part we associate interesting images to the images that are memorable, so a correlation between memorability and physiological signals will be found. The second objective is to deal with the egocentric paradigm. Some recent works show that machine learning algorithms that have been trained with human-taken images cannot be extended to egocentric images due to the image construction. Based on MIT (Massachusetts Institute of Technology) previous work I built a visual game that allows me to manually annotate the memorability of the images with a simple user interaction (the user doesn't know that he is annotating images during the game). From this game I have computed memorability scores and I have obtained predicted scores from a convolutional neural network that MIT presents in his work: MemNet. From both results I have compared the results in order to decide if the application of the algorithms on egocentric images is possible.

## Table of Contents

<b>ACKNOWLEDGEMENTS .....</b>	<b>II</b>
<b>DECLARATION .....</b>	<b>III</b>
<b>ABSTRACT .....</b>	<b>IV</b>
<b>CHAPTER 1 - INTRODUCTION .....</b>	<b>1</b>
1.1 OBJECTIVES .....	2
1.1.1 Add physiological signals for retrieval .....	2
1.1.2 Study egocentric memorability and the application of current algorithms .....	2
1.2 EQUIPMENT AND SOFTWARE .....	3
1.2.1 Capture devices .....	3
1.2.1.1 Egocentric camera .....	3
1.2.1.2 Physiological signals .....	3
1.2.2 Software .....	4
1.2.2.1 Matlab .....	4
1.2.2.2 Caffe framework .....	5
1.2.2.3 Graphical processing unit .....	5
1.2.2.4 Web application and Docker image .....	6
1.3 PRIVACY CONSIDERATIONS .....	6
<b>CHAPTER 2 - TECHNICAL BACKGROUND.....</b>	<b>7</b>
2.1 INTRODUCTION .....	7
2.2 MACHINE LEARNING AND DEEP LEARNING .....	7
2.2.1 Convolutional neural network .....	7
2.3 CLUSTERING .....	9
2.4 MEMORABILITY .....	10
2.5 RELEVANT KEYFRAMES .....	11
2.6 SNAP POINTS .....	12
2.7 CONCLUSIONS .....	14
<b>CHAPTER 3 – PHYSIOLOGICAL SIGNALS CORRELATION METHOD .....</b>	<b>15</b>
3.1 INTRODUCTION .....	15
3.2 DATASET .....	16

3.3 METHOD .....	19
3.4 CONCLUSIONS .....	20
<b>CHAPTER 4 - RESULTS CNN AND CORRELATION BETWEEN SIGNALS .....</b>	<b>21</b>
4.1 INTRODUCTION .....	21
4.2 ANALYSING OBTAINED DATA .....	21
4.3 VISUALIZING MEMORABILITY PREDICTION RESULTS .....	23
4.4 SEARCHING A CORRELATION .....	24
4.5 CONCLUSIONS .....	32
<b>CHAPTER 5 – GOING DEEPER WITH EGOCENTRIC IMAGE MEMORABILITY .....</b>	<b>33</b>
5.1 INTRODUCTION .....	33
5.2 MEMORABILITY .....	33
5.3 METHOD .....	33
5.4 EGOCENTRIC VISUAL MEMORY GAME .....	34
5.4.1 JavaScript algorithm .....	35
5.4.2 Image selection for game .....	38
5.4.3 Docker platform .....	39
5.5 CONCLUSIONS .....	41
<b>CHAPTER 6 – VISUAL MEMORY GAME RESULTS .....</b>	<b>42</b>
6.1 INTRODUCTION .....	42
6.2 RESULTS .....	42
6.3 CONCLUSIONS .....	44
<b>CHAPTER 7 – CNN FINE-TUNE .....</b>	<b>45</b>
7.1 INTRODUCTION .....	45
7.2 FINE-TUNE A CONVOLUTIONAL NEURAL NETWORK .....	45
7.3 CNN MODELS EVALUATION .....	46
7.4 EVALUATION RESULTS .....	47
7.5 CONCLUSIONS .....	49
<b>CHAPTER 8 – CONCLUSIONS .....</b>	<b>50</b>
<b>REFERENCES .....</b>	<b>52</b>
<b>GLOSSARY .....</b>	<b>54</b>

## Table of Figures

Figure 1. Convolutional neural network layer structure .....	8
Figure 2. Convolutional neural network design.....	8
Figure 3. Convolutional neural network layers learning .....	9
Figure 4. Visual features for a layer of a convolutional neural network .....	9
Figure 5. Different examples of snap point prediction .....	13
Figure 6. Examples of snap points and non-snap points.....	14
Figure 7. Example of a text file with metadata information that includes physiological signals .....	16
Figure 8. Plot of galvanic skin response at left and heart rate values at right for this new dataset .....	17
Figure 9. Filtered heart rate values.....	18
Figure 10. Filtered galvanic skin response values.....	18
Figure 11. Filtered heart rate and galvanic skin response for a certain interval of samples	18
Figure 12. Memorability score variation for the dataset analyzed .....	21
Figure 13. High and low memorability scores. In red the five hundred highest values and in blue the five hundred lowest values.....	22
Figure 14. Memorability score histogram for the dataset .....	22
Figure 15. Most memorable images at top and less memorable images at bottom .....	23
Figure 16. Plot of heart rate value of the most (in red) and less (in blue) memorable images .....	24
Figure 17. Plot of sorted heart rate values for the most (red) and less (blue) memorable images .....	25
Figure 18. Memorability scores per bin, mean heart rate score per bin and mean galvanic skin response per bin for bin size of 0.05 points .....	26
Figure 19. Memorability scores per bin, mean heart rate score per bin and mean galvanic skin response per bin for bin size of 0.03 points .....	27
Figure 20. Memorability scores per bin, mean heart rate score per bin and mean galvanic skin response per bin for bin size of 0.001 points .....	28

Figure 21. Memorability scores per bin, median heart rate score per bin and median galvanic skin response per bin for bin size of 0.05 points .....	29
Figure 22. Memorability scores per bin, mean heart rate score per bin and mean galvanic skin response per bin for bin size of 0.05 points applying a temporal averaging of scores after bin clustering .....	30
Figure 23. Images with low heart rate value at top and images with high heart rate value at bottom.....	31
Figure 24. Example of fillers and targets .....	38
Figure 25. Dockerfile snapshot.....	40
Figure 26. Predicted memorability scores against manual annotated scores .....	42
Figure 27. Image with less exact prediction (maximum difference) at left and image with most exact prediction (minimum difference) at right.....	43
Table 1. Evaluation results between convolutional neural network models.....	47-48



## Chapter 1 - Introduction

In this chapter a brief introduction of the problem will be presented, highlighting the main objectives of the project and required equipment and software.

An egocentric camera is a device that takes images in first person vision (FPV). There are many brands and models of egocentric cameras, with different image quality, lenses... but the purpose of this devices is the same, capture people's life.

*“What happens when that user's camera is always on, worn at eye-level, and has the potential to capture everything he sees throughout the day?”<sup>1</sup>*

So, a lifelogger is a person that captures all his daily life in order to create a virtual and digital memory. Maite Garolera<sup>2</sup>, from the neuropsychology unit of Terrassa's hospital (located in Spain) said: “The brain is build to forget, we need to forget to survive, because we can't live remembering in each moment all that we have lived”.

The main problem is that our daily life has an average of 12 hours, meaning that there are 720 minutes in one day. This kind of devices uses to take a photo every 30 seconds, so we have two photos per minute. It implies that we will have about 1.000-1.400 images per day in average. This is a big number, also if we think that this amount of images could imply about two gigabytes per day of image storage.

A part from this significant problem, we have to think that having all this image involves to expend a lot of time if we want to find an specific moment or scene. We have to know that these images are objectives, that we are not implicated in the capture of the images, so a lot of these images might be blurred (because a camera wearer is moving quickly), with low information, fast illumination changes...

It might be very useful to deal with the problem of big (and noisy) data and help users to explore his life in a comfortable and efficient way.

## **1.1 Objectives**

In this section of the project I will present two main objectives.

### **1.1.1 Add physiological signals for retrieval**

As is well-known, egocentric images presents some problems due to them objective point of view and his automatic capture. The main problem is big data: taking care that egocentric devices capture images every few seconds, in some ours they have generated a big amount of images, having a big potential but also means a storage problem and a retrieval problem too, because it's not easy to remember at what day something occurs and from a certain moment, there are a lot of images, quite different between them but without the necessity of store all these images.

Some current works deal with the retrieval offering tools and algorithms to find what user want to retrieve. But these works use to base the retrieval in visual features, understanding the content and applying some rules that have been defined previously. But at anytime the algorithm takes cares of what the user that wears the camera feels. These feelings can be catch with some devices, each one with a different level of body invasion.

The goal of part of this project will be to study the correlation between images, them memorability (as I consider an important and relevant image the one that is memorable) and physiological signals associated to the user.

### **1.1.2 Study egocentric memorability and the application of current algorithms**

There are some studies that concludes that current machine learning algorithms, which have been trained with human-taken images don't work well and can not be applied to egocentric images since a clear difference in the image composition exist between human-taken images and egocentric images. To demonstrate this affirmation, I will perform an experiment based on previous studies from MIT, assuming that with human-taken images, a manual annotation of memorability have to be equal with the memorability score predicted with trained algorithms for this purpose.

## **1.2 Equipment and software**

The following sections will introduce us to the technical context: devices used and available resources for that project.

### **1.2.1 Capture devices**

#### **1.2.1.1 Egocentric camera**

An egocentric camera is the main device used in this project and the motivation of this project. With very reduced dimensions this electronic device is capable of capture a full day, every few seconds, generating thousands of images.

There are few companies that sell this kind of cameras, but are growing now and the quality of these devices has been perform since nowadays there are more “lifeloggers” in the world, and offers us to store our life in a digital format.

It's not only the camera, this companies offer software and tools to organize all the images. Narrative for example, is a Sweden company founded in 2012 and now are selling them second model of egocentric camera. The first camera they offer, called Narrative Clip 1, had low resolutions, enough memory to store a whole day and takes images every 30 seconds. The second model outperforms the first one adding video and audio, allowing us to have a nice egocentric experience. This company offer to them customs free software that allows them to upload the images from the egocentric camera to the cloud or storing them in local storage. Before upload the images to the cloud, the application have some techniques that, using accelerometer information (include in a JSON file) rotate and crop the image to present a best point of view. Also the application uploads only the images that have minimum requirements (lightness, blur...).

Another important company is Autographer, that has been pioneering the wearable camera market since the launch of the world's first wearable camera in 2012. Due to user interaction and feedback, this company had modified the model since the first appearance. This device has two times the dimension of Narrative Clip.

#### **1.2.1.2 Physiological signals**

The number of wearable devices that monitors our signals is growing. Few years ago, wristbands that measure our heart rate, calories, quilometers walked... start to appear

and nowadays, with the appearance of *smartwatches* the monitorization of our body is possible.

Maybe this fact is bad known for the population, but all this data are very useful for research community, because we can capture what user feel, vital signs, allowing us to try to offer solutions every time near humans feeling, doing an individual customization.

For this project basically two physiological signals have been explored. The signal we can catch easily is heart rate. This signal corresponds to a value that expresses the number of heartbeats per unit of time, usually per minute. This can report the status of a person in relation with his environment or a situation he was living, and will be interesting to relate it to image relevance.

The other signal I have studied is galvanic skin response (gsr). This signal, also known as electrodermal activity are common used by clinical researchers are can be acquired with non-invasive devices. It is a property of the human body that causes continuous variation in the electrical characteristics of the skin, and some that happens to a user can motivate this variation. This feature of human body can be used for my study.

## **1.2.2 Software**

The project uses some machine learning and deep learning techniques and algorithms that allows us to explore and process the data.

### **1.2.2.1 Matlab**

Matlab, matrix laboratory, is a multi-paradigm numerical computing environment and fourth-generation programming language. A proprietary programming language developed by MathWorks, MATLAB allows matrix manipulations, plotting of functions and data, implementation of algorithms, creation of user interfaces, and interfacing with programs written in other languages, including C, C++, Java, Fortran and Python.

Since all metadata is written in text files and similar, this environment offers us to structure the data and plot values.

In this project, the program will be used to plot some signals, build matrix for understanding data meaning and for apply deep learning techniques. This software allow us to charge libraries easily, and researchers tends to write them algorithms in that language. Particularly, I will use Caffe framework.

#### **1.2.2.2 Caffe framework**

Caffe is a deep learning framework made with expression, speed, and modularity in mind. It is developed by the Berkeley Vision and Learning Centre (BVLC) and by community contributors.

Many deep learning techniques use this framework and can be applied without too much effort using standard functions created.

#### **1.2.2.3 Graphical processing unit**

Each computer has its own CPU (central processing unit), an electronic circuitry within a computer that carries out the instructions of a computer program. This part is essential and enough for normal computing uses, but deep learning techniques require a lot of processing and computation, so only with a GPU this algorithm takes long time of execution and might have a memory problem during computation.

#### **1.2.2.4 Web application and Docker image**

Since I decided to adapt MIT work for my project, a web application was needed and I have not any access to the original visual memory game source code. After some advices, the final implementation uses HTML combined with JavaScript languages to achieve the desired interface and the application was running in a server from Docker image.

