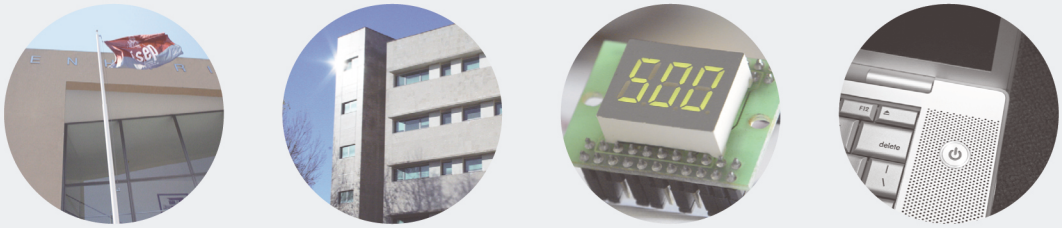
INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO
MESTRADO EM ENGENHARIA ELECTROTÉCNICA E DE COMPUTADORES



Comparação de Optimizadores de Deep Learning em Reconhecimento de Expressões Faciais

MIGUEL ANTÓNIO MADUREIRA FONTOURA ALVES
Outubro de 2022

POLITÉCNICO
DO PORTO

Politécnico do Porto
Instituto Superior de Engenharia do Porto

Deep Learning Optimizers Comparison in Facial Expression Recognition

Miguel Alves

Master in Electrical and Computer Engineering
Specialization Area of Automation and Systems

Departamento de Engenharia Eletrotécnica
Instituto Superior de Engenharia do Porto

October, 2022

This dissertation partially satisfies the requirements of the Thesis/Dissertation course of the program Master in Electrical and Computer Engineering, Specialization Area of Automation and Systems.

Candidate: Miguel Alves, No. 1170728, 1170728@isep.ipp.pt

Scientific Guidance: Paula Viana, pmv@isep.ipp.pt

Scientific Co-Guidance: Pedro Carvalho, pms@isep.ipp.pt

Company: INESC TEC

Advisor: Paula Viana, pmv@isep.ipp.pt

Departamento de Engenharia Eletrotécnica
Instituto Superior de Engenharia do Porto
Rua Dr. António Bernardino de Almeida, 431, 4200–072 Porto

October, 2022

Acknowledgements

I would first like to thank my supervisors Paula Viana and Pedro Carvalho for guiding me and making possible for me to learn and work with a concept that has always fascinated me.

I want also to thank my family for always being there for me and helping in any way they can, in the best and the worst of times.

Finally, I would like to thank everyone that has influenced me in any type of way and made me the person I am today.

Abstract

Artificial Intelligence is everywhere we go, whether it is programming an interactive cleaning robot or detecting a bank fraud. Its rise is inevitable. In the last few decades, many new architectures and approaches were brought up, so it becomes hard to know what is the best approach or architecture for a certain area. One of such areas is the detection of emotion in the human face, most commonly known by Facial Expression Recognition (or FER).

In this work we started by doing an intensive collection of data concerning the theories that explain the existence of emotions, how they are distinguished from one another, and how they are recognized in a human face. After this, we started to develop deep learning models with different architectures as to compare their performances when used for Facial Expression Recognition.

After developing the models, we took one of them and tested it with different deep learning optimizer algorithms, as to verify the difference among them, thus figuring out the best optimizing algorithm for this particular case.

Keywords: Artificial Intelligence, motion, FER, deep learning, optimizer.

Resumo

A Inteligência Artifical encontra-se presente em todo o lado, quer seja a programar um robô de limpeza interativo ou a detetar uma fraude bancária. A sua ascensão é inevitável. Nas últimas décadas, foram criadas inúmeras novas arquiteturas e abordagens e, por isso, torna-se difícil saber qual a melhor abordagem ou arquitetura para uma certa área. Uma dessas áreas é a deteção de emoção na cara humana, também conhecida como Reconhecimento de Expressão Facial.

