

---

EAGLeNet – A DEEP MULTI-TASK LEARNING CNN FOR EMOTION,  
AGE AND GENDER RECOGNITION

**João Santos Crispiniano Vieira**

---

Dissertation  
Master in Data Analytics

---

Supervised by  
**Professor Doutor João Manuel Portela da Gama**  
**Doutor Brais Cancela Barizo**

---

2018



# Abstract

Facial analysis systems have become increasingly relevant to a growing number of applications. Since the recent advent of Deep Learning, particularly Convolutional Neural Networks, facial analysis tasks have been a trendy topic, as developed models are consistently breaking accuracy records on various tasks, many breaking the human-level barrier.

Accurate systems have been proposed for the tasks of emotion recognition, age estimation and gender recognition on face images, often at the expense of heavy models that require much processing power and time. The present work focuses on a solution for the three tasks deploying an efficient and fast system. To achieve this, various models derived from the same fundamental architecture were developed and compared, which perform the three tasks simultaneously on a single Convolutional Neural Network, through the use of a Multi-task learning approach.

The final models are efficient and fast, achieving successful results comparable to the Single-task models, supporting the Multi-task Learning approach.

**Keywords:** Facial analysis, Deep Learning; Convolutional Neural Networks; Multi-task Learning.

# Resumo

Os sistemas de análise facial têm vindo a ganhar relevância num número crescente de aplicações. Com o advento recente do Deep Learning, particularmente das Convolutional Neural Networks, as tarefas de análise facial têm-se tornado um tópico de interesse geral, muitas ultrapassando a performance do humano.

Sistemas precisos têm sido propostos para as tarefas de reconhecimento de emoção, estimação de idade e reconhecimento de género em imagens faciais, muitas vezes recorrendo a modelos pesados que requerem forte poder e tempo de processamento. O presente estudo foca-se em desenvolver uma solução rápida e eficiente para as três tarefas. Para o alcançar, vários modelos derivados da mesma arquitectura base foram desenvolvidos e comparados, realizando as três tarefas numa única Convolutional Neural Network. Para isso foi usada uma abordagem de aprendizagem Multi-task.

Os modelos obtidos são eficientes e rápidos, alcançando bons resultados, comparáveis aos dos modelos Single-task, assim suportando a abordagem Multi-task.

**Palavras-chave:** Análise facial, Deep Learning; Convolutional Neural Networks; Aprendizagem Multi-task.

# Table of Contents

|                                                             |           |
|-------------------------------------------------------------|-----------|
| <b>CHAPTER 1 – INTRODUCTION .....</b>                       | <b>1</b>  |
| 1.1. PROBLEM DEFINITION .....                               | 2         |
| 1.2. MOTIVATION .....                                       | 2         |
| 1.3. STRUCTURE OF THE DISSERTATION .....                    | 3         |
| <b>CHAPTER 2 – LITERATURE REVIEW.....</b>                   | <b>4</b>  |
| 2.1. DEEP LEARNING AND CNNs .....                           | 4         |
| 2.1.1. <i>Deep Learning: definition and evolution</i> ..... | 4         |
| 2.1.2. <i>Convolutional Neural Networks</i> .....           | 5         |
| 2.1.2.1. Layers .....                                       | 7         |
| Convolution layers .....                                    | 7         |
| Pooling layers .....                                        | 8         |
| Fully Connected layers.....                                 | 9         |
| Non-linear Activation Functions .....                       | 11        |
| 2.1.3. <i>Advances in CNNs</i> .....                        | 11        |
| 2.1.3.1. Convolution layer .....                            | 11        |
| 2.1.3.2. Pooling layer.....                                 | 13        |
| 2.1.3.3. Activation function .....                          | 13        |
| 2.1.3.4. Regularization .....                               | 14        |
| 2.1.3.5. Optimization .....                                 | 14        |
| 2.1.3.6. Structure.....                                     | 15        |
| 2.2. FACIAL ANALYSIS .....                                  | 18        |
| 2.2.1. <i>Gender Recognition</i> .....                      | 18        |
| 2.2.2. <i>Age Estimation</i> .....                          | 18        |
| 2.2.3. <i>Emotion Recognition</i> .....                     | 19        |
| 2.3. MULTI-TASK LEARNING ARCHITECTURES .....                | 20        |
| 2.3.1. <i>Deeply supervised MTL architecture</i> .....      | 21        |
| 2.3.2. <i>Parallel MTL architecture</i> .....               | 23        |
| 2.4. DAGER – A COMMERCIAL APPROACH .....                    | 24        |
| <b>CHAPTER 3 – METHODOLOGY.....</b>                         | <b>26</b> |
| 3.1. TASK FORMULATION .....                                 | 26        |

|                                                                 |           |
|-----------------------------------------------------------------|-----------|
| 3.1.1. <i>Emotion recognition</i> .....                         | 26        |
| 3.1.2. <i>Age estimation</i> .....                              | 27        |
| 3.1.3. <i>Gender recognition</i> .....                          | 28        |
| 3.2. DATABASES .....                                            | 29        |
| 3.2.1. <i>Image Pre-processing</i> .....                        | 29        |
| 3.3. MODEL DESIGN .....                                         | 31        |
| 3.3.1. <i>Baseline Network Architecture</i> .....               | 32        |
| 3.3.2. <i>Training procedure</i> .....                          | 34        |
| 3.3.2.1 Single-task Training.....                               | 35        |
| Architectures .....                                             | 35        |
| Model training – parameters, procedures and data handling ..... | 37        |
| 3.3.2.1. Multi-task Training.....                               | 38        |
| Architectures .....                                             | 38        |
| Model training – parameters, procedures and data handling ..... | 41        |
| 3.4. SOFTWARE.....                                              | 42        |
| <b>CHAPTER 4 – RESULTS AND EVALUATION .....</b>                 | <b>43</b> |
| 4.1. SINGLE-TASK MODELS.....                                    | 43        |
| 4.1.1. <i>Emotion Recognition</i> .....                         | 43        |
| 4.1.1.1. Class Unweighted.....                                  | 44        |
| Training phase.....                                             | 44        |
| Testing phase .....                                             | 45        |
| 4.1.1.2. Class weighted.....                                    | 46        |
| Training phase.....                                             | 46        |
| Testing phase .....                                             | 47        |
| 4.1.2. <i>Age Estimation</i> .....                              | 48        |
| Training phase.....                                             | 48        |
| Testing phase .....                                             | 49        |
| 4.1.3. <i>Gender Recognition</i> .....                          | 50        |
| 4.1.3.1. Class unweighted.....                                  | 50        |
| Training phase.....                                             | 50        |
| Testing phase .....                                             | 51        |
| 4.1.3.2. Class weighted.....                                    | 52        |
| Training phase.....                                             | 52        |
| Testing phase .....                                             | 53        |

|                                                   |           |
|---------------------------------------------------|-----------|
| 4.2. MULTI-TASK MODELS .....                      | 54        |
| 4.2.1. <i>Triple-Module Sharing (v1)</i> .....    | 55        |
| Training phase.....                               | 55        |
| Testing phase .....                               | 57        |
| 4.2.2. <i>Dual-Module Sharing (v2)</i> .....      | 60        |
| Training phase.....                               | 60        |
| Testing phase .....                               | 62        |
| 4.2.3. <i>Single-Module Sharing (v3)</i> .....    | 65        |
| Training phase.....                               | 65        |
| Testing phase .....                               | 67        |
| 4.2.4. <i>Zero-Module Sharing (v4)</i> .....      | 69        |
| Training phase.....                               | 69        |
| Testing phase .....                               | 72        |
| 4.3. COMPARISONS.....                             | 74        |
| 4.3.1. <i>Emotion task model comparison</i> ..... | 74        |
| 4.3.1. <i>Age task model comparison</i> .....     | 76        |
| 4.3.1. <i>Gender task model comparison</i> .....  | 77        |
| 4.3.1. <i>Global model comparison</i> .....       | 79        |
| <b>5. CONCLUSIONS AND FINAL THOUGHTS .....</b>    | <b>81</b> |
| <b>REFERENCES.....</b>                            | <b>83</b> |

# List of Tables

|                                                            |    |
|------------------------------------------------------------|----|
| Table 1 – Emotion dataset class distribution .....         | 29 |
| Table 2 – Gender dataset train-validation-test split ..... | 30 |
| Table 3 - Emotion dataset train-validation-test split..... | 30 |
| Table 4 - Emotion training class weights.....              | 38 |
| Table 5 - Gender training class weights .....              | 38 |
| Table 6 – Class metrics.....                               | 45 |
| Table 7 – Model metrics.....                               | 46 |
| Table 8 – Class metrics.....                               | 48 |
| Table 9 – Model metrics.....                               | 48 |
| Table 10 – Class metrics.....                              | 52 |
| Table 11 – Model metrics .....                             | 52 |
| Table 12 – Class metrics.....                              | 54 |
| Table 13 – Model metrics .....                             | 54 |
| Table 14 – Emotion class metrics.....                      | 58 |
| Table 15 – Emotion metrics .....                           | 58 |
| Table 16 – Gender class metrics .....                      | 59 |
| Table 17 – Gender metrics .....                            | 60 |
| Table 18 – Emotion class metrics.....                      | 63 |
| Table 19 – Emotion metrics .....                           | 63 |
| Table 20 – Gender class metrics .....                      | 64 |
| Table 21 – Gender metrics .....                            | 64 |
| Table 22 – Emotion class metrics.....                      | 68 |
| Table 23 – Emotion metrics .....                           | 68 |
| Table 24 – Gender class metrics .....                      | 69 |
| Table 25 – Gender metrics .....                            | 69 |
| Table 26 – Emotion class metrics.....                      | 72 |
| Table 27 – Emotion metrics.....                            | 72 |
| Table 28 – Gender class metrics .....                      | 73 |
| Table 29 – Gender metrics .....                            | 73 |
| Table 30 – Emotion class model comparison.....             | 75 |
| Table 31 – Emotion model comparison.....                   | 76 |

|                                                |    |
|------------------------------------------------|----|
| Table 32 – Age model comparison.....           | 77 |
| Table 33 – Gender class model comparison ..... | 78 |
| Table 34 – Gender model comparison .....       | 79 |

# List of Figures

|                                                                                                       |    |
|-------------------------------------------------------------------------------------------------------|----|
| Figure 1 - Traditional pattern recognition (LeCun et al., 1998) .....                                 | 6  |
| Figure 2 - General CNN architecture (Guo et al., 2016) .....                                          | 7  |
| Figure 3 - Mathematical model of a neuron (Cadène et al., 2016).....                                  | 10 |
| Figure 4 - Residual Connection (He et al., 2016).....                                                 | 16 |
| Figure 5 – Global Average Pooling (Cook, 2017) .....                                                  | 17 |
| Figure 6 - Deeply supersized MTL architecture (Ranjan et al.,2017) .....                              | 23 |
| Figure 7 - Parallel MTL architecture (Xing et al., 2017) .....                                        | 24 |
| Figure 8 - DAGER framework (Dehghan et al., 2017).....                                                | 25 |
| Figure 9 - Gender dataset train-validation-test split.....                                            | 30 |
| Figure 10 – Emotion dataset train-validation-test split .....                                         | 30 |
| Figure 11 - Age Distribution for the training set, validation set, test set and the merged sets ..... | 31 |
| Figure 12 - Framework Comparison (Operations per Accuracy) (Canziani et al., 2017).....               | 32 |
| Figure 13 – Baseline Architecture .....                                                               | 34 |
| Figure 14 - Emotion Recognition Model.....                                                            | 35 |
| Figure 15 - Age Estimation Model .....                                                                | 36 |
| Figure 16 - Gender Recognition Model.....                                                             | 36 |
| Figure 17 - V1 model architecture .....                                                               | 39 |
| Figure 18 - V2 model architecture .....                                                               | 40 |
| Figure 19 - V3 model architecture .....                                                               | 40 |
| Figure 20 - V4 model architecture .....                                                               | 41 |
| Figure 21 - Training and validation loss (real and smoothed) .....                                    | 44 |
| Figure 22 - Training and validation metrics (real and smoothed).....                                  | 44 |
| Figure 23 – Confusion matrix.....                                                                     | 45 |
| Figure 24 - Training and validation loss (real and smoothed) .....                                    | 46 |
| Figure 25 - Training and validation metrics (real and smoothed).....                                  | 47 |
| Figure 26 - Confusion matrix .....                                                                    | 47 |
| Figure 27- Training and validation loss (real and smoothed) .....                                     | 49 |
| Figure 28 - Training and validation metrics (real and smoothed).....                                  | 49 |
| Figure 29 – MAE per age results.....                                                                  | 50 |
| Figure 30 - Training and validation loss (real and smoothed) .....                                    | 51 |
| Figure 31 - Training and validation metrics (real and smoothed).....                                  | 51 |
| Figure 32 - Confusion matrix .....                                                                    | 52 |
| Figure 33 - Training and validation loss (real and smoothed) .....                                    | 53 |
| Figure 34- Training and validation metrics (real and smoothed) .....                                  | 53 |

|                                                                                              |    |
|----------------------------------------------------------------------------------------------|----|
| Figure 35 - Confusion matrix .....                                                           | 53 |
| Figure 36 - Emotion training and validation loss (real and smoothed) .....                   | 55 |
| Figure 37 - Emotion training and validation metrics (real and smoothed) .....                | 56 |
| Figure 38 - Age training and validation loss (real and smoothed) .....                       | 56 |
| Figure 39 - Age training and validation metrics (real and smoothed) .....                    | 56 |
| Figure 40 - Gender training and validation loss (real and smoothed) .....                    | 56 |
| Figure 41 - Gender training and validation metrics (real and smoothed) .....                 | 57 |
| Figure 42 - Aggregated gender and age training and validation loss (real and smoothed) ..... | 57 |
| Figure 43 - Total training and validation loss (real and smoothed) .....                     | 57 |
| Figure 44 - Emotion confusion matrix .....                                                   | 58 |
| Figure 45 - MAE per age results .....                                                        | 59 |
| Figure 46 - Gender confusion matrix .....                                                    | 59 |
| Figure 47 - Emotion training and validation loss (real and smoothed) .....                   | 60 |
| Figure 48 - Emotion training and validation metrics (real and smoothed) .....                | 61 |
| Figure 49 - Age training and validation loss (real and smoothed) .....                       | 61 |
| Figure 50 - Age training and validation metrics (real and smoothed) .....                    | 61 |
| Figure 51 - Gender training and validation loss (real and smoothed) .....                    | 61 |
| Figure 52 - Gender training and validation metrics (real and smoothed) .....                 | 62 |
| Figure 53 - Aggregated gender and age training and validation loss (real and smoothed) ..... | 62 |
| Figure 54 - Total training and validation loss (real and smoothed) .....                     | 62 |
| Figure 55 - Emotion confusion matrix .....                                                   | 63 |
| Figure 56 - Mae per age results .....                                                        | 64 |
| Figure 57 - Gender confusion matrix .....                                                    | 64 |
| Figure 58 - Emotion training and validation loss (real and smoothed) .....                   | 65 |
| Figure 59 - Emotion training and validation metrics (real and smoothed) .....                | 65 |
| Figure 60 - Age training and validation loss (real and smoothed) .....                       | 66 |
| Figure 61 - Age training and validation metrics (real and smoothed) .....                    | 66 |
| Figure 62 - Gender training and validation loss (real and smoothed) .....                    | 66 |
| Figure 63 - Gender training and validation metrics (real and smoothed) .....                 | 66 |
| Figure 64 - Aggregated gender and age training and validation loss (real and smoothed) ..... | 67 |
| Figure 65 - Total training and validation loss (real and smoothed) .....                     | 67 |
| Figure 66 - Emotion confusion matrix .....                                                   | 68 |
| Figure 67 - MAE per age result .....                                                         | 68 |
| Figure 68 - Gender confusion matrix .....                                                    | 69 |
| Figure 69 - Emotion training and validation loss (real and smoothed) .....                   | 70 |
| Figure 70 - Emotion training and validation metrics (real and smoothed) .....                | 70 |

|                                                                                              |    |
|----------------------------------------------------------------------------------------------|----|
| Figure 71 - Age training and validation loss (real and smoothed) .....                       | 70 |
| Figure 72 - Age training and validation metrics (real and smoothed) .....                    | 70 |
| Figure 73 - Gender training and validation loss (real and smoothed).....                     | 71 |
| Figure 74 - Gender training and validation metrics (real and smoothed).....                  | 71 |
| Figure 75 - Aggregated gender and age training and validation loss (real and smoothed) ..... | 71 |
| Figure 76 - Total training and validation loss (real and smoothed).....                      | 71 |
| Figure 77 - Emotion confusion matrix .....                                                   | 72 |
| Figure 78 – MAE per age results.....                                                         | 73 |
| Figure 79 – Gender confusion matrix.....                                                     | 73 |

# Chapter 1 – Introduction

A change of paradigm is currently underway. Computers have been consistently surpassing humans in many tasks of the day-to-day life in the past decades, with us relying more and more on them. If a bank's database servers would for some reason shutdown and wipe out all of its information, the world economy could collapse. We are in the age of information, and that information is increasingly being collected and stored by machines. Machine Learning uses this information to learn patterns. It can be used to predict the weather, to help on credit approval, or to recommend a visit to the doctor if an irregular heartbeat is detected.

But this change of paradigm is currently occurring in a specific subfield of computer science – computer vision, fueled by the advent of Deep Learning and Convolutional Neural Networks. Over the last decade, the rate of image uploads to the internet has been increasing exponentially. Motivated by this and the fact that high computing power is becoming increasingly accessible to the common user, more and more research is being done on computer vision. Problems that were once considered impossible to study due to lack of data or computing power are being attempted now (Ekmekji, 2016). Cars can drive themselves nowadays, detecting pedestrians and predicting/avoiding accidents. Despite self-driving cars being a hot-topic today, there is perhaps one even hotter – facial analysis.

Facial analysis comprehends several tasks, which can be used for several purposes. There are models for the task of face detection, pose estimation, age estimation, face recognition, smile detection, gender recognition, and so on. In many cases, machine-estimation has been consistently breaking the human level, achieving outstanding results. There is a sense of urgency in the quest to investigate and develop the best model, fueled by Artificial Intelligence competitions and the tech business world. These systems can be applied to security systems, person identification, human-computer interaction platforms, marketing, etc.

We are getting closer, step by step, to become fully connected to technology, which can now learn and estimate our needs and characteristics. Facial analysis plays a key role in this scenario, as a significant part of human communication is nonverbal.

## 1.1. Problem Definition

The present dissertation seeks to propose a new framework of a single Convolutional Neural Network for the simultaneous tasks of age estimation, gender recognition and emotion recognition on face images. With the purpose of being able to perform all three tasks in a single network, a multi-task learning architecture is used, jointly learning the various tasks.

Using a baseline Convolutional Neural Network architecture, modifications are made to incorporate the three tasks, changes are made to the baseline architecture processes and layers, and various loss functions are considered. This has the objective of attaining an efficient and fast Multi-task working model. Various models are created with different shared layers across tasks, evaluating and comparing them on a test set. The final models should be able to provide insights on a face image concerning the age of the individual outputted in a real valued variable, the emotion of the individual outputted in one of seven classes (anger, disgust, happiness, fear, surprise, sadness and neutrality), and the gender of the individual outputted in one of two classes (male or female).

## 1.2. Motivation

Facial analysis systems can ultimately change the way we live. Allied to Artificial Intelligence, it can provide personal assistance in every aspect of life. It can estimate our mood upon waking and, connecting it to our habits, age and other factors, provide the perfect meal for the moment, the best music to accompany that meal, and the perfect outfit, linked to the weather prediction. Likewise, upon entering a clothing store, it can automatically suggest the best piece of clothing for you, according to your age and gender, and send to your smartphone a picture of you virtually dressed in such suggestions.

Machines will be capable of taking care of the uninteresting part of life, end-to-end. Supported by computer vision and facial analysis, the full connection and understanding of humans will soon happen. In fact, many state-of-the-art models have already outpaced human recognition in many facial analysis tasks.

Deep Learning and Convolutional Neural Networks are a hot-topic in the field of Machine Learning. Many breakthroughs are being reached every other day. Being a part of that is what motivates me to develop this work. As far as it is possible to determine, a Deep

Multi-Task Learning Convolutional Neural Network for Emotion, Age and Gender Recognition is still to be developed, and such an endeavor would certainly constitute a valuable contribution to Machine Learning.

### **1.3. Structure of the dissertation**

The present dissertation is organized as follows: an introductory chapter, a literature review, the methodology approach, the results and evaluation and the concluding thoughts.

In the first chapter, the topic of the dissertation is introduced, as well as the problem description. The main motivations to the development of the study are then presented, followed by a brief description of the structure of the dissertation.

The second chapter covers the literature review, where firstly a description of Deep Learning and Convolutional Neural Networks is presented, followed by a review on facial analysis and the tasks proposed as the problem in this dissertation.

In the third chapter the methodology is described, explaining the task formulation and detailing the intended models. The databases for training, model design and training, testing procedure, and finally the used software are also explained.

A fourth chapter presents the results and evaluation of the developed models, each with interpretations over the training and testing phase, finalizing with a thorough model comparison section.

The fifth and final chapter outlines the final thoughts and conclusions on the developed work.

## Chapter 2 – Literature Review

The present chapter introduces the concepts and descriptions relevant to the present study. Firstly, the definition of Deep Learning is introduced, followed by a more exhaustive description of Convolution Neural Networks, an application of such method which is the baseline concept of the present dissertation. Secondly, the topic of facial analysis is reviewed, focusing on the main tasks to be developed in the course of the study. The third section introduces the multi-task learning concept, where some examples are drawn to better expose the topic. Finally, a specific commercial related work is mentioned.

### 2.1. Deep Learning and CNNs

In this section, an introduction to Deep Learning is presented, briefly describing its evolution. Secondly, in a more exhaustive way, Convolutional Neural Networks are reviewed, defining and explaining each layer, elucidating the main differences between a CNN and regular Neural Network. Finally, the main advances achieved in each layer and process are outlined, with special emphasis on the ones used throughout this work.

#### 2.1.1. Deep Learning: definition and evolution

There is a subfield of Machine Learning which aims to discover multiple levels of distributed representations – such field is called Deep Learning. Deep Learning relies on hierarchical architectures to learn high-level abstractions in data (Y. Guo et al., 2016). Various traditional machine learning tasks have been improved through the use of deep learning algorithms, such as semantic parsing (e.g. Bordes et al., 2012), transfer learning (e.g. Ciresan et al., 2012b), natural language processing (e.g. Mikolov et al., 2013) and computer vision (e.g. Ciresan et al., 2012a; Krizhevsky et al., 2012) among others.

Traditional Machine Learning techniques are limited in tasks such as object recognition in images, mainly because of their capacity for processing natural data – data in its raw form – hence the need to develop and use Deep Learning, a more advanced technique in representation learning (LeCun et al., 2015).

Deep Learning is a class of Artificial Neural Networks (NNs). Standard NNs, or Shallow NNs, have been around at least since the 1960s. They consist of many simple connected

neurons, which can be viewed as simple processors, where each one produces an arrangement of real-valued activations. They are divided in two sections: the input neurons, activated through the perception of the environment through sensors, and other neurons, activated through weighted connections to previously activated neurons. The environment might be influenced by actions deployed by some neurons. Shallow NNs and Deep NNs are distinguished by the depth of these chains of connected neurons, which are chains of possibly learnable causal links between action and effects (Schmidhuber, 2015).

Deep Learning became feasible in some areas during the 1990s and saw many improvements during the 1990s and 2000s. In the 21st century came the boom of Deep Learning, attracting wide-spread attention, due to the development of models that outperformed alternative machine learning methods, and the fact that supervised deep NNs have won various international machine learning competitions since 2009, even achieving better results than human visual pattern recognition in some domains (Schmidhuber, 2015). The boom in the development of DL methods is due to three main reasons: increased chip processing capacity, lowered cost of computing hardware, and the considerable achievements mentioned previously (Y. Guo et al., 2016).

Early supervised Neural Networks were variants of linear regression methods, dating back to the early 1800s in the work of Gauss. But simple Artificial Neural Networks architectures were firstly developed in the 1940s, and in the following decades the use of supervised learning techniques were introduced to these architectures. During the 1960s, the discovery of simple and complex cells in the cat's visual cortex later inspired deep NN architectures used in award-winning models. Throughout this decade, NNs trained by the GMDH method – Group Method of Data Handling – were the first kind of Deep Learning systems. In 1979 Fukushima introduced the Neocognitron, the first truly deep NN, a convolutional NN. The first efficient application of backpropagation in NNs was described in 1981 (Schmidhuber, 2015).

### **2.1.2. Convolutional Neural Networks**

Having presented the definition of Deep Learning, Convolutional Neural Networks (CNNs) are now addressed. CNNs are shown to outperform all the other techniques in image pattern recognition tasks, exemplified in Figure 1. They are one of the best Deep

Learning techniques with a robust multiple layer training (LeCun et al., 1998). Proving to be highly effective, they are the most commonly used in computer vision and speech recognition problems (Y. Guo et al., 2016).

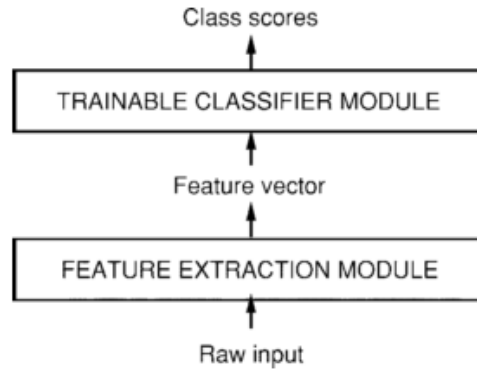


*Figure 1 - Traditional pattern recognition (LeCun et al., 1998)*

Multilayer networks' ability to learn complex, high dimensional, nonlinear mappings through large datasets makes them candidates for image recognition tasks. This can be done with regular fully connected feedforward networks with some success, but there are mainly two problems (LeCun et al., 1998).

Firstly, images are often large, with several thousands or millions of pixels, which would work as variables, and a fully connected first layer with, for example, one hundred hidden units in the first layer would comprehend several tens of thousands of weights. Several parameters increase the capacity of the system, requiring a larger training set. Furthermore, the toll taken on memory from storing so many weights may exclude some hardware implementations. Perhaps the greater problem of unstructured nets for image or speech applications is the fact that the feeding of the input layer, the signals, such as images, must be approximately size normalized and centered (LeCun et al., 1998). This can be automatically achieved in CNNs, as will be described later.

Secondly, there is the problem concerning the topology of the input: it is entirely ignored – the outcome of the training is not affected by the input order of the variables. This poses a problem because images have a strong two-dimensional local structure, being that pixels spatially close are highly correlated. CNNs perform better relative to this problem since they search and extract local features restricting the receptive fields of hidden units locally (LeCun et al., 1998), as will be later described.

A typical CNN (Fig. 2) comprehends three main neural layers: convolutional layers, pooling layers, and fully connected layers. Usually, the convolutional layers alternate with pooling layers, followed by some fully connected ones at the end of the network. All of them perform distinct roles. There are two training stages of the network, a forward stage and a backward stage. During the first, the effort is on the representation of the input image with the current parameters, weights and biases, in each layer, comparing the predicted to the ground truth labels for the computation of the loss cost. Secondly, and using this loss cost, the network's backward stage computes the gradients of each parameter with chain rules to update all the parameters of the network, preparing it for the next forward computation. With enough iterations of the two stages, the network learning can be stopped (Y. Guo et al., 2016).

In the following section the description of each layer is presented.

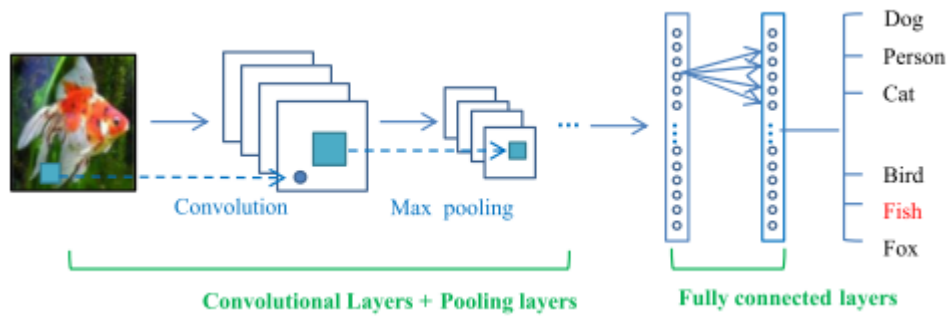


Figure 2 - General CNN architecture (Guo et al., 2016)

### 2.1.2.1. Layers

#### Convolution layers

The purpose of convolution layers is to build feature maps by convolving the whole image, as well as the intermediate feature maps, with the use of various kernels (Y. Guo et al., 2016). In image processing, a kernel is a two-dimensional matrix of numbers used to compute new matrices by performing some calculations between the kernel and the input picture or feature map. The input image is nothing but a matrix as well, such as in an 8-bit RGB image, for example, in which each pixel (each cell of the matrix) is represented by three numbers with a value ranging from 0 to 255 (red, green and blue information). There are multiple kernels for various tasks in image processing, such as blurring and edge detection (Cope, 2017).

In an image, close pixels exhibit many spatial relationships among them. The location of such neighboring pixels within the image should not affect these spatial relationships. A convolution layer is instructed with a set of filters (or kernels), which are analogous to templates. A number of these filters are convolved with the input image to generate feature maps, by matching the learned filters to every possible patch of the image. If the filter has a high correlation with a region of the input image, the resulting feature map will have a strong response in the location of such region (Zeiler, 2014).

So, the convolutional layer focuses on learning feature representations of the inputs. One convolution layer is composed of several convolution kernels to compute different feature maps. A new feature map is obtained by first convolving the input image or feature map with a learned kernel and then applying a nonlinear activation function, specified later in the study, on the convolved results. On a newly obtained feature map, each neuron is connected to a set of neighboring neurons in the previous layer, called the receptive field of this neuron in the previous layer. During the generation of each feature map, all the spatial locations of the input share the same kernel. The complete set of feature maps generated at each convolution layer is obtained through the use of several different kernels (Gu et al., 2017).

A spatial convolution is defined by (1) the number of filters, as the number of output channels, (2) by the properties of such filters, as the number of input channels and its width and height, (3) and the properties of the convolution process, like the padding or stride size (Cadène et al., 2016).

The use of convolution layers provides three main advantages: (1) for the same feature map, there is a mechanism for weight sharing which decreases the number of parameters; (2) correlations among neighboring pixels is learnt through local connectivity; (3) equivariance - the location of the object is not important, there is an invariance towards it (Zeiler, 2014).