Docker containers wrap up a piece of software in a complete file system that contains everything it needs to run: code, runtime, system tools, system libraries – anything you can install on a server. This guarantees that it will always run the same, regardless of the environment it is running in. Containers running on a single machine all share the same operating system kernel so they start instantly and make more efficient use of RAM. Images are constructed from layered file systems so they can share common files, making disk usage and image downloads much more efficient. Docker containers are based on open standards allowing containers to run on all major Linux distributions and Microsoft operating systems with support for every infrastructure. Containers include the application and all of its dependencies, but share the kernel with other containers. They run as an isolated process in user space on the host operating system. They're also not tied to any



Detect snap points in egocentric images with physiological signals – Marc Carné Herrera  
specific infrastructure – Docker containers run on any computer, on any infrastructure and  
in any cloud.

A benefit of that technology is that a docker image (built from a Dockerfile) can run  
on any operating system without having incompatibilities.

### **1.3 Privacy considerations**

Egocentric cameras capture what our eyes look; images are our digital memory. In  
our memory we can remember the face of someone and sometimes his visualization can  
differ from the reality because our brain always try to do a reconstruction of what has seen  
before. With digital memory we can remember a moment with total definition and  
likelihood with the reality. It is a clear advantage for us, but there are some privacy  
problems.

In egocentric images, taken with any intentionality often faces, car plates... appear  
and we can store these images without any permission. This is not illegal, but the problem  
appear if we share this images, so it is critical to submit any research results where this kind  
of images can be shown. So many cares have to be considered when we decide to works  
with a certain database of images.

## **Chapter 2 - Technical Background**

### **2.1 Introduction**

In this section a brief description of the actual state of the art in this field will be reviewed to understand the objective of this work.

Previous works try to deal with the retrieval paradigm, avoiding big data problem. Using visual features and machine learning techniques these works filter all the data and try to classify it.

### **2.2 Machine learning and deep learning**

A part from review previous works related with this project, we have also to review some contents to understand the field of this work.

Machine learning can be defined as sets of techniques automatically learn a model of the relationship between a set of descriptive features (or attributes, the input) and a target feature (or attributes, the output, what we want to predict) from a set of historical examples. We use a machine learning to induce a prediction model from a training dataset.

This algorithms learn from image features, and there are a lot of different features we can select from images and different ways to manage this data in order to construct a model. But for all these techniques, features are handcrafted and we define a set of rules that the algorithm must apply from data, defining arbitrary threshold and operations.

A recent technique called deep learning outperforms all existing techniques. The implementation of deep learning known as convolutional neural network (CNN) have been extensively applied for image recognition, detection and segmentation problems. CNNs are capable of automatically learning complex features that cannot be designed for the programmer of the algorithm, features learned by the own structure, which tries to simulate brain architecture, achieve superior results performance to hand-crafted features.

#### **2.2.1 Convolutional neural network**

Convolutional neural networks are biologically inspired variants, as we know the brain contains a complex arrangement of cells. These cells are sensitive to small sub-

regions of visual field, called receptive field. These cells act as local filters and are connected between them, structured in many layers, as shown in the next figure.

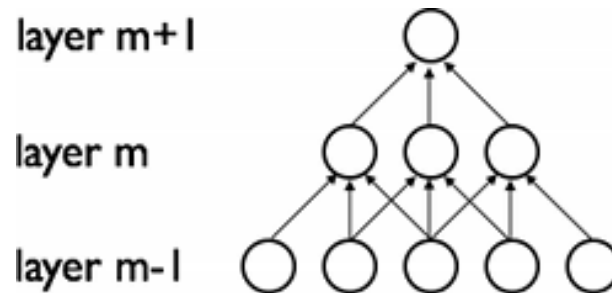


Figure 1. Convolutional neural network layer structure

As we know that each cell is a filter, all cells that belong to a same layer shared the same weights, weights that will be applied to the input. In computer vision the input is an image and each cell corresponds to a convolutional filter that act to a certain part of the image. Many layers are connected between them, where the output of a layer is the input of the next layer, as we can see in the next figure:

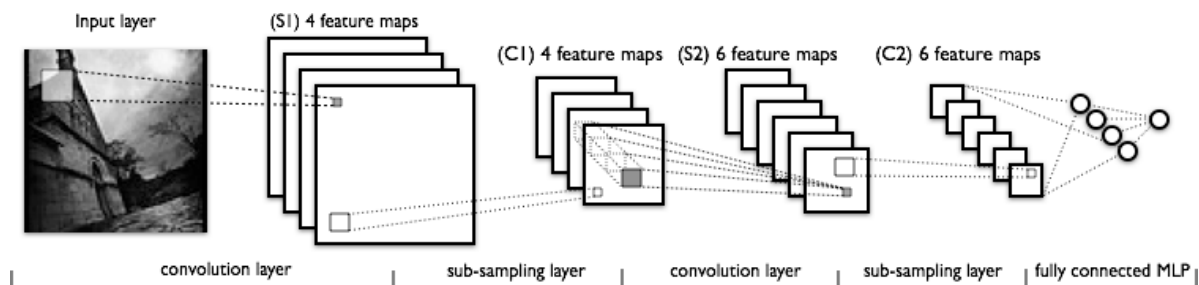
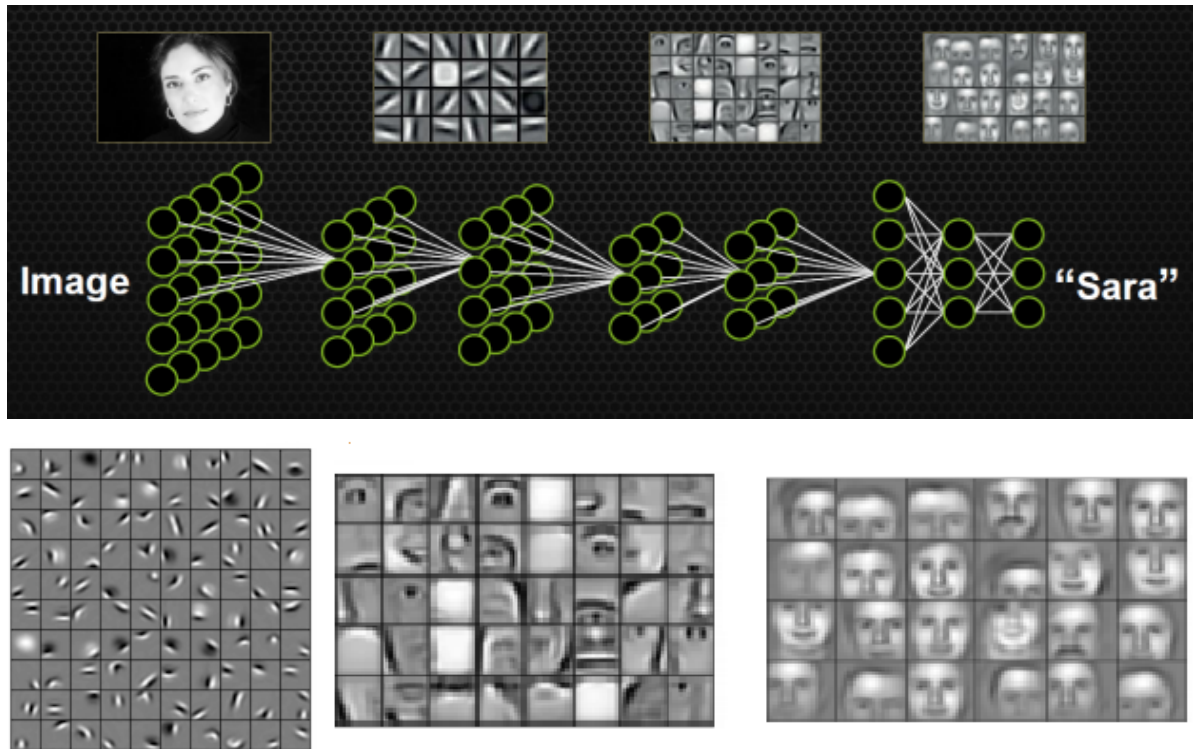


Figure 2. Convolutional neural network design

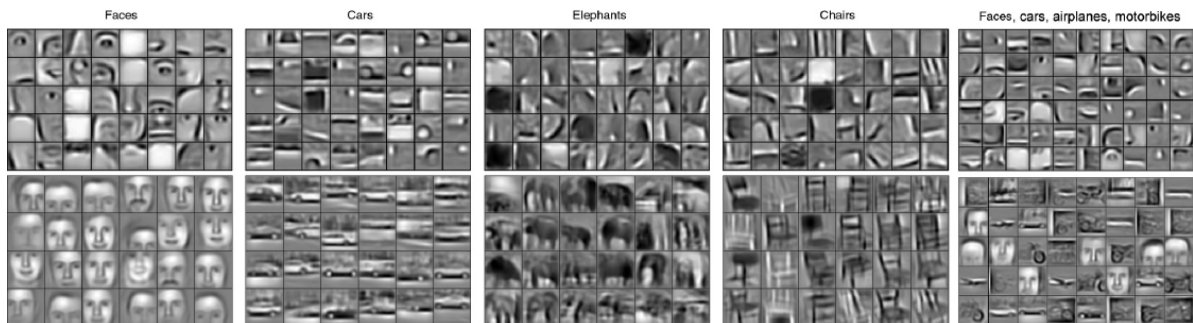
How the network understand an image or process the image is difficult, but it is known that features obtained in different layers of a network works better than human-built features. As a need of understanding what a CNN see, some works have tried to represent the output of each layer, showing that if we go deeper in the network, features are more complex, and more details of image are understand, starting understanding textures, shapes and finally objects.





*Figure 3. Convolutional neural network layers learning*

The previous example show what convolutional network see in case of faces, but in the next figure we can see what it sees for different categories:



*Figure 4. Visual features for a layer of a convolutional neural network*

## 2.3 Clustering

From a day, we can have thousand of images. These images correspond to different moments in a day, and some of them are similar or corresponds to the same event. The idea of these works is to try to segment a whole day in different moments, doing clusters of images, similar between them but different from images that belong to another cluster.

A simple approach to do segmentation is an algorithm called k-means. This algorithm tries to make clusters, calculating some centroids and assigning any point in the features space to a certain cluster. The algorithm start in the raw features space. ‘K’ initial centroids are randomly placed in this space. Each point is assigned to a cluster selecting the nearest centroid to the point. After do a first classification of the points (feature vector) news centroids are computed. This process is repeated until the centroids not change. This algorithm uses visual features and the well choose of features are a key point.

From this baseline algorithm, more elaborated have been created. There are lot of works for clustering. In the University of Barcelona (UB) the Image Group has develop an algorithm that allows us to do a clustering based on visual features and also semantic features, called SR-clustering<sup>1</sup>. They compute semantic features from *Imagga*<sup>2</sup>, an algorithm for image tagging. The results outperform them version without using semantic features.

## 2.4 Memorability

The memorability of an image is a topic that has appeared recently in the research community. These works tries to takes cares of what makes an image memorable and explore visual features and composition of the image.

Decide when an image is memorable or not can be interesting on publicity fields, photography...

In my case, memorability is useful because from a day, we want to pick these images that are relevant and we can think that image is relevant if we can remember it. So, work from MIT define that an image is memorable if when we see an image for a second time, we can detect that it is a repetition, that we have seen this image before. This assumption seems to be coherent. The authors of the research did an experiment with many users. The task for the users was to do a visual memory game. The game consists in an application that shows images during 5 minutes. They define two types of images: targets, are the images that they want to supervise, to see if the users had detect the image on his second repetition. Fillers are images that they put between targets. Targets have only two visualizations, and the fillers can be repeated many times. Some of fillers are considered for vigilance, to ensure

---

<sup>1</sup> Talavera, E., Dimiccoli, M., Bolaños, M., Aghaei, M., & Radeva, P. (2015). “R-clustering

<sup>2</sup> Imagga web for more information: <https://imagga.com>

that a user are paying attention in the game and the results of the game can be used in the research.

After do the game, they compute a memorability score for each image that consist in:

$$memorability = \frac{\#detections}{\#users}$$

For each image they obtained a score value. For example, if the value of the memorability score was 0.6, that means that 60% of users had detected the repetition of that image.

In the first work, they trained a machine learning algorithm that assign an output to an image after compare visual features of test image with features of training images. But with deep learning and his performance over any handcraft descriptor, they decide to create a convolutional neural network which is the fine-tune of Places convolutional neural network, and that new network predicts memorability score for an image. The output of the network is a number between 0 o 1, where 0 is a non-memorable image and 1 is the maximum score for a memorable image.

All this works can be revised on them web site<sup>3</sup>. There it is possible to find all the papers and an on-line web application that computes the memorability score from an image. Also there is some information about the API. The code for the convolutional neural network is publicly available and has been used for this project.

## 2.5 Relevant keyframes

Another interesting research field is the ones that define when an image can be relevant. This topic is being well investigated. There are some different definitions about a relevant image, but a work from Polytechnical University of Catalonia (UPC) takes care of some image feature when define a relevant frame. In that work<sup>4</sup> the authors, from a day

---

<sup>3</sup> <http://memorability.csail.mit.edu>

<sup>4</sup> Semantic Summarization of Egocentric Photo Stream Events. Lidon, A., Bolaños, M., Dimiccoli, M., Radeva, P., Garolera, M. & Giró-i-Nieto, X. (2015) Special Issue on Wearable and Ego-vision Systems for Augmented Experience. In Transactions on Human-Machine Systems Journal

segmentation previously done; they present an approach to select the most relevant frames. To do that they first did a pre-filtering of the images to avoid some computing when after they use convolutional neural networks. To do this pre-filtering, they delete images that are non-informativeness. The prediction of how informativeness an image is, is possible using a trained convolutional neural network. This network assign a score to each image between 0 to 1 and a threshold define which images they have to delete. Images deleted in that step are blurred images, with only one colour...