Neste trabalho começámos por realizar uma coleta intensiva de dados acerca das teorias que explicam a existência de emoções, como as mesmas são distinguidas umas das outras e como podem ser identificadas numa cara humana. Posteriormente, começámos a desenvolver modelos de *deep learning* com diferentes arquiteturas para comparar os respetivos desempenhos quando usadas em Reconhecimento de Expressão Facial.

Após desenvolver os modelos, pegámos num dos mesmos e testámo-lo com diferentes algoritmos de otimização *deep learning* de forma a verificar quais as diferenças entre os mesmos, percebendo assim qual o mais indicado para uso neste caso em particular.

Keywords: Inteligência Artificial, emoção, Reconhecimento de Expressão Facial, *deep learning*, otimizador.

Contents

List of Figures **vii**

List of Tables **xi**

List of Acronyms **xiii**

1 Introduction **1**
1.1 Context 1
1.2 Definition Of The Problem 1
1.2.1 Goals 2
1.3 Organization Of The Dissertation 2

2 Human Emotion **3**
2.1 The Emotion Theory 4
2.1.1 Discrete Emotion Theory 5
2.1.2 Dimensional Theory of Emotion 6
2.1.3 Emotions elements 7
2.2 Paul Ekman and the creation of FACS 8
2.3 Wheels of Emotion 11
2.3.1 Plutchik's Wheel of Emotion 11
2.3.2 Geneva Emotion Wheel 13
2.3.3 Difference between Plutchik's Wheel of Emotions and the GEW 14

3 Facial Expression Recognition **15**
3.1 Facial Expression Recognition Datasets 15
3.2 Data Augmentation 20
3.3 Neural Networks 24
3.3.1 Elements and Architectures 24
3.3.2 Types of neural networks 27
Perceptron 27
Feed Forward Network 28
Multi-Layer Perceptron 28
Radial Basis Network 29
Convolutional Neural Networks 29

Recurrent Neural Networks 30
Long Short-Term Memory Networks 31
3.3.3 Supervised and Unsupervised Learning 32
3.3.4 Train vs Validation vs Test 33
3.3.5 Performance Metrics 33
3.4 Related Work 34

4 Implementation and Results **39**
4.1 Definition of the Problem 39
4.2 Implementation 39
4.2.1 Dataset analysis 40
4.2.2 Data Augmentation 41
4.2.3 CNN architecture 42
4.2.4 Optimizer Training 54
4.2.5 Results Comparison 68

5 Conclusions **69**

References **71**

Appendix A Action Units **75**

List of Figures

2.1 Whissel's Wheel of Emotion[14] 7
2.2 Duchenne's Experiment[16] 10
2.3 Plutchik's Wheel of Emotion[19] 12
2.4 The Geneva Wheel of Emotion[21] 14

3.1 (a) Input image with Gaussian noise (b) Input image with Salt and Pepper noise (c) Input image with Speckle noise. (d), (e), and (f) Output denoised images[22] 20
3.2 Cropping an image or portions of it[23] 21
3.3 Flipping an image[23] 21
3.4 Rotating an image[23] 21
3.5 Different scales of the same image[23] 22
3.6 Different scales of the same image[23] 22
3.7 Change of brightness intensity of an image[23] 22
3.8 Changing the contrast of an image[23] 23
3.9 Changing an image to grayscale[23] 23
3.10 Changing the saturation of an image[23] 23
3.11 Example of a neural network[26] 25
3.12 Example of an Artifical Neuron[26] 25
3.13 Example of a Feed Forward Network[27] 28
3.14 Example of a Multi-Layer Perceptron[27] 29
3.15 Example of a Radial Basis Network[27] 29
3.16 Example of a Convolutional Neural Network[27] 30
3.17 Example of a Recurrent Neural Network[27] 31
3.18 Example of a Long Short-Term Memory Network[27] 31
3.19 Complexity Rate of FER Techniques[37] 37
3.20 Accuracy Rates of FER Techniques[37] 37

4.1 FER2013 class distribution 41
4.2 Example of an ImageDataGenerator configuration 42
4.3 architecture 43
4.4 Adam model's configuration 44
4.5 Adam model's Accuracy Graph 45