### **Pooling layers**

Pooling layers are used in CNNs to provide invariance in cases where the input images have just minute differences, and to reduce the dimensionality of the feature maps. A pooling function is used over a region of pixels. One of the most common functions is Max pooling, where, during backpropagation, the gradient is placed in a single location, making this function one of the preferred ones as it dodges cancellation of negative elements and prevents blurring of the activations and gradients throughout the network (Cadène et al.,

2016). Another common pooling strategy is average pooling, but max pooling is capable of achieving faster convergence of the network, improve the generalization requiring less training examples selecting better invariant features (Scherer et al., 2010). Most of the recent CNN implementation nowadays utilize max-pooling (e.g. Ciresan et al., 2011; Krizhevsky et al., 2012).

A pooling layer usually follows a convolutional layer and has the ability to reduce the dimensions of feature maps and networks parameters. As previously stated, and similarly to convolutional layers, they are translation invariant, as their computations consider the neighboring pixels. This type of layer is the most extensively studied (Y. Guo et al., 2016). The three more relevant new approaches on the pooling method, aside from the common max-pooling and average-pooling: (1) the Stochastic pooling, which works similarly to standard max pooling, but tries to overcome a common problem of this method, which is overfitting to the training dataset (making it harder to generalize), by creating many copies of the input image each having small local deformations (Zeiler & Fergus, 2013); (2) the Spatial pyramid pooling, developed by He et al. (2015b), a solution to deal with different scales handling, sizes and aspect ratios, as the other methods require a fixed-sized input image, which can be applied to any CNN architecture replacing the last pooling layer (Liu et al., 2014); (3) the Def-pooling, which tries to deal with the deformation of visual patterns more efficiently than max-pooling (Ouyang et al., 2015). The various strategies can be used in combination in order to boost the CNN's performance.

In sum, the aggregation function, the dimensions of the area where it is applied and the convolution's properties are the elements that define the spatial pooling layer (Cadène et al., 2016).

### **Fully Connected layers**

The fully or linear connected layer motivation is the basic processing unit of the brain, the neuron (Fig. 3). It contains several neurons and can be mathematically seen as a function that applies a linear transformation on an input vector, and outputs another vector with another dimension, usually having a bias parameter. In each layer, every neuron is connected to the same input layer – hence fully connected - and outputs a signal (Cadène et al., 2016).

CNNs high-level reasoning is performed in these layers. Every neuron in a fully connected layer is connected to all the feature maps in the previous layer. Usually the last layer of the

network is the loss layer, where the penalty of the deviation between predicted values and true ones is specified, during the training of the network. There are various loss functions available, and the choice of which to use relies on the type of problem. Softmax loss is the most commonly used, and it is applied on classification problems where the prediction is a single class between a set of mutually exclusive ones. Sigmoid cross-entropy loss is commonly used in classification tasks of  $N$  independent classes with probability between zero and one. Euclidean loss is mostly used on regression problems with real-valued labels (Gomez et al., 2015). The CNN training process is a global optimization process: the best fitting set of parameters are obtained by trying to minimize the loss function (Gu et al., 2017).

Thus, the last pooling layer in the network is usually followed by these layers of fully connected neurons, where the two-dimensional feature map is converted into a one-dimensional feature vector. These layers are the same as in traditional neural networks and hold 90% of the parameters of a CNN (Y. Guo et al., 2016). After the feed forwarding process, the resultant vector could be used for image classification category selection (e.g. Krizhevsky et al., 2012), or it can be used as a feature vector for follow-up processing (e.g. Girshick et al., 2014).

The fact that these layers hold so many parameters comes at the cost of large computational effort for training them. One way to deal with this is by removing the layers altogether, or by decreasing the connections between neurons with some method (Y. Guo et al., 2016). The GoogLeNet model handled this problem by switching these layers from fully connected to sparsely connected, maintaining the computational budget constant, using an Inception module (Szegedy et al., 2015), which is described in the next section.

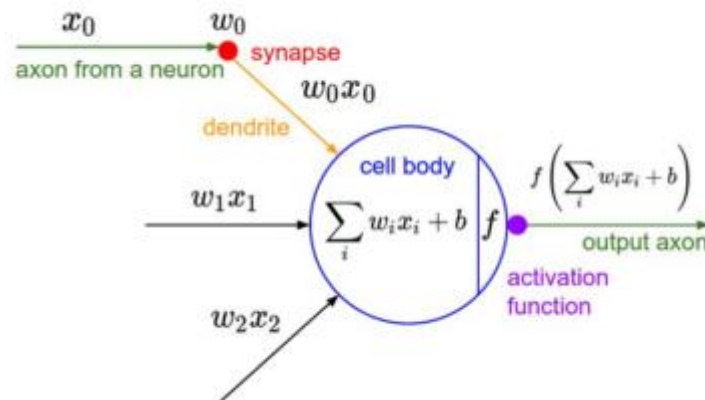


Figure 3 - Mathematical model of a neuron (Cadène et al., 2016)

## **Non-linear Activation Functions**

The activation function is the identity function responsible for the output of real values. The ability for NNs to approximate any functions comes from the non-linear activation functions. In CNNs it is desirable to detect nonlinear features. There are three typical activation functions: the sigmoid, the hyperbolic tangent and the Rectified Linear Unit functions. The sigmoid function (1) takes as an input a real value and crushes it, with its output being between 0 and 1 – when the activation is at its limits, low or high, these regions gradient is almost zero, making the backpropagation stage fail at modifying the parameters of this layer and the preceding ones. The hyperbolic tangent function (2) works similarly to the sigmoid function but the output is between -1 and 1, and has the same problem. The rectified linear unit function or ReLU (3), developed by Nair & Hinton (2010), works differently as its transformation does not crush the input value. Aside from removing all the negative information from the previous layer, making it inadequate to some datasets, ReLU has become quite popular in the last years as it accelerates the convergence of stochastic gradient descent (compared to the previous functions) due to its form being linear and non-saturating. As the mathematical operations are simpler, compared to the other two functions, ReLU is a more reliable and efficient activation function (Cadène et al., 2016). In fact, research revealed that ReLU achieves better performances than those two functions (e.g. Maas et al., 2013).

In the following section the recent advances in some of the CNNs' methods are outlined.

### **2.1.3. Advances in CNNs**

Since the success of AlexNet in 2012, various improvements have been applied to CNNs, as noted by Gu et al. (2017). In the next paragraphs, the most relevant and recent methods of the different layers of CNNs are introduced, which were considered in the development of this study.

#### **2.1.3.1. Convolution layer**

In the convolutional layer, various methods have been suggested to enhance its representation ability, three of which are now outlined. The first (1) is the Network in Network (NIN), a method where the linear filter of the convolutional layer is replaced by a

small network – the original method proposed a multilayer perceptron convolution (mlpconv) – making it possible to better approximate more abstract representations (Lin et al., 2014). The second (2) is the Depthwise Separable Convolution, a method developed by Laurent Sifre while interning at Google Brain with the main purpose of improving the convergence speed and reduction in model size of the AlexNet. This method performs Depthwise convolutions followed by Pointwise convolutions over the input feature maps. The main advantage is the reduction of computation time and number of parameters. Instead of performing convolutions for each kernel over all the channels of the input feature map and summing them (like a traditional convolution) – where each kernel has the same number of channels as the input feature map – this method employs a different method. Firstly it performs Depthwise convolutions over each channel of the input feature maps – the kernel has only one channel – and then performs Pointwise convolutions over its output – which uses a kernel with size  $1 \times 1$  and the number of channels of the output. Compared to a traditional convolution layer, a Depthwise Separable convolution has much less parameters as it performs much less multiplications for the same input and output size. It tends to output comparable results, with just a slight deterioration, but with a sizeable reduction of computation time (Chollet, 2017; Vanhoucke, 2014). Finally (3), the Inception Module, is a method where there is an application of flexible filter sizes to capture varied visual patterns of different sizes and approximation to the optimal sparse structure by the inception module, where one pooling and three types of convolution operations take place. This allows for larger CNNs, increasing the depth and the width, without increasing the computational cost. Compared to AlexNet’s 60 million parameters and ZFNet 75 million, the application of the inception module permits the reduction of the network’s parameters to 5 million (Szegedy et al., 2015). Such method, under the name GoogLeNet, obtained state-of-the-art results in the ImageNet 2014 challenge. The newest inception module CNN proposed by the same author, attempting to find a high-performance network considering the computation cost and inspired by the ResNet architecture (He et al., 2016), combines the inception architecture with shortcut connections. It is called the Inception-V4, and the use of such shortcut connections enables faster training regimens (Szegedy, Ioffe, et al., 2017).

### 2.1.3.2. Pooling layer

In the pooling layer, various methods have been proposed to enhance the provided invariance and lowered computation cost (by reducing the number of connections between convolutional layers through the reduction of the feature maps). Four of these methods are briefly outlined, which could be considered more relevant: (1) Lp Pooling, a method inspired by the biology of complex cells that can provide better generalization than max-pooling (Bruna et al., 2014); (2) Mixed Pooling, a combination of max-pooling and average-pooling that can provide better results than each of them, and can improve the overfitting problems (Yu et al., 2014); (3) Spectral Pooling, a form of pooling where the dimensionality reduction is achieved by cropping the representation of the input in frequency domain. This method can maintain more information than max-pooling for the same output dimensionality, avoiding its sharp reduction like other pooling strategies (Rippel et al., 2015). Finally, (4) Multi-scale Orderless Pooling, a method that tries to improve the invariance of CNNs without the loss of information. The extraction of deep features is done globally and also locally with different scales (Gong et al., 2014).

### 2.1.3.3. Activation function

Various activation functions have been experimented, as a proper implementation can significantly improve the performance of a CNN (Gu et al., 2017). Four of such activation functions are now delineated, all of which achieve successful results. The first, (1) Leaky ReLU is a derivative of ReLU that compresses the negative part, contrary to transforming it to zero, losing less information than ReLU (Maas et al., 2013). A similar function to LReLU, (2) Parametric ReLU, tries to adaptively learn the parameters of the rectifiers, improving accuracy, and the extra computational cost is negligible (He et al., 2015a). (3) Randomized ReLU, another derivative of Leaky ReLU, assigns the parameters of negative parts randomly sampling from a uniform distribution in training, and then fixing its values in testing. The author compared ReLU, LReLU, PReLU and RReLU on an image classification task and concluded that those with a non-zero slope in the negative part in rectified activation units (all of them except ReLU) improved the performance of such task (Xu et al., 2015). Finally, (4) ELU - Exponential Linear Unit, is also a derivative of ReLU where a saturation function is employed for the negative part. It can enable the deep neural networks to train faster and can lead to higher performances (Clevert et al., 2016).

#### 2.1.3.4. Regularization

Regularization methods are used to reduce overfitting in deep CNNs. Of the various methods available, four are subsequently described, considered the most relevant. The first (1), Lp-norm Regularization, a method that alters the objective function (activation or loss) by adding a term that penalizes the model complexity, which can reduce overfitting through the addition of variance (Gu et al., 2017). The second (2), Dropout, proved to be effective in reducing overfitting. The authors applied it to the fully-connected layers, aiming to prevent the network from relying too much on specific neurons, allowing it to be accurate even without full information. (Lu et al., 2016). (3) SpatialDropout, a derivative of Dropout developed to further prevent overfitting which works particularly well if the training data size is small (Tompson et al., 2015). Finally, (4) DropConnect, a method inspired by Dropout that does not change the value of the outputs of the neurons, but instead modifies the weights between them (Wan et al., 2013).

#### 2.1.3.5. Optimization

The loss function at the end of a CNN quantifies the quality of the network with a defined set of weights for the task at hand. The goal of the process of optimization is to find the best set of weights in order to minimize the loss function.

The most commonly used optimization method is the gradient descent. To find the best direction in the weight-space that would minimize the loss function by improving the weight vector, a random search algorithm would be impractical. Instead, one could follow the gradient of the loss function. The gradient of the loss function is a vector of numbers containing the derivatives for each dimension in the input space, the partial derivatives. Following the gradient makes it possible to search in the direction of the best set of parameters, guaranteeing the best possible result in an infinite amount of time. It is possible to compute the gradient either numerically, calculating the partial derivative of the loss function for each dimension by making a small change and noticing how much the function changed, or analytically, using Calculus to derive a direct formula for the gradient. Gradient Descent is the process of repeatedly evaluating the gradient and updating the parameters of the network, following its direction in a defined adaptive or not amount – learning rate – one of the hyperparameter settings in the training phase of a neural network (Karpathy, 2018).

There are several algorithms to optimize the gradient descent mechanisms, counteracting some challenges: difficulty choosing a proper learning rate, applying the same learning rate to all the parameter updates even when the features have very different frequencies, and finally the difficulty of avoiding getting trapped in saddle points - points where one dimension slopes up and another slopes down, usually surrounded by a plateau of the same error as the gradient is very small in all dimensions (Dauphin et al., 2014). (Ruder, 2017a).

The Adaptive Moment Estimation – Adam – is a method that tries to work around these challenges. Combining the advantages of two popular methods, AdaGrad, which performs well with sparse gradients, and RMSProp, which performs well in on-line and non-stationary settings. Adam is forthright to implement, is computationally efficient, has minute memory requirements and performs well on problems with large datasets and large networks (many parameters) (Kingma & Ba, 2017).

#### **2.1.3.6. Structure**

In 2015 Microsoft Research introduced Deep residual networks, a kind of network that uses residual blocks. These networks achieved state-of-the-art results on ImageNet and COCO 2015 competitions, which comprehended image classification, object detection and semantic segmentation.

Network depth is of critical importance in neural networks architectures. In fact, it has been recently demonstrated a mathematical proof revealing the utility of having deeper networks rather than wider networks (Eldan & Shamir, 2016). Motivated by this insight, the residual learning framework was developed. By applying it, it is possible to train deeper networks, leading to improved performances. Residual networks are deeper than their non-residual equivalents, yet they use a similar number of parameters.

Increasing the depth of a regular network leads to a saturated accuracy, which eventually degrades. This degradation is not caused by overfitting, and increasing the number of layers to a capable deep model leads to higher training error. But considering a shallower architecture, there is a solution to a deeper counterpart (with more layers) model by construction: copying the layers from the shallower model, and the added layers are identity mapping. In this case, the deeper model should produce an error not greater than its shallower counterpart. So, the degradation problem is addressed through the use of a deep

residual learning framework, in which each few stacked layers fits a residual mapping (He et al., 2016). The basic framework can be seen in Figure 4.

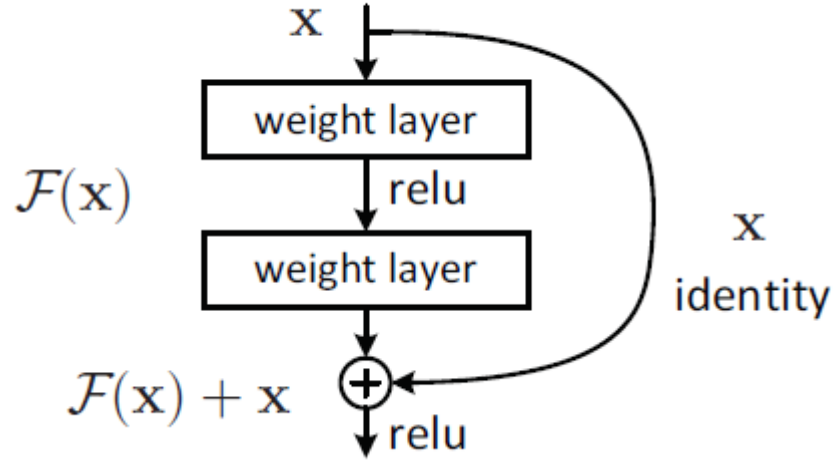


Figure 4 - Residual Connection (He et al., 2016)

Another great advance in the CNN structure is the Global Average Pooling. Typically, a CNN architecture is a sequence of convolution layers followed by pooling layers, concluding in two or more densely connected layers, with the last one having a softmax activation function and a neuron for each class. These last fully connected layers usually contain most of the parameters of the network. This leads to overfitting to the training set, mitigated by the use of regularization methods and datasets as large as possible.

In recent years Global Average Pooling (GAP) was introduced to minimize overfitting by reducing the total number of parameters in the model. This fact has consequences like less computation required and faster models. Similar to max pooling layers, GAP layers are used to reduce the spatial dimensions of a 3D tensor. They take a tensor with size  $h \times w \times d$  and reduce it to  $1 \times 1 \times d$ , by averaging each feature map,  $h \times w$ , to a single number (Fig. 5). GAP can be seen as a structural regularizer, as it explicitly enforces each feature map to be directly connected to a category of the classification task (Lin et al., 2014).

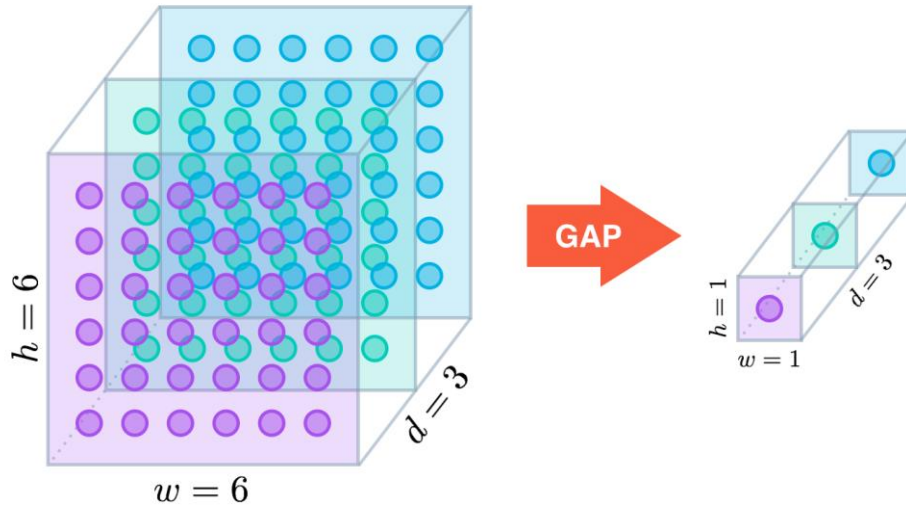


Figure 5 – Global Average Pooling (Cook, 2017)

Another recently proposed mechanism accelerated the training of Deep Neural Networks by as much as fourteen times fewer training steps in a state-of-the-art image classification model, aside from allowing a significantly better accuracy margin. This development, Batch Normalization, works by reducing the internal covariance shift of each layer's inputs. The distribution of each layer's inputs changes as the parameters of the previous layer are updated during the training process: every time the optimization process occurs and updates the parameters of each layer. The changing input distribution is referred as Internal Covariate Shift. This is a problem as it slows down the training process by requiring lower training rates and careful parameter initialization. To counteract it, a mechanism of normalizing each layer's inputs for each training mini-batch – fixating the means and variances of layer inputs – is performed. This process allows for higher learning rates, speeding up the training process, and it acts as regularizer, inducing some noise to each hidden layer, reducing overfitting and reducing the need to use other regularization methods such as Dropout and  $L_2$  weight regularization (Ioffe & Szegedy, 2015).

Having provided a broad description of the CNN process and internal mechanisms, and outlined its most relevant and recent advances, it has been established that the Convolutional Neural Network is the state-of-the-art method in almost every subfield of computer vision. Hence, an overview of the three facial analysis' tasks (based on this method?) is presented in the following section.

## **2.2. Facial Analysis**

In recent years, the status quo of perception science has been challenged. For a long time, biological systems were the only ones with the ability of perception, as it was little understood in its inner workings. With the evolution of Deep Learning systems, however, neural networks are routinely outperforming humans in recognition and estimation tasks (VanRullen, 2017).

In the next three sections, a brief introduction to the tasks of emotion and gender recognition and age estimation through computer vision are outlined, with the most common or classic methods. In every task, evidence is presented to show that CNNs achieve the state-of-the-art results.

### **2.2.1. Gender Recognition**

Recently, computer vision researchers have been giving increased attention to identifying demographic attributes, such as age and gender. These attributes are termed as soft biometrics and have applications in areas such as human-computer interaction, surveillance, biometrics, content-based retrieval, demographics collection and intelligent marketing. Gender recognition by humans achieve accuracies above 95% based only on face images, but it is still a challenging task for computer vision (Ng et al., 2015).

The early methods of gender recognition used many different classifiers to solve the problem, such as Neural Networks (e.g. Golomb et al., 1991), SVM (e.g. Moghaddam & Yang, 2002) and AdaBoost (e.g. Baluja & Rowley, 2007). Recently, the advances in CNN have outperformed every other method, making it the state-of-the-art in most benchmarks (Levi & Hassner, 2015).

### **2.2.2. Age Estimation**

The human age estimation process is not flawless. In fact, it is usually a more complex and challenging task than determining other facial information features, such as identity, expression and gender. Hence, developing automatic machine processes to do so, capable of equaling or even outperforming the human ability to estimate age became an attractive and challenging topic in recent years (Abousaleh et al., 2016).

Predicting the age from a face image has been a complex challenge in computer vision. There are many applications to this task, such as precision advertising, intelligent surveillance, face retrieval and recognition, etc. The variability of the images, the gender differences and other factors, impose subtle but relevant complexity to the age estimation task (Xing et al., 2017).

The classic age estimation methods usually perform two consecutive but independent procedures: given an input face image they start by performing a facial feature extraction, where the objective is to extract invariant facial features representing aging information, and then, in a second phase, perform the age estimation task. The feature extraction phase has been performed with numerous methods, such as the Gabor features (e.g. Gao & Ai, 2009) or the local binary pattern (e.g. Yang & Ai, 2007). Using the extracted features, a wide range of Machine Learning algorithms have been used, such as Support Vector Machines (e.g. Luu et al., 2009). The accuracy of such classic models greatly depends on the developed features and learning algorithms, and many experiments are usually required. But, with the development of CNNs, feature representation and classification models can be developed in an end-to-end framework. Although CNNs have achieved a state-of-the-art state in most computer vision problems, there are few studies on how to develop a high accuracy deep age estimation model (Xing et al., 2017).

### **2.2.3. Emotion Recognition**

Research on computer science has been increasingly focused on the task of emotion recognition. The advent of Human-computer interaction (HCI) fueled this growing interest in the area, which is focused on the interface between computers and human users, where emotions have a significant impact in this interactivity, making its recognition by a computer system a major area of interest. Automated facial emotion recognition can be applied in a wide variety of areas, such as data-driven animation, interactive games, entertainments, humanoid robots, surveillance, crowd analytics, marketing, etc. Though there is extensive research on the area with high-performance achieving methods proposed, it is considered a difficult task due to its complexity and variability (Mehdi Ghayoumi, 2017).

There are two main steps for the task of automatic facial emotion analysis: facial data extraction (feature extraction) and representation (classification). Many models with different classification methods have been proposed for the task of facial emotion recognition, the

last step, such as SVM (support vector machine), BN (Bayesian network), KNN (K-Nearest Neighbors), LDA (Linear Discriminant Analysis) and many others. Deep Learning - using a CNN - the most recently proposed method achieves the best results (Mehdi Ghayoumi, 2017).

The developed models and most databases on emotion recognition use the six key human emotions originally presented in the work of Ekman et al. (1971): anger, disgust, fear, happiness, sadness, surprise, with the addition of a seventh – neutrality (Savoie & Wong, 2017).

As it can be seen, for every task proposed as the objective of the present dissertation, CNNs achieve the state-of-the-art results. Nowadays, a vast number of state-of-the-art results on the most important benchmarks are achieved through the use of a multi-task learning framework applied to the CNN. The framework details and benefits are explained in the following section.

## 2.3. Multi-task Learning Architectures

The vast majority of models developed with several tasks of facial analysis treat them as separate problems. This kind of approach makes the integration of such models inefficient in end-to-end systems (Ranjan, Sankaranarayanan, et al., 2017). Furthermore, developing systems where correlated tasks are simultaneously learned can boost the performance of individual tasks (e.g. Chen et al., 2014; G. Guo & Mu, 2014; Ranjan, Patel, et al., 2017; Xing et al., 2017).

In end-to-end systems, this approach is more efficient time and memory wise – solving all the tasks in the same process and storing a single CNN contrary to storing a CNN for each task (Ranjan, Sankaranarayanan, et al., 2017). The motivations to develop multi-task learning (MTL) systems can be expressed in three ways: biologically, as a baby first learns to recognize faces and then learns to identify other objects with the prior knowledge; pedagogically, as Karate Kid learned first to sand the floor and wax a car as these are invaluable skills to master karate; as a machine learning improvement on learning, as it can be seen as a form of inductive transfer (Ruder, 2017b).

The mostly used MTL architectures in facial analysis tasks are parallel multi-task learning and deeply supervised multi-task learning. The parallel multi-task learning architecture has been extensively used in previous deep learning models (e.g. Ren et al., 2017; Sun et al., 2015;

Z. Zhang et al., 2014). It is a method where the tasks are fused concurrently, at the end of the baseline network. The deeply supervised multi-task architecture fuses the different tasks progressively in each layer, according to their need for more abstract representations (Xing et al., 2017).

### **2.3.1. Deeply supervised MTL architecture**

The All-in-one CNN model developed by Ranjan et al. (2017) is an example of a deeply supervised multi-task learning model implemented with remarkable success. The authors developed a multi-purpose algorithm, where face detection, face alignment, pose estimation, gender recognition, smile detection, age estimation and face recognition are performed simultaneously in a single deep CNN – the first of its kind - using a multi-task learning framework (MTL) which regularizes the parameters of the network. They achieve state-of-art results, using unconstrained datasets, in most of the proposed tasks (Ranjan, Sankaranarayanan, et al., 2017).

The method implemented in this paper has its basis on the MTL framework proposed by Caruana (Caruana, 1997). The lower layers parameters of the CNN are shared between all the tasks, as in this way, the lower layer learn a general and common representation to all the tasks performed, and the upper layers learn more abstract representations for the specific tasks. This approach has the potential to reduce over-fitting in the shared layers, and the process of learning robust features for different tasks makes it possible for the network to learn the correlations between data from different distributions in an effective way. The network has seven convolutional layers, followed by three fully connected ones. The first six convolution layers are used for the other face analysis tasks, sharing the parameters. The different specific tasks of the model are sub-models derived from different layers of the network. These sub-models are divided based on two definitions: whether the specific tasks rely on local – subject-independent - or global – subject-dependent - information of the face. The subject-independent tasks are face detection, key-points localization and visibility, pose estimation and smile prediction, and the subject-dependent tasks are age estimation, gender prediction and face recognition. The sub-model for subject-independent tasks fuse the first, third and fifth convolutional layers, as they rely more on local information, which can be seen as a more fundamental representation, available from the lower layers of the network. After the fused layers, two convolution, a pooling, and a fully-connected layer is added

specifically to these branched-out sub-model. After all these shared layers, a specific fully-connected layer is added to each different task, followed by the output layer. The sub-model for subject-dependent tasks are derived from the sixth convolutional layer, and three fully-connected layers are added to each specific task (Fig. 6) (Ranjan, Sankaranarayanan, et al., 2017).

The training of the All-in-one CNN contains five sub-networks with parameters shared among some of the tasks. Face detection, key-points localization and visibility, and pose estimation are trained as a single sub-model, as the same dataset is used for training them. The MTL approach used in the training of this sub-model is 'Task-based Regularization, where the optimal parameters for each task is obtained by minimizing the weighted sum of loss functions for each different task in the sub-model – where the weights are empirically assigned by the authors. The remaining tasks, smile detection, gender recognition, age estimation and face recognition are trained as separate sub-networks, as there is not a single large dataset with information regarding each task. The MTL approach in this case is Domain-based Regularization. With this method, the training is performed as if each model were a different CNN, but sharing the weights and updating them for each task. Doing so, the weights tend to adapt to the complete set of domains, instead of fitting to a task-specific domain, reducing over-fitting and making the model better at generalization (Ranjan, Sankaranarayanan, et al., 2017).

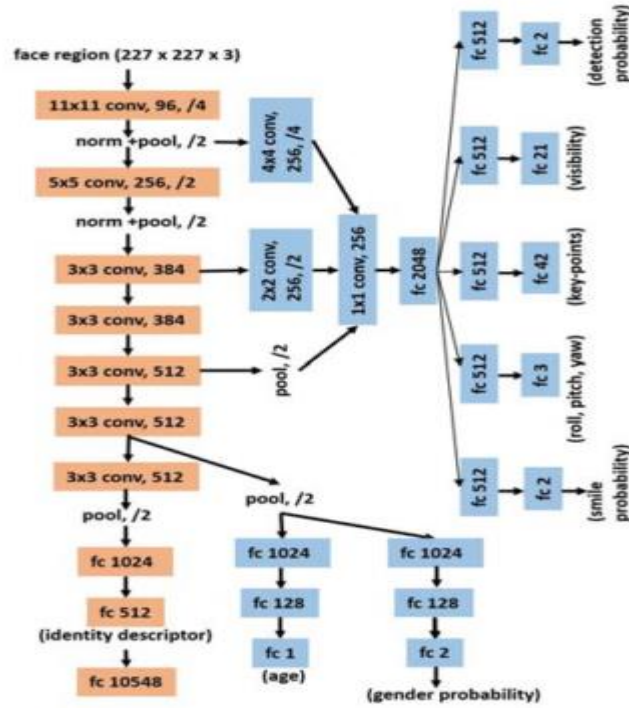


Figure 6 - Deeply supersized MTL architecture (Ranjan et al., 2017)

### 2.3.2. Parallel MTL architecture

In the work of Xing et al. (2017), one of the proposed models has a parallel multi-task learning architecture. The proposed method sought to develop an age estimation model. During the development of the model, the authors proposed the incorporation of gender and race recognition in a multi-task learning framework as a means to further improve the age prediction task. By accumulating the insights of the proposed tasks, they obtained a very deep age estimation model that outperformed all the previous methods by a large margin (Xing et al., 2017).

They started with a baseline architecture based on AlexNet (Krizhevsky et al., 2012), which uses large convolution kernel and stride in the first layers, obtaining large feature maps, and gradually reducing its size as the layer goes deeper. The architecture has three convolutional layers, all of them consecutive with a max pooling layer, followed by two fully-connected layers (Fig. 7). The activation used is ReLu – Rectified Linear Units.

Having described the two most commonly used MTL approaches and outlined the main concepts and methods considered in the development of this study's model with this literature review, a final section briefly presents a commercially available model.