Some algorithms process images not deleted in the previous step, and the results of these algorithms are combined strategically to obtain a way to compare images and determine which images are the relevant ones. Algorithms used for that task are: faces detection, LSDA (large scale detection through adaptation), saliency maps... Results of this algorithm are combined with different weights.

Despite this approach consider many visual features, the results are not always successful, because sometimes, some moments or segments not contains any relevant frame, and this algorithms can detect erroneously a relevant image due to thresholds or algorithms errors.

## 2.6 Snap points

A recent work<sup>5</sup> start to work with egocentric images, while the others approaches explained in the others sections were designed for human-taken images. In this study, they want to detect snap points. This concept appear in them paper and they define a snap point as an image that look like intentionally taken.

The authors assume that an image with intentionality are these images that we can find in the social networks, images that have been done by the owners, with a camera and with the desired point of view. For the work, they select images from the web as positive examples. Them approach is to assign a value to an egocentric image that defines how this image look like intentionality taken. To do that they defines a kernel, geodesic flow kernel,

---

<sup>5</sup> B. Xiong and K. Grauman. "Detecting Snap Points in Egocentric Video with a Web Photo Prior". In ECCV, 2014. [bibtex]

$$K_{GFK}(\mathbf{x}_i, \mathbf{x}_j) = \langle \mathbf{z}_i^\infty, \mathbf{z}_j^\infty \rangle = \int_0^1 (\phi(t)^T \mathbf{x}_i)^T (\phi(t)^T \mathbf{x}_j) dt,$$

an operation between images features that transform feature space in the desired one. The new space shared by human-taken and egocentric images give us a score that are proportional to images likelihood.

Features selected for this approach are which that allows us to distinguish from egocentric image and human-taken image. These features are: motion blur, line alignment (to study image composition, the main difference between these two kinds of images... and a dimensionality reduction are needed to combine all features into a single vector.

From this subspace they approach detect these snap points placing a novel egocentric image in the feature space, computing all kernel values and selecting the ‘k’ highest values. With these values they do a sum,

$$S(\mathbf{x}^e) = \sum_{j=1}^k K_{GFK}(\mathbf{x}^e, \mathbf{x}_{n_j}^w),$$

and finally, from all egocentric images they select the ones that have the high value. Next figure shows with a red bar the result of that value, where a long bar means more likelihood to human-taken image.



*Figure 5. Different examples of snap point prediction*

Them approach suggests two different applications: the first one is object detection. They demonstrated that current algorithm for object detection, that had been trained with human-taken images cannot be applied to egocentric images because this two kind of images have different composition. So they decide to outperform human-taken existing algorithm adapting them for egocentric images. The approach was to train the algorithm with snap points. A second application is summarization of a set of images, the application that inspire my project. This part of the work suggest that from a previously done segmentation of a whole set, if they pick some snap points from each segments, that would be a good summarization. They compare that assumption with picking random image from each segment and they concluded that them approach obtains best results.

The next figure show us the results of a selection of snap points:



*Figure 6. Examples of snap points and non-snap points*

## 2.7 Conclusions

In this chapter we have seen different works previously done for image summarization. Current memorability works will inspire this project encouraging extending it to egocentric images. Deep learning, a new machine learning techniques, presents a performing over other vision techniques.



## Chapter 3 – Physiological signals correlation method

### 3.1 Introduction

As we have seen previously, current works try to deal with the objectives detailed in this project with visual features, hand-crafted or obtained with deep learning techniques. This works well with human-taken images, but egocentric images are related with a certain persons and as a digital memory, not all images present the same level of memorability or relevance. Applying existing works to this kind of images, we can identify some of the desired images, but may be the detection of this key frames can be done with more precision using physiological signals.

This idea appear knowing that egocentric images are taken with a constant interval of time so that images are objective, because have been taken without any criteria or intentionality. Comparing these images with human-taken images, which present certain intentionality, we can try to involve people to them own egocentric images but without any direct interaction. This idea is to use physiological signals at the same time that visual features and detect that snap points<sup>6</sup>.

But not all physiological signals can be used for this purpose because we need signals that could be easily measures and be less invasive as possible. Taking advantage of technology improvements, I decided to explore what wearable devices can offer to us and study if this kind of features can be easily used for detect snap points.

Actually, there are not much datasets of egocentric images and sets that contain physiological signals and images are very difficult to retrieve. In DCU (Dublin City University) there is a centre for data analytics called Insight Centre. At this centre, where I had develop this project there are experts on egocentric topics, like Dr. Cathal Gurrin, supervisor of the project, who from many years ago wears an egocentric camera with him all the time, having a valuable database. One of his databases consists in a big number of images that have physiological signals. Gently Dr. Gurrin gave his dataset for this project.

---

<sup>6</sup> Image that look like intentionally taken.

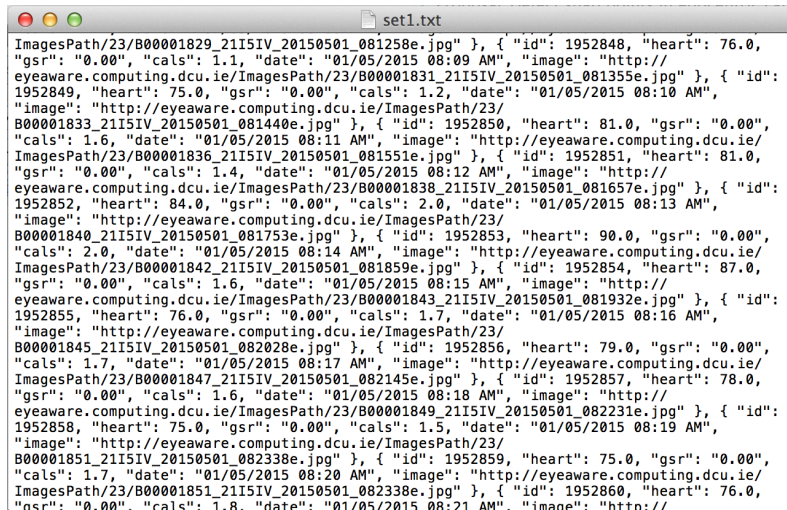
## 3.2 Dataset

The dataset available for this research consist in over than 20 days of images and physiological signals. Images present a lot of activities and actions. All this egocentric images had been taken with Autographer egocentric image.

Dataset belongs to EyeAware a tool for egocentric images developed in the same centre and allows researchers to explore data. EyeAware is a platform technology that supports any Google Glass (or equivalent) developer to create context-aware apps that understand what the user sees and make this searchable historically. This is achieved through developing the EyeAware API, based on extensive DCU know-how, IP and experience. EyeAware accepts images uploaded from apps and generates meaningful contextual information from the images.

The dataset contains a lot of images but restrictions of privacy have to be considered. For that reason the dataset was reviewed and cleaned and finally a total of 16228 images could be used.

With these images a text file was given. This text file contains some information per image: image identification number in EyeAware, heart rate, galvanic skin response, calories, date, image name in the database.



```

set1.txt
ImagesPath/23/B00001829_21I5IV_20150501_081258e.jpg" }, { "id": 1952848, "heart": 76.0,
"gsr": "0.00", "cals": 1.1, "date": "01/05/2015 08:09 AM", "image": "http://
eyeaware.computing.dcu.ie/ImagesPath/23/B00001831_21I5IV_20150501_081355e.jpg" }, { "id":
1952849, "heart": 75.0, "gsr": "0.00", "cals": 1.2, "date": "01/05/2015 08:10 AM",
"image": "http://eyeaware.computing.dcu.ie/ImagesPath/23/
B00001833_21I5IV_20150501_081440e.jpg" }, { "id": 1952850, "heart": 81.0, "gsr": "0.00",
"cals": 1.6, "date": "01/05/2015 08:11 AM", "image": "http://eyeaware.computing.dcu.ie/
ImagesPath/23/B00001836_21I5IV_20150501_081551e.jpg" }, { "id": 1952851, "heart": 81.0,
"gsr": "0.00", "cals": 1.4, "date": "01/05/2015 08:12 AM", "image": "http://
eyeaware.computing.dcu.ie/ImagesPath/23/B00001838_21I5IV_20150501_081657e.jpg" }, { "id":
1952852, "heart": 84.0, "gsr": "0.00", "cals": 2.0, "date": "01/05/2015 08:13 AM",
"image": "http://eyeaware.computing.dcu.ie/ImagesPath/23/
B00001840_21I5IV_20150501_081753e.jpg" }, { "id": 1952853, "heart": 90.0, "gsr": "0.00",
"cals": 2.0, "date": "01/05/2015 08:14 AM", "image": "http://eyeaware.computing.dcu.ie/
ImagesPath/23/B00001842_21I5IV_20150501_081859e.jpg" }, { "id": 1952854, "heart": 87.0,
"gsr": "0.00", "cals": 1.6, "date": "01/05/2015 08:15 AM", "image": "http://
eyeaware.computing.dcu.ie/ImagesPath/23/B00001843_21I5IV_20150501_081932e.jpg" }, { "id":
1952855, "heart": 76.0, "gsr": "0.00", "cals": 1.7, "date": "01/05/2015 08:16 AM",
"image": "http://eyeaware.computing.dcu.ie/ImagesPath/23/
B00001845_21I5IV_20150501_082028e.jpg" }, { "id": 1952856, "heart": 79.0, "gsr": "0.00",
"cals": 1.7, "date": "01/05/2015 08:17 AM", "image": "http://eyeaware.computing.dcu.ie/
ImagesPath/23/B00001847_21I5IV_20150501_082145e.jpg" }, { "id": 1952857, "heart": 78.0,
"gsr": "0.00", "cals": 1.6, "date": "01/05/2015 08:18 AM", "image": "http://
eyeaware.computing.dcu.ie/ImagesPath/23/B00001849_21I5IV_20150501_082231e.jpg" }, { "id":
1952858, "heart": 75.0, "gsr": "0.00", "cals": 1.5, "date": "01/05/2015 08:19 AM",
"image": "http://eyeaware.computing.dcu.ie/ImagesPath/23/
B00001851_21I5IV_20150501_082338e.jpg" }, { "id": 1952859, "heart": 75.0, "gsr": "0.00",
"cals": 1.7, "date": "01/05/2015 08:20 AM", "image": "http://eyeaware.computing.dcu.ie/
ImagesPath/23/B00001851_21I5IV_20150501_082338e.jpg" }, { "id": 1952860, "heart": 76.0,
"gsr": "0.00", "cals": 1.8, "date": "01/05/2015 08:21 AM", "image": "http://

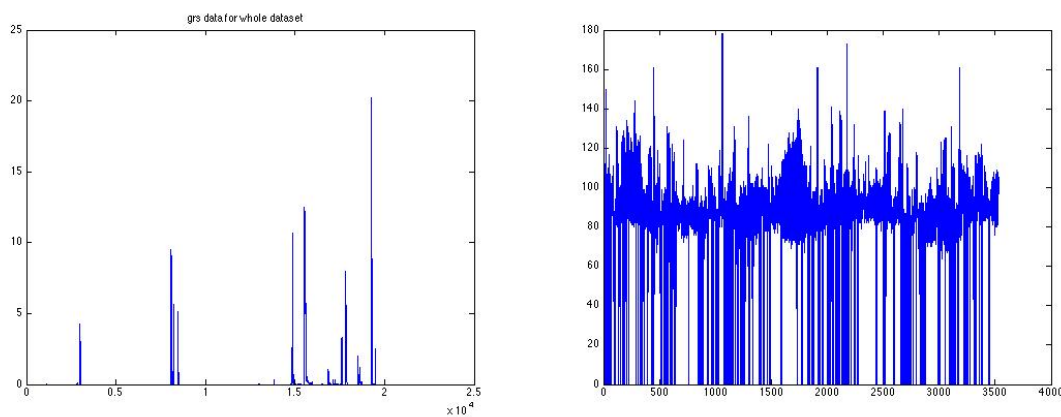
```

Figure 7. Example of a text file with metadata information that includes physiological signals



The first problem that appears was empty image names of many samples, so it means that not all physiological signals had image related, for example because the user only wears the wearable device and not the egocentric camera. For that reason, I had to retrieve and store only images that had them physical signals associated. All this data exploration was do with Matlab, creating my own scripts. After select images that satisfy: have signals associated, belong to image that follows privacy statements... a total of 8474 images remain for the research.