4.6 Adam model's Loss Graph 45
4.7 Adam model's Confusion Matrix 46
4.8 Adam model's Classification Report 46
4.9 MobileNetV2 Model's Accuracy and Loss Graphs 47
4.10 MobileNetV2 Model's AUC, Precision and F1-Score Plots 47
4.11 MobileNetV2 Model's Confusion Matrix 47
4.12 Xception Model's Accuracy and Loss Graphs 48
4.13 Xception Model's AUC, Precision and F1-Score Graphs 48
4.14 Xception model's Confusion Matrix 49
4.15 Xception model's Classification Report 49
4.16 ResNet-50 model's Accuracy Graph 50
4.17 ResNet-50 model's Loss Graph 50
4.18 ResNet-50 model's ROC Curves 51
4.19 ResNet-50 model's Confusion Matrix 51
4.20 VGG19 model's Accuracy Graph 52
4.21 VGG19 model's Loss Graph 52
4.22 VGG19 model's ROC Curves 53
4.23 VGG19 model's Confusion Matrix 53
4.24 AMSGrad Initialization 56
4.25 AMSGrad model's Accuracy Graph 57
4.26 AMSGrad model's Loss Graph 58
4.27 AMSGrad model's Confusion Matrix 58
4.28 AMSGrad model's Classification Report 59
4.29 Adagrad Initialization 59
4.30 Adagrad model's Accuracy Graph 60
4.31 Adagrad model's Loss Graph 60
4.32 Adadelta model's Confusion Matrix 61
4.33 Adagrad model's Classification Report 61
4.34 Adadelta Initialization 62
4.35 Adadelta model's Accuracy Graph 62
4.36 Adadelta model's Loss Graph 63
4.37 Adadelta model's Confusion Matrix 63
4.38 Adadelta model's Classification Report 63
4.39 SGD model's Accuracy Graph 64
4.40 SGD model's Loss Graph 65
4.41 SGD model's Confusion Matrix 65
4.42 SGD model's Classification Report 65
4.43 SGD with Nesterov Momentum model's Initialization 66
4.44 SGD with Nesterov Momentum model's Accuracy Graph 66
4.45 SGD with Nesterov Momentum model's Loss Graph 67

4.46 SGD with Nesterov Momentum model's Confusion Matrix 67
4.47 SGD with Nesterov Momentum model's Classification Report 68

List of Tables

2.1 Emotions and their respective Action Units[18] 11

3.1 Datasets and their respective contents 19

4.1 Versions of software used 40
4.2 Times per step 68

A.1 Main Action Units 76
A.2 Head Movement Action Units 77
A.3 Eye Movement Action Units 77
A.4 Emotions and respective description 78

List of Acronyms

AFEW Acted Facial Expressions In The Wild

AFEW-VA Acted Facial Expressions In The Wild - Valence and Arousal

AI Artificial Intelligence

AM-FED+ Extended Affectiva-MIT Facial Expression Dataset

ANN Artificial Neural Networks

API Application Programming Interface

AU Action Unit

AUC Area Under Curve

CK+ Extended Cohn-Kanade

CNN Convolutional Neural Network

EMFACS Emotion Facial Action Coding System

FACS Facial Action Coding System

FER Facial Expression Recognition

FFN Feed Foward Network

GD Gradient Descent

GEM Group Expression Model

GF Geometric Facial

HOG Histogram Of Gradients

ILSVRC ImageNet Large Scale Visual Recognition Challenge

JAFFE Japanese Female Facial Expression

KDEF Karolinska Directed Emotional Faces

LSTM Long Short-Term Memory

NN Neural Networks

RBF Radial Basis Function

RBN Radial Basis Network

ReLU Rectified Linear Unit

ResNet-50 ResidualNetwork-50

RGB Red, Green and Blue

RNN Recurrent Neural Network

ROC Receiver Operating Characteristic

ROI Region Of Interest

SFEW Static Facial Expression In The Wild

SGD Stochastic Gradient Descent

SPI Strictly Person Independent

SVM Support Vector Machines

VGG Visual Geometry Group

Chapter 1

Introduction

1.1 Context

Emotions are a big part of our everyday life. They are part of our human nature and they influence almost every aspect of our lives. Emotion can be used to emphasize the way we talk, to show that you like something or you hate it, to lie, to motivate. However, sometimes it becomes hard, if not impossible, to identify other people's emotions. Since we all basically share the same set of possible emotions, why can't everyone easily perceive everyone's emotions?