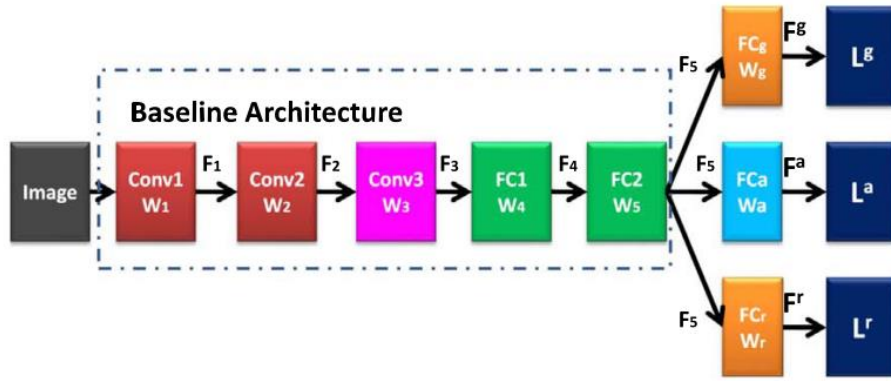


Figure 7 - Parallel MTL architecture (Xing et al., 2017)

## 2.4. DAGER – a commercial approach

DAGER - Deep Age, Gender and Emotion Recognition, is a model developed by Dehghan et al. (2017). This is the only developed model with a somewhat similar output to the present dissertation. The model's research paper describes the results of an end-to-end system developed for commercial use. Their system consists of several deep CNNs with the output tasks of Age, Gender and Emotion recognition on unconstrained real-world images (Fig. 8), without the use of any MTL approach. The CNNs internal mechanics, design, concepts and functions of choice are not shared, as this is a commercial software available through API calls to Sighthound's server, with a cost up to 5000\$ a month. They obtain state-of-the-art results on every task compared to other commercially available models, but the comparison is mostly done on their one proprietary dataset. The training data is proprietary and not available.

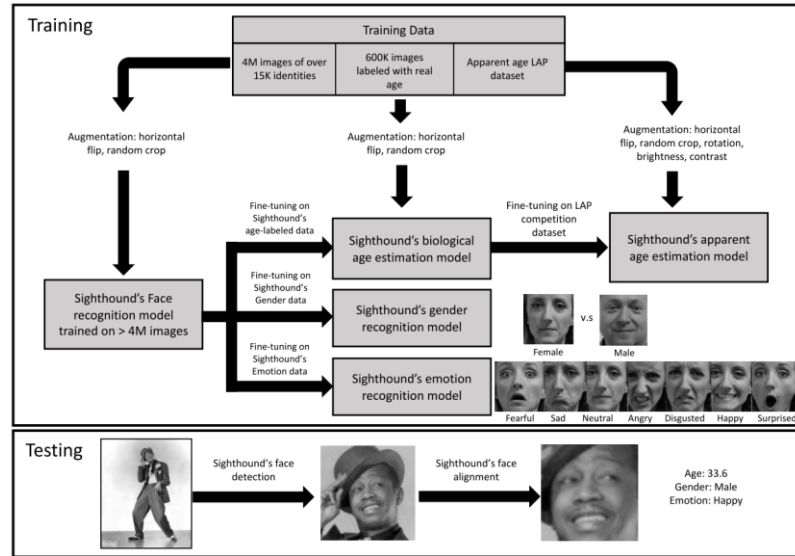


Figure 8 - DAGER framework (Dehghan et al., 2017)

## Chapter 3 – Methodology

After a complete review of the main concepts and techniques which were considered in the course of this study, the model proposition is outlined in this chapter. First, the model framework is described in respect to the conception of all the tasks, followed by the design of the network. Furthermore, the datasets used are outlined, with a description of the chosen split scheme and image processing. Lastly, the software for the implementation of this dissertation is discussed.

### 3.1. Task Formulation

During the next chapters, the methodology proposition for each task is outlined and explained. There are several ways to formulate each task, and several ways to design the model architecture. The final choice of model formulation for each task has a direct implication on the output of the network and can too have a direct implication on the model performance as some activation and loss functions are optimized for such tasks. First, a problem formulation for each task is given, explaining each choice, and then the score and loss functions are proposed. The choice of the baseline model architecture is, then, presented and described, as well as each single-task learning and multi-task learning frameworks.

#### 3.1.1. Emotion recognition

The emotion recognition task is performed as a multi-class classification problem. As reviewed in the literature section, most databases incorporate seven distinct emotion classes: fear, anger, surprise, happiness, sadness, disgust and neutrality.

In the output layer, a loss function is needed as the training objective, to update the weights during the backpropagation stage. The most commonly used loss function deployed to train a multi-class mutually-exclusive problem is the multi-class cross-entropy loss. It indicates the distance between the output distribution. According to (Ranjan, Sankaranarayanan, et al., 2017), the cross-entropy loss is defined by equation 1:

$$\mathcal{L}_{Emotion} = \sum_{c=0}^{N-1} -y_c \cdot \log(p_c) \quad (1)$$

Where  $N$  is the number of classes,  $y_c$  is the ground label of the sample, where  $y_c = 1$  if the sample belongs to emotion  $c$ , otherwise 0, and at last  $p_c$  is the predicted probability that the sample belongs to emotion  $c$ . In this particular case, the equation (2) is modified.

$$\mathcal{L}_{Emotion} = \sum_{c=0}^6 -y_c \cdot \log(p_c) \quad (2)$$

Knowing that the ground truth label of the sample  $y$  can only be one of the seven emotions (it is not a probability distribution), the other classes can be ignored and one can just use the matching term from the estimates  $\hat{y}$  (equation 3):

$$\mathcal{L}_{Emotion} = -y_{emotion} \cdot \log(\widehat{y_{emotion}}) \quad (3)$$

Applying this to a determined number of samples  $N$ , as the input is a mini-batch (discussed in the next chapter), the loss is the average of all the input samples (equation 4):

$$\mathcal{L}_{Emotion} = -\frac{1}{N} \sum_{i=1}^N y_{emotion_i} \cdot \log(\hat{y}_i) \quad (4)$$

The evaluation metrics used in this problem are accuracy, precision, recall and F1 score. Accuracy is used as it is the most common used metric in the development of classification problem CNNs, usually being the only model comparison method (in classification competitions such as Kaggle’s Facial Expression Recognition Challenge). Precision and recall are used to better evaluate the models, searching for biased accuracy results, and the F1 score, a function of precision and recall, which balances both results and is used with uneven class distributions.

### 3.1.2. Age estimation

The age estimation task can be drawn in three different formulations: as a multi-class classification problem, as a regression problem and as an ordinal regression problem. In the work of Xing et al. (2017), a comparison between these formulations is performed on the Morph II dataset using the same baseline architecture, and the best results are obtained treating the task as a regression problem. Aside from these results, treating the task as a regression problem is somewhat more natural, as the age of a human is measured in time after birth, therefore it is a real valued variable. The output layer of the task is composed of

only one neuron, contrary to the set of neurons needed if the problem was treated differently. Thus, the task of age estimation is treated as a regression problem.

The most used loss function in age estimation regression problems is the Mean Squared Error (MSE) –loss which is defined by equation 5:

$$\mathcal{L}_{Age} = (\widehat{y_{age}} - y_{age})^2 \quad (5)$$

Where  $N$  is the number of samples,  $\widehat{y_{age}}$  is the prediction and  $y_{age}$  is the ground truth label.

Applying this to  $N$  number of samples, the equation is 6:

$$\mathcal{L}_{Age} = \frac{1}{N} \sum_{i=1}^N (\widehat{y_{age}_i} - y_{age_i})^2 \quad (6)$$

Where  $\widehat{y_{age}_i}$  is the prediction and  $y_{age_i}$  is the ground truth for the  $i^{th}$  sample.

The evaluation metric used for this task is the Mean Absolute Error (MAE) metric, the most commonly used for regression problems and this task in particular.

### 3.1.3. Gender recognition

The task of gender recognition, as the task of emotion recognition, is a multi-class problem. There are two mutually exclusive classes – male or female. The loss function considered in this task is the same as in the emotion recognition task, the cross-entropy loss. Following the same basis described in the emotion recognition task, the loss is defined by the following equation:

$$\mathcal{L}_{Gender} = -\frac{1}{N} \sum_{i=1}^N y_{gender_i} \cdot \log(\widehat{y_{gender}_i}) \quad (7)$$

Where  $N$  is the number of samples,  $\widehat{y_i}$  is the prediction and  $y_i$  is the ground truth for the  $i^{th}$  sample.

The evaluation metrics used in this problem are accuracy, precision, recall and F1 score, with the same motivation as explained in the emotion recognition task.

## 3.2. Databases

For the tasks of age estimation and gender recognition the same dataset is used - Morph II database (“MORPH,” 2018). It contains about 55000 face images of more than 13000 subjects, it is a multi-ethnic dataset and has metadata on age, gender and race of the subject. 77% of the dataset is composed of black faces, 19% white faces, and the remaining 4% include Hispanic, Indian, Asian, etc. Ages range from 16 to 77 with a median age of 33 and there are 8489 female faces (15%) and 46645 male faces (85%). The images resolution is either 200x240 or 400x480, and the color coding is RGB (Ricanek & Tesafaye, 2006).

For the task of emotion recognition, Kaggle’s FER-2013 dataset (“FER-2013,” 2018) is used. FER-2013 - Facial Expression Recognition Challenge of 2013 - contains about 36000 face images labelled with the seven key emotions (Table 1). The images resolution is 48x48, and they are grayscale.

| EMOTION  | NUMBER OF FACES (%) |
|----------|---------------------|
| ANGRY    | 4953 (14)           |
| DISGUST  | 547 (2)             |
| FEAR     | 5121 (14)           |
| HAPPY    | 8989 (25)           |
| SAD      | 6077 (17)           |
| SURPRISE | 4002 (11)           |
| NEUTRAL  | 6198 (17)           |
|          | 35887 (100)         |

*Table 1 – Emotion dataset class distribution*

### 3.2.1. Image Pre-processing

To build the models, 35000 images from each dataset were selected, so there are 35000 images labeled with age and gender data, and 35000 images labeled with emotion data. The samples were divided into three groups: train, validation and test sets. The percentage of each group is, accordingly, 70%, 20% and 10% - 24500 train images, 7000 validation images and 3500 test images.

The gender labels are not well balanced, there are much more male than female labels (Table 2). The distribution across splits is approximately maintained (Fig. 9).

|        | TRAIN SPLIT | VALIDATION SPLIT | TEST SPLIT |
|--------|-------------|------------------|------------|
| MALE   | 20710       | 5894             | 2953       |
| FEMALE | 3790        | 1106             | 547        |

Table 2 – Gender dataset train-validation-test split

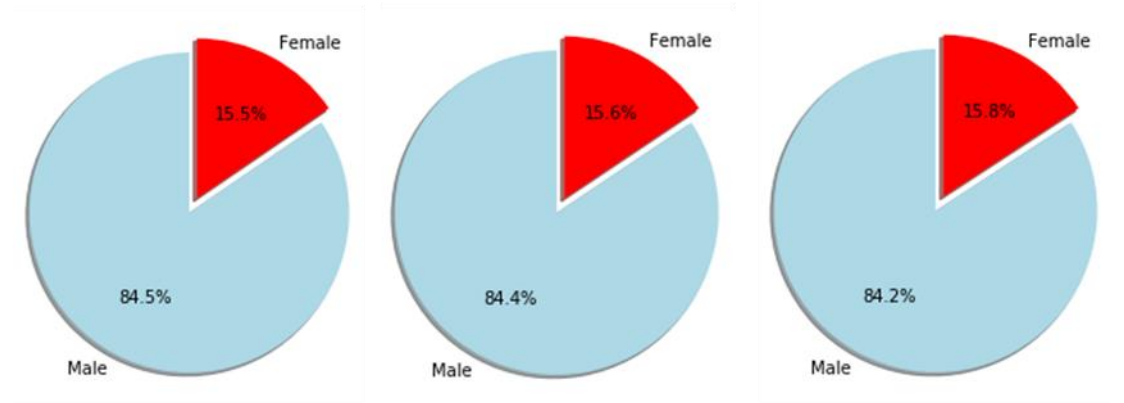


Figure 9 - Gender dataset train-validation-test split

The emotion classes are somewhat more balanced, but there are still two apart classes: Happy with the most samples, and Disgust with very few (Table 3). The distribution can be viewed in Figure 10.

|          | TRAIN SPLIT | VALIDATION SPLIT | TEST SPLIT |
|----------|-------------|------------------|------------|
| ANGRY    | 3360        | 979              | 474        |
| DISGUST  | 373         | 105              | 59         |
| FEAR     | 3488        | 1012             | 493        |
| HAPPY    | 6235        | 1708             | 857        |
| SAD      | 4088        | 1211             | 621        |
| SURPRISE | 2734        | 775              | 381        |
| NEUTRAL  | 4222        | 1210             | 615        |

Table 3 - Emotion dataset train-validation-test split

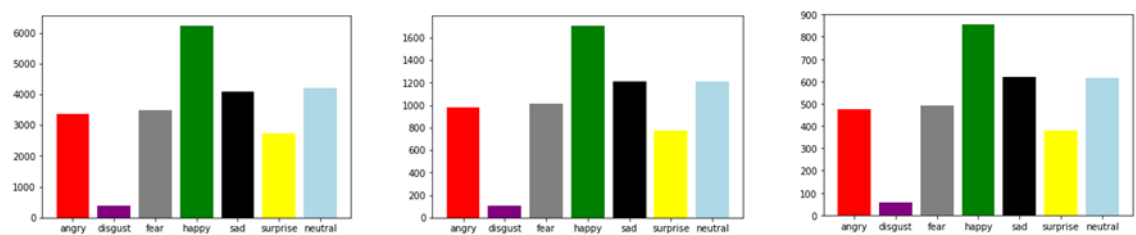


Figure 10 – Emotion dataset train-validation-test split

The age dataset distribution is not constant, with most samples in the interval 20-50 years old. The age distribution per split can be seen in Figure 11.

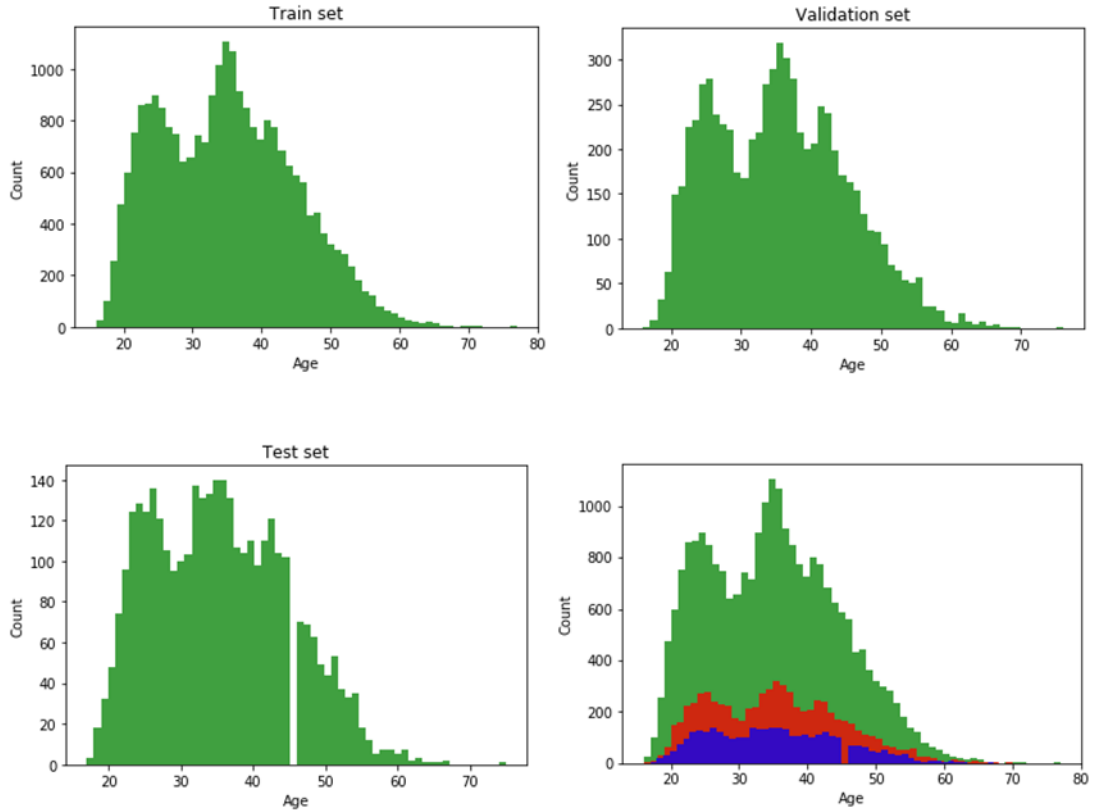


Figure 11 - Age Distribution for the training set, validation set, test set and the merged sets

The Age and Gender dataset was resized to 48x48 in order to share the input resolution of the FER-2013 dataset, and it was converted to grayscale as well. There is an incentive on using such small images and grayscale - to be able to train the models hastily.

Having described the dataset and image pre-processing, the model design is now presented, explaining the baseline architecture and each built framework.

### 3.3. Model Design

In this chapter, the baseline network architecture is described and justified, followed by the newly created network for each developed model of the present dissertation. Upon defining each network, the training procedure of the single-task and the multi-task models is outlined and explained, with its choice of parameters, hyperparameters and data handling.

### 3.3.1. Baseline Network Architecture

It has been recently shown that jointly learning correlated tasks boosts the performance of individual tasks. The explanation for this and some examples are outlined during the chapter two of this dissertation. Consequently, a parallel multi-task learning architecture was developed on top of a baseline model architecture.

Very deep CNNs have been successful in multiple computer vision tasks. There are a number of well-established network architectures, like the VGGNet-16 (Simonyan & Zisserman, 2015), the ResNet-50 (He et al., 2016), the GoogLeNet (Szegedy et al., 2015) and Inception-V4 (Szegedy, Liu, et al., 2017) among others. Some of these can be seen in Figure 12.

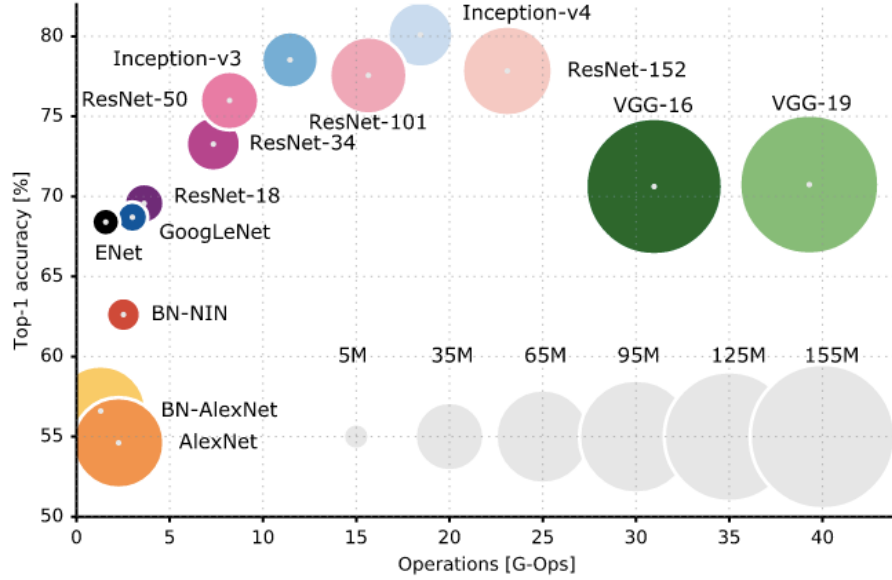


Figure 12 - Framework Comparison (Operations per Accuracy) (Canziani et al., 2017)

The baseline architecture used in this work is a variant of the Xception (Chollet, 2017) architecture with far fewer parameters and less depth, called mini-Xception (Arriaga et al., 2017).

The Extreme Inception architecture, dubbed “Xception” by its creator, is a linear stack of Depthwise Separable Convolution layers with residual connections, operating in repeated modules. The framework has 36 convolutional layers, the baseline for the feature extraction, organized in 14 modules with residual connections around them except for the first and last one (Chollet, 2017).

A number of developments and past work made this architecture possible: the VGG-16 architecture influenced the schematics of the Xception architecture, which is similar; the advantages of factoring convolutions into multiple branches successively on channels and then on space was taken from the Inception architectures; the development of depthwise separable convolutions, which is the main subject on this architecture; and the Residual connections, extensively used in the Xception architecture (Chollet, 2017). This architecture is simple to implement and modify on high-level deep learning libraries.

The mini-Xception derivative architecture was developed with the objective of achieving the best accuracy over number of parameters ratio. Reducing the number of parameters has two main advantages: reducing the processing power needed to run the network (this architecture was developed to use in robots), and fewer parameters yield a better generalization under Occam’s razor framework – the network is simpler (Arriaga et al., 2017).

This architecture is inspired by Xception, combining the use of residual modules and depthwise separable convolutions. Furthermore, it eliminates the last fully-connected layers in order to reduce the number of parameters, through the usage of a Global Average Pooling layer (Arriaga et al., 2017). All the methods and layers used in this framework are explained during the Recent Advances section in the Literature Review.

The motivation to choose this baseline architecture is simple: achieving good results without the need for great processing power and time. This allows the present work to be implemented in a live feed setting (video), which requires a quick and computationally inexpensive model. The mini-Xception main development purpose was exactly this: to be small, inexpensive, and efficient. Furthermore, it is a small enough model to train speedily, making it possible to test on a bigger number of settings and hypothesis.

The chosen baseline architecture, the mini-Xception framework, comprehends four residual depthwise separable convolution modules. Every convolution is followed by a Batch Normalization operation and a ReLU activation function. Lastly, a global average pooling operation followed by a softmax activation function are implemented to output a prediction (Figure 13).

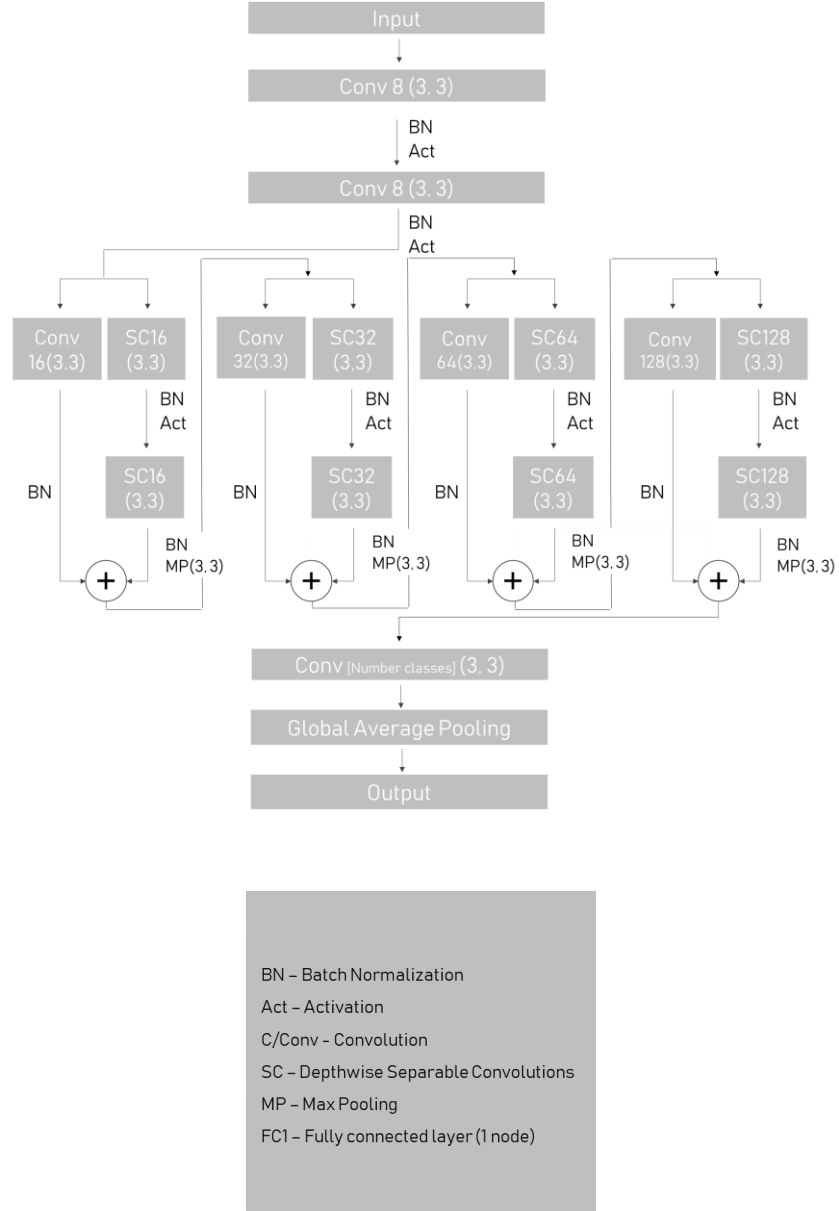


Figure 13 – Baseline Architecture

### 3.3.2. Training procedure

A model for each solo task and some models for the three tasks were trained. This rationale makes it possible to compare the multi-task learning models' performance on each task with each single-task model. In the next sections, the description of each model – single-task and multi-task – is outlined, along with the training parameters and chosen systems.

### 3.3.2.1 Single-task Training

#### Architectures

After defining the baseline architecture, a concrete framework was defined for every task. The difference for each model appears in the last layers: the number of filters in the last convolution layer, and the output layer.

For the emotion recognition task, the last convolutional layer has seven filters, as this is the number of possible outputs, succeeded by a global average pooling layer. Finally, an output layer where a softmax activation takes place. The model can be seen in Figure 14.

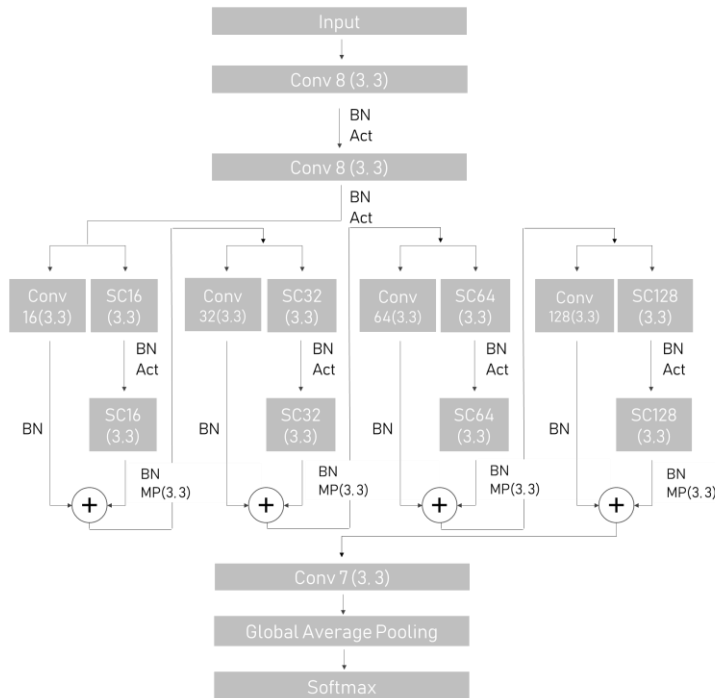


Figure 14 - Emotion Recognition Model

The task of age estimation is a little different as it is treated as a regression problem. This task has a last convolution layer with only one filter, succeeded by a global average pooling layer and a dense layer of only one neuron yielding the results. The model is seen in Figure 15.

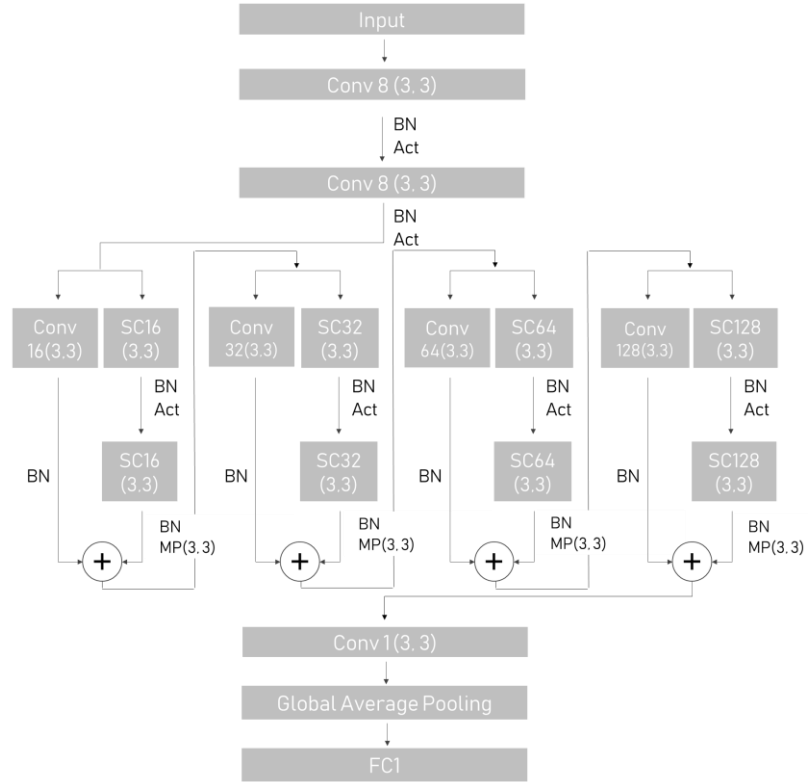


Figure 15 - Age Estimation Model

The task of gender recognition is similar to the emotion recognition task, but the last convolution has only two filters, as can be seen in Figure 16.