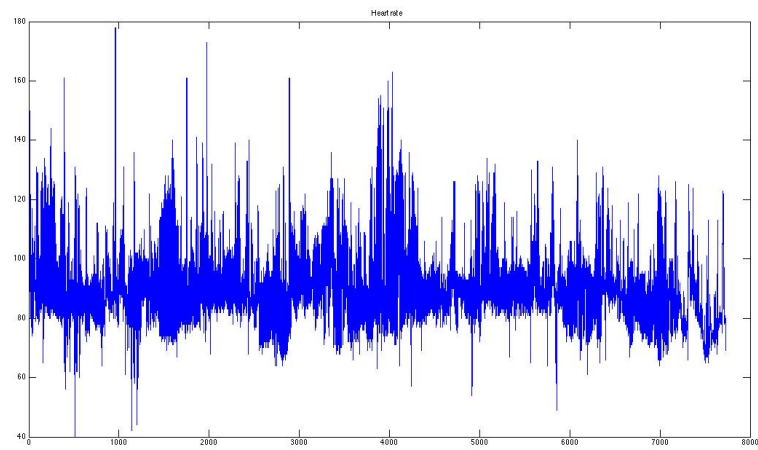
Before start exploring data, I wanted to visualize how the signals look like. Using Matlab I plotted raw signals as we seen below:



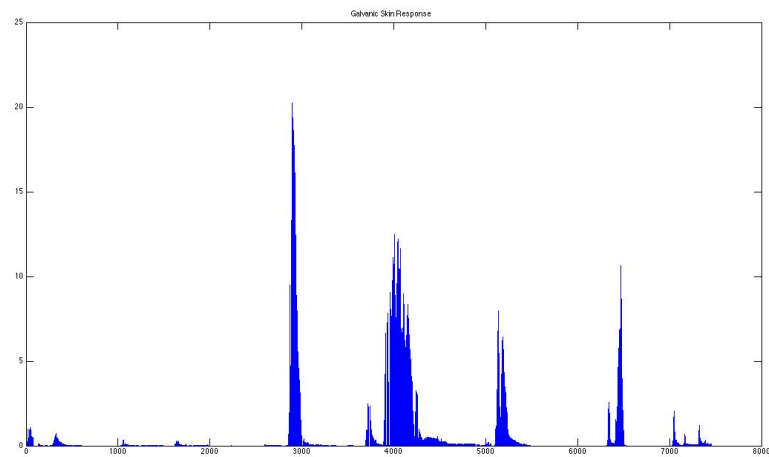
*Figure 8. Plot of galvanic skin response at left and heart rate values at right for this new dataset*

First impression of data is that is noisy and low informative, but it is because data had to be cleaned. As heart rate was very explored by clinical research compared with galvanic skin response, I decided to filter heart rate data. A simple filtering process was applied, deleting all sample that has a heart rate value associated equals to 0, because it means the person is not alive. This fact can be related with sensor error.

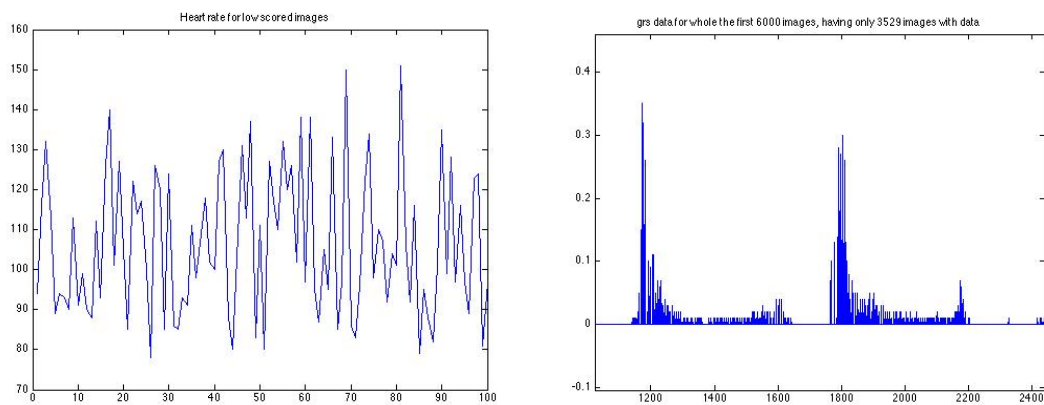
After do that, data presented other structure,



*Figure 9. Filtered heart rate values*



*Figure 10. Filtered galvanic skin response values*



*Figure 11. Filtered heart rate and galvanic skin response for a certain interval of samples*

So I considered I could use this data for this research, a total of 7652 images.

### 3.3 Method

The retrieve of relevant images can be done by different ways. In this project, I define that snap points is an image that has been taken with intentionality and that should be memorable for the owner. So I had related snap point with memorable image. Assuming that I used previous works in memorability for predict memorability score for egocentric images. The code for convolutional neural network such is available<sup>7</sup> for the research. This code run in Caffe framework, available for Matlab, and this implementation is faster with GPU use. In this project, I have used a virtual machine, where I installed Ubuntu and Caffe. Once installed I import Caffe network architecture and Caffe network model. The first file corresponds to a text file that specify how many layers the network have and how many convolutional filters (cells) are in each layer. Also include the size of the input and the output of the network. These two parameters are very important because I had to create a Matlab script to initialize and run the CNN, so input requirements have to be considered.

Matlab script uses functions of Caffe to initialize the CNN and with a simple command we can obtain the result for a simple image, but we have to take care of the next paragraph in the deploy.txt, the file that specify the architecture:

```
name: "MemNet"
input: "data"
input_dim: 10
input_dim: 3
input_dim: 227
input_dim: 227
layers {
  name: "conv1"
  type: CONVOLUTION
  bottom: "data"
```

---

<sup>7</sup> Source code available - <http://memorability.csail.mit.edu/download.html>

```
top: "conv1"  
convolution_param {  
    num_output: 96  
    kernel_size: 11  
    stride: 4  
}  
}
```

We have to see that this implementation requires an input image of 227 height pixels, 227 width pixels and 3 channels. This is image requirements but a forth number, ‘10’, means that we have to give 10 images to network. After review the work from MIT, I realized that they obtained the best results when use 10 transformations per image and did the mean of 10 memorability score to compute the global memorability score for an image. I decided to follow the same criteria and using a common used function I did 10 transformation of each egocentric image I processed with that convolutional neural network.

These transformations are: crop of the centre for specified dimensions, crop of 4 corners and their x-axis flips. The results were saved in a text file where the first field was the name and the other field the score obtained.

After process all images, the follow step is to plot this memorability score against heart rate value and galvanic skin response value in order to find some correlation.

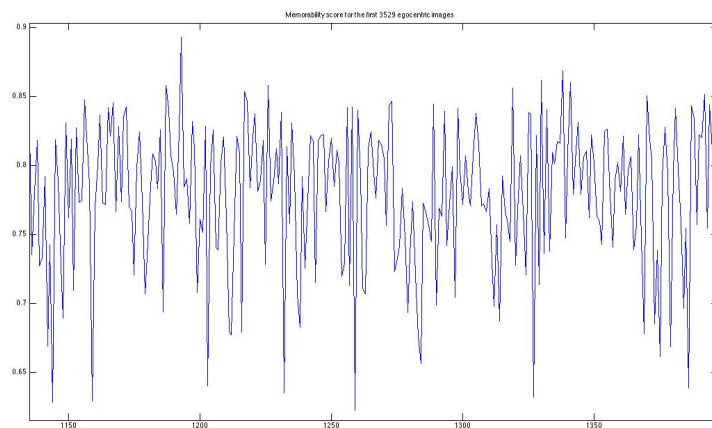
### 3.4 Conclusions

In this chapter a new dataset have been prepared for his data analytics and using current algorithm based on deep learning techniques for memorability score prediction, for all images in the dataset this score has been computed.

## Chapter 4 - Results CNN and correlation between signals

### 4.1 Introduction

After compute memorability score for all dataset images described before, a single value was obtained for each image, as the next figure shown. In the x-axis we see the index of image in the list and in the y-axis the value of memorability score. This plot allows us to see the variability of the memorability score.

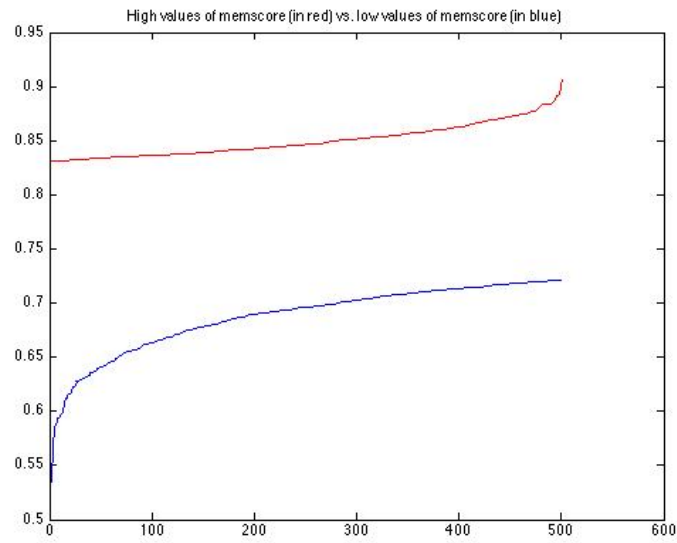


*Figure 12. Memorability score variation for the dataset analysed*

With predicted memorability score results and physiological signals a matrix was created, where each row corresponds to a sample of the dataset and each column to: image name, heart rate value, galvanic skin response value, memorability score and identification of each image in this experiment. The identification number is very important because was assigned in temporal order and when a sort rows was done to the matrix, I could identify well the images.

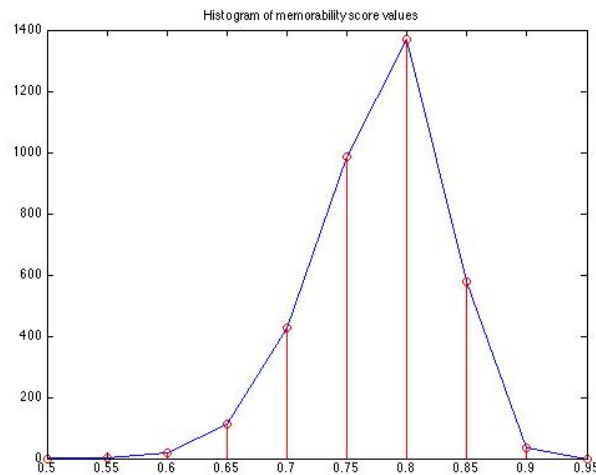
### 4.2 Analysing obtained data

A first step was to sort the matrix by memorability score. Doing that a new matrix was obtained. I started to explore the most memorable and less memorable images predicted by the convolutional neural network. I selected the 500 images of both cases and plotted the value of memorability score, to see if most and less memorable images had different values.



*Figure 13. High and low memorability scores. In red the five hundred highest values and in blue the five hundred lowest values*

From the graphic we can look that the values differs but we can see also that there are a lot of images that have more or less the same values of memorability score, and these images can be very different between them. For that reason I compute the memorability histogram, to have a general perspective of the values,

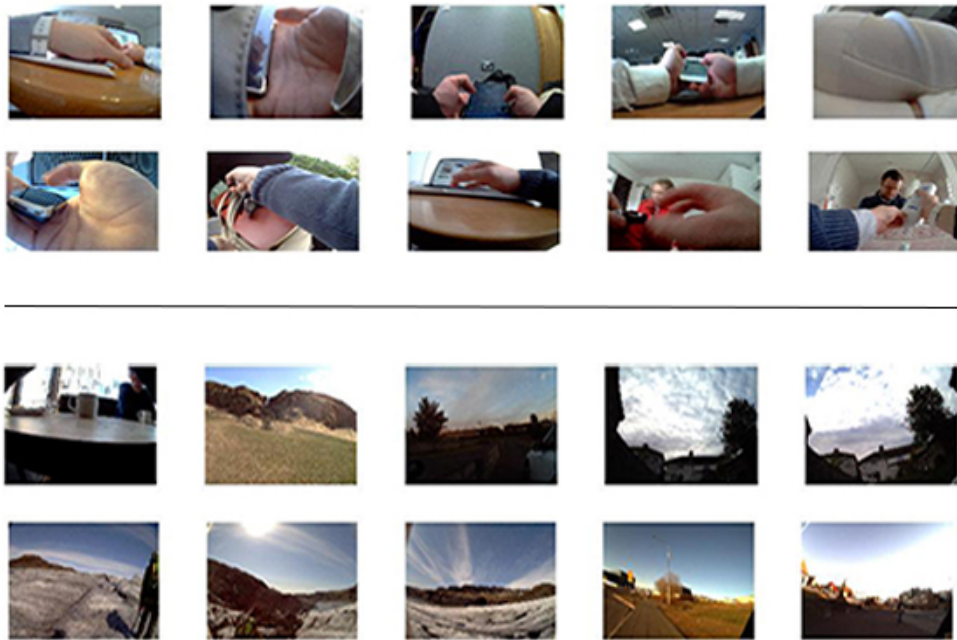


*Figure 14. Memorability score histogram for the dataset*

and I could see a lot of images had a memorability score value near 0.8. This means 80 people of 100 can remember this image in a second repetition, as authors of memorability works (from where I have based this study) said. This fact made me think that the algorithm

### 4.3 Visualizing memorability prediction results

Before continue with the research a visualization of images with high and low information was needed.