In fact, this subject intrigued many brilliant minds throughout History that exhaustively tried to unravel the existence and meaning of emotions because, if they could be one day mastered, society could greatly benefit from the fact that everyone could finally better understand each other.

The work developed intends to deepen the knowledge on Artificial Intelligence and how emotions can be perceived thanks to this innovative field of Science.

1.2 Definition Of The Problem

Given the impact that emotion recognition could bring into several situations of everyday life, emerging new approaches to emotion recognition, new model architectures and new theories about emotions in general have been discussed. Faces and the power they have to express sentiments and emotions have long raised the attention of researchers. Several experiments have been described making it hard

to choose, from a long list of methods, a Facial Expression Recognition model. As such, this work pretends to perform a study on different optimizing algorithms on the deep learning model as to achieve greater performance.

1.2.1 Goals

This project has as its main goals the study of emotion theories, the identification and classification of facial expressions, the study of different emotion datasets and respective comparison, the development of deep learning models with different architectures and respective performance comparison, and finally the development of models with the same architecture but with different optimizing algorithms, with the respective performance comparison. This study is expected to contribute to identify the better models and optimization approaches for increasing the accuracy of deep learning approaches for Facial Emotion Recognition.

1.3 Organization Of The Dissertation

This dissertation is divided into 5 chapters:

- Chapter 1 - Introduction: This chapter includes the context of the work, definition of the problem, goals and expected results.
- Chapter 2 - Human Emotion: This chapter includes the study performed on human emotions, the history of the first researches on emotions, and the difference between different emotion theories.
- Chapter 3 - Facial Expression Recognition: This chapter provides an overview of what are the foundations of emotion recognition, what are the datasets relevant to this area and the techniques that may be used to improve the quality of datasets. It also introduces different types of neural networks, as well as some of machine/deep learning concepts.
- Chapter 4 - Implementation and Results: In this chapter the experiments implemented are described and the different models' results are presented and compared.
- Chapter 5 - In this final chapter, some ideas on how to improve the results and future applications for the work done are presented.

Chapter 2

Human Emotion

Some of the earliest mentions to what we can call emotions date back to Ancient Greece. Ancient Greek literature is quite rich in emotions, whether it is in the behaviour of characters, heroes or gods, or emotions triggered in readers or audiences. The first word referring to emotions in Homer's Iliad (probably the earliest Greek poem to survive, dating to the 8th century BC) is "wrath". Aristotle considered that one of the most defining features of a tragedy is being able to trigger the feeling of pity and fear. Ancient Greece orators, such as Isocrates or Demosthenes (considered the greatest Greek orator), aimed to arouse pity or anger in their listeners. There is a particularly extensive vocabulary for emotions when it comes to Ancient Greece, since there were individuals who stated there were differences, even if only slight ones, in certain synonymous terms[1].

According to the Oxford dictionary, an emotion consists of an instinctive or intuitive feeling, distinct from knowledge or reasoning. More specifically, emotions are reactions that human beings experience in response to situations or events, as such, the type of emotion is usually defined by the type of situation that prompts that emotion. For example, a person feels joy or happiness when they receive good news and feels sadness or sorrow when they receive bad news.

Knowledge about emotions was, for a long time, limited due to the fact that emotions are inherently complex and also to the fact that emotions have for long been seen as hurdles to rationality, instead of an independent psychological role.

2.1 The Emotion Theory

Charles Darwin is most known for his writings and research on biology and evolution. However, not many know that Darwin was also an early experimental psychologist. In 1872, he published The Expression of the Emotions in Man and Animals, in which he stated that all humans, and even