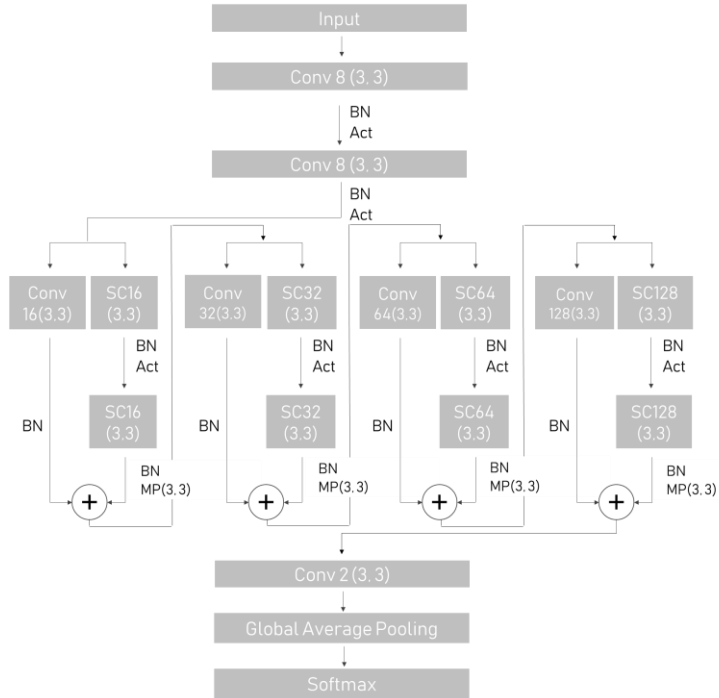


Figure 16 - Gender Recognition Model

### Model training – parameters, procedures and data handling

Except for the choice of the optimizer used in the training process, every other parameter and configuration was chosen after a phase of searching and testing, with different values for each parameter. The final parameters and handling choices are as follows:

|                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| The models are trained using a batch of size 32.                                                                                                                                                                                                                                                                                                                                                                                             |
| The initial random weights of each layer are set drawing samples from a truncated normal distribution, with center 0 and standard deviation $= \sqrt{\frac{2}{inputU}}$ , where inputU is the number of input units in the weight tensor. This is commonly known as He normal initial random weights.                                                                                                                                        |
| Adam is the chosen optimizer, the same used by the creator of the baseline network, using the same initial learning rate 0,001 and the default beta 1 and 2 parameters provided in the original study (Kingma & Ba, 2017). <ul style="list-style-type: none"><li>– The learning rate was reduced by a factor of 0.5 for every 20 epochs without improvement on the validation set loss;</li></ul>                                            |
| The stop of the training process relied on a patience of 100 epochs without improvement on the validation set loss.                                                                                                                                                                                                                                                                                                                          |
| Both datasets were rescaled (each byte representing a pixel was divided by 255, so the final pixel map ranges from 0 to 1).                                                                                                                                                                                                                                                                                                                  |
| Data augmentation was used to provide changeability to the input images: <ul style="list-style-type: none"><li>– Random rotation with a range up to 15 degrees;</li><li>– Random width shift with a range up to 15% (of the width of the image);</li><li>– Random height shift with a range up to 15% (of the height of the image);</li><li>– Random zoom with a range up to 10%;</li><li>– Possible horizontal flip (50% chance).</li></ul> |
| L2 regularization and dropout were used in the preliminary stages of the development of the models, but later disregarded.                                                                                                                                                                                                                                                                                                                   |
| The emotion task is trained with and without class weights to balance the classes distribution. The weights are presented in Table 4.                                                                                                                                                                                                                                                                                                        |
| The gender task is trained with and without class weights to balance the male-female ratio (Table 5)                                                                                                                                                                                                                                                                                                                                         |

|              | ANGER   | DISGUST | FEAR    | HAPPY    | SAD      | SURPRISE | NEUTRAL  |
|--------------|---------|---------|---------|----------|----------|----------|----------|
| CLASS WEIGHT | 1.04167 | 9.38338 | 1.00344 | 0.561347 | 0.856164 | 1.28018  | 0.828991 |

Table 4 - Emotion training class weights

|              | MALE     | FEMALE  |
|--------------|----------|---------|
| CLASS WEIGHT | 0.591502 | 3.23219 |

Table 5 - Gender training class weights

### 3.3.2.1. Multi-task Training

#### Architectures

The multi-task training for the three different tasks is performed using two different methods, since there is no available dataset with metadata for all the tasks. The work of Ranjan et al. (2017) explains the two different methods: Task-based Regularization and Domain-based Regularization. The prior method consists of a way to find the best parameters during the training of the network minimizing the weighted sum of the loss functions for each task. This method is used for the tasks of gender recognition and age estimation, as the MORPH dataset contains metadata for both tasks. The aggregating loss function is defined in equation 8:

$$\mathcal{L}(X) = \alpha L^{gender} + \beta L^{age} \quad (8)$$

Where  $L^{gender}$  is the value of the loss function used for gender recognition,  $L^{age}$  is the value of the loss function used for age estimation,  $\alpha$  and  $\beta$  are hyper-parameters to tune the importance of each task.

The latter method, Domain-based Regularization, is used for the task of emotion recognition, as it is trained with a dataset with only that particular metadata. The approach is that of training multiple CNNs with respect to task-related datasets but sharing the parameters between such tasks. The shared parameters adapt to the complete set of tasks instead of fitting to a task-specific domain, inducing a better generalization. It has the advantage of increasing the total number of training samples for the whole network to 70000.

The complete training is performed simultaneously for the three tasks, batch by batch: for each epoch the network is trained sequentially in batches of images containing data for age and gender first, and emotion next. Accordingly, a batch of images labeled with both age and gender metadata is fed to the network, a weight update is performed minimizing the aggregating loss function described above. Then, a batch of images labeled with emotion

metadata is fed, updating the weights based on this batch. This process occurs until the end of every batch of data (which has the same number on both datasets, as the number of images is equal). The weight update over each batch logically only targets the specific parameters of the task being trained, which can be in a shared layer and/or not (Figures 17 to 20). Thus, the parameters of the shared layers of the network are always updated for every batch, and the specific ones are updated in alternating batches, a method developed specifically for the present work.

Four different networks were developed: a triple-module sharing network – V1 (Fig. 17), a double-module sharing network – V2 (Fig.18), a single-module sharing network – V3 (Fig.19) and a zero-module sharing network – V4 (Fig.20).

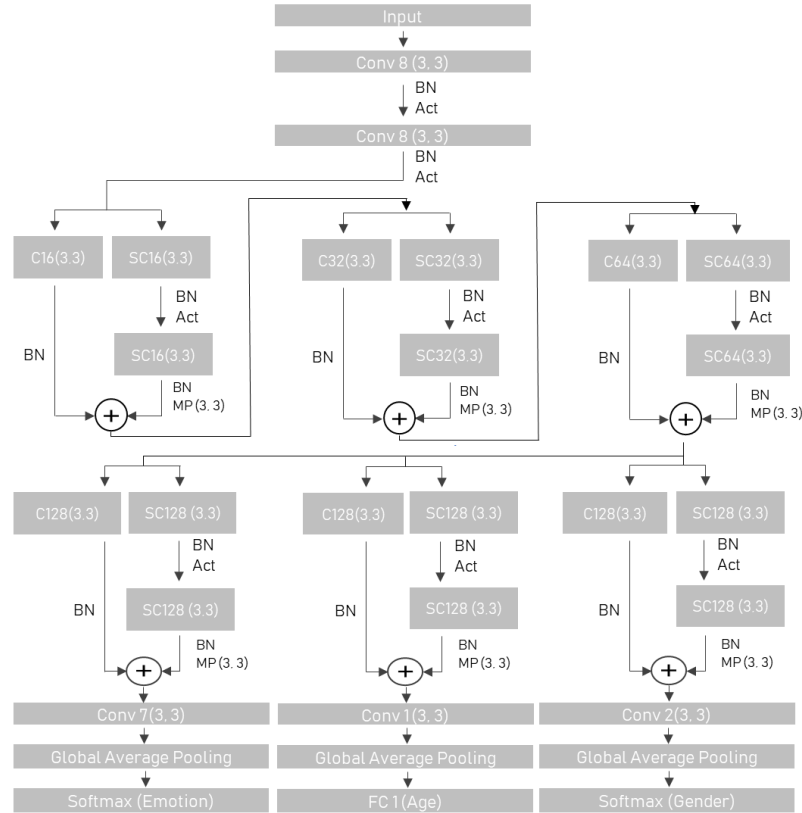


Figure 17 - V1 model architecture



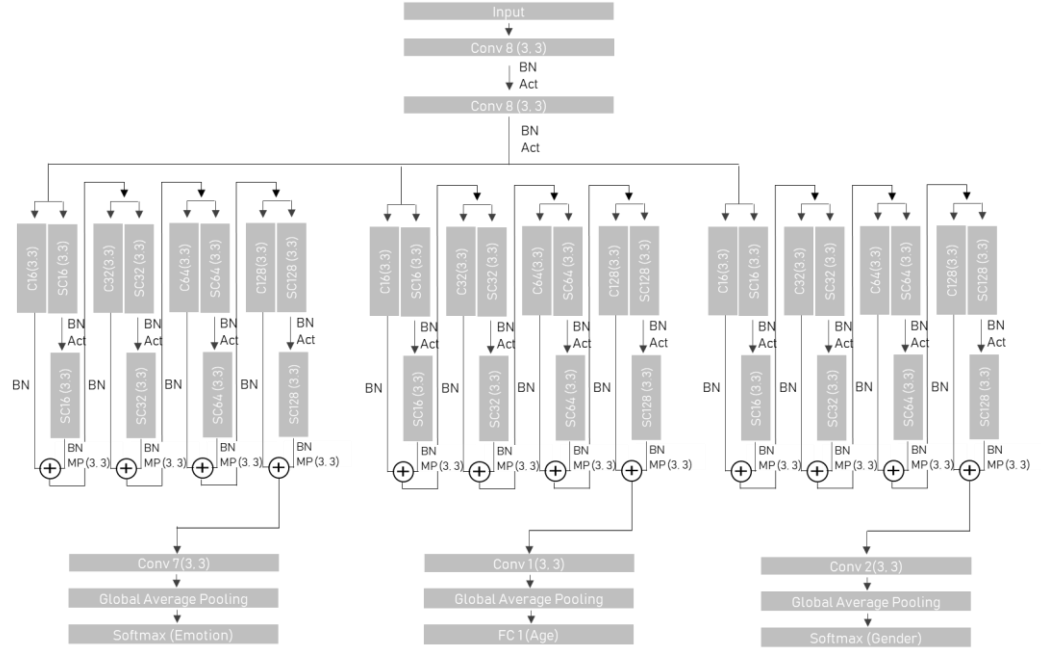


Figure 20 - V4 model architecture

### Model training – parameters, procedures and data handling

After experimentation, the best results were obtained with the aggregating loss function with the following weights:

$$\mathcal{L}(X) = L_{gender} + 0.01L_{age} \quad (9)$$

During the training phase, the total loss achieved for the three tasks is computed, but does not interfere with the weight updating, as the tasks of age and gender recognition are not trained simultaneously with the emotion task. The computation of the total loss is done to better visualize the performance evolution of the complete model, and to decay the learning rate of the optimizer and stop the learning process, discussed subsequently. The formulation of the Total loss is shown in equation 10:

$$\mathcal{L}_{Total}(X) = L_{gender} + 0.01L_{age} + L_{emotion} \quad (10)$$

The same approach was used as the single-task models training which can be seen in the previous section. The only difference is the decay of the learning rate and the stop of the learning process, which occurs based on the computed Total Loss.

### 3.4. Software

The present work was developed using Python programming language (“Python,” 2018), version 3.6. The following packages are mainly used: Keras (“Keras,” 2018), TensorFlow (“TensorFlow,” 2018), Scikit-learn (“scikit-learn,” 2018), NumPy (“NumPy,” 2018), Pandas (“pandas,” 2018) and Matplotlib (“Matplotlib,” 2018).

Keras is a high-level neural networks API capable of running on top of TensorFlow, CNTK or Theano, which allows easy and fast prototyping. TensorFlow is an open source library for numerical computation using data flow graphs, developed with the purpose of conducting machine learning and deep neural networks research. Scikit-learn is a package focused on simple functions and procedures for machine learning, data mining and data analysis. NumPy is a scientific package for Python, designed to deal with array objects, linear algebra capabilities, etc. It is extensively used as a container of generic data and is one of the most used packages in Python programming. Pandas is a library specialized in data structures and data analysis. Matplotlib is a package focused on 2D plotting.

## Chapter 4 – Results and Evaluation

After discussing the methodology used in this thesis, where the models, learning framework, parameter choices, data and software were explained, the results of the training and testing phases for each model are outlined in this chapter. First, the results for all the single task models are shown and explained, followed by the results and evaluation of the multi-task models. The results are characterized by the metrics and loss learning curves in the training phase, and by the confusion matrices and evaluation metrics task by task and overall in the testing phase. The Precision, Recall and F1 score are computed using the Macro-average method.

At the end of the chapter, a model comparison is performed, evaluating each model strengths and weaknesses, and comparing task by task every model in respect to their performances measured in different ways.

### 4.1. Single-task Models

In this section the results and evaluation for the single-task models are described, with different evaluation metrics compared and explained.

#### 4.1.1. Emotion Recognition

Two models are shown, one with class weights and one without (for reference to class weights, see 3.3.2. Training Procedure). The training phase results provided are the loss and metrics learning curves, where it is possible to see the evolution of the results on the training set and validation set for every trained epoch. The same process is performed for every model.

The testing phase provides the results on the test set, with two confusion matrices, one normalized and one absolute. The normalized confusion matrix is obtained through the normalization of the absolute numbers by class support size (number of elements in each class). This normalized confusion matrix is provided to aid the visual interpretation, as the classes are not well balanced, and with the normalized values it is easier to see which classes are better classified. Evaluation metrics are provided per class and overall. The same process

is repeated for the gender recognition model, but for fewer classes as there are only two (male/female).

#### 4.1.1.1. Class Unweighted

##### Training phase

Observing Figures 21 and 22, it is possible to conclude that either the loss and the accuracy measures tend to drop, meaning the network training is successful. The validation loss stalls at about the 100<sup>th</sup> epoch, and overfitting starts to happen. This is expected to occur, as it is defined the training of the network stops if there are 100 consequent epochs without improvements on the validation loss, and for every 20 epochs there is a drop in the learning rate. This is performed in order to thoroughly search the solution space, and the model with the best achieved validation loss is the one used during the testing phase.

The training lasted 182 epochs, and the model with the lowest validation loss was achieved in the 82<sup>nd</sup> epoch. This model reached 0.670 accuracy and 0.882 cross-entropy loss in the training set, and 0.630 and 0.996 in the test set.

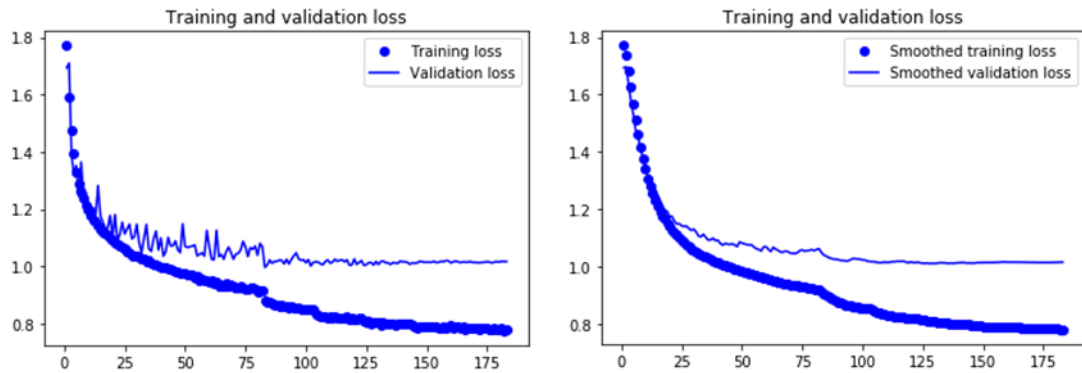


Figure 21 - Training and validation loss (real and smoothed)

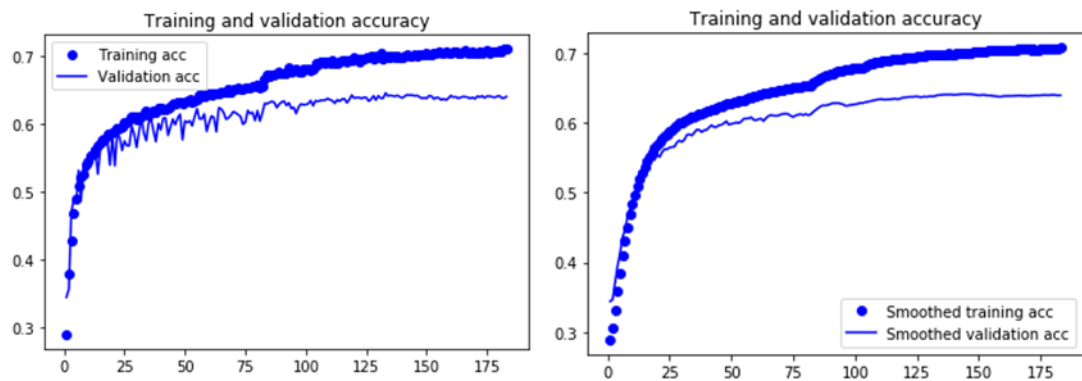


Figure 22 - Training and validation metrics (real and smoothed)

## Testing phase

Looking at the confusion matrix (Fig. 23), the model is capable of distinguishing well between classes, except for the second class – disgust. There are more individuals predicted as angry – 25 out of 59 (42%) – than those correctly predicted as disgusted – 16 out of 59 (27%). This was expected, as this class has fewer elements, and this model does not balance the classes through weighting.

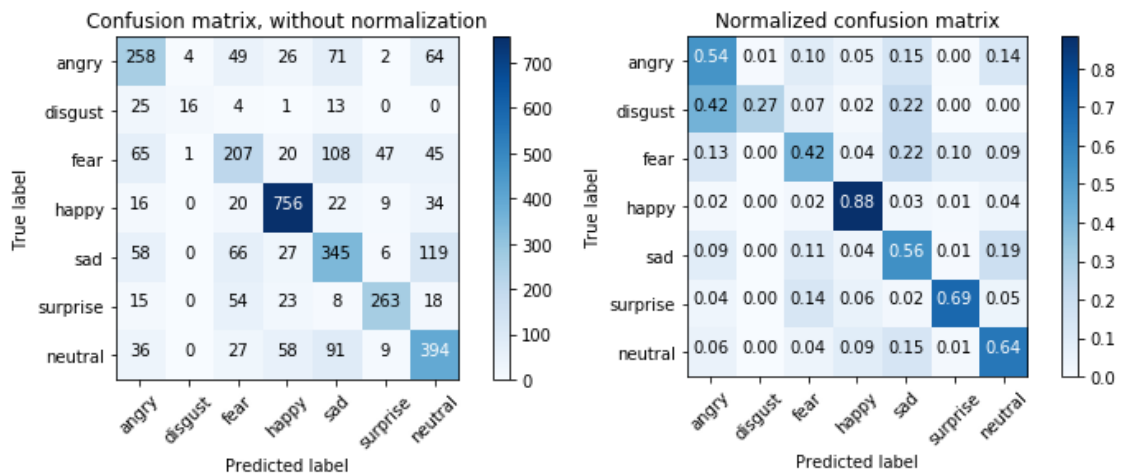


Figure 23 – Confusion matrix

Examining the evaluation metrics class by class (Table 6) it is possible to see the classes Happy and Surprise are the most precise and the same classes have the highest recall. Turning to the class disgust, it has relatively great precision and the lowest recall measure. This means that for every sample predicted as Disgust, 76% were correctly labeled – high precision - and out of all the samples with the ground truth label Disgust, only 27% were predicted as such – low recall. The F1 measure, a function of precision and recall, yields the same conclusions, Disgust achieving the lowest score and Happy achieving the highest.

|          | PRECISION | RECALL | F1     |
|----------|-----------|--------|--------|
| ANGRY    | 0.5455    | 0.5443 | 0.5449 |
| DISGUST  | 0.7619    | 0.2712 | 0.4000 |
| FEAR     | 0.4848    | 0.4199 | 0.4500 |
| HAPPY    | 0.8299    | 0.8821 | 0.8552 |
| SAD      | 0.5243    | 0.5556 | 0.5395 |
| SURPRISE | 0.7827    | 0.6903 | 0.7336 |
| NEUTRAL  | 0.5846    | 0.6407 | 0.6114 |

Table 6 – Class metrics

The overall and weighted results (Table 7) show similar values for the metrics, with a difference worth reporting: the overall recall is somewhat lower than the weighted version, easily explained by the weight of the class Disgust, which is the lowest weight and recall value.

|                 | ACCURACY | PRECISION | RECALL | F1     |
|-----------------|----------|-----------|--------|--------|
| <b>OVERALL</b>  | 0.6397   | 0.6448    | 0.5720 | 0.5907 |
| <b>WEIGHTED</b> | -        | 0.6392    | 0.6397 | 0.6363 |

Table 7 – Model metrics

The evaluation on the test set yielded a 0.6397 accuracy and a 0.9782 cross-entropy loss. The evaluation took approximately  $9,03 \times 10^{-4}$  seconds per image.

#### 4.1.1.2. Class weighted

##### Training phase

Similar results for the weighted version can be seen in the loss and accuracy learning curves (Figures 24 and 25). The validation loss stalls at about the 150<sup>th</sup> epoch.

The training lasted 286 epochs, and the model with the lowest validation loss was achieved in the 186<sup>th</sup> epoch. This model reached 0.679 accuracy and 0.783 cross-entropy loss in the training set, and 0.632 and 1.016 in the test set.

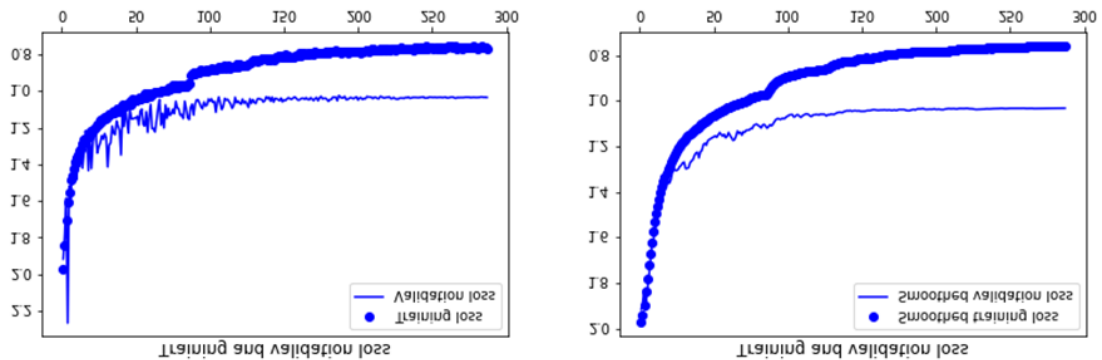


Figure 24 - Training and validation loss (real and smoothed)

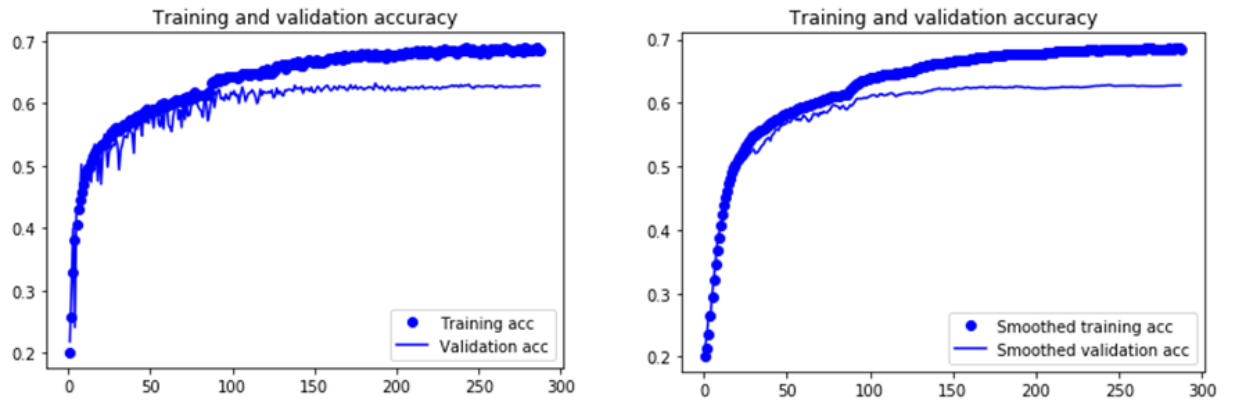


Figure 25 - Training and validation metrics (real and smoothed)

## Testing phase

The confusion matrix (Fig. 26) yields distinguishable classes, this time capable of distinguishing the class Disgust, compared to the unweighted version. The class Happy has the greatest number of individuals correctly predicted. It is curious to see that 99 individuals with the ground truth label Fear are predicted as Sad (about half of those correctly predicted).

During the learning phase the class Disgust has the highest weight as it is the class with fewest samples, with the intent of balancing the class, which again explains the much higher recall measure compared to the unweighted model (Table 8).

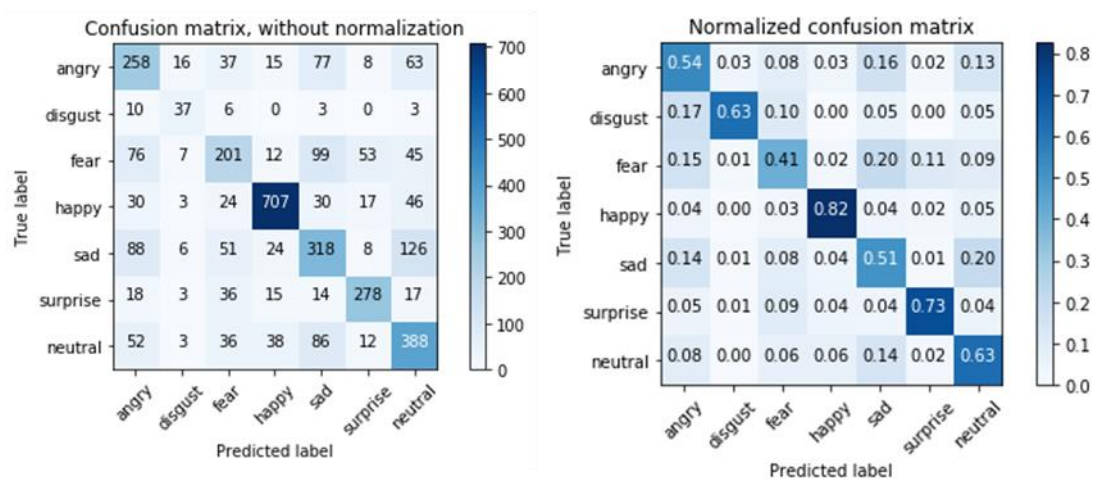


Figure 26 - Confusion matrix

|                 | PRECISION | RECALL | F1     |
|-----------------|-----------|--------|--------|
| <b>ANGRY</b>    | 0.4850    | 0.5443 | 0.5129 |
| <b>DISGUST</b>  | 0.4933    | 0.6271 | 0.5522 |
| <b>FEAR</b>     | 0.5141    | 0.4077 | 0.4548 |
| <b>HAPPY</b>    | 0.8718    | 0.8250 | 0.8478 |
| <b>SAD</b>      | 0.5072    | 0.5121 | 0.5096 |
| <b>SURPRISE</b> | 0.7394    | 0.7297 | 0.7345 |
| <b>NEUTRAL</b>  | 0.5640    | 0.6309 | 0.5956 |

Table 8 – Class metrics

|                 | ACCURACY | PRECISION | RECALL | F1     |
|-----------------|----------|-----------|--------|--------|
| <b>OVERALL</b>  | 0.6249   | 0.5964    | 0.6110 | 0.6011 |
| <b>WEIGHTED</b> | -        | 0.6295    | 0.6249 | 0.6254 |

Table 9 – Model metrics

The evaluation on the test set yielded a 0.6249 overall accuracy and a 1.03 cross-entropy loss. The evaluation took  $8,17 \times 10^{-4}$  seconds per image, approximately 3 seconds for the entire set.

For the Multi-task models the weighted version of the emotion recognition models is used. The unweighted version achieves a slightly higher score (about 1% better accuracy and F1), but the difference is not significant, and the weighted version is preferred with the rationale of better treatment of class imbalance.

After describing the results of the emotion models, the age estimation model is presented next.

### 4.1.2. Age Estimation

The age estimation model results are described and explained in this sub-section. As explained in the first model, the same protocol is followed in every model.

The testing phase is slightly different than the other tasks, as the age estimation task is not a multi-class problem. Instead of a confusion matrix, a graphic is provided with the MAE per true age label.

#### Training phase

Compared to the emotion models, the loss and metrics learning curve is more abrupt in this model (Figures 27 and 28).

The training lasted 224 epochs, and the model with the lowest validation loss was achieved in the 124<sup>th</sup> epoch. This model reached 3.875 MAE and 24.433 MSE loss in the training set, and 4.416 and 31.409 in the test set.

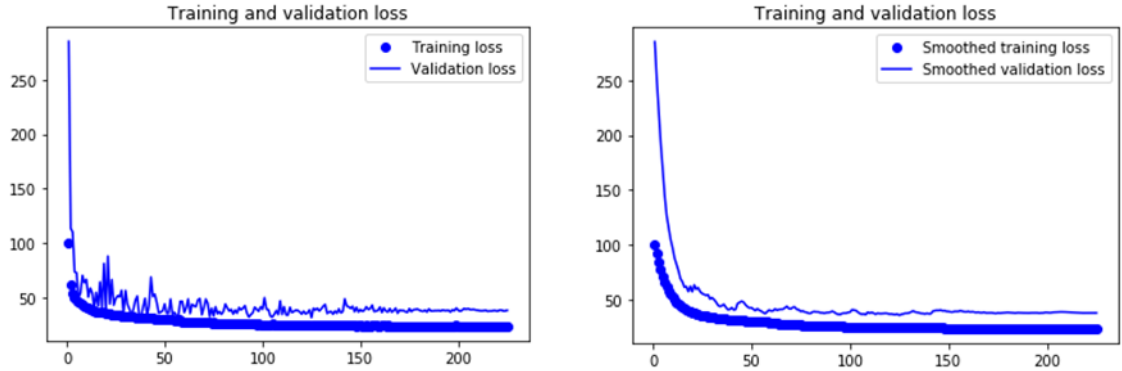


Figure 27- Training and validation loss (real and smoothed)

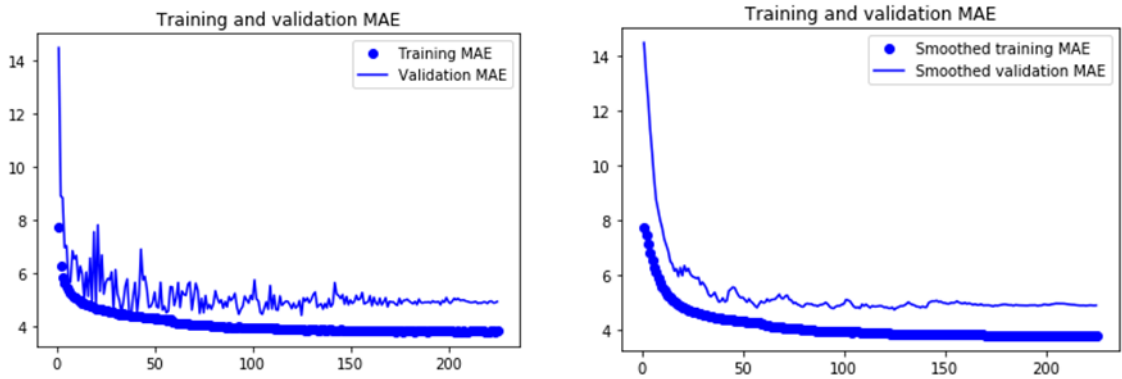


Figure 28 - Training and validation metrics (real and smoothed)

## Testing phase

Reading the MAE per age chart (Fig. 29) it is possible to conclude that the model is constant between the ages 20-55. Above the age 60 the MAE score spikes, which is expectable looking at the dataset age distribution (see Chapter 3) – there are very few examples above the age of 55, which makes the training of the model very unreliable for those ages.