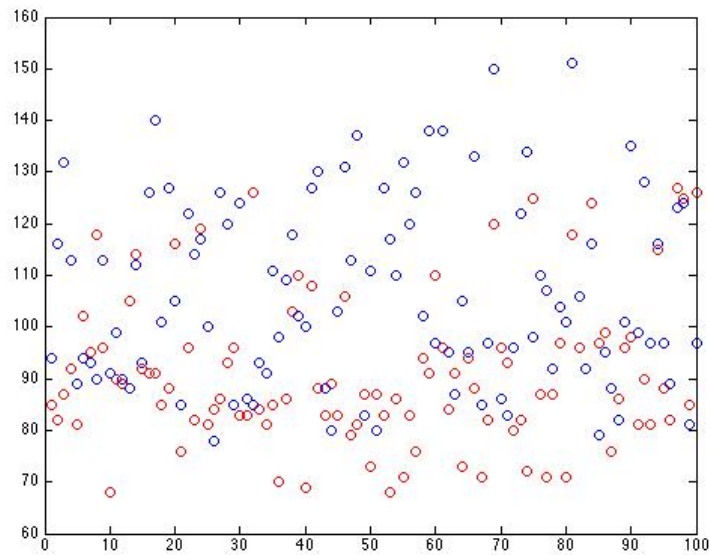
*Figure 15. Most memorable images at top and less memorable images at bottom*

Last image shows us ten images with high memorability score at top and ten images with less memorability score at bottom. This images had been taken by the next method: the most memorable image was taken and then the next memorable image in the sorted list with a slope (image not taken) of ten images was taken, and this process was repeated until had ten images. The same method was done for less memorable images. As we can observe in the images, most memorable images are these images that contains objects, persons, centred scenes... despite images with low memorability score are these images that contains landscapes, outdoor scenes, etc. This results are the same that authors of memorability study obtains, so I assumed that prediction of network was valid. Although many images with intermediate memorability score was analysed sampling all dataset and a general pattern was identified.

From that results next sections of project were inspired, being a good idea to outperform CNN for egocentric images.

## 4.4 Searching a correlation

Having these predicted memorability scores and physiological signals I searched a pattern in heart rate and galvanic skin response in relation to memorability score. A plot of memorability score against heart rate for high and low memorable images in the same graphic shows us that for these two subsets of image, predicted values are mixed.

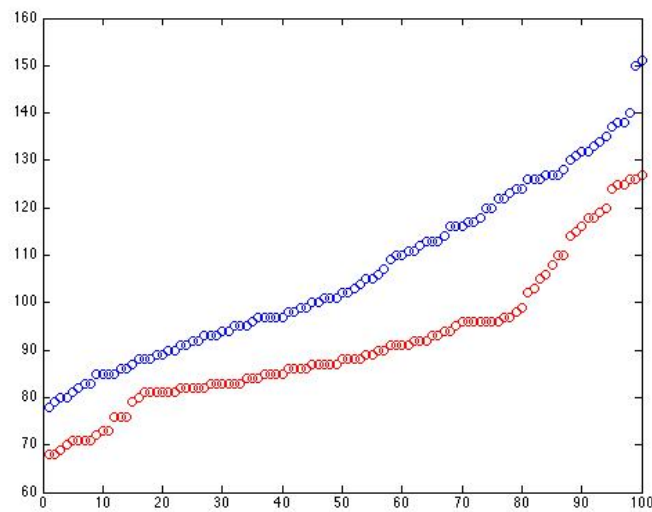


*Figure 16. Plot of heart rate value of the most (in red) and less (in blue) memorable images*

Red dots are the most memorable images and blue dots are less memorable images. For example, if we want to decide if an image with a heart rate of 100, from this plot we cannot have an answer. From that plot we can think that as prediction might have an error and sensors for data acquisition can introduce noise, try to find a correlation image-by-image has no sense. From here, the idea of organizes data into groups started.

Following previous steps I did the same graphic as before but considering that all most memorable images have the same memorability score, without establish an order between them, doing the same assumption for low memorable images, two new matrix were constructed for high and low memorable images and the value of heart rate was sorted and plotted.





*Figure 17. Plot of sorted heart rate values for the most (red) and less (blue) memorable images*

In this figure we see that a pattern start to appear, and knowing that red values are high memorable images and blue values are low memorable images the graphic shows how the range of values for high and low values are similar but with an offset. That fact encouraged me to definitely group samples. To do that, data had been separated into bins. A bin is a group of images that have same value or are similar in something. In my case I decided to do the bins from memorability score, doing quantification of the predicted value.

To divide data into groups I defined a bin size, defining how to group the data. The first division was into 8 groups or bins, dividing the range of memorability score between 0.5 and 1 each 0.05. After clustering was done, each bin had a different number of samples. It has sense because in memorability score histogram we saw a lot of images have the same predicted value.

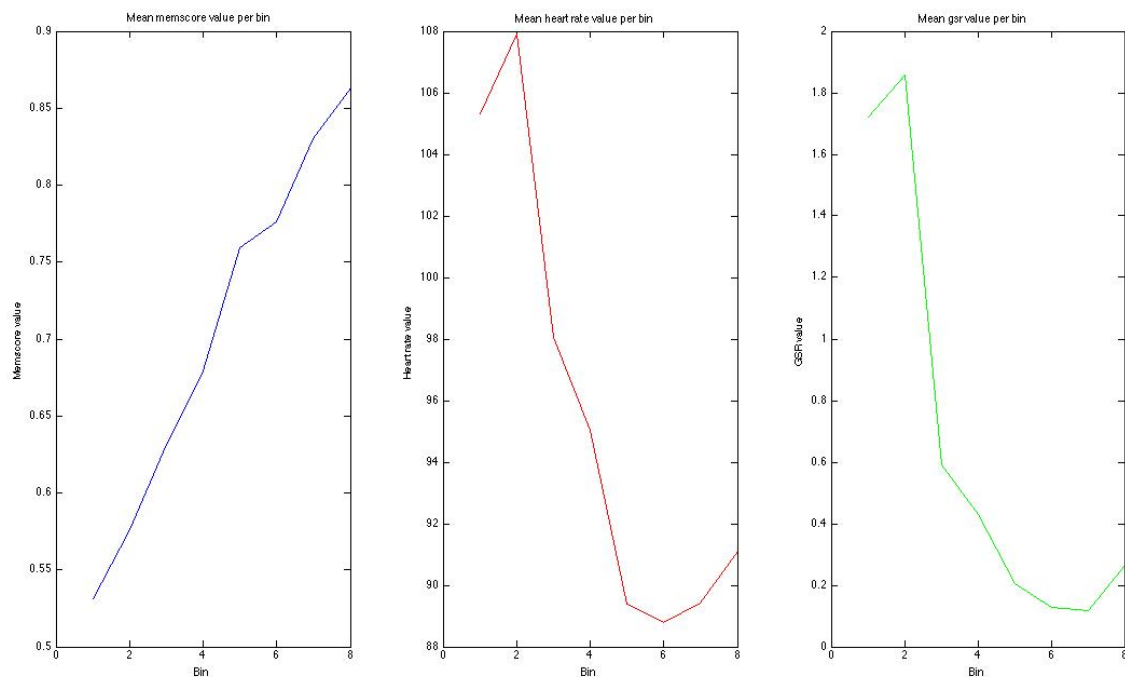
As I mentioned before, I clean the data deleting images with heart rate associated equal to 0, but it's possible that images with very high value of heart rate are noise. So a second filtering of data was needed, but this time I applied a different filtering technique. In image processing, a filter is a convolutional filter that applied to an image modifies the content. A mean image filter is a convolutional filter of  $N \times N$  dimensions with the same value in each position. This is the same that do the mean of a matrix. In my case I inspired the filtering of clustered data by applying the mean to the bin. Doing that we can “delete” noise having that correct values (that I supposed are most of samples) have more weight.

The mean value of a bin is not the only filtering technique I used in this project. There is another filter called median filter. This is not a linear filter; there is any operation with the values of the bin. This filter consists in sorting the values of a bin and selects the central value. With this method we can discard noisy values and take that value that is common in data.

For each bin the mean value and the median value was computed. This two values obtained was very similar in some cases. The use of these two values is motivated due to find a pattern valid in these two cases.

To have a general point of view of cluster data, the follow graphic was plotted. In this graphic there are 3 plots.

First mean values per bin had been explored. The first graphic corresponds to the mean value of memorability score of each bin. The second graphic is related with the mean value of heart rate for each bin and the same is plotted in the third graphic, in this case with galvanic skin response.

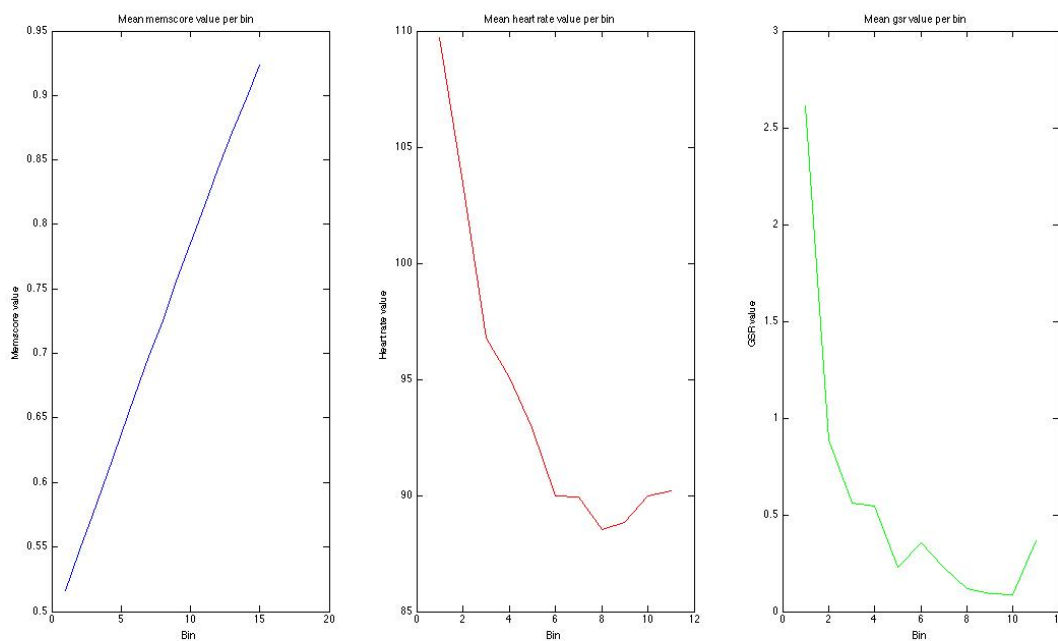


*Figure 18. Memorability scores per bin, mean heart rate score per bin and mean galvanic skin response per bin for bin size of 0.05 points*

First of all, we can directly observe that memorability score grows, as we expect because each bin is a group of images that has a high value of memorability score each time. With only eight bins we data seems to follow a pattern. Results seem to tell that when

memorability score grows, heart rate and galvanic skin response tend to decrease. Despite that observation, our dataset have about 8.000 images, so eight bins cannot be significant to split data. We have to remember that eight bins means a slope of 0.05 when the bins were done and that all bins have not the same number of samples.

To go deeper, high number of bins was done. To do that I decided to decrease the size of each bin, assuming that the difference of memorability score between the samples of a bin are very small. Next plot shows what happen when a bin size was decreased to 0.03. That means have eleven bins.

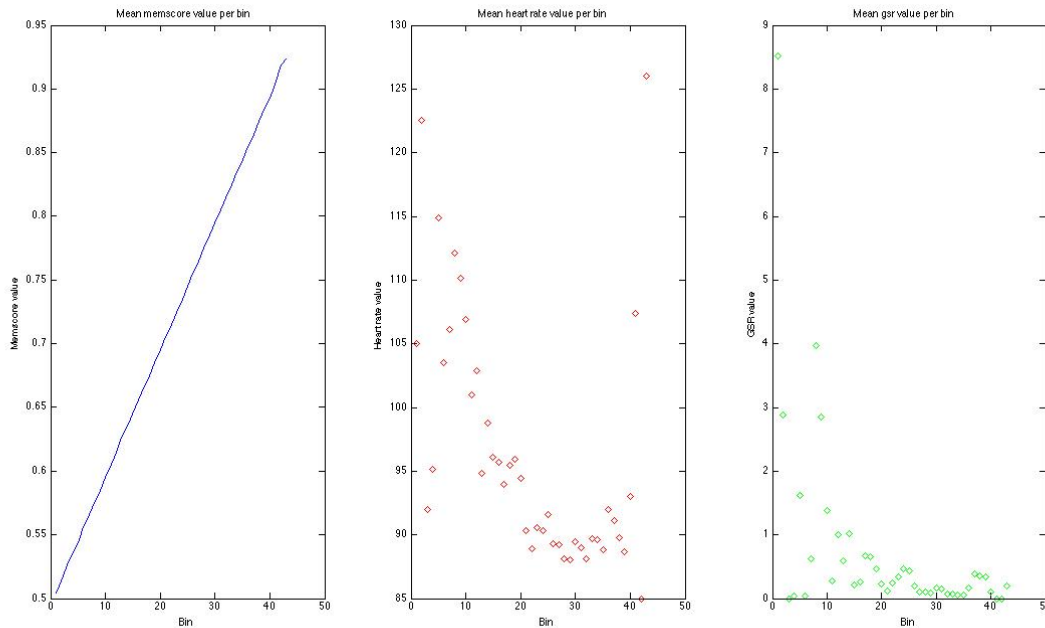


*Figure 19. Memorability scores per bin, mean heart rate score per bin and mean galvanic skin response per bin for bin size of 0.03 points*

As results show, the same pattern had retrieved, so has sense to think that memorability score are related with heart rate and galvanic skin response, where this two physiological signals seem to follow the same pattern for memorability score. Despite that observation I have to advise that this results have been obtained filtering data, not considering sample-by-sample, so it seems to be clear that we cannot work directly with these two signals values, so may be a process of original signal must be needed to work with a single sample.