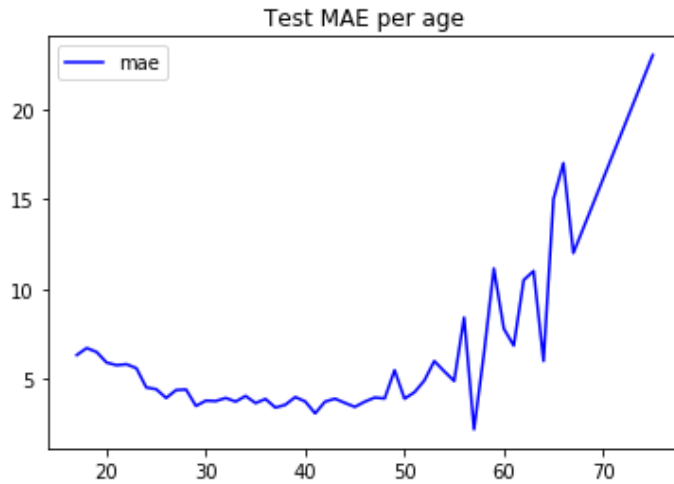


Figure 29 – MAE per age results

The evaluation on the test set yielded a 4.2468 Mean Absolute Error and a 29.5313 Mean Squared Error loss. The evaluation took approximately  $8,84 \times 10^{-4}$  seconds per image.

### 4.1.3. Gender Recognition

The gender recognition model results are described and explained in this sub-section.

The testing phase is similar to the one used in the emotion recognition models, except for the number of classes.

#### 4.1.3.1. Class unweighted

##### Training phase

Looking at the loss and accuracy learning curves (Figures 30 and 31) it is possible to see that during the first 100 epochs there are spikes. After the 100<sup>th</sup> epoch the spikes stop. This may be due to the learning rate being somewhat high during the beginning of the training (for this task), and after the process of reducing it kicks in, the spikes stop. Because the process of reducing the learning rate occurs at least four times (if there is no improvement in the validation loss), there is no problem of underfitting caused by the high learning rate. It can also be seen there is no underfitting in the charts, where after around the 125<sup>th</sup> epoch the validation loss stalls.

The training lasted 208 epochs, and the model with the lowest validation loss was achieved in the 108<sup>th</sup> epoch. This model reached 0.991 accuracy and 0.024 cross-entropy loss in the training set, and 0.985 and 0.048 in the test set.

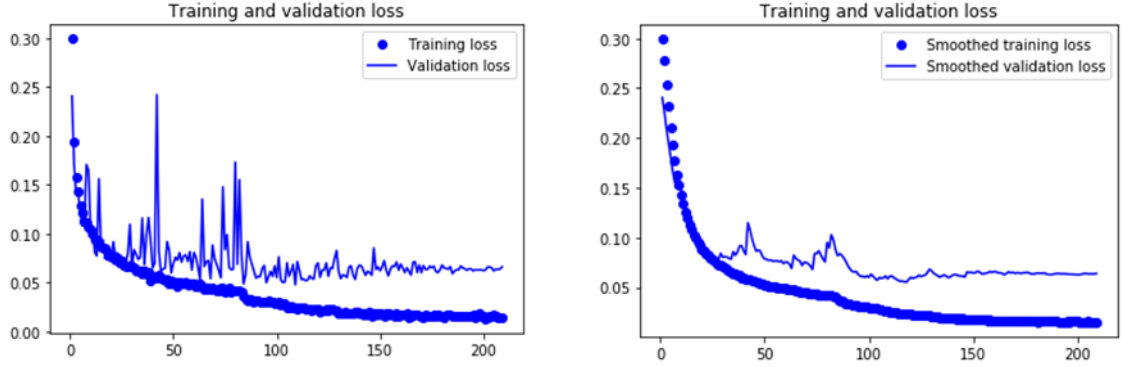


Figure 30 - Training and validation loss (real and smoothed)

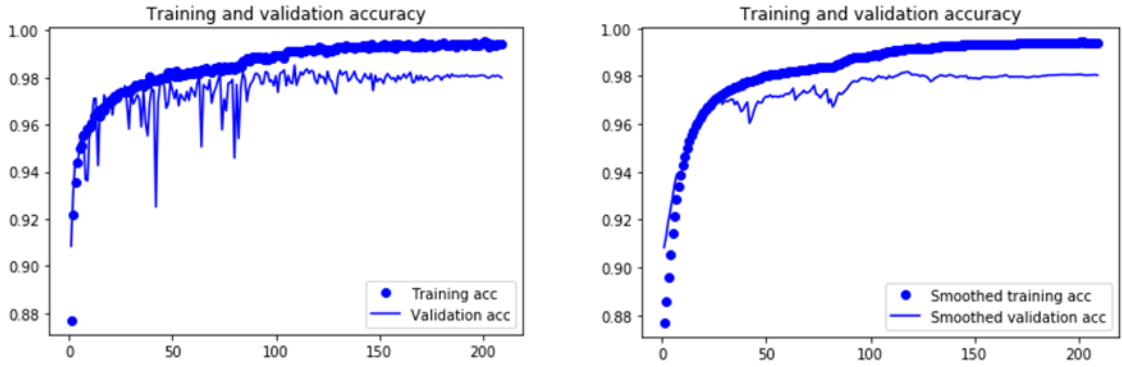


Figure 31 - Training and validation metrics (real and smoothed)

## Testing phase

The model behaves very well when distinguishing the two classes (Fig. 32), even though they are quite unbalanced. While this model is unweighted, not trying to correct this imbalance, there is almost the same number of samples wrongfully predicted on both classes.

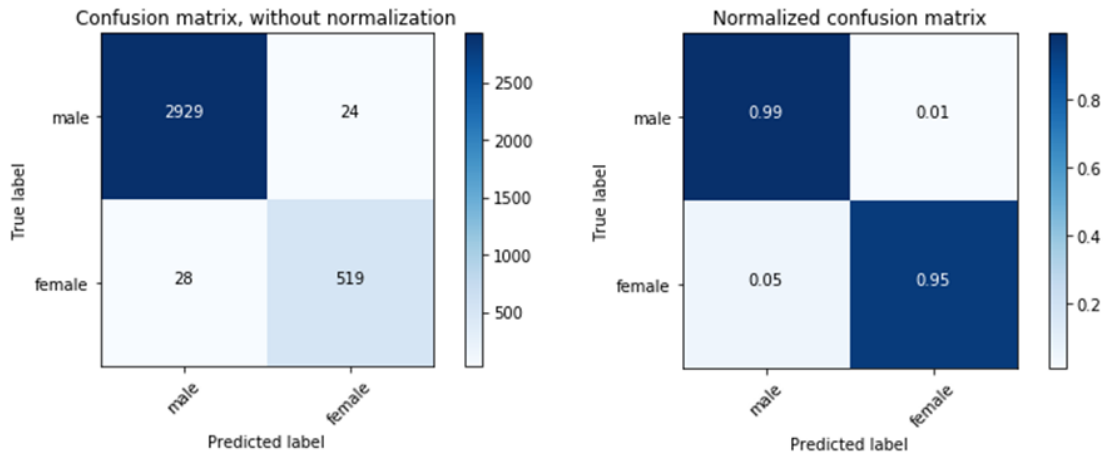


Figure 32 - Confusion matrix

The three evaluation measures are quite similar for each class (Table 10). The male class has better precision, recall and F1 as expected, as the dataset is not balanced.

|               | PRECISION | RECALL | F1     |
|---------------|-----------|--------|--------|
| <b>MALE</b>   | 0.9905    | 0.9919 | 0.9912 |
| <b>FEMALE</b> | 0.9558    | 0.9488 | 0.9523 |

Table 10 – Class metrics

The model achieves a high accuracy (Table 11). The precision, recall and F1 measures are, again, quite similar and suitably high.

|                 | ACCURACY | PRECISION | RECALL | F1     |
|-----------------|----------|-----------|--------|--------|
| <b>OVERALL</b>  | 0.9851   | 0.9732    | 0.9704 | 0.9717 |
| <b>WEIGHTED</b> | -        | 0.9851    | 0.9852 | 0.9851 |

Table 11 – Model metrics

The evaluation on the test set yielded a 0.9851 accuracy and a 0.0446 cross-entropy loss. The evaluation took about  $9,18 \times 10^{-4}$  seconds per image.

#### 4.1.3.2. Class weighted

##### Training phase

The same behavior described in the unweighted model can be seen in this model (Figures 33 and 34): learning spikes. The same explanation is suggested. The validation loss stalls approximately at the 100<sup>th</sup> epoch.

The training lasted 244 epochs, and the model with the lowest validation loss was achieved in the 144<sup>th</sup> epoch. This model reached 0.988 accuracy and 0.030 cross-entropy loss in the training set, and 0.984 and 0.051 in the test set.

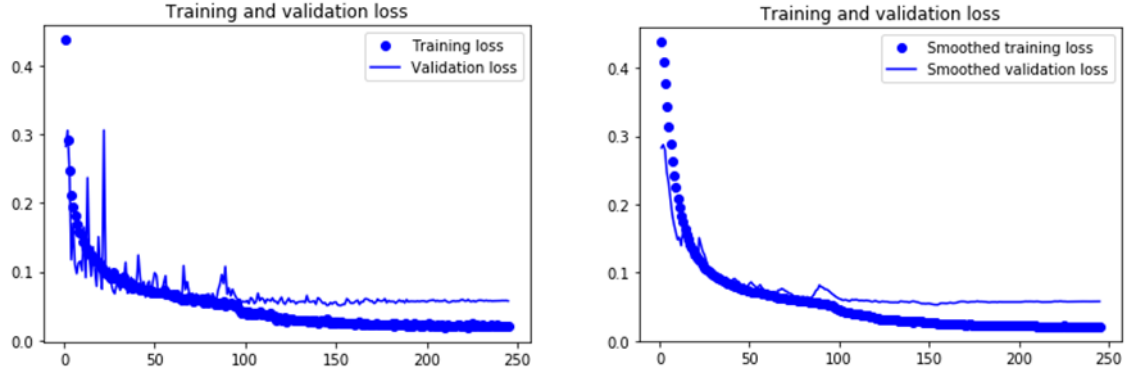


Figure 33 - Training and validation loss (real and smoothed)

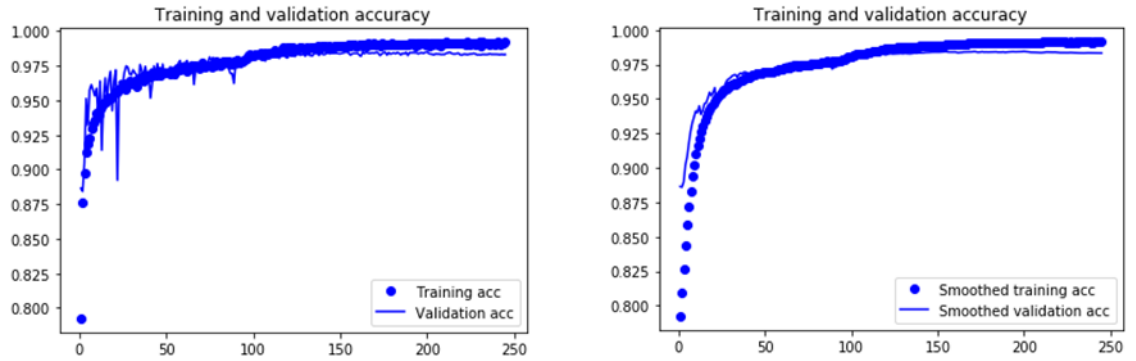


Figure 34- Training and validation metrics (real and smoothed)

## Testing phase

The obtained confusion matrix (Figure 35) is fundamentally the same, with slight better values in this model.

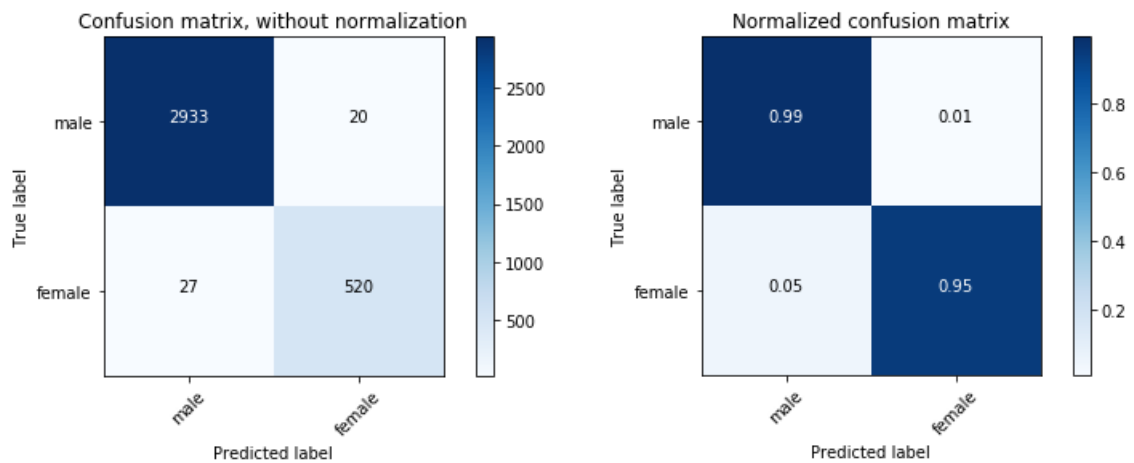


Figure 35 - Confusion matrix

The evaluation measures are quite similar per class too (Table 12), and this model has somewhat better results for all the metrics.

|               | <b>PRECISION</b> | <b>RECALL</b> | <b>F1</b> |
|---------------|------------------|---------------|-----------|
| <b>MALE</b>   | 0.9909           | 0.9932        | 0.9920    |
| <b>FEMALE</b> | 0.9630           | 0.9506        | 0.9568    |

*Table 12 – Class metrics*

In comparison with the unweighted model, this model has, again, slightly better values in all the metrics (Table 13). With this in mind, and with the same motivation explained in the emotion recognition model, the Multi-task learning models employ the weighted version of the gender recognition task.

|                 | <b>ACCURACY</b> | <b>PRECISION</b> | <b>RECALL</b> | <b>F1</b> |
|-----------------|-----------------|------------------|---------------|-----------|
| <b>OVERALL</b>  | 0.9866          | 0.9770           | 0.9719        | 0.9744    |
| <b>WEIGHTED</b> | -               | 0.9865           | 0.9865        | 0.9865    |

*Table 13 – Model metrics*

The evaluation on the test set yielded a 0.9866 accuracy and a 0.0410 cross-entropy loss. The evaluation took nearly  $7,82 \times 10^{-4}$  seconds per image.

## 4.2. Multi-task Models

In this section the results and evaluation of the multi-task models are outlined. There are four sub-sections, one for each model. Since the difference between the models is the number of shared residual modules (described in 3.3. Model Design), these are presented, in order, from the model with the most shared modules (v1) to the model with the fewest shared modules (v4). After this section, a model comparison is performed, observing the different evaluation metrics.

The training and testing phase results and evaluation are performed in the same way as the single-model task models, but every multi-task model comprehends all of the single-models' charts. Thus, during the training phase, the metrics and loss learning curves epoch by epoch are presented. There are charts for the age, gender and emotion metrics and losses, with added charts of the joint learnt gender and age, which were trained simultaneously using an aggregated loss, and a Total loss chart.

The testing phase provides the model results on the test set for all the tasks, and the same results presented in the single-tasks are provided – confusion matrices for the tasks of gender and emotion recognition, along with class by class and overall accuracy, precision, recall and F1 metrics; and an MAE distribution by real age for the task of age estimation.

### 4.2.1. Triple-Module Sharing (v1)

#### Training phase

Observing the charts (Figures 36 to 43), it is possible to conclude the training was successful, the learning curve shows a tendency to a smaller validation loss. A stall occurs approximately at the 125<sup>th</sup> epoch.

The training lasted 181 epochs, and the model with the lowest Total loss was achieved in the 81<sup>st</sup> epoch. This model reached 0.5991 accuracy and 1.002 cross-entropy loss in the emotion training set, and 1.122 and 0.051 in the emotion validation set. This model reached 4.358 MAE and 30.750 MSE loss in the age training set, and 4.685 and 35.927 in the age validation set. It reached 0.976 accuracy and 0.057 cross-entropy loss in the gender training set, and 0.976 and 0.065 in the gender validation set. The aggregated gender and age loss reached 0.365 and 0.425 training and validation loss respectively, and finally, the total loss reached, in the 81<sup>st</sup> epoch, 0.681 and 0.774 training and validation loss respectively.

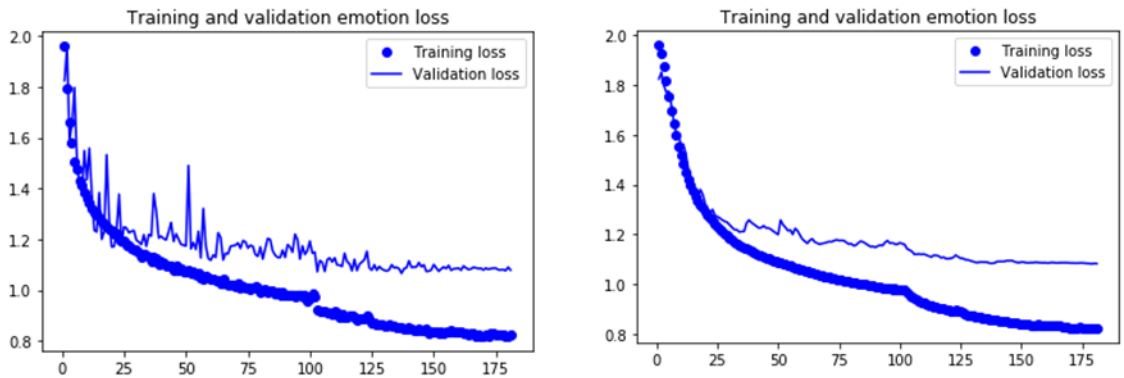


Figure 36 - Emotion training and validation loss (real and smoothed)

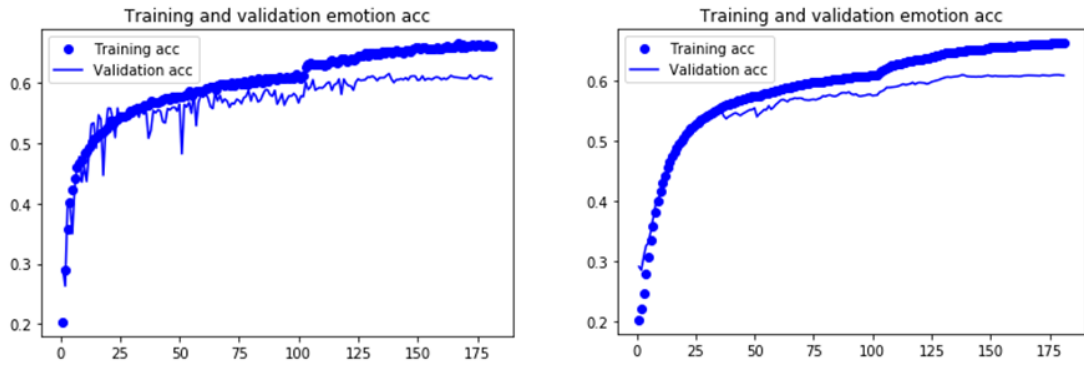


Figure 37 - Emotion training and validation metrics (real and smoothed)

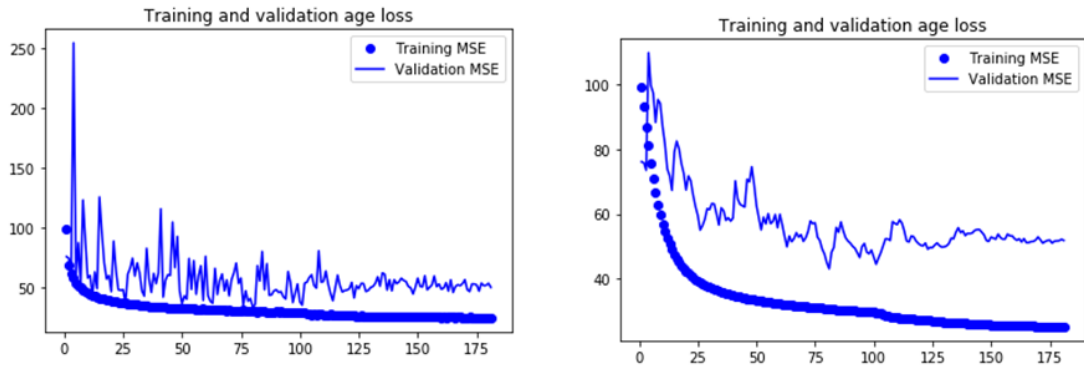


Figure 38 - Age training and validation loss (real and smoothed)

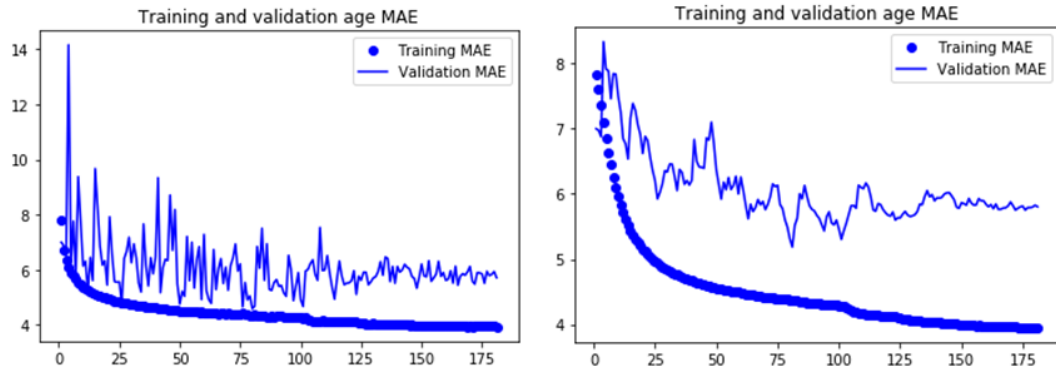


Figure 39 - Age training and validation metrics (real and smoothed)

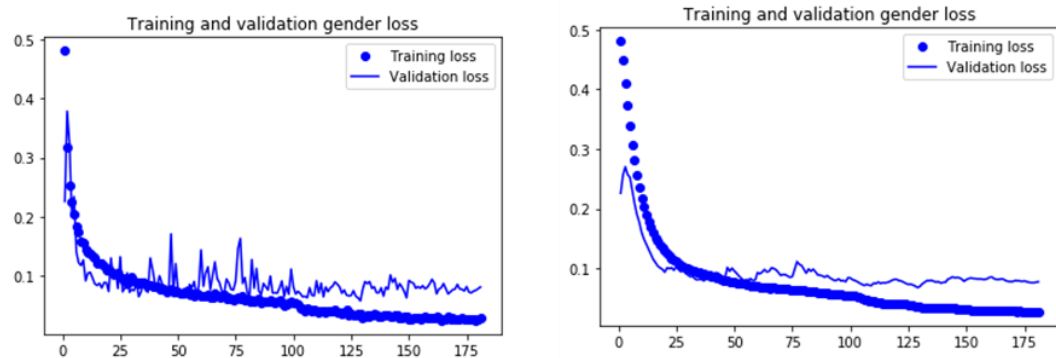


Figure 40 - Gender training and validation loss (real and smoothed)

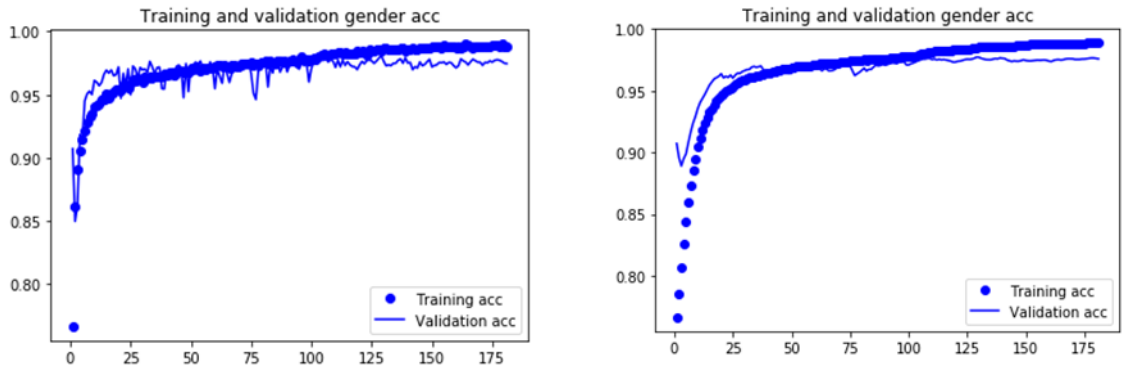


Figure 41 - Gender training and validation metrics (real and smoothed)

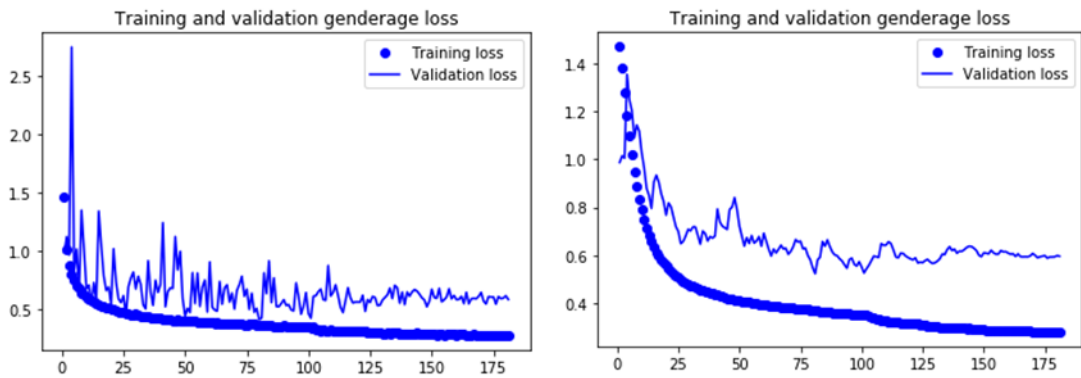


Figure 42 - Aggregated gender and age training and validation loss (real and smoothed)

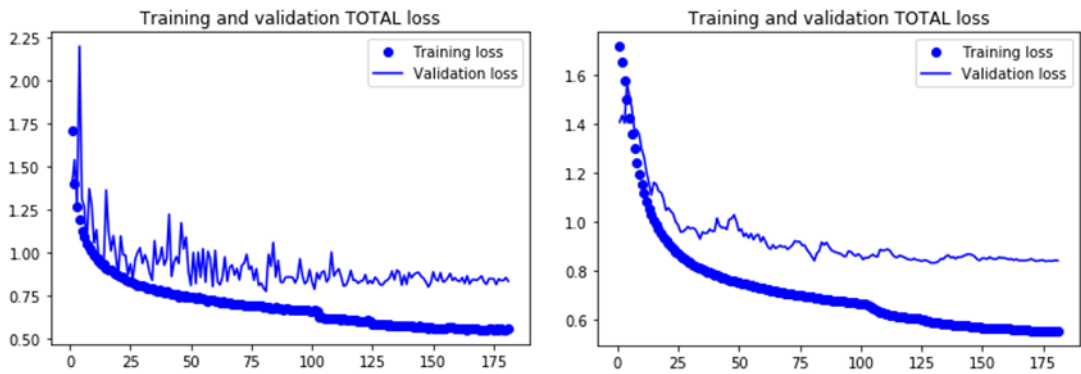


Figure 43 - Total training and validation loss (real and smoothed)

## Testing phase

Observing Figure 44 and Table 14, it is possible to conclude this model does not behave well for the classes Angry, Disgust and Fear. Disgust is expected to have the lowest performance, as explained in the emotion model analysis, but one can see the F1 metric is lowest for the class Fear.

The overall accuracy and weighted F1 score (Table 15) show the model has a good performance, as it is comparable to the single-task emotion model.

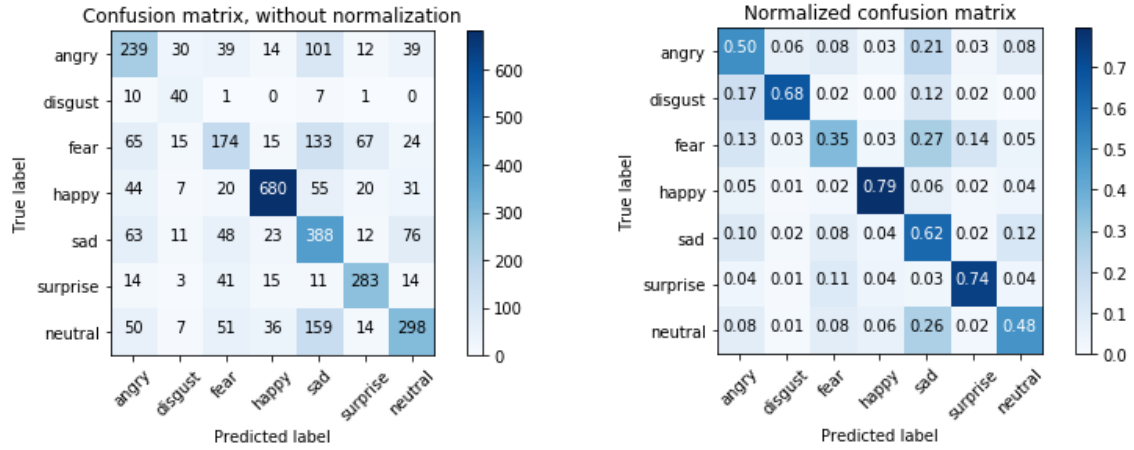


Figure 44 – Emotion confusion matrix

|          | PRECISION | RECALL | F1     |
|----------|-----------|--------|--------|
| ANGRY    | 0.4928    | 0.5042 | 0.4984 |
| DISGUST  | 0.3540    | 0.6780 | 0.4651 |
| FEAR     | 0.4652    | 0.3529 | 0.4013 |
| HAPPY    | 0.8685    | 0.7935 | 0.8293 |
| SAD      | 0.4543    | 0.6248 | 0.5261 |
| SURPRISE | 0.6919    | 0.7428 | 0.7164 |
| NEUTRAL  | 0.6183    | 0.4846 | 0.5433 |

Table 14 – Emotion class metrics

|          | ACCURACY | PRECISION | RECALL | F1     |
|----------|----------|-----------|--------|--------|
| OVERALL  | 0.6006   | 0.5636    | 0.5973 | 0.5686 |
| WEIGHTED | -        | 0.6155    | 0.6006 | 0.6017 |

Table 15 – Emotion metrics

Observing Figure 45, the same conclusion is reached as the single-task model. The model is capable of predicting well between the ages 20 and 55, but then the performance decays abruptly, explained by the lower number of samples above the age of 55.

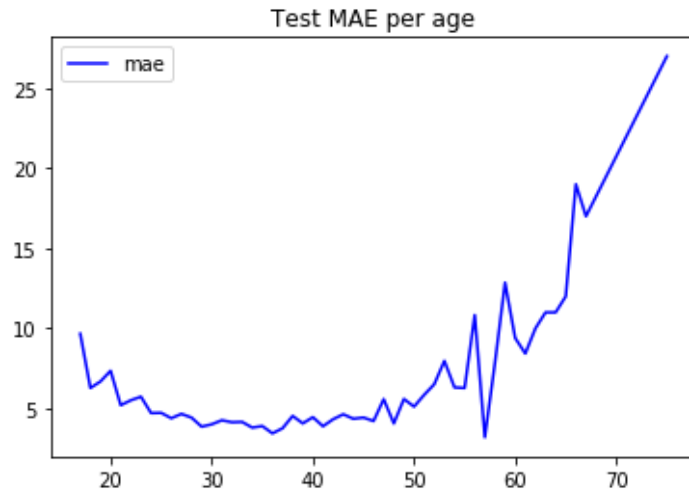


Figure 45 – MAE per age results

Examining the gender results (Fig. 46 and Tables 16 and 17), the model is better at accessing the male class, as expected, as all the measures are higher for that class.

The evaluation on the emotion test set yielded a 0.6006 accuracy and a 137.1691 cross-entropy loss. The evaluation took about  $9,46 \times 10^{-4}$  seconds per image. On the age test set the model yielded a 4.6825 MAE and a 35.4815 MSE loss taking around  $1,00 \times 10^{-3}$  seconds per image. Lastly, the evaluation on the gender test set yielded a 0.9766 accuracy and a 0.064439 cross-entropy loss taking approximately  $1,00 \times 10^{-3}$  seconds per image.

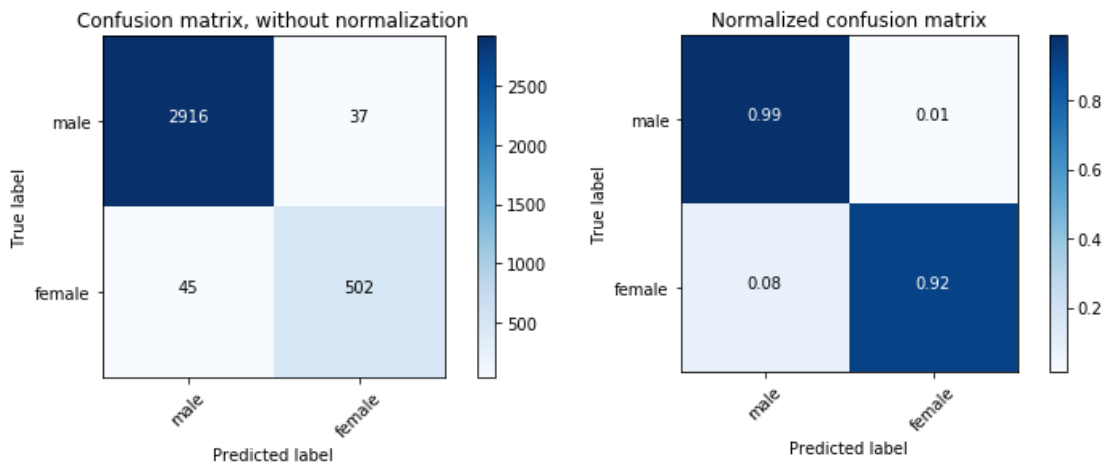


Figure 46 – Gender confusion matrix

|               | PRECISION | RECALL | F1     |
|---------------|-----------|--------|--------|
| <b>MALE</b>   | 0.9848    | 0.9875 | 0.9861 |
| <b>FEMALE</b> | 0.9314    | 0.9526 | 0.9419 |

Table 16 – Gender class metrics

|                 | ACCURACY | PRECISION | RECALL | F1     |
|-----------------|----------|-----------|--------|--------|
| <b>OVERALL</b>  | 0.9766   | 0.9581    | 0.9701 | 0.9640 |
| <b>WEIGHTED</b> | -        | 0.9765    | 0.9820 | 0.9792 |

Table 17 – Gender metrics

## 4.2.2. Dual-Module Sharing (v2)

### Training phase

Examining Figures 47 to 54, it is possible to conclude the training was successful, as there is a tendency to achieve a lower validation for all the tasks.

The emotion charts show a stall at the 100<sup>th</sup> epoch, and the age and gender tasks at the 75<sup>th</sup> epoch, approximately.

The training lasted 180 epochs, achieving the lowest total loss on the validation set on the 80<sup>th</sup> epoch, with a value of 0.741. The emotion task achieved a cross-entropy loss of 0.927 and 1.086, and accuracy 0.630 and 0.625 on the training set and validation set respectively. The age task achieved an MAE of 4.129 and 4.591, and an MSE loss of 27.535 and 34.092 on the training and validation set respectively. The gender task achieved an accuracy of 0.980 and 0.981, and a cross-entropy loss of 0.047 and 0.055. The aggregated gender and age loss reached 0.323 and 0.396 on the training and validation set, and finally the total loss at the 80<sup>th</sup> epoch was 0.625 and 0.741.

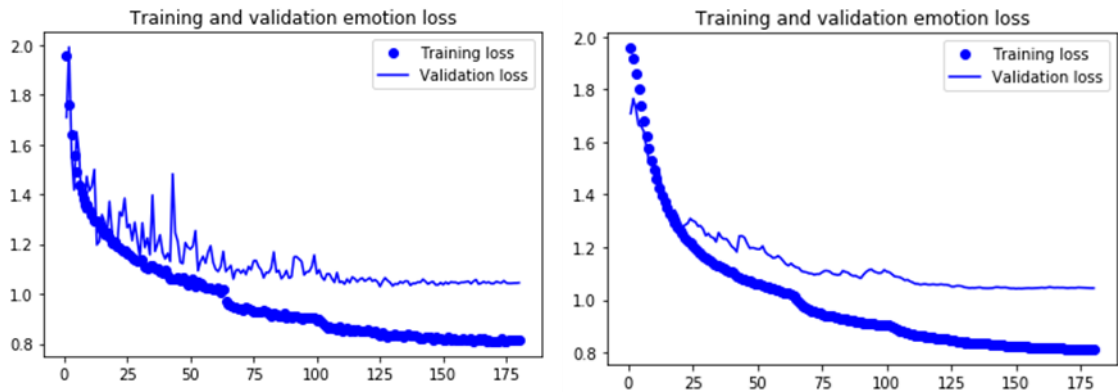


Figure 47 - Emotion training and validation loss (real and smoothed)

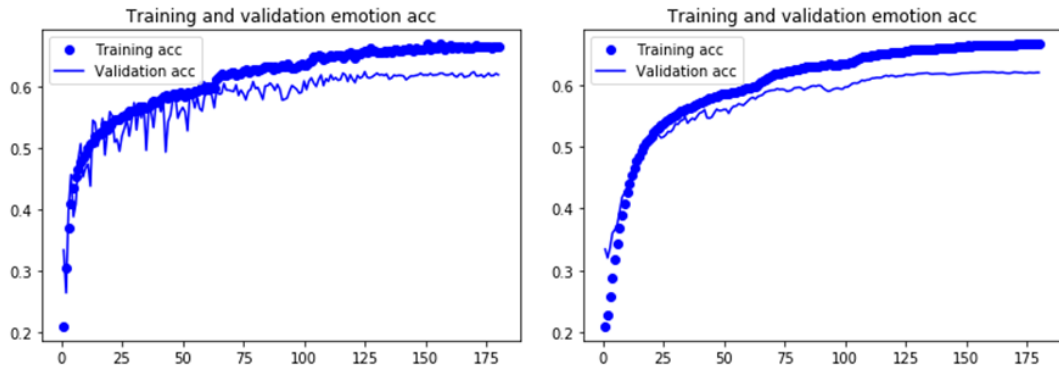


Figure 48 - Emotion training and validation metrics (real and smoothed)

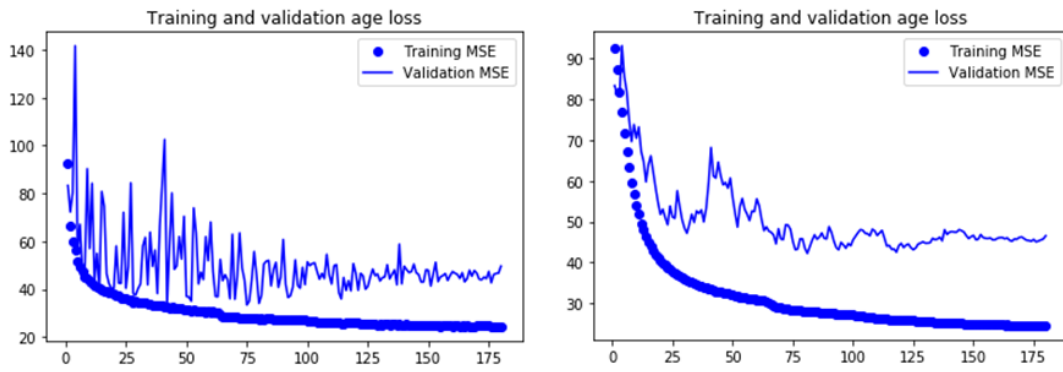


Figure 49 - Age training and validation loss (real and smoothed)

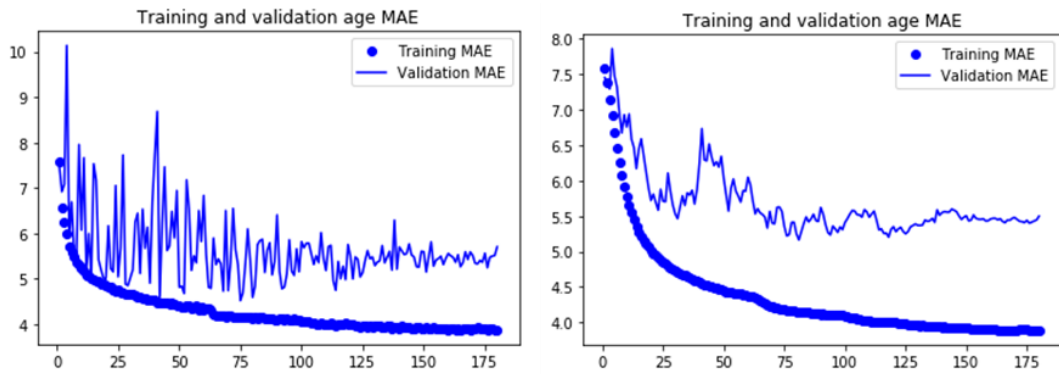


Figure 50 - Age training and validation metrics (real and smoothed)

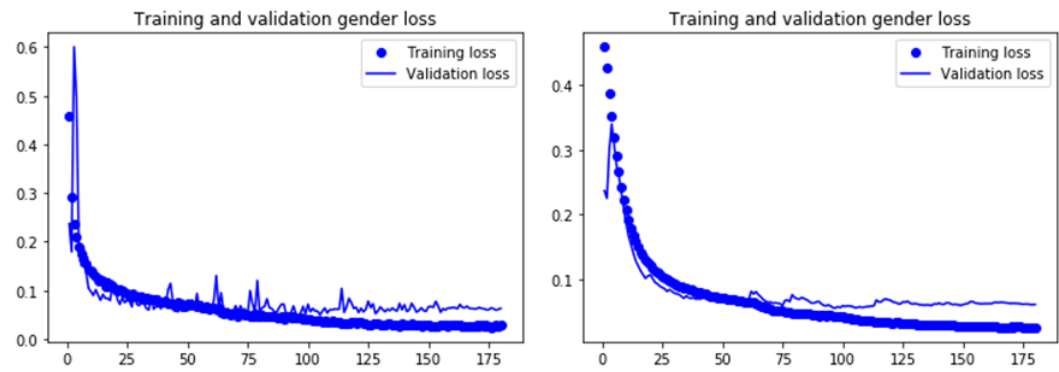


Figure 51 - Gender training and validation loss (real and smoothed)

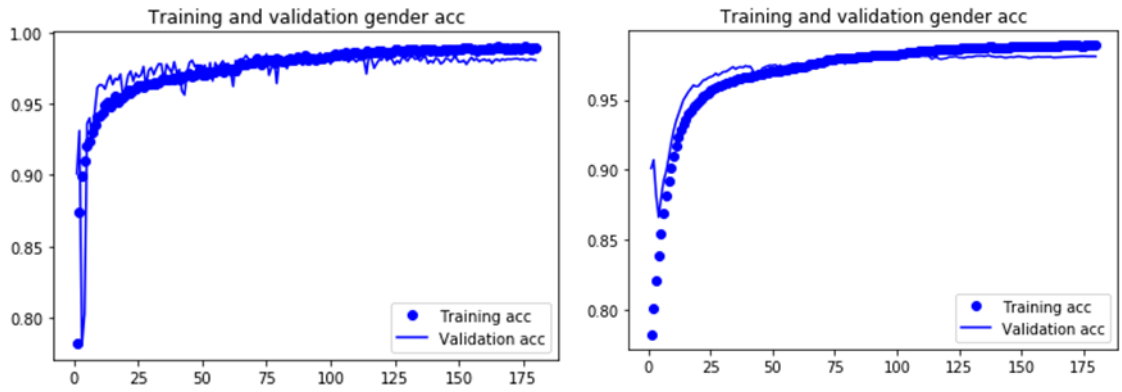


Figure 52 - Gender training and validation metrics (real and smoothed)

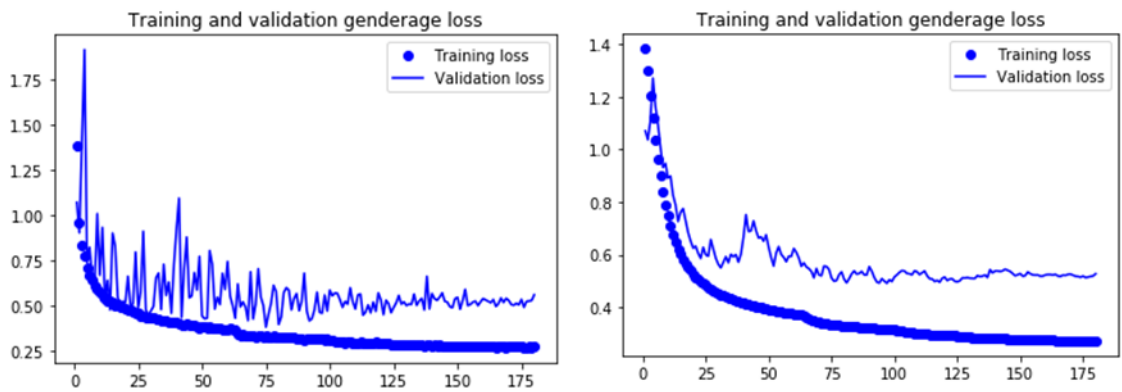


Figure 53 - Aggregated gender and age training and validation loss (real and smoothed)

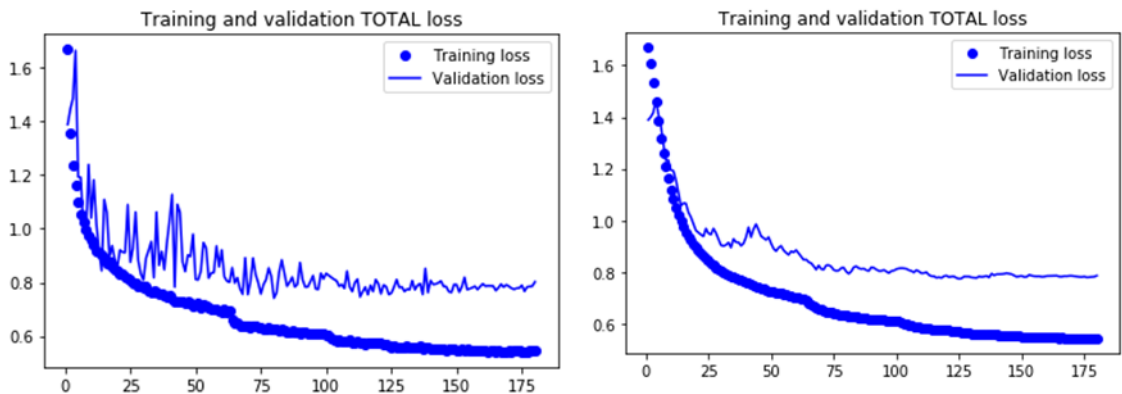


Figure 54 - Total training and validation loss (real and smoothed)

## Testing phase

The obtained results for the emotion test (Fig.55 and Tables 18 and 19) are expectable, again with the Disgust and Fear classes achieving the worst performances looking at the F1 score. The age (Fig. 56) and gender (Fig.57 and Tables 20 and 21) results are consistent with the other models.

The evaluation on the emotion test set yielded a 0.5874 accuracy and a 1.0983 cross-entropy loss. The evaluation took about  $1 \times 10^{-3}$  seconds per image. On the age test set, the model yielded a 4.5901 MAE and a 33.8809 MSE loss, taking around  $2 \times 10^{-3}$  seconds per image. The gender task reached 0.9811 accuracy and a 0.0573 cross-entropy loss taking approximately  $2 \times 10^{-3}$  seconds per image.

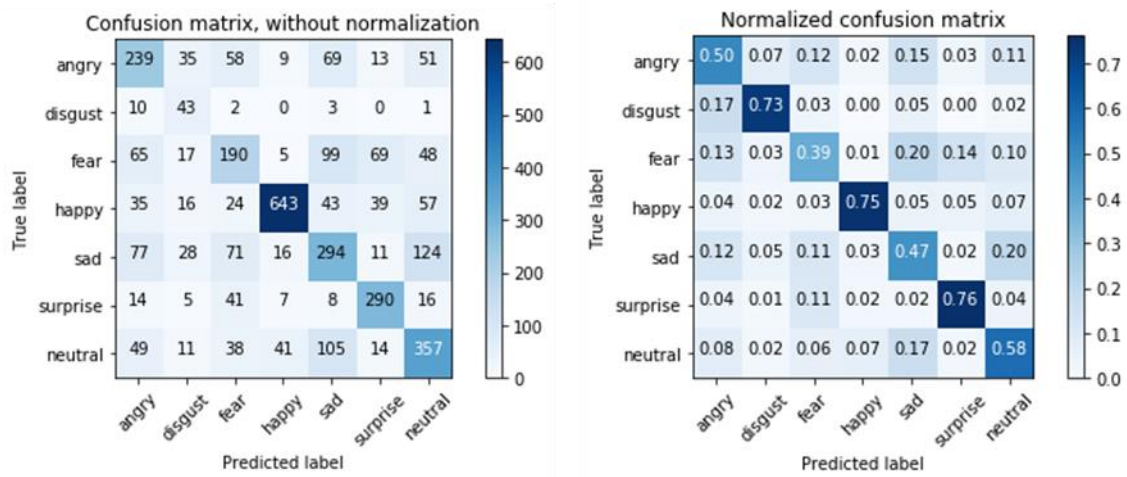


Figure 55 - Emotion confusion matrix

|          | PRECISION | RECALL | F1     |
|----------|-----------|--------|--------|
| ANGRY    | 0.4888    | 0.5042 | 0.4964 |
| DISGUST  | 0.2774    | 0.7288 | 0.4018 |
| FEAR     | 0.4481    | 0.3854 | 0.4144 |
| HAPPY    | 0.8918    | 0.7503 | 0.8150 |
| SAD      | 0.4734    | 0.4734 | 0.4734 |
| SURPRISE | 0.6651    | 0.7612 | 0.7099 |
| NEUTRAL  | 0.5459    | 0.5805 | 0.5627 |

Table 18 – Emotion class metrics

|          | ACCURACY | PRECISION | RECALL | F1     |
|----------|----------|-----------|--------|--------|
| OVERALL  | 0.5874   | 0.5415    | 0.5977 | 0.5534 |
| WEIGHTED | -        | 0.6047    | 0.5874 | 0.5921 |

Table 19 – Emotion metrics

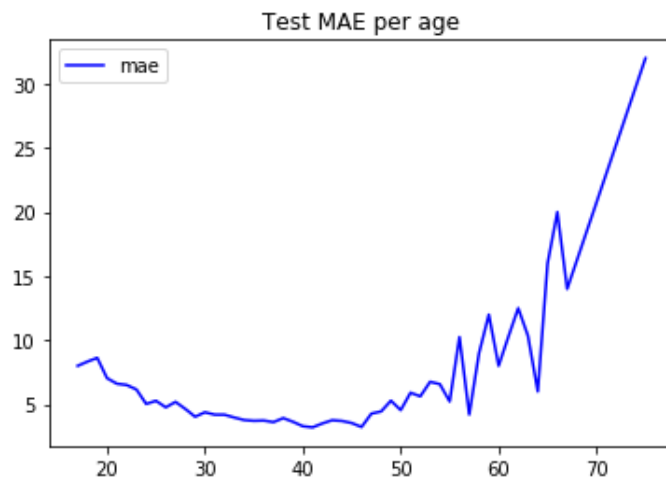


Figure 56 – Mae per age results

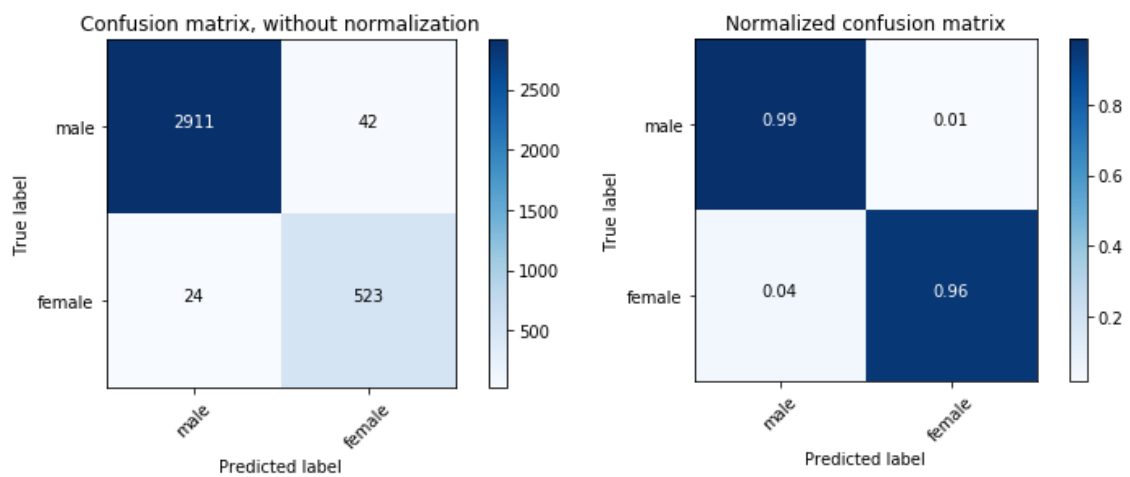


Figure 57 – Gender confusion matrix

|        | PRECISION | RECALL | F1     |
|--------|-----------|--------|--------|
| MALE   | 0.9918    | 0.9858 | 0.9888 |
| FEMALE | 0.9257    | 0.9561 | 0.9407 |

Table 20 – Gender class metrics

|          | ACCURACY | PRECISION | RECALL | F1     |
|----------|----------|-----------|--------|--------|
| OVERALL  | 0.9811   | 0.9588    | 0.9710 | 0.9647 |
| WEIGHTED | -        | 0.9815    | 0.9812 | 0.9813 |

Table 21 – Gender metrics

### 4.2.3. Single-Module Sharing (v3)

#### Training phase

Figures 58 to 65 show a successful tendency to learn during the training phase. Consistent with the other models, the various losses tend to decrease.

The training procedure lasted 197 epochs, with the lowest validation total loss achieved during the epoch 97, with a value of 0.730. The emotion task achieved a cross-entropy loss of 0.893 and 1.087, and accuracy 0.637 and 0.599 on the training set and validation set respectively. The age task achieved an MAE of 4.004 and 4.422, and an MSE loss of 26.150 and 31.670. The gender task achieved an accuracy of 0.982 and 0.982, and a cross-entropy loss of 0.041 and 0.055. The aggregated gender and age loss reached 0.303 and 0.372 on the training and validation set, and at last the total loss at the 80<sup>th</sup> epoch was 0.598 and 0.730.

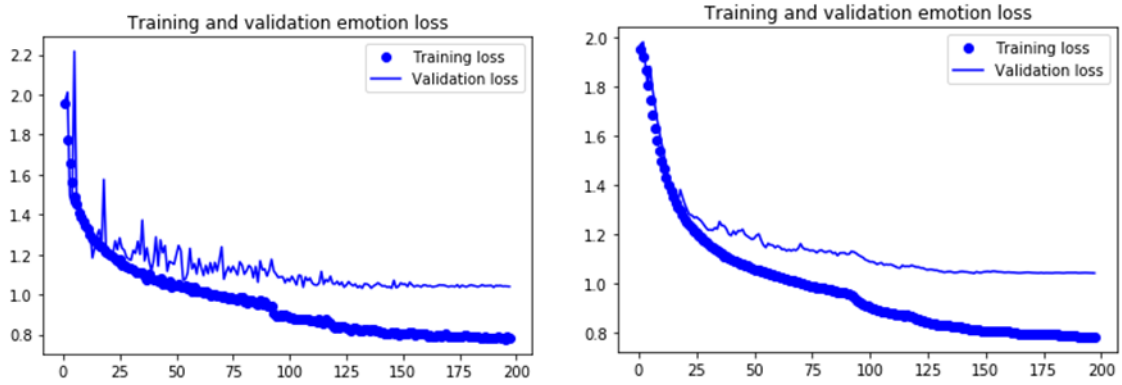


Figure 58 – Emotion training and validation loss (real and smoothed)

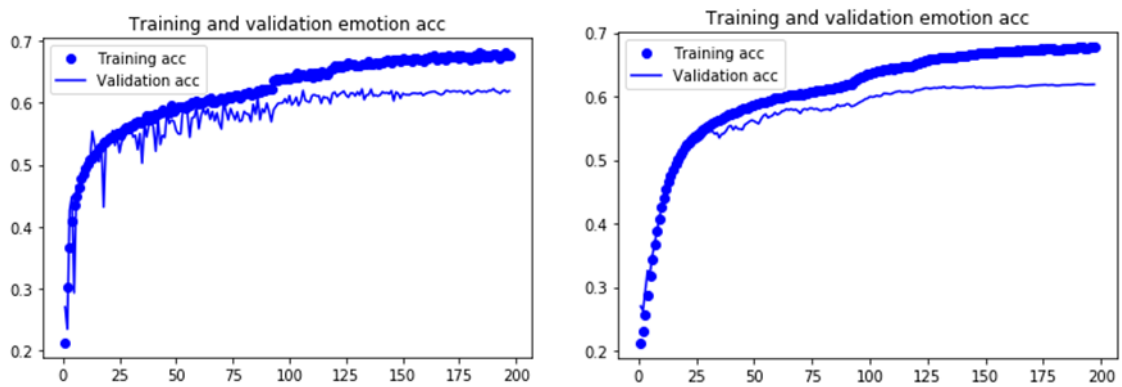


Figure 59 – Emotion training and validation metrics (real and smoothed)

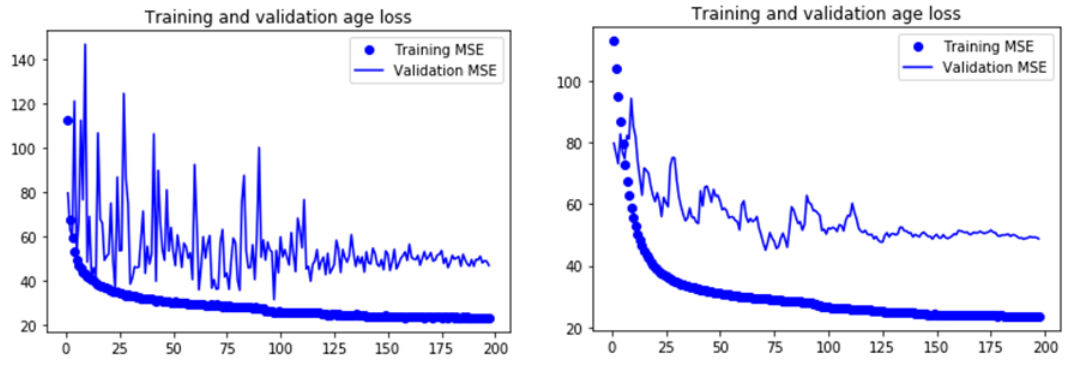


Figure 60 – Age training and validation loss (real and smoothed)

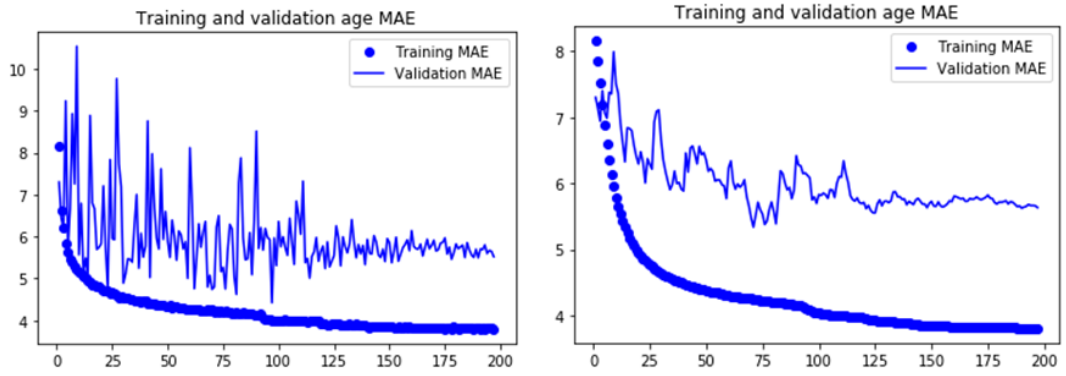


Figure 61 – Age training and validation metrics (real and smoothed)

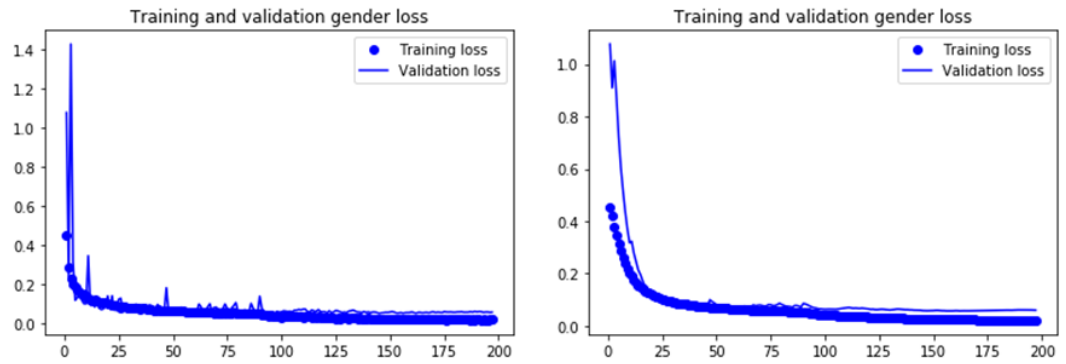


Figure 62 – Gender training and validation loss (real and smoothed)

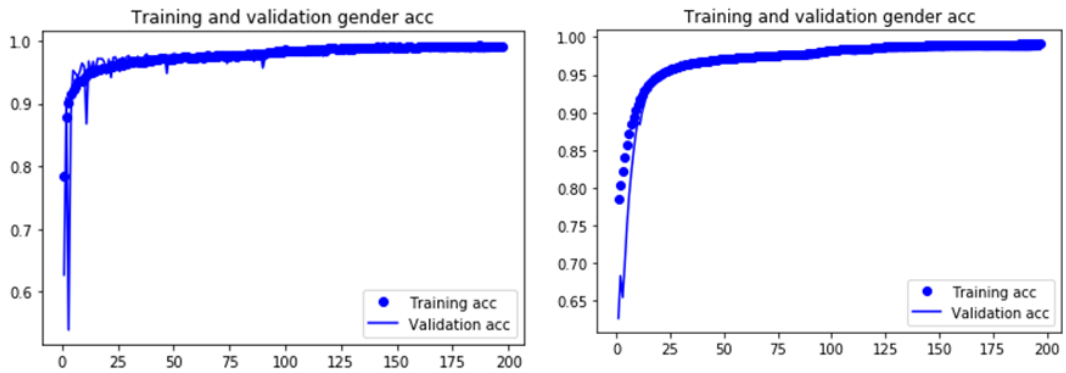


Figure 63 – Gender training and validation metrics (real and smoothed)

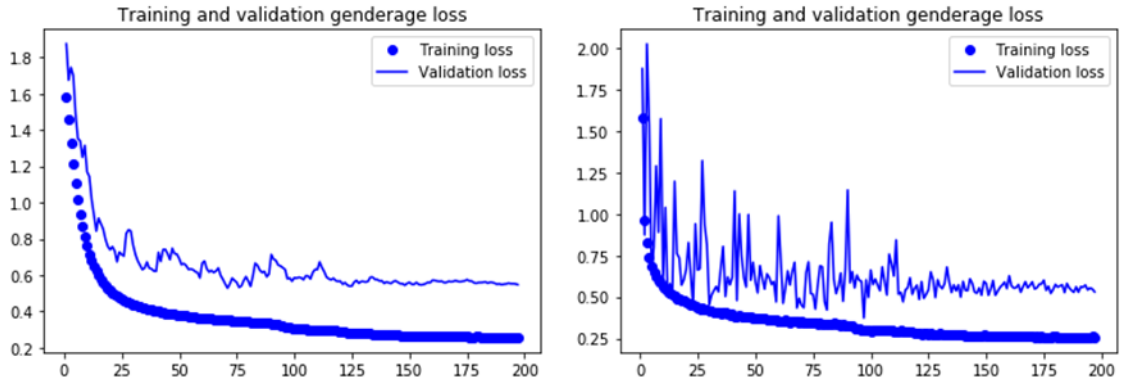


Figure 64 – Aggregated gender and age training and validation loss (real and smoothed)

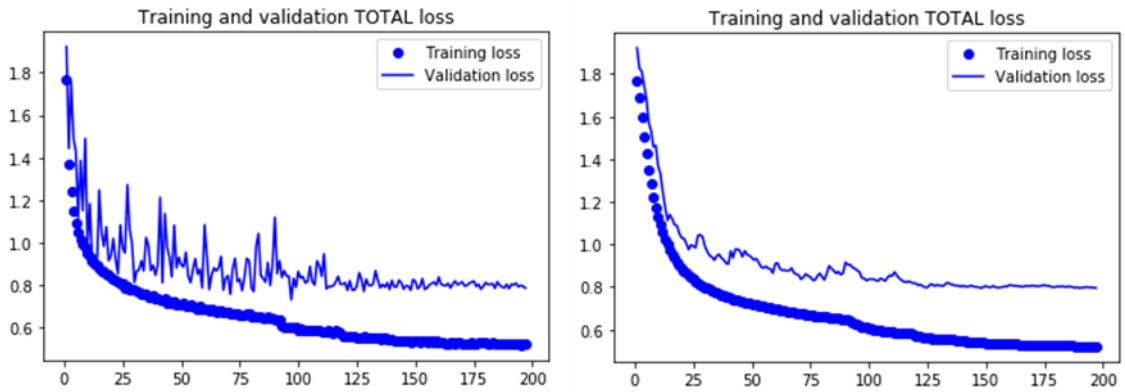


Figure 65 – Total training and validation loss (real and smoothed)

## Testing phase

Looking at the emotion results (Fig. 66 and Tables 22 and 23) there is a difference from the other models, the worst performance belongs to the class Fear and not to the class Disgust, through the F1 metric result comparison. The age graph (Fig. 67) also looks to some extent steadier, with a prolonged interval of low MAE – 20 to about 60 years old. The gender (Fig. 68 and Tables 24 and 25) results look consistent with the other models.