To contrast these results and be sure that the results are valid, a fine exploration was done. It implies to reduce the bin size to 0.001, a very low value and see if data keep the

same pattern. The results of that last assumption were plotted with dots to offer clear values because each cluster or bin has only a single value, that in this case is the mean value.



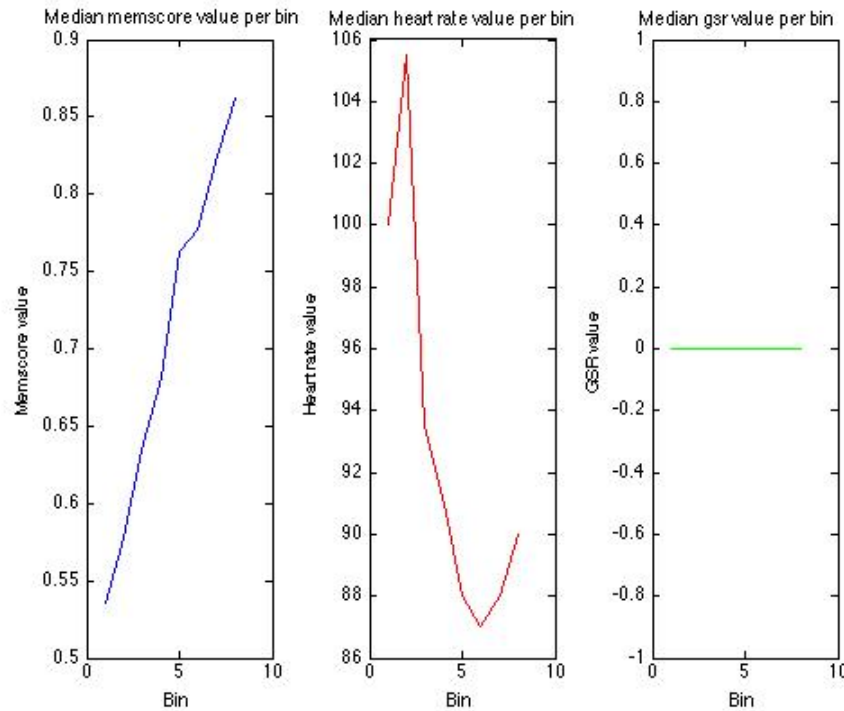
*Figure 20. Memorability scores per bin, mean heart rate score per bin and mean galvanic skin response per bin for bin size of 0.001 points*

By reducing the range of a bin, the results seem to be sparse and unclear, but in general we can conclude that with an increase of memorability score value (in average), heart rate and galvanic skin response tend to decrease and this result cannot be applied to each single sample because with only one value of these signals for images are not enough. At time of data acquisition a processing of the value might has to be done, to achieve a mean value of signals at each time, as I had done for this analysis of data.

We can also see that in the last plot, the first graphic is a line, so that means that the mean value of memorability score for each bin is the same value of the mean value of the range or the domain of each bin, meaning that data are good distributed along the bin.

A clear pattern has appeared when I filtered data with a mean filter for each bin and with different bin size the results are similar, but to ensure these conclusions an observation of the results when a median filter was applied is a good idea.

Next graphic shows results for median filter application to data cleaning,



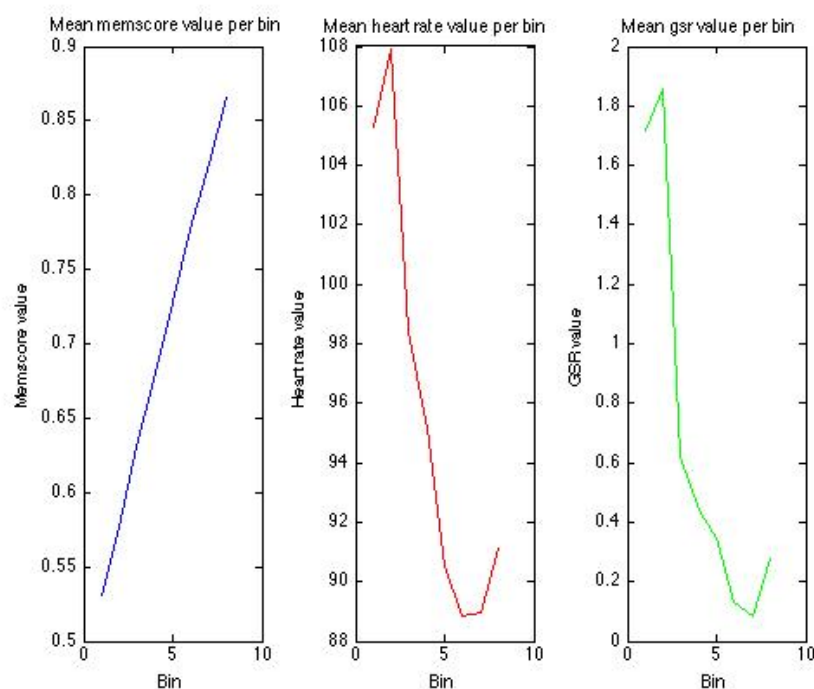
*Figure 21. Memorability scores per bin, median heart rate score per bin and median galvanic skin response per bin for bin size of 0.05 points*

One more time, the results obtained are the same, despite taking median value galvanic skin response in all bins the result is equal to 0, meaning that the central value of this signal is 0. Those made me think that maybe this signal is not appropriate for being related with memorability score.

As mentioned before a good idea when data have to be acquired a good idea is to compute a mean value in the temporal domain. In this work data is acquired so a simulation of do that in a real time was done with the following approach: from a matrix with all clean samples without sort rows, a temporal feature is kept, so we can suppose that we are at data acquisition moment, having a difference between samples of about thirty seconds. In real situation for data acquisition this interval of time has to be small, but for my assumption is enough. With Matlab an average of the values for each sample was done. So for each sample I took his real physiological signals and the value of some neighbours and compute the mean value, assigning that value to the sample. Different number of neighbours was taken, obtaining different mean value in each case. As high is the number of neighbours a low pass filtering is applied, so assuming that sensors can be very noisy, I applied a very

restrictive low pass filter. It means that the number of neighbours had to be high. After test with this value, I decided to take a value of 50 neighbours per sample. This number offers us a good influence of neighbours but keeping the value the original sample. This means to compute a value with 25 minutes data for only one sample (because data are separated by 30 seconds approximately). In real data acquisition separation between samples must be small to compute a value for a small range of time.

With that approach, computing a temporal value of the heart value and galvanic skin response, I follow the same proceed that before, grouping data in bins and compute mean and median value.



*Figure 22. Memorability scores per bin, mean heart rate score per bin and mean galvanic skin response per bin for bin size of 0.05 points applying a temporal averaging of scores after bin clustering*

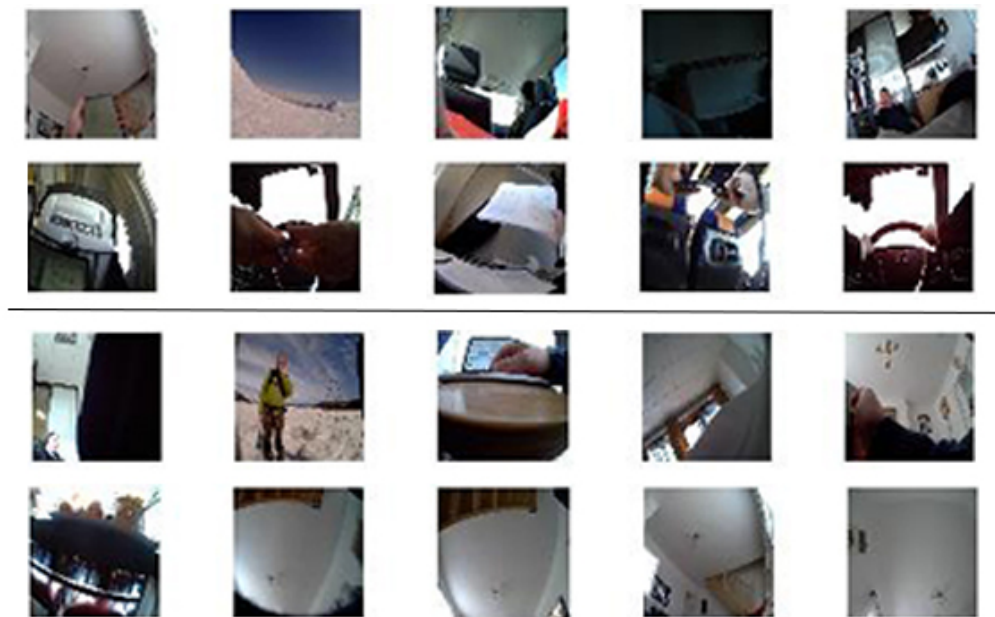
Results are similar taking mean value or median. In last graphic value taken was from a mean filter, low pass filter, and the results are quite similar as the results that not take care the temporal values of signals. The conclusions for this approach are the same of the other procedure. For that approach we can conclude that a temporal filtering at data acquisition moment is a good option and would be implemented in future work.

Median value or mean value of heart rate can be computed without having significant changes but with galvanic skin response we have to be work with caution. For an

easy acquisition and success results in general heart rate might be used instead of galvanic skin response.

Clearly heart rate is related with memorability score but for a direct relation a process of this signal value have to be done in the temporal domain.

To end with this part, a plot of the images with high heart rate and low heart rate has been done to see if the memorability rule can be definitely applied. If images with high heart rate have the same visual composition of low memorable images and images with low heart rate have the same visual composition of high memorable images, a direct use of signals for a single sample can be used.



*Figure 23. Images with low heart rate value at top and images with high heart rate value at bottom*

In the previous image we can show at top images with low heart rate value, images that with my data exploration conclusion are more memorable; in the bottom images with high heart rate are shown, and this might imply that these images have a low value of memorability score, being less memorable. As we can observe in general top images contains objects, people, presents well-composition, etc. as memorable images look like, but despite general issue landscapes or outdoor scenes appear, meaning that this assumption introduce some noise that can be avoided doing temporal averaging for physiological signals in data acquisition. With bottom images the same happen: a lot of them follow low

memorability rules as being landscapes, low informativeness scenes, etc. but many images that should be memorable appear in that group.

## 4.5 Conclusions

From all results obtained a pattern has been identified, relating memorability image prediction to physiological signals. Physiological signals (that corresponds to a single value for each image) cannot be used directly for memorability correlation, so a temporal processing is needed. General pattern in data shows that when memorability score increase physiological signals as heart rate or galvanic skin response tends to decrease. Finally, a lot of values equal to zero for galvanic skin response was detecting, suggesting that relate this signal with memorability in real time can be difficult and uncertain.



## **Chapter 5 – Going deeper with egocentric image memorability**

### **5.1 Introduction**

In the previous chapter after use MemNet, a convolutions neural network for memorability prediction, and visualize the results the use of this algorithm that has been trained with human-taken images is not a bad idea but maybe a networks adaptation for egocentric images can outperform this network.

To deal with a conclusion of that fact a manual annotation of egocentric images memorability might be used. If networks prediction is quite similar to manual human annotation, an adaptation of the network will not needed.

This part of the project was inspired from the work of MIT mentioned before (in previous sections).

### **5.2 Memorability**

The main assumption of this section is that an image is memorable is a user can detect his repetition when he see the image for a second time. As demonstrated in MIT research, not all the images are equal memorable and that images that are more memorable have a similar structure or composition, allowing us to define some visual rules for estimate image memorability. With that assumption and manual annotation of images the convolutional neural network that has been used in this work was trained.

### **5.3 Method**

After consider all previous works and the meaning of memorability and manual annotation an affirmation can be done: the score for images that authors used to train CNN have to be equal from manual annotation and algorithm prediction, so if the CNN are well trained for egocentric images, the scores that I had obtained in previous section must be the same (or very similar) to manual annotation.

The approach for that section will be compute manual annotation and compare it with predicted memorability score.

To do that we have to remember how to compute manual memorability:

$$memorability = \frac{\#detections}{\#users}$$

As we can see in the equation, memorability score value computed with human interaction not consist only in a single user interaction, is an average of multiple users. In the previous MIT work, they mentioned having 78 users for image on average.

The method for obtain manual memorability annotation consist in a visual memory game. MIT team presented a game in them research work, but the code for that game is not publicly available, so I had to create a game similar to the original work to allow me to compare results.

After play myself with the original game<sup>8</sup>, it catch the main features that the game had to have. Images for the game are divided in two groups: the first one is “targets”, images that we want to analyse. These images are the ones that we want to detect user recognition when this image is shown for second time. The second group is “fillers”, images that are places between targets repetitions and we don’t want to control, only few of these images, called “vigilance fillers” will be analysed to see if user are paying attention to the game and we can use these results for this study. Despite this clear structure of data, is difficult to know how the sequence of image has been done. In the next section I will explain how I created my own game, and how the algorithm works, for if any future reproduction or use is needed.

## 5.4 Egocentric visual memory game

This egocentric visual memory game was inspired in MIT research and consist in a web application for manually annotate image memorability. The implementation is a HTML source code that a part from text contains a command to show an image. Source image are not constant, changes every 1.2 seconds controlled by JavaScript algorithm. So, the main complexity of the game is in this script.

---

<sup>8</sup> Visual memory game from MIT research group - <http://facemem.csail.mit.edu>

### 5.4.1 JavaScript algorithm

At the beginning of the algorithm there are two text files that are read and stored in two different variables. These text files contain the name of images, one file for targets and other for fillers. As part from charge these names to a variable, these new variables have more fields. This variable consists in a table that at the end of the game will be stored in a local storage.

The structure of this table is the following: a first field indicates the name of the image including the format, the others fields are numerical and allow us to have a control of these images, in particular, target images. These numerical fields are:

- Second field: is a binary value, 0 means that this image have not been shown during the game, that user have not seen; and 1 means that this image have appeared at least one time, but not ensure the image have been repeated.
- Third field: is a positive integer number that indicates the position in where this image has appeared. This number is the position in the whole sequence including fillers images.
- Forth field: is a positive integer number that indicates the position in where this image has been repeated. It means that the image appeared before and that is the first repetition. This value helps us to not to show this image another time, because we have to ensure targets appeared only two times (this is not mandatory for fillers images).
- Fifth field: is a binary value that indicates if the user has detected the repetition of this image. A JavaScript function takes care of that fact.

This table has also a final line that is different from the rest. This line consists in five more values referred to vigilance fillers:

- First value: number of vigilance fillers that have been shown during the game, to control user attention.
- Second value: number of vigilance fillers that have been detected for the user.
- Third value: number of vigilance fillers that have not been detected for the user.
- Forth value: relation between detected vigilance fillers against non-detected vigilance fillers.

- Fifth value: binary value that define if we can use these results for this research (1) or we have to discard them (0).

The criteria for evaluate user attention is that the relation between vigilance fillers detected and non-detected must be equal or high than 0.5. This relation is computed as:

$$AC \text{ (Attention control)} = \frac{\text{correct vigilance fillers detections}}{\text{vigilance fillers non - detected}}$$

From this value, utility value is defined as:

$$Utility = 0, \text{ if } AC < 0.5$$

$$Utility = 1, \text{ if } AC \geq 0.5$$

Once this variables have been created a *\$interval* function, from JavaScript library called *Jquery* start a cycling repetition. This repetition is every 1.2 second and can be stopped by two ways: there are a “stop” button and after press it, this repetition stops immediately; the other option is the normal use of the game, after 9-10 minutes the application stop automatically. This is equivalent to 440 cycles of the algorithm. This not means that 440 images appear during the game because images are separated by blank image. This image consists in a white images with a little black square in the middle with the purpose of focus user gaze in this part of image and makes that user pay more attention in where the images are shown.

For the game, images have been resized to 300x300 pixels images with Matlab script. With this dimensions we ensure user can have a global view of image and also that object not appear enough big. Blank image have the same dimensions and the black square is 20x20 pixels, allowing to user to focus the gaze at this point.

The first step of the algorithm is to compute a random number between 0 and 3 and shows this amount of fillers selecting them randomly from fillers list. After this visualization a target image are shown and the control table are update, changing the values of the second and third fields, indicating the image has appeared and in what position. After that, the algorithm computes a new random number between 0 and 3 for fillers visualization. This proceed is repeated 220 times and between each image visualization a blank image are

shown. So the mechanism is: show N (random) fillers, show random target and repeat. But the algorithm is more complex because the first fillers of the N fillers array created in every iteration of the algorithm is the vigilance filler. This image have to be repeated in a show interval of images because is for user attention control, so that image will be shown in the next iteration of the fillers, as the first filler. Also target selection is not trivial, because in every iteration we have to compute a random number between 0 and the length of the target list, in this case: fifty. The algorithms look if the image correspondent to that list position has been shown in the game. If the image has not been shown, it has to be shown. If the algorithm looks that this image has been shown and since the first repetition number of images seen are between 8 and 40 it show this image. If the number of frames shown is not in this interval the algorithm computes a new random value and do the same inspection of the image. There are 10 trials for select a target for not to collapse the algorithm. For the new random value could happen that it have not appeared before. But a critical situation may occur: a target image has been shown 40 images ago and the algorithm has not selected it yet. So, a condition in the algorithm avoid that possibility and when the algorithm has to select a new target, it first explores all positions in the list (all targets) for retrieve if any target presents this problem.

After these 440 iterations of the algorithm (including iterations for blank images) the algorithm stops automatically and stores locally the results in a text file. This result is the target table.

The most important part of this game is the user, who annotates image memorability score. The task for the users is to press button “start” to run the algorithm in the web application and if they think that an image has appeared for a second time they have to press key “d”.

Some important thinks to take in mind that that web application has been designed to be used in Google Chrome browser and that the duration of the game is between 9 and 10 minutes. In comparison with the original game that takes 5 minutes to complete, this game allows us to annotate a high number of images in each hit (each time that the game is done). It's important that users have not seen these images before because it can cause some confusion.

### 5.4.2 Image selection for the game

Egocentric images are different from human-taken images and present some problems. The main problem of these images is that neighbour images can be very similar, so if there is a target or filler that is similar to another target user can detect a repetition erroneously. So, a first step is to select these two groups of images with some criteria.

For targets we need a total amount of fifty, so what I did is a uniform sampling of dataset in order to take images well temporal separated. By this way I ensure that targets are enough different between them. Few targets can be similar sometimes but contains some details that allow us to find a difference between them.

For fillers, to ensure that are very different from targets I used another dataset of images. This dataset is available publicly: UT Egocentric that was created for a research by Grauman et al. to detect snap points. That dataset consist in four videos of about four hours each one. What I did is to sample this video extracting frames with a fixed interval of time, as were captured by an egocentric camera. The quality of these frames is more or less similar than dataset I used for this work. By this way we can ensure that any filler are similar to targets, but these images had to have the same point of view than targets to make that users cannot distinguish fillers and targets. The number of fillers is higher than targets to achieve a non-repetition of fillers and focus user attention to targets.

In the next figure we can show some samples of targets and fillers, to have an idea of how these images look like.



*Figure 24. Example of fillers and targets*

### 5.4.3 Docker platform

As the use of these images was allowed, memorability game was shared in social networks for data acquisition, as we need to have many users to compute memorability score.

To share the game a server was required. In this case, computation services in UPC (Polytechnic University of Catalonia) deal with that. A server was set up with the game configuration to allow an online use of the game.

A common implementation of this kind of servers consists in run the algorithm in server machine and starts a server process. A problem may occur if there are some incompatibilities with software versions. To avoid that problem, a new way to set up a server was used. Based on the idea of a virtual machine, a virtual computer that run on the machine but with his own configuration and with independence, a new concept has appeared: *docker*<sup>9</sup>.

A “*docker*” is a container where an operative system is placed and all files and software required for the use are added. With that system I created a *docker* image, virtual configuration of the required machine. With *docker* tools this image was run.

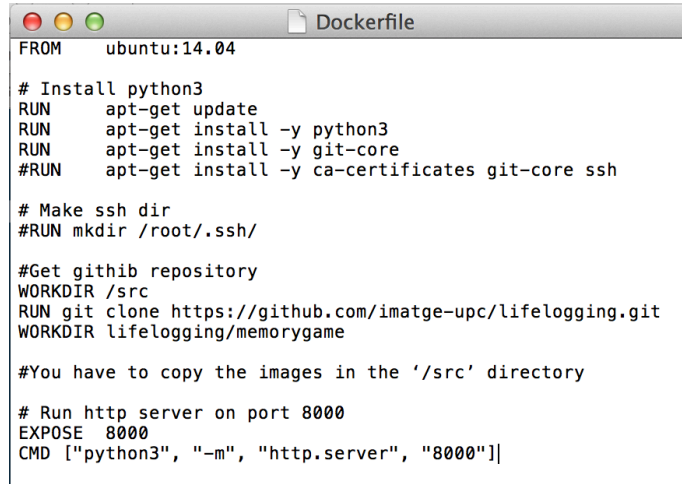
The installation of docker environment starts in his website, downloading two different software:

- Kitematik: is a software that allows us to download docker images that other users have upload to dockerhub. We can use these docker images for free and give us some examples. It is useful for example to see some *Dockerfile* examples.
- Docker Quick Start Terminal: application that opens a system terminal and start docker virtual machine in order to use docker commands.

After installation, the first step is to create a *Dockerfile* that contains the structure of the *docker* and his functions. In the next figure we can see an example of Dockerfile for this work.

---

<sup>9</sup> Docker web site - <https://www.docker.com>



```
FROM    ubuntu:14.04

# Install python3
RUN     apt-get update
RUN     apt-get install -y python3
RUN     apt-get install -y git-core
#RUN    apt-get install -y ca-certificates git-core ssh

# Make ssh dir
#RUN    mkdir /root/.ssh/

#Get github repository
WORKDIR /src
RUN     git clone https://github.com/imatge-upc/lifelogging.git
WORKDIR lifelogging/memorygame

#You have to copy the images in the '/src' directory

# Run http server on port 8000
EXPOSE  8000
CMD     ["python3", "-m", "http.server", "8000"]
```

Figure 25. Dockerfile snapshot

Commands used to create the image of the *docker* are:

- FROM: indicates the operative system that virtual image has to be.
- RUN: in the bash of the virtual machine, this command starts the process that follows. In a *Dockerfile* can be many of these issues. It is to set up the virtual machine as install packages, change directories, etc.
- WORKDIR: docker command for change the work directory, similar to ‘cd’ command from Linux.
- EXPOSE: when a server function is required for *docker image*, we have to specify the port where the process that we want to execute as server will use in the virtual machine.
- CMD: command to execute the main action of the docker virtual machine. Only one of this command can be used in the Dockerfile. Comma split must be used to indicate to the operative system what to do.
- ADD: command to add some files from physical machine to virtual machine.

As in others programming languages a comment of a line is done adding ‘#’ at the beginning of the line.

Once the configuration file has been created a virtual image has to be built. To do that, after execute *Docker Quick Start Terminal* and change the directory to the place where the Dockerfile is located, next command in the terminal must be used:

**\$ docker – t built *image\_name***



After built, this image is available for his use. In dockerhub we can upload this image to allow other users to use it. An easy idea was to install *docker tools* in UPC server and download the image from the hub, but this way to deal with the problem is inefficient. The best way we considered is to use only Dockerfile for the implementation, as it is a 500bytes file.

With only a simple text file we could create a full virtual machine. In this file, as we saw previously, different steps for the set up was specified. The efficiency of this implementation is based on github<sup>10</sup> use. This webpage is where programmers can store, share and structure their code, so with a simple command we can clone a whole repository of code in a machine, as I did in this work. The code for the game is publicly available<sup>11</sup>.

In the server computing services built the Dockerfile and run in a physical port, linking virtual machine port with it. To run a virtual docker image next command has to be used:

**\$ docker run – it *image\_name***

If we need to use the terminal of the virtual machine to supervise any process or execute something in the container, the command is the follow:

**\$ docker run – it *image\_name bash***

As a result of these steps the visual memory game for this work was available online<sup>12</sup> and open to everyone that wants to collaborate with this research.

## 5.5 Conclusions

In this chapter we have presented a visual memory game, a tool for manual memorability annotation of egocentric images. Technical features have been described and server setup has been presented as a Docker implementation.

---

<sup>10</sup> Github webpage - <https://github.com>

<sup>11</sup> Visual game code webpage –  
<https://github.com/imatge-upc/lifelogging/tree/master/memorygame>

<sup>12</sup> Online visual memory game - <http://imatge.upc.edu:8000>

## Chapter 6 – Visual memory game results

### 6.1 Introduction

After made the visual game publicly, a set of results was obtained. Not all results were valid because the last line of the text file obtained as results of the game was used for control user attention. Despite website had instructions for users, some of them might not understand the task well. Also since this visual game was run in a server some problems can occur during his use, for that reason a supervision of the results is required.

### 6.2 Results

For this part of the project, fifty images were explored and studied, using them as targets for the visual game. After obtain results and to compute memorability score, a number of twenty-five users were considered. By this way, if the definition of memorability is used, the resolution of memorability score is 0.04 points (as the score was computed from twenty-five users).

Once score was computed for each image, a quick comparative of the scores can be done as next figure shows, where a 2D graphic plot predicted memorability score (with convolutional neural network model) against manual annotated score (with visual memory game).

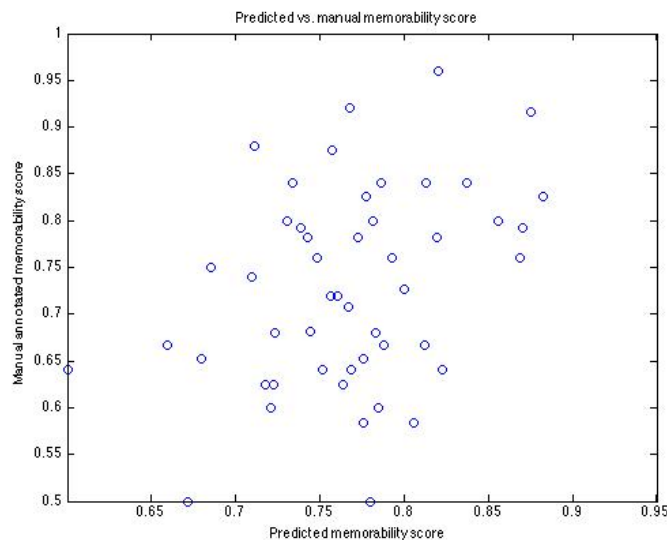


Figure 26. Predicted memorability scores against manual annotated scores

From the figure a difference between predicted and manual memorability score is clear since points are not following a linear pattern as we expect if a correlation of values exist, but some statistical results can be explored:

**Average annotated score: 0.7294**

**Average predicted score: 0.76868**

**Max score difference: 0.27972**

**Min score difference: 0.00237**

**Mean square error: 0.57133**

As we can show, the average score for all this images is similar between annotated and predicted, but it has not a correlated meaning. An interesting value is maximum and minimum scores differences, taking the absolute value of that difference to compare between images. This measure show us how similar is predicted memorability score versus manual annotated memorability score. The lowest difference is near zero, meaning that the score is well predicted for this image, and the highest difference means that for some images the predictor not works properly.

Remembering the idea of this approach, the assumption was that if predicted memorability score (using a CNN) was similar to manually annotated score, this algorithm could be extended to egocentric images. Before extract a conclusion, visualize images with high and low difference is useful, as next figure:



*Figure 27. Image with less exact prediction (maximum difference) at left and image with most exact prediction (minimum difference) at right*

The first image corresponds to the image that has the highest difference of scores. It means that the real score (score obtained with user manual annotation) are very different from the score assigned (or predicted) by the convolutional neural network. This image, like the others, was taken in an egocentric point of view. The second image has the same score in the two ways. Despite be a first person vision image, it can look intentional, like images that was used in MIT work to train MemNet (convolutional neural network used in this work).

If the mean value of the difference between the scores is computed, this value is near 0.1 points. This variation of the score is a 10% of the score range, and this offset don't follow any pattern. For these reasons, if we want to predict the memorability of egocentric images, an approximation of it can be done using MemNet, but the correct procedure for deal with that objective is to perform the current network giving annotated egocentric examples to retrain the algorithm, without forgot previous knowledge.

In the next section, with the results from the game the way to perform a current convolutional neural network will be explained and evaluated.

## 6.3 Conclusions

Analysing the results from the visual game (manual annotations) and predicted scores from a convolutional neural network, in this chapter has been demonstrated that current algorithms cannot be applied for egocentric images due to their different composition.

## Chapter 7 – CNN fine-tune

### 7.1 Introduction

A convolutional neural network is an algorithm that automatically learns features from a set of samples and their correspondent labels. In the first part of this project a pre-trained model was used to predict memorability score of an image set. But after compare manual memorability scores obtained with a visual memory game and memorability scores predicted with this model, an adaptation of that CNN is necessary

### 7.2 Fine-tune a convolutional neural network

Train a convolutional neural network is computationally expensive and requires a lot of examples to allow that network to generalize with any image, because this kind of models have a lot of parameters to compute. If we train a convolutional neural network with few images an overfitting of the model can occur, resulting in a low error prediction in the training data but high error in test set, images that the algorithm has not seen before.

An option to avoid that problem is to fine-tune the model. In a convolutional neural network each layer of the structure have a weight matrix and that values are computed during training. A fine-tune consist in, from a pre-trained model update these weights giving to the algorithm new examples with their correspondent labels. By this way with a less amount of samples we can perform a current model, doing an adaptation of it.

For this project I decide to fine-tune MIT memorability model. Although, from the visual memory game I had few annotated images, so new model might overfit. There are many parameters of CNN to change and we can update all layers weights or only some layers.

As I had few images to fine-tune the algorithm I proposed three different strategies: the first strategy was fine-tune the CNN giving only fifty annotated images. The second strategy was to do visual data augmentation. This process of an image consists in ten transformations: from an image, five crops were done (centre and the four corners) and from those crops, x-axis flip was applied. For this second way I had 400 images to fine-tune (due to set split) the network. The last strategy applied was to do a temporal data augmentation.

The idea is the same as the second strategy, increase the number of samples available to fine-tune, but instead to use an image and apply a set of transformations I took advantage of egocentric problem to increase the number of samples. This problem is the visual similarity of temporal neighbour images, so for each image similar temporal neighbours were manually selected and for them, the same memorability score was considered due to visual similarity.

These three strategies were applied to MemNet model. Also, different models for each strategy were created. These models differ in the number of iterations of the training algorithm (how many time the image set is shown to the algorithm) and also the number of layer weights updated.

The algorithm to fine-tune the convolutional neural network model runs in Python using Caffe framework. A GPU (graphics processing unit) was used during training time because the creation of the model requires a lot of memory.

I have to highlight that this is not a classification problem; it is a regression problem so we have to select the correct way to fine-tune the model because for that kind of models, the label is not an integer. The way to deal with that fact, algorithm input has to be in hdf5 or lmdb format. The first format requires more memory as all images are charged first, instead of lmdb format, where it can be processed with a certain batch size (number of images processed at the same time).

### 7.3 CNN models evaluation

After many configurations of fine-tune algorithm many models are obtained. These models have been compared using different metrics. I considered SAD (sum of absolute differences) and MSE (mean square error) as metric based on the predicted score against the score manually annotated. Also a metric based on the rank images was used to focus the evaluation of each model in the rank position instead of focus the evaluation just in the memorability predicted value. This measure is useful since this model can be used to summarize a set of images taking images with the highest memorability score (top N rank images). The metric based on rank is rank correlation; in special as considered in MIT work I compute Spearman's rank correlation. This metric between two ranks computes the position of an image in a rank and computes a correlation between these positions as next equation shows:

$$r_s = 1 - \frac{6 \sum d^2}{n(n^2 - 1)}$$

Where ‘ $n$ ’ is the number of samples in the rank (pairs of samples) and ‘ $d$ ’ is the difference of positions of each sample between the two ranks.

To evaluate all models a test set was needed and this set is built from images that the algorithm has not seen before. To do that I split fifty images set into forty images to train and 10 images to test. Data augmentation was done for forty images and test set did not had any transformation.

To compute Spearman’s rank correlation I used Matlab tools:

- To compute the rank position of each sample:

**[R, T]=tiedrank(rank\_list);**

- To compute the Spearman’s correlation:

```
diff=0;
for jj=1:length(R)
    diff=diff+(R(jj)-Ro(jj))^2;
end
spear=1-((6*diff)/(pairs*(pairs^2-1)));
```

## 7.4 Evaluation results

Next table shows results of fine-tune process:

Model	SAD	MSE	Spearman’s coef.
MemNet	0.7478	0.00846	0.7727
Finetune_50_iter_25	0.7031	0.00766	0.7182
Finetune_50_iter_50	0.7542	0.00900	0.7061
Finetune_50_iter_75	0.7579	0.00910	0.7061
Finetune_50_iter_100	0.7582	0.00911	0.7001
Finetune_50_fc7_iter_25	1.0605	0.01574	0.7606
Finetune_50_fc7_iter_50	1.1067	0.01699	0.7606
Finetune_50_fc7_iter_75	1.1094	0.01706	0.7606
Finetune_50_fc7_iter_100	1.1096	0.01707	0.7606

Finetune_VA_fc7_iter_25	1.1854	0.01977	0.7182
Finetune_VA_fc7_iter_50	1.5864	0.03100	0.7182
Finetune_VA_fc7_iter_75	1.6248	0.03233	0.7182
Finetune_VA_fc7_iter_100	1.6276	0.06242	0.7182
Finetune_TA_fc7_iter_25	1.0930	0.01635	0.8152
Finetune_TA_fc7_iter_50	1.1346	0.01751	0.8152
Finetune_TA_fc7_iter_75	1.1362	0.01756	0.8152
Finetune_TA_fc7_iter_100	1.1363	0.01756	0.8152
Finetune_TA_fc6_iter_25	0.8723	0.01085	0.8394

*Table 1. Evaluation results between convolutional neural network models.*

In the previous table models name corresponds to:

- Finetune\_50\_iter\_X: model fine-tuned with forty annotated images. This number is due to set split (train and test sets) ‘X’ determines the number of iterations done. All layers had been updated.
- Finetune\_50\_fc7\_iter\_X: model fine-tuned with fifty annotated images where only the layer fc7, the last layer before regressor, was updated (his weights). ‘X’ determines the number of iterations done.
- Finetune\_VA\_fc7\_iter\_X: model fine-tuned with a visual data augmentation. A total of 500 annotated images were used for that. As in the previous model only layer fc7 was updated. ‘X’ determines the number of iterations done.
- Finetune\_TA\_fc7\_iter\_X: the same procedure as last model but changing the set for temporal data augmentation image set.
- Finetune\_TA\_fc6\_iter\_25: fine-tune with the temporal data augmentation but updating at this time fc6 layer, the sixth convolutional layer.

From results table we can see that the best result in Spearman’s rank correlation value is for the model crated with a temporal data augmentation set and updating weights for layer fc6 (sixth layer).

Due to the low number of samples, as more iteration was done the error between predicted scores and manually annotated scores increase. We see that update (fine-tune) some layers are better than fine-tune all layers because as many layers there are the number of parameters increase. Also we can see that updating weights from only the sixth layer in



the network is better than doing the same for the last layer (seventh layer) before regressor.

Note that last layer (regressor) must not be updated because we can obtain values out of the normalised range (that goes from 0 to 1).

## 7.5 Conclusions

After try different fine-tune configurations results shows the best one is fine-tune current model (MemNet) for memorability prediction updating only the weights for sixth layer (fc6, fully connected layer) training the CNN with a training set built from apply temporal data augmentation to forty annotated egocentric images, performing results from MemNet CNN evaluating both models with Spearman's rank correlation.

## Chapter 8 – Conclusions

Egocentric images open a new branch for the research as they present a different image composition. First of all, egocentric devices provide us a large number of images per day that can be sometimes very similar between them and due to their non-intentional acquisition can be blurred, with no information, etc. Many previous works proposes approaches to try to select the most relevant frames from a set of images. In the first part of this project a correlation between memorability of images (predicted with a trained convolutional neural network trained for that purpose) and physiological signals have been retrieve. From a new dataset that contains images and their correspondent physiological signals (heart rate, galvanic skin response and calories) all this data were processed as described in previous sections and the results shows that a pattern between these physiological signals and memorability exists. A general rule is that when memorability of an image increases the value of heart rate tends to decrease, and the same happen with galvanic skin response value. Also results show that median value of galvanic skin response is equal to 0, so that value only change if there is a special event occurs. For that reason this signal should not be used for relate it directly to memorability score. Despite the general pattern shown the value of heart rate value cannot be directly related to memorability score, so a temporal processing in data acquisition must be done to not only consider one value of a physiological signal, a temporal change is more valuable.

First person vision images have another problem when algorithms process them. Some previous studies highlight current algorithms cannot be applied to egocentric images because they have been trained with human-taken images. To demonstrate that a visual memory game was created and allows us to compute manually the memorability score for a set of images. After compare it with predicted value (obtained with trained convolutional neural network) the results highlight an adaptation was needed to apply current techniques to egocentric images. To do that fine-tune of an existing CNN for memorability prediction was done. Some models were created and compare between them with Spearman's rank correlation because for summarization application (the main application of the model) the most important is the rank position of the images. Results show models trained with temporal data augmentation from annotated set performs current model. The best metric



Detect snap points in egocentric images with physiological signals – Marc Carné Herrera obtained belongs to a model fine-tuned with temporal data augmentation (taking advantage of temporal neighbours) and updating only weights from sixth layer. Also the results shows that the number of iterations is important for the error between scores but not for rank correlation.

## References

[1]

B. Xiong and K. Grauman. "Detecting Snap Points in Egocentric Video with a Web Photo Prior". In ECCV, 2014. [bibtex]

[2]

Talavera, E., Dimiccoli, M., Bolaños, M., Aghaei, M., & Radeva, P. (2015). "R-clustering for egocentric video segmentation". In Pattern Recognition and Image Analysis (pp. 327-336). Springer International Publishing.

[3]

Lidon, A., Bolaños, M., Dimiccoli, M., Radeva, P., Garolera, M. & Giró-i-Nieto, X. (2015) "Semantic Summarization of Egocentric Photo Stream Events". Special Issue on Wearable and Ego-vision Systems for Augmented Experience. In Transactions on Human-Machine Systems Journal

[4]

Isola, P., Xiao, J., Torralba, A., Oliva, A. What makes an image memorable? IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2011. Pages 145-152.

[5]

Isola, P., Parikh, D., Torralba, A., Oliva, A. Understanding the intrinsic memorability of images. To appear in Advances in Neural Information Processing Systems (NIPS), 2011.

[6]

Image Memorability and Visual Inception, Aditya Khosla, Jianxiong Xiao, Phillip Isola, Antonio Torralba, Aude Oliva. In SIGGRAPH Asia, 2012 (*invited paper, technique briefs section*)

[7]

Memorability of Image Regions [paper] Aditya Khosla, Jianxiong Xiao, Antonio Torralba and Aude Oliva

*Advances in Neural Information Processing Systems (NIPS)*, 2012.

[8]

The Intrinsic Memorability of Face Photographs, Wilma A. Bainbridge, Phillip Isola, Aude Oliva. In *Journal of Experimental Psychology: General (JEPG)*, 2013

[9]

Mukul Bisht et al. (2007) “Context-Coded Memories: "Who, What, Where, When, Why?"”. In Workshop "Supporting Human Memory with Interactive Systems", HCI Conference, September 4th, 2007, Lancaster, UK

[10]

Ceren Ergorul and Howard Eichenbaum, “The Hippocampus and Memory for “What,” “Where,” and “When””. In Center for Memory and Brain, Program in Neuroscience, Boston University, Boston, Massachusetts 02215, USA

## Glossary

CNN – convolutional neural network.

Fc6 – fully connected layer of a convolutional neural network, sixth layer.

MSE – mean square error.

SAD – sum of absolute difference.

Fine-tune – from a pre-trained model of a convolutional neural network update weights from learning new examples.