The evaluation on the emotion test set yielded a 0.6009 accuracy and a 1.0804 cross-entropy loss taking around  $1 \times 10^{-3}$  seconds per image. On the age test set, yielded a 4.2883 MAE and a 29.6632 MSE loss. The evaluation took about  $2 \times 10^{-3}$  seconds per image. The model reached a 0.9800 accuracy and a 0.0598 cross-entropy loss on the gender test set evaluation taking approximately  $2 \times 10^{-3}$  seconds per image.

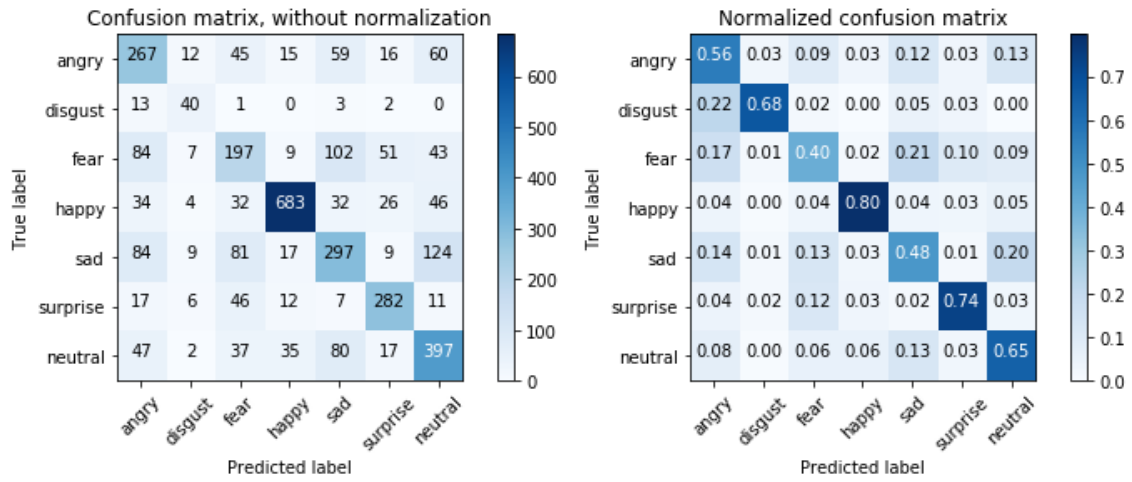


Figure 66 – Emotion confusion matrix

|          | PRECISION | RECALL | F1     |
|----------|-----------|--------|--------|
| ANGRY    | 0.4332    | 0.5401 | 0.4808 |
| DISGUST  | 0.3194    | 0.7797 | 0.4532 |
| FEAR     | 0.4673    | 0.2901 | 0.3580 |
| HAPPY    | 0.8450    | 0.8460 | 0.8455 |
| SAD      | 0.5690    | 0.3849 | 0.4592 |
| SURPRISE | 0.6593    | 0.7874 | 0.7177 |
| NEUTRAL  | 0.5427    | 0.6407 | 0.5876 |

Table 22 – Emotion class metrics

|          | ACCURACY | PRECISION | RECALL | F1     |
|----------|----------|-----------|--------|--------|
| OVERALL  | 0.6009   | 0.5480    | 0.6098 | 0.5574 |
| WEIGHTED | -        | 0.6049    | 0.6009 | 0.5931 |

Table 23 – Emotion metrics

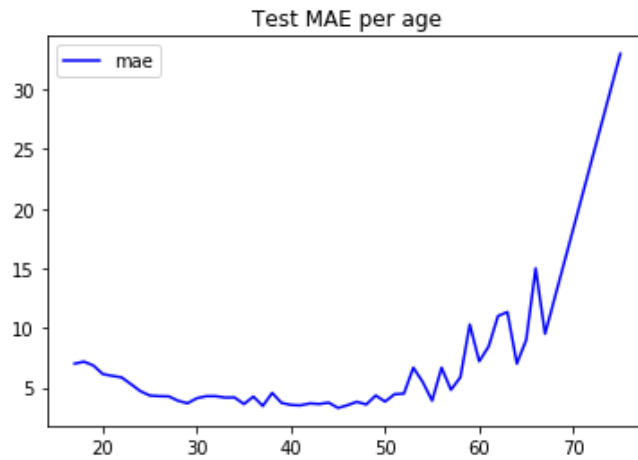


Figure 67 - MAE per age result

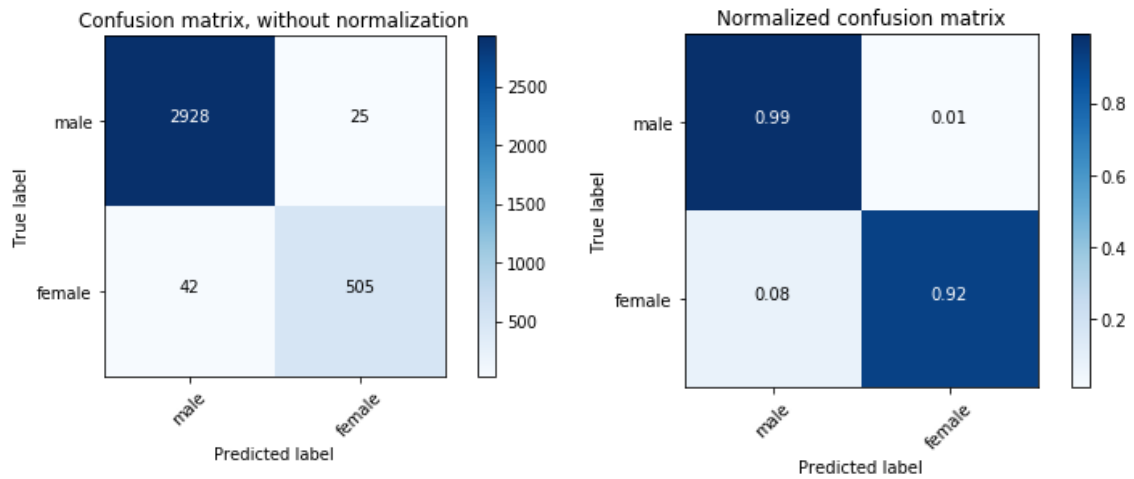


Figure 68 – Gender confusion matrix

|               | PRECISION | RECALL | F1     |
|---------------|-----------|--------|--------|
| <b>MALE</b>   | 0.9865    | 0.9898 | 0.9881 |
| <b>FEMALE</b> | 0.9441    | 0.9269 | 0.9354 |

Table 24 – Gender class metrics

|                 | ACCURACY | PRECISION | RECALL | F1     |
|-----------------|----------|-----------|--------|--------|
| <b>OVERALL</b>  | 0.9800   | 0.9653    | 0.9584 | 0.9618 |
| <b>WEIGHTED</b> | -        | 0.9799    | 0.9800 | 0.9799 |

Table 25 – Gender metrics

#### 4.2.4. Zero-Module Sharing (v4)

##### Training phase

Figures 69 to 76 are consistent with the other models, without any visual remarks worth noting: the training was successful.

The training phase underwent 181 epochs. The lowest validation total loss was achieved in the epoch 81, with a value of 0.723. The emotion task achieved a cross-entropy loss of 0.899 and 1.045, and accuracy 0.637 and 0.613 on the training set and validation set respectively. The age task achieved an MAE of 4.132 and 4.552, and an MSE loss of 27.687 and 34.108. The gender task achieved an accuracy of 0.981 and 0.980, and a cross-entropy loss of 0.048 and 0.060. The aggregated gender and age loss reached 0.325 and 0.402 on the training and validation set, and at last the total loss at the 81<sup>st</sup> epoch was 0.612 and 0.723.

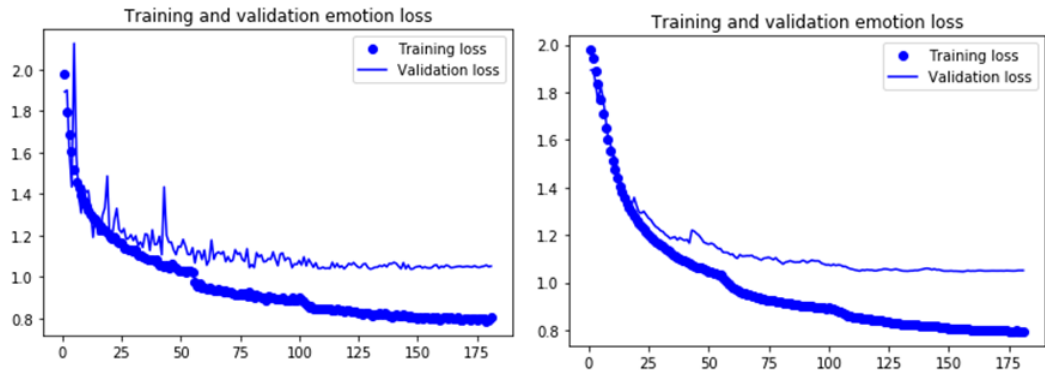


Figure 69 - Emotion training and validation loss (real and smoothed)

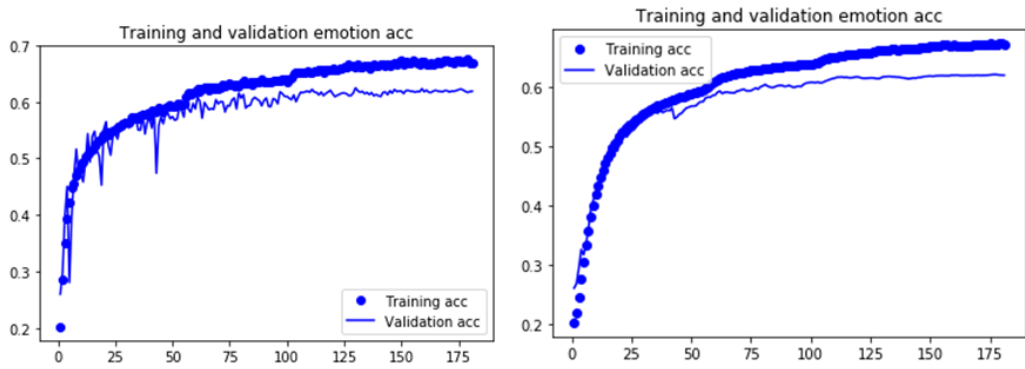


Figure 70 - Emotion training and validation metrics (real and smoothed)

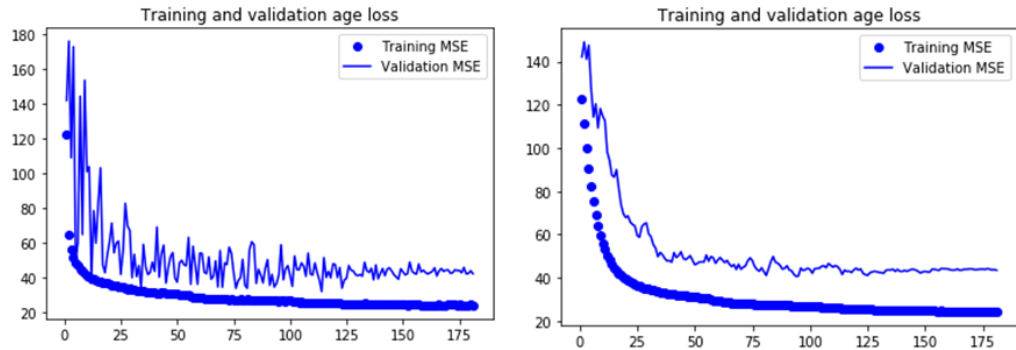


Figure 71 - Age training and validation loss (real and smoothed)

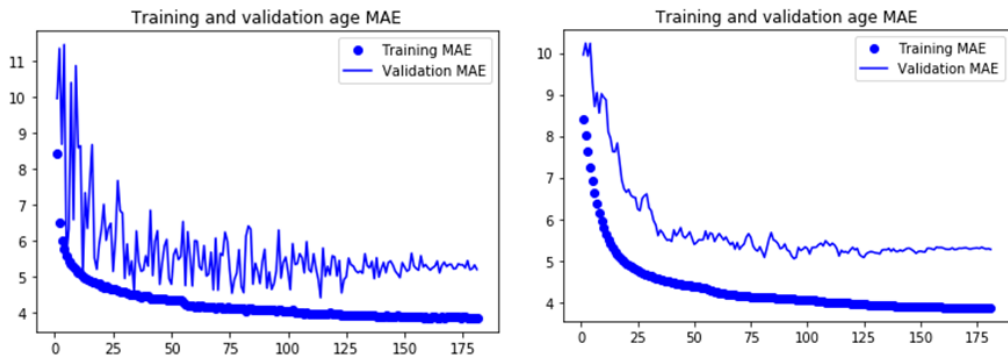


Figure 72 - Age training and validation metrics (real and smoothed)

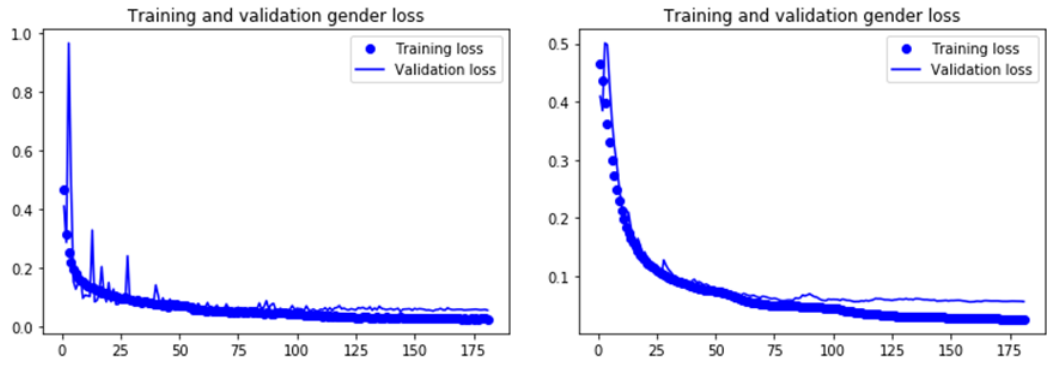


Figure 73 - Gender training and validation loss (real and smoothed)

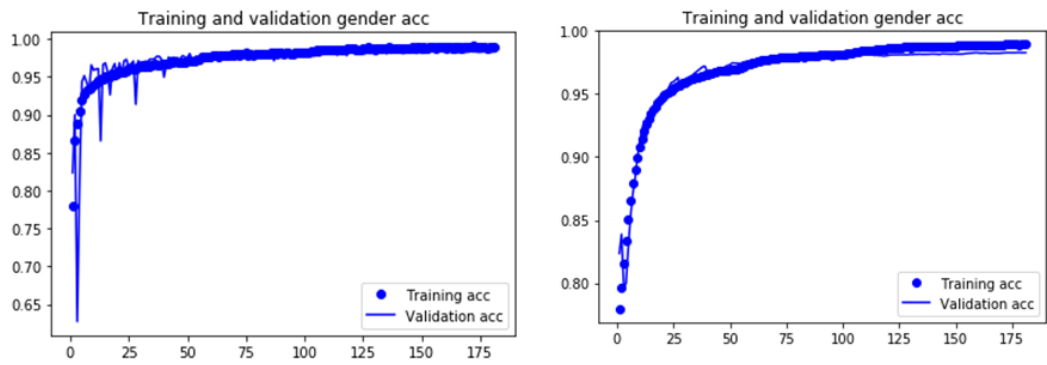


Figure 74 - Gender training and validation metrics (real and smoothed)

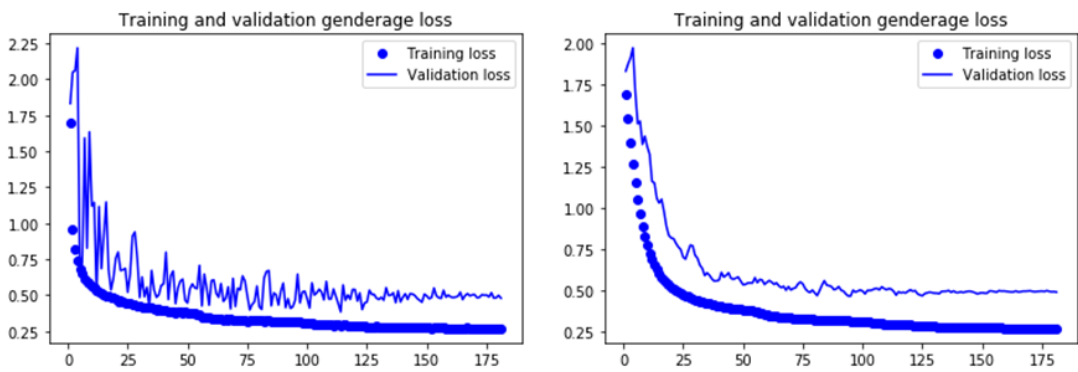


Figure 75 - Aggregated gender and age training and validation loss (real and smoothed)

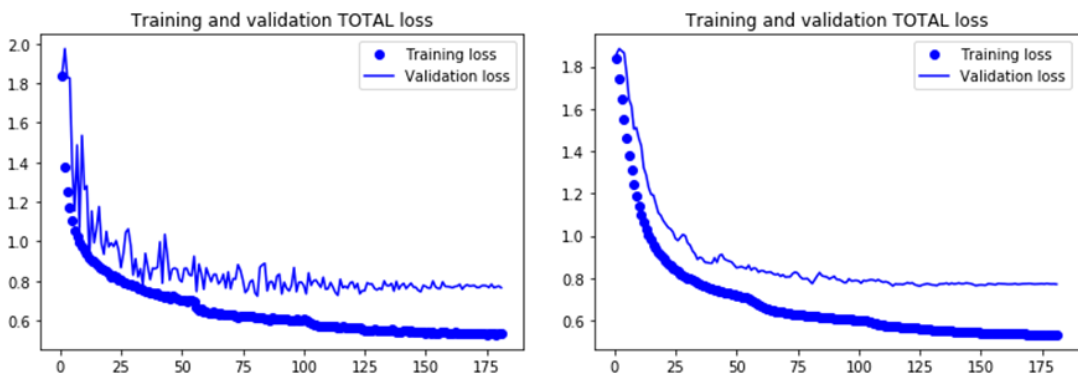


Figure 76 - Total training and validation loss (real and smoothed)

## Testing phase

The model achieves similar results as the previous one on the emotion task (Fig. 77 and Table 26 and 27). The class Fear is again the worst F1 metric score, and Happy the most distinguishable class throughout all the metrics. The age (Fig. 78) and gender (Fig. 79) results are consistent with the other models.

An accuracy of 0.6080 and cross-entropy loss of 1.0525 were achieved on the emotion test set evaluation. It took around  $2 \times 10^{-3}$  seconds per image. The evaluation on the age test set yielded a 4.4367 MAE and a 32.2955 MSE loss, taking approximately  $2 \times 10^{-3}$  seconds per image. The gender task yielded a 0.9826 accuracy and a 0.0501 cross-entropy loss on the test set evaluation, which took about  $2 \times 10^{-3}$  seconds per image.

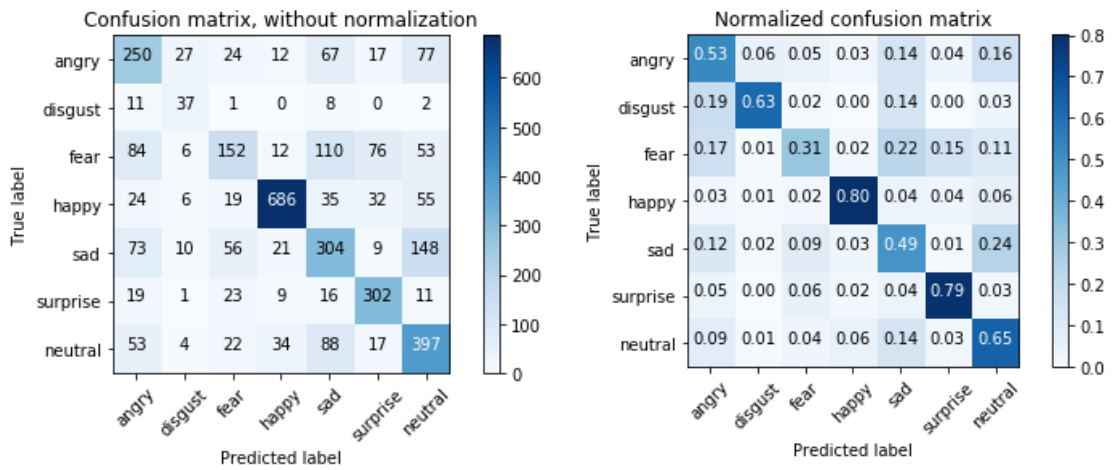


Figure 77 - Emotion confusion matrix

|          | PRECISION | RECALL | F1     |
|----------|-----------|--------|--------|
| ANGRY    | 0.4864    | 0.5274 | 0.5061 |
| DISGUST  | 0.4066    | 0.6271 | 0.4933 |
| FEAR     | 0.5118    | 0.3083 | 0.3848 |
| HAPPY    | 0.8863    | 0.8005 | 0.8412 |
| SAD      | 0.4841    | 0.4895 | 0.4868 |
| SURPRISE | 0.6667    | 0.7927 | 0.7243 |
| NEUTRAL  | 0.5343    | 0.6455 | 0.5847 |

Table 26 – Emotion class metrics

|          | ACCURACY | PRECISION | RECALL | F1     |
|----------|----------|-----------|--------|--------|
| OVERALL  | 0.6080   | 0.5680    | 0.5987 | 0.5744 |
| WEIGHTED | -        | 0.6142    | 0.6080 | 0.6050 |

Table 27 – Emotion metrics

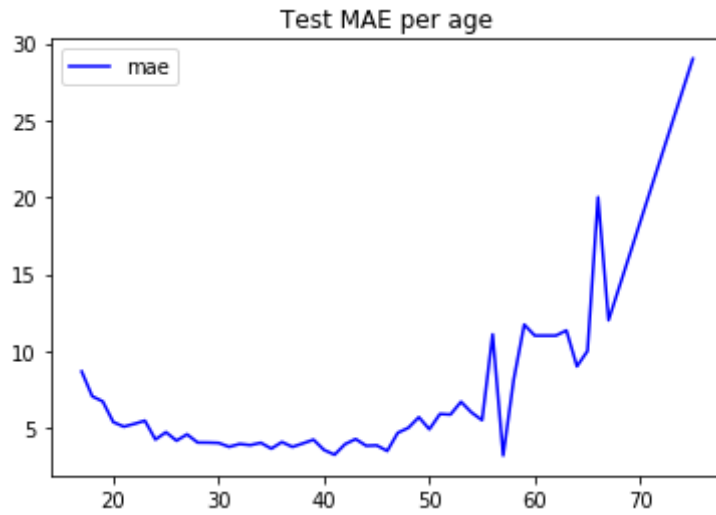


Figure 78 – MAE per age results

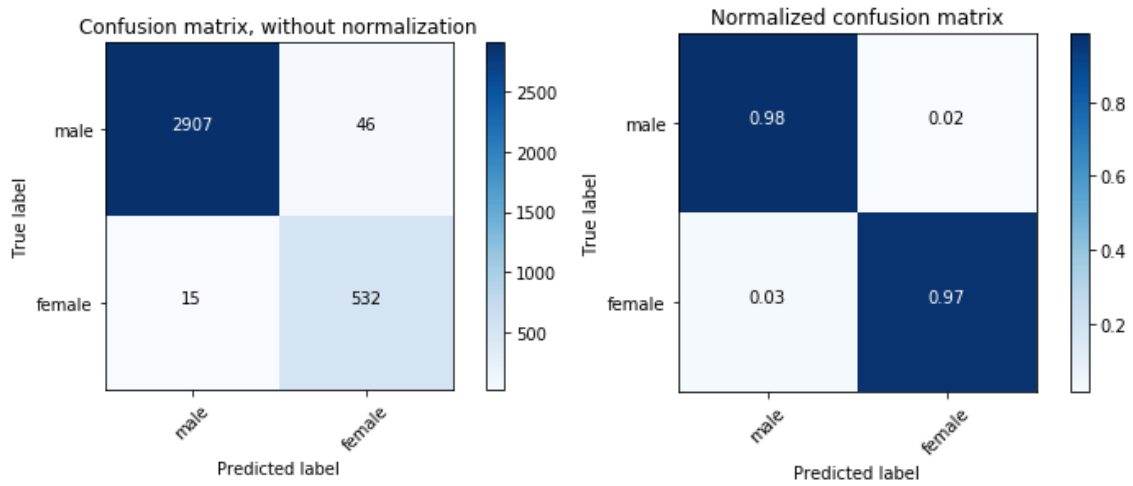


Figure 79 – Gender confusion matrix

|        | PRECISION | RECALL | F1     |
|--------|-----------|--------|--------|
| MALE   | 0.9949    | 0.9844 | 0.9896 |
| FEMALE | 0.9204    | 0.9726 | 0.9458 |

Table 28 – Gender class metrics

|          | ACCURACY | PRECISION | RECALL | F1     |
|----------|----------|-----------|--------|--------|
| OVERALL  | 0.9826   | 0.9577    | 0.9785 | 0.9677 |
| WEIGHTED | -        | 0.9833    | 0.9826 | 0.9828 |

Table 29 – Gender metrics

After completing the outline of the single-task and multi-task models results of the training and testing phase, a model metrics comparison is provided in the next section.

## 4.3. Comparisons

### 4.3.1. Emotion task model comparison

Through careful examination of Tables 30 and 31, some conclusion can be attained. It is important to state that most differences are minute.

Interpreting Table 30, the general notion is that the emotion single-task model is consistently the model with higher metrics. There are some exceptions, such as the highest precision score for the Anger class, it is reached by model V1. But closely looking at the other metrics, recall and F1 score, the emotion single-task model outperforms V1 by a slight margin. Thus, given all the predicted labels for class Anger, V1 model is the best with the correctly predicted samples – has the highest precision. Nonetheless, for all the ground truth Anger samples, the emotion single-task model correctly predicts the most – has the highest recall. The F1 measure is highest for the single-task model, consequently overall it may be a better choice (as this measure is a function of precision and recall). The Disgust class has the better precision and F1 scores with the single-task model, and the higher recall with model V3. This model achieves one of the lowest precisions. The best model for the class Fear is the single-task. Class Sad is the only one in which the single-task model does not attain the highest F1 score, achieved by model V1, as the highest recall. Class Happy and Surprise highest recall is reached by model V3, as the highest precision for the class Sad.

Looking at the global classes results, the highest metrics scores are consistent across the models for class Happy. This is expected as this class is the one with most samples. Surprise is the second most distinguishable class. Class Disgust has the lowest F1 performance in the model V2, and has the worst precision in models V1, V2, V3 and V4. This is the class with the fewest samples. However, from the top results, the lowest F1 is reached by class Fear, so this is the least distinguishable class.

Table 31 shows the emotion task overall scores. Again, most differences are small. The highest results are obtained in the single-task emotion model. Comparing the multi-task models, the highest results tend to be achieved by model V4, closely followed by model V1. The lowest scores are achieved by model V2. The different scores are not significantly different, no model is much better than the others.

| MODELS                     | ANGER         |               |               | DISGUST       |               |               | FEAR          |               |               | HAPPY         |              |               | SAD           |               |               | SURPRISE      |               |               | NEUTRAL       |               |               |
|----------------------------|---------------|---------------|---------------|---------------|---------------|---------------|---------------|---------------|---------------|---------------|--------------|---------------|---------------|---------------|---------------|---------------|---------------|---------------|---------------|---------------|---------------|
|                            | Prec          | Rec           | F1            | Prec          | Rec           | F1            | Prec          | Rec           | F1            | Prec          | Rec          | F1            | Prec          | Rec           | F1            | Prec          | Rec           | F1            | Prec          | Rec           | F1            |
| EMOTION                    | 0.485         | <b>0.5443</b> | <b>0.5129</b> | <b>0.4933</b> | 0.6271        | <b>0.5522</b> | <b>0.5141</b> | <b>0.4077</b> | <b>0.4548</b> | 0.8718        | 0.825        | <b>0.8478</b> | 0.5072        | 0.5121        | 0.5096        | <b>0.7394</b> | 0.7297        | <b>0.7345</b> | 0.564         | 0.6309        | <b>0.5956</b> |
| TRIPLE-MODULE SHARING (V1) | <b>0.4928</b> | 0.5042        | 0.4984        | 0.354         | 0.678         | 0.4651        | 0.4652        | 0.3529        | 0.4013        | 0.8685        | 0.7935       | 0.8293        | <b>0.4543</b> | <b>0.6248</b> | <b>0.5261</b> | 0.6919        | 0.7428        | 0.7164        | <b>0.6183</b> | 0.4846        | 0.5433        |
| DOUBLE-MODULE SHARING (V2) | 0.4888        | 0.5042        | 0.4964        | 0.2774        | 0.7288        | 0.4018        | 0.4481        | 0.3854        | 0.4144        | <b>0.8918</b> | 0.7503       | 0.8150        | 0.4734        | 0.4734        | 0.4734        | 0.6651        | 0.7612        | 0.7099        | 0.5459        | 0.5805        | 0.5627        |
| SINGLE-MODULE SHARING (V3) | 0.4332        | 0.5401        | 0.4808        | 0.3194        | <b>0.7797</b> | 0.4532        | 0.4673        | 0.2901        | 0.3580        | 0.845         | <b>0.846</b> | 0.8455        | <b>0.569</b>  | 0.3849        | 0.4592        | 0.6593        | <b>0.7874</b> | 0.7177        | 0.5427        | 0.6407        | 0.5876        |
| ZERO-MODULE SHARING (V4)   | 0.4864        | 0.5274        | 0.5061        | 0.4066        | 0.6271        | 0.4933        | 0.5118        | 0.3083        | 0.3848        | 0.8863        | 0.8005       | 0.8412        | 0.4841        | 0.4895        | 0.4868        | 0.6667        | 0.7927        | 0.7243        | 0.5343        | <b>0.6455</b> | 0.5847        |

Table 30 – Emotion class model comparison

| MODELS                     | ACCURACY | MACRO-AVERAGED PRECISION | MACRO-AVERAGED RECALL | MACRO-AVERAGED F1 |
|----------------------------|----------|--------------------------|-----------------------|-------------------|
|                            |          | Weighted Precision       | Weighted Recall       | Weighted F1       |
| EMOTION                    | 0.6249   | 0.5964                   | 0.6110                | 0.6011            |
| TRIPLE-MODULE SHARING (V1) | 0.6006   | 0.6295                   | 0.6249                | 0.6254            |
| DOUBLE-MODULE SHARING (V2) | 0.5874   | 0.5636                   | 0.5973                | 0.5686            |
| SINGLE-MODULE SHARING (V3) | 0.6009   | 0.6155                   | 0.6006                | 0.6017            |
| ZERO-MODULE SHARING (V4)   | 0.6080   | 0.5415                   | 0.5977                | 0.5534            |
|                            |          | 0.6047                   | 0.5874                | 0.5921            |
|                            |          | 0.5480                   | 0.6098                | 0.5574            |
|                            |          | 0.6049                   | 0.6009                | 0.5931            |
|                            |          | 0.5680                   | 0.5987                | 0.5744            |
|                            |          | 0.6142                   | 0.6080                | 0.6050            |

Table 31 – Emotion model comparison

#### 4.3.1. Age task model comparison

Table 32 shows the age task comparison. There is no outstanding model, as again, the differences are subtle. Nonetheless, the lowest MAE is achieved in the single-task age model, closely followed by model V3. The lowest performance is achieved by model V1.

| MODELS                        | MEAN ABSOLUTE ERROR |
|-------------------------------|---------------------|
| AGE                           | 4.2468              |
| TRIPLE-MODULE<br>SHARING (V1) | 4.6825              |
| DOUBLE-MODULE<br>SHARING (V2) | 4.5901              |
| SINGLE-MODULE<br>SHARING (V3) | 4.2883              |
| ZERO-MODULE<br>SHARING (V4)   | 4.4367              |

Table 32 – Age model comparison

### 4.3.1. Gender task model comparison

The gender class model comparable can be seen in Table 33. All the models are quite similar, again with no major differences. The highest male precision and female recall are attained by model V4, but the other metrics are highest for the single-task model. The F1 scores are greater in this model.

Table 34 shows the gender task comparison. Consistent with the previous table, all the models are alike, particularly the single-task gender model and the V1, with slightly better results achieved by the first.

| MODELS                            | MALE   |        |        | FEMALE |        |        |
|-----------------------------------|--------|--------|--------|--------|--------|--------|
|                                   | Prec   | Rec    | F1     | Prec   | Rec    | F1     |
| <b>GENDER</b>                     | 0.9909 | 0.9932 | 0.9920 | 0.963  | 0.9506 | 0.9568 |
| <b>TRIPLE-MODULE SHARING (V1)</b> | 0.9848 | 0.9875 | 0.9861 | 0.9314 | 0.9526 | 0.9419 |
| <b>DOUBLE-MODULE SHARING (V2)</b> | 0.9918 | 0.9858 | 0.9888 | 0.9257 | 0.9561 | 0.9407 |
| <b>SINGLE-MODULE SHARING (V3)</b> | 0.9865 | 0.9898 | 0.9881 | 0.9441 | 0.9269 | 0.9354 |
| <b>ZERO-MODULE SHARING (V4)</b>   | 0.9949 | 0.9844 | 0.9896 | 0.9204 | 0.9726 | 0.9458 |

Table 33 – Gender class model comparison

| MODELS                            | ACCURACY | MACRO-AVERAGED PRECISION | MACRO-AVERAGED RECALL | MACRO-AVERAGED F1 |
|-----------------------------------|----------|--------------------------|-----------------------|-------------------|
|                                   |          | Weighted Precision       | Weighted Recall       | Weighted F1       |
| <b>GENDER</b>                     | 0.9866   | 0.9770                   | 0.9719                | 0.9744            |
|                                   |          | 0.9865                   | 0.9865                | 0.9865            |
|                                   |          |                          |                       |                   |
| <b>TRIPLE-MODULE SHARING (V1)</b> | 0.9766   | 0.9581                   | 0.9701                | 0.9640            |
|                                   |          | 0.9765                   | 0.9820                | 0.9792            |
| <b>DOUBLE-MODULE SHARING (V2)</b> | 0.9811   | 0.9588                   | 0.9710                | 0.9647            |
|                                   |          | 0.9815                   | 0.9812                | 0.9813            |

|                                   |        |        |        |        |
|-----------------------------------|--------|--------|--------|--------|
| <b>SINGLE-MODULE SHARING (V3)</b> | 0.9800 | 0.9653 | 0.9584 | 0.9618 |
|                                   |        | 0.9799 | 0.9800 | 0.9799 |
| <b>ZERO-MODULE SHARING (V4)</b>   | 0.9826 | 0.9577 | 0.9785 | 0.9677 |
|                                   |        | 0.9833 | 0.9826 | 0.9828 |

Table 34 – Gender model comparison

### 4.3.1. Global model comparison

Table 35 shows the overall model comparison, with all the tasks. The single-task models are consistently the top performers in the respective task. The differences are not very significant, though. The age estimation single-task model MAE is very close to the V1 MAE score, and the Gender single-task scores are close to the V4 ones.

As noted, comparing the multi-task models one does not note an outstanding model, results are quite similar. The task of emotion recognition and gender recognition is slightly better performed by model V4, and the age estimation task is best performed by model V3.

| MODELS                     | EMOTION RECOGNITION |           |        |        | AGE ESTIMATION | GENDER RECOGNITION |           |        |        |
|----------------------------|---------------------|-----------|--------|--------|----------------|--------------------|-----------|--------|--------|
| EMOTION                    | Accuracy            | Precision | Recall | F1     | MAE            | Accuracy           | Precision | Recall | F1     |
|                            | 0.6249              | 0.6295    | 0.6249 | 0.6254 | -              | -                  | -         | -      | -      |
| AGE                        | -                   | -         | -      | -      | 4.2468         | -                  | -         | -      | -      |
| GENDER                     | -                   | -         | -      | -      | -              | 0.9866             | 0.9865    | 0.9865 | 0.9865 |
| TRIPLE-MODULE SHARING (V1) | 0.6006              | 0.6155    | 0.6006 | 0.6017 | 4.6825         | 0.9766             | 0.9765    | 0.9820 | 0.9792 |
| DOUBLE-MODULE SHARING (V2) | 0.5874              | 0.6047    | 0.5874 | 0.5921 | 4.5901         | 0.9811             | 0.9815    | 0.9812 | 0.9813 |
| SINGLE-MODULE SHARING (V3) | 0.6009              | 0.6049    | 0.6009 | 0.5931 | 4.2883         | 0.9800             | 0.9799    | 0.9800 | 0.9799 |
| ZERO-MODULE SHARING (V4)   | 0.6080              | 0.6142    | 0.6080 | 0.6050 | 4.4367         | 0.9826             | 0.9833    | 0.9826 | 0.9828 |

Table 35 – Model overall comparison (Weighted method)

As noted during the interpretation of all the task scores throughout the section, the results tend to be quite comparable with no outstanding model. This is exciting, because even though the single-task models are focused on a solo task, a model with three shared modules attains similar results. This means that developing a multi-task model for the tasks of emotion recognition, age estimation and gender recognition, with fewer parameters than three single-task models for such tasks, sharing much of the network framework and training simultaneously achieves comparable results. Accordingly, the computation expense of deploying three different models for three different tasks can be diminished by developing a single multi-task model. Alongside this ease of the computation demand, the inference time is obviously faster, as there is no redundancy, and the training process is faster as well.

A final conclusions chapter follows, discussing the dissertation overall achievements, some limitations and encountered problems, and future work.

## 5. Conclusions and final thoughts

The present study on facial analysis, particularly emotion recognition, age estimation and gender recognition, proposed developing a Multi-task learning framework capable of solving the three tasks simultaneously in a fast and efficient way.

Various models were developed using a chosen CNN as the baseline architecture, derived from the known Xception, altering and adapting the network to various Multi-task structures. The difference between the various developed models relies on the number of shared residual modules. Having two datasets, one with emotion labels and one with gender and age labels, the networks were trained using two multi-task learning methods: Task-based Regularization and Domain-based Regularization. The Task-based Regularization approach uses an aggregated loss function as the objective function to train the CNN. This approach was used to train the gender and age tasks. The Domain-based Regularization approach implies that sharing the parameters between tasks trained with task-related datasets makes the shared parameters adapt to the complete set of tasks, instead of fitting to a task-specific domain, inducing a better generalization capacity to the network. The method to train in such a way, batch by batch training, was developed specifically for this dissertation.

The models' results are successful, supporting the developed multi-task learning method success. Furthermore, the results state comparable results between the multi-task learning models and the single-task ones, providing evidence that a model with shared layers – more efficient, faster at inference times and lighter – is a better choice if one seeks solving different tasks simultaneously.

The present research is nonetheless affected by faults and limitations. One of the limitations concerns the used gender and age dataset, which is biased towards dark-skinned individuals, males, and the age interval of 25 to 55 years old, approximately. Also, the Disgust class of the emotion dataset has quite few samples.

Future work would benefit from testing the framework with different baseline architectures and different datasets. Additionally, the development of an end-to-end system would be appropriate, as in this way it would be possible to deploy the model in a live (or just video footage) setting. The model is very capable of such, as it is small, fast and efficient. There are various face detection and alignment CNNs available, making the development of an end-to-end system quite simple: by implementing an available system, i.e. the MTCNN (K. Zhang et al., 2016), and feeding the outputted aligned face image to the developed model.

Finally, it would be interesting to implement this model on a live business service. There are various situations in which it could be implemented, such as crowd analytics, intelligent marketing, and HCI (human-computer interaction). This is the current tendency: one small step on the Computer Vision field, a giant leap for the fusion between humankind and technology.

## References

- Abousaleh, F. S., Lim, T., Cheng, W.-H., Yu, N.-H., Hossain, M. A., & Alhamid, M. F. (2016). A novel comparative deep learning framework for facial age estimation. *EURASIP Journal on Image and Video Processing*, 2016(1), 47. <https://doi.org/10.1186/s13640-016-0151-4>
- Arriaga, O., Valdenegro-Toro, M., & Plöger, P. (2017). Real-time Convolutional Neural Networks for Emotion and Gender Classification. Retrieved from <http://arxiv.org/abs/1710.07557>
- Baluja, S., & Rowley, H. A. (2007). Boosting sex identification performance. *International Journal of Computer Vision*. <https://doi.org/10.1007/s11263-006-8910-9>
- Bordes, A., Glorot, X., Weston, J., & Bengio, Y. (2012). Joint learning of words and meaning representations for open-text semantic parsing. *Artificial Intelligence and Statistics*, 127–135. <https://doi.org/10.1109/CVPR.2015.7298594>
- Bruna, J., Szlam, A., & LeCun, Y. (2014). Signal Recovery from Pooling Representations. *ArXiv Preprint*. Retrieved from <http://arxiv.org/abs/1311.4025>
- Cadène, R., Thome, N., & Cord, M. (2016). Master's Thesis : Deep Learning for Visual Recognition. Retrieved from <http://arxiv.org/abs/1610.05567>
- Canziani, A., Paszke, A., & Culurciello, E. (2017). An Analysis of Deep Neural Network Models for Practical Applications, 1–7. Retrieved from <http://arxiv.org/abs/1605.07678>
- Caruana, R. (1997). Multitask learning. In *Machine learning* (pp. 41–75). <https://doi.org/10.1109/TCBB.2010.22>
- Chen, D., Ren, S., Wei, Y., Cao, X., & Sun, J. (2014). Joint cascade face detection and alignment. In *European Conference on Computer Vision* (pp. 109–122). [https://doi.org/10.1007/978-3-319-10599-4\\_8](https://doi.org/10.1007/978-3-319-10599-4_8)
- Chollet, F. (2017). Xception: Deep learning with depthwise separable convolutions. *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, 2017-Janua*, 1800–1807. <https://doi.org/10.1109/CVPR.2017.195>
- Ciresan, D., Meier, U., Masci, J., Gambardella, L. M., & Schmidhuber, J. (2011). High-Performance Neural Networks for Visual Object Classification. *Advances In Neural Information Processing Systems*, 12. <https://doi.org/http://arxiv.org/abs/1102.0183>
- Ciresan, D., Meier, U., & Schmidhuber, J. (2012a). Multi-column Deep Neural Networks for

- Image Classification. *International Conference of Pattern Recognition*, (February), 3642–3649. <https://doi.org/10.1109/CVPR.2012.6248110>
- Ciresan, D., Meier, U., & Schmidhuber, J. (2012b). Transfer Learning for Latin and Chinese Characters with Deep Neural Networks. *Neural Networks (IJCNN), The 2012 International Joint Conference On*, 1–6. <https://doi.org/10.1109/IJCNN.2012.6252544>
- Clevert, D.-A., Unterthiner, T., & Hochreiter, S. (2016). Fast and Accurate Deep Network Learning By Exponential Linear Units (Elus). *Iclr 2016*, 1–14.
- Cook, A. (2017). Batch Normalization. Retrieved August 10, 2018, from <https://alexisbcook.github.io/2017/global-average-pooling-layers-for-object-localization/>
- Cope, G. (2017). Kernels in Image Processing. Retrieved November 30, 2017, from <http://www.naturefocused.com/articles/photography-image-processing-kernel.html>
- Dauphin, Y., Pascanu, R., Gulcehre, C., Cho, K., Ganguli, S., & Bengio, Y. (2014). Identifying and attacking the saddle point problem in high-dimensional non-convex optimization, 1–14. Retrieved from <http://arxiv.org/abs/1406.2572>
- Dehghan, A., Ortiz, E. G., Shu, G., & Masood, S. Z. (2017). DAGER: Deep Age, Gender and Emotion Recognition Using Convolutional Neural Network. Retrieved from <http://arxiv.org/abs/1702.04280>
- Ekmekji, A. (2016). Convolutional Neural Networks for Age and Gender Classification. *Technical Report*.
- Eldan, R., & Shamir, O. (2016). The Power of Depth for Feedforward Neural Networks, 1–33. Retrieved from <http://arxiv.org/abs/1512.03965>
- FER-2013. (2018). Retrieved December 12, 2017, from <https://www.kaggle.com/c/challenges-in-representation-learning-facial-expression-recognition-challenge/data>
- Gao, F., & Ai, H. (2009). Face age classification on consumer images with gabor feature and fuzzy LDA method. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* (Vol. 5558 LNCS, pp. 132–141). [https://doi.org/10.1007/978-3-642-01793-3\\_14](https://doi.org/10.1007/978-3-642-01793-3_14)
- Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (pp. 580–587). <https://doi.org/10.1109/CVPR.2014.81>

- Golomb, B. A., Lawrence, D. T., & Sejnowski, T. J. (1991). Sexnet: A neural network identifies sex from human faces. *Advances in Neural Information Processing Systems 3*, (July), 572–7. Retrieved from <http://dl.acm.org/citation.cfm?id=118953>
- Gomez, V., Cortes, A., & Noguer, F. (2015). Object Detection for Autonomous Driving Using Deep Learning. *Meeting of the Universitat Politecnica de Catalunya, Spain*, (December).
- Gong, Y., Wang, L., Guo, R., & Lazebnik, S. (2014). Multi-scale orderless pooling of deep convolutional activation features. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* (Vol. 8695 LNCS, pp. 392–407). [https://doi.org/10.1007/978-3-319-10584-0\\_26](https://doi.org/10.1007/978-3-319-10584-0_26)
- Gu, J., Wang, Z., Kuen, J., Ma, L., Shahroudy, A., Shuai, B., ... Wang, G. (2017). Recent Advances in Convolutional Neural Networks. *ArXiv*, 1–14. <https://doi.org/10.3389/fpsyg.2013.00124>
- Guo, G., & Mu, G. (2014). A framework for joint estimation of age, gender and ethnicity on a large database. *Image and Vision Computing*, 32(10), 761–770. <https://doi.org/10.1016/j.imavis.2014.04.011>
- Guo, Y., Liu, Y., Oerlemans, A., Lao, S., Wu, S., & Lew, M. S. (2016). Deep learning for visual understanding: A review. *Neurocomputing*, 187, 27–48. <https://doi.org/10.1016/j.neucom.2015.09.116>
- He, K., Zhang, X., Ren, S., & Sun, J. (2015a). Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE International Conference on Computer Vision* (Vol. 2015 Inter, pp. 1026–1034). <https://doi.org/10.1109/ICCV.2015.123>
- He, K., Zhang, X., Ren, S., & Sun, J. (2015b). Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(9), 1904–1916. <https://doi.org/10.1109/TPAMI.2015.2389824>
- He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 770–778). <https://doi.org/10.1109/CVPR.2016.90>
- Ioffe, S., & Szegedy, C. (2015). Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. Retrieved from <http://arxiv.org/abs/1502.03167>
- Karpathy, A. (2018). Convolutional Neural Networks for Visual Recognition. Retrieved June 19, 2018, from <http://cs231n.github.io/optimization-1/#optimization>
- Keras. (2018). Retrieved January 15, 2018, from <https://keras.io/>

- Kingma, D. P., & Ba, J. (2017). Adam: A Method for Stochastic Optimization, 1–15.  
Retrieved from <http://arxiv.org/abs/1412.6980>
- Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet Classification with Deep Convolutional Neural Networks. *Advances In Neural Information Processing Systems*, 1–9.  
<https://doi.org/http://dx.doi.org/10.1016/j.protcy.2014.09.007>
- LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, 521(7553), 436–444.  
<https://doi.org/10.1038/nature14539>
- LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11), 2278–2323.  
<https://doi.org/10.1109/5.726791>
- Levi, G., & Hassner, T. (2015). Age and gender classification using convolutional neural networks. In *2015 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)* (pp. 34–42). <https://doi.org/10.1109/CVPRW.2015.7301352>
- Lin, M., Chen, Q., & Yan, S. (2014). Network In Network. *ArXiv Preprint*, 10.  
<https://doi.org/10.1109/ASRU.2015.7404828>
- Liu, C., Yuen, J., Torralba, A., Viorica, P., Jégou, H., Douze, M., ... Patrick, P. (2014). Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(11), 346–361.  
<https://doi.org/10.1109/TPAMI.2015.2389824>
- Lu, Y., Kim, S. S.-M. S., Hovy, E., Kim, S. S.-M. S., Hovy, E., Hovy, E., ... Bengio, Y. (2016). Improving neural networks by preventing co-adaptation of feature detectors. *Proceedings of the 30th Conference on Artificial Intelligence (AAAI 2016)*, 48(1), 1–9.  
<https://doi.org/10.1007/s10107-014-0839-0>
- Luu, K., Ricanek, K., Bui, T. D., & Suen, C. Y. (2009). Age estimation using Active Appearance Models and Support Vector Machine regression. *Proc. IEEE Intl. Conf. on Biometrics: Theory, Applications, and Systems*, 1–5.  
<https://doi.org/10.1109/BTAS.2009.5339053>
- Maas, A. L., Hannun, A. Y., & Ng, A. Y. (2013). Rectifier Nonlinearities Improve Neural Network Acoustic Models. *Proceedings of the 30 Th International Conference on Machine Learning*, 28, 6. Retrieved from [https://web.stanford.edu/~awni/papers/relu\\_hybrid\\_icml2013\\_final.pdf](https://web.stanford.edu/~awni/papers/relu_hybrid_icml2013_final.pdf)
- Matplotlib. (2018). Retrieved June 15, 2018, from <https://matplotlib.org/index.html>
- Mehdi Ghayoumi. (2017). A Quick Review of Deep Learning in Facial Expression. *Journal of*

- Communication and Computer*, 14(1), 34–38. <https://doi.org/10.17265/1548-7709/2017.01.004>
- Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems* (pp. 3111–3119).
- Moghaddam, B., & Yang, M. H. (2002). Learning gender with support faces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5), 707–711. <https://doi.org/10.1109/34.1000244>
- MORPH. (2018). Retrieved December 12, 2017, from <https://www.faceaginggroup.com/morph/>
- Nair, V., & Hinton, G. E. (2010). Rectified Linear Units Improve Restricted Boltzmann Machines. *Proceedings of the 27th International Conference on Machine Learning*, (3), 807–814. <https://doi.org/10.1.1.165.6419>
- Ng, C. B., Tay, Y. H., & Goi, B. M. (2015). A review of facial gender recognition. *Pattern Analysis and Applications*, 18(4), 739–755. <https://doi.org/10.1007/s10044-015-0499-6>
- NumPy. (2018). Retrieved June 15, 2018, from <http://www.numpy.org/>
- Ouyang, W., Wang, X., Zeng, X., Qiu, S., Luo, P., Tian, Y., ... Tang, X. (2015). DeepID-Net: Deformable deep convolutional neural networks for object detection. *Computer Vision and Pattern Recognition (CVPR), 2015 IEEE Conference On*, 2403–2412. <https://doi.org/10.1109/CVPR.2015.7298854>
- pandas. (2018). Retrieved June 15, 2018, from <https://pandas.pydata.org/>
- Python. (2018). Retrieved January 15, 2018, from <https://www.python.org/>
- Ranjan, R., Patel, V. M., & Chellappa, R. (2017). HyperFace: A Deep Multi-task Learning Framework for Face Detection, Landmark Localization, Pose Estimation, and Gender Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. Retrieved from <http://arxiv.org/abs/1603.01249>
- Ranjan, R., Sankaranarayanan, S., Castillo, C. D., & Chellappa, R. (2017). An All-In-One Convolutional Neural Network for Face Analysis. *Proceedings - 12th IEEE International Conference on Automatic Face and Gesture Recognition, FG 2017 - 1st International Workshop on Adaptive Shot Learning for Gesture Understanding and Production, ASLAGUP 2017, Biometrics in the Wild, Bwild 2017, Heteroge*, 17–24. <https://doi.org/10.1109/FG.2017.137>
- Ren, S., He, K., Girshick, R., & Sun, J. (2017). Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Transactions on Pattern Analysis and*

- Machine Intelligence*, 39(6), 1137–1149. <https://doi.org/10.1109/TPAMI.2016.2577031>
- Ricanek, K., & Tesafaye, T. (2006). MORPH: A longitudinal image database of normal adult age-progression. In *FGR 2006: Proceedings of the 7th International Conference on Automatic Face and Gesture Recognition* (Vol. 2006, pp. 341–345). <https://doi.org/10.1109/FGR.2006.78>
- Rippel, O., Snoek, J., & Adams, R. P. (2015). Spectral Representations for Convolutional Neural Networks, (2013), 1–10. Retrieved from <http://arxiv.org/abs/1506.03767>
- Ruder, S. (2017a). An overview of gradient descent optimization algorithms, 1–14. Retrieved from <http://arxiv.org/abs/1609.04747>
- Ruder, S. (2017b). An Overview of Multi-Task Learning in Deep Neural Networks, (May). Retrieved from <http://arxiv.org/abs/1706.05098>
- Savoiu, A., & Wong, J. (2017). Recognizing Facial Expressions Using Deep Learning, 67.
- Scherer, D., Müller, A., & Behnke, S. (2010). Evaluation of pooling operations in convolutional architectures for object recognition. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* (Vol. 6354 LNCS, pp. 92–101). [https://doi.org/10.1007/978-3-642-15825-4\\_10](https://doi.org/10.1007/978-3-642-15825-4_10)
- Schmidhuber, J. (2015). Deep Learning in neural networks: An overview. *Neural Networks*. <https://doi.org/10.1016/j.neunet.2014.09.003>
- scikit-learn. (2018). Retrieved January 15, 2018, from <http://scikit-learn.org/stable/>
- Simonyan, K., & Zisserman, A. (2015). Very Deep Convolutional Networks for Large-Scale Image Recognition, 1–14. Retrieved from <http://arxiv.org/abs/1409.1556>
- Sun, Y., Wang, X., & Tang, X. (2015). Deeply learned face representations are sparse, selective, and robust. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (Vol. 07–12–June, pp. 2892–2900). <https://doi.org/10.1109/CVPR.2015.7298907>
- Szegedy, C., Ioffe, S., Vanhoucke, V., & Alemi, A. (2017). Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning. *Association for the Advancement of Artificial Intelligence*, 4278–4284. <https://doi.org/10.1016/j.patrec.2014.01.008>
- Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., ... De Lathauwer, L. (2017). Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning. *Proceedings of the IEEE International Conference on Computer Vision*, 115(3), 4278–4284. <https://doi.org/10.1145/3038912.3052569>
- Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., ... Rabinovich, A. (2015).

- Going deeper with convolutions. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (Vol. 07–12–June, pp. 1–9). <https://doi.org/10.1109/CVPR.2015.7298594>
- TensorFlow. (2018). Retrieved January 15, 2018, from <https://www.tensorflow.org/>
- Tompson, J., Goroshin, R., Jain, A., Lecun, Y., & Bregler, C. (2015). Efficient Object Localization Using Convolutional Networks. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 648–656. <https://doi.org/10.1109/CVPR.2015.7298664>
- Vanhoucke, V. (2014). Learning Visual Representations at Scale. *ICLR Invited Talk*, (April).
- VanRullen, R. (2017). Perception science in the age of deep neural networks. *Frontiers in Psychology*, 8(FEB). <https://doi.org/10.3389/fpsyg.2017.00142>
- Wan, L., Zeiler, M., Zhang, S., LeCun, Y., & Fergus, R. (2013). Regularization of neural networks using dropconnect. *International Conference on Machine Learning*, (1), 1058–1066. <https://doi.org/10.1109/TPAMI.2017.2703082>
- Xing, J., Li, K., Hu, W., Yuan, C., & Ling, H. (2017). Diagnosing deep learning models for high accuracy age estimation from a single image. *Pattern Recognition*, 66, 106–116. <https://doi.org/10.1016/j.patcog.2017.01.005>
- Xu, B., Wang, N., Chen, T., & Li, M. (2015). Empirical Evaluation of Rectified Activations in Convolution Network. *ICML Deep Learning Workshop*, 1–5.
- Yang, Z., & Ai, H. (2007). Demographic Classification with Local Binary Patterns. *International Conference on Biometrics*, 464–473. [https://doi.org/10.1007/978-3-540-74549-5\\_49](https://doi.org/10.1007/978-3-540-74549-5_49)
- Yu, D., Wang, H., Chen, P., & Wei, Z. (2014). Mixed pooling for convolutional neural networks. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* (Vol. 8818, pp. 364–375). [https://doi.org/10.1007/978-3-319-11740-9\\_34](https://doi.org/10.1007/978-3-319-11740-9_34)
- Zeiler, M. (2014). Hierarchical Convolutional Deep Learning in Computer Vision. Retrieved from <http://arxiv.org/abs/1207.0151>
- Zeiler, M., & Fergus, R. (2013). Stochastic Pooling for Regularization of Deep Convolutional Neural Networks, 1–9. Retrieved from <http://arxiv.org/abs/1301.3557>
- Zhang, K., Zhang, Z., Li, Z., Member, S., Qiao, Y., & Member, S. (2016). Joint Face Detection and Alignment using Multi - task Cascaded Convolutional Networks. *Signal Processing Letters*, 23(10), 1499–1503. <https://doi.org/10.1109/LSP.2016.2603342>

Zhang, Z., Luo, P., Loy, C. C., & Tang, X. (2014). Facial landmark detection by deep multi-task learning. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* (Vol. 8694 LNCS, pp. 94–108). [https://doi.org/10.1007/978-3-319-10599-4\\_7](https://doi.org/10.1007/978-3-319-10599-4_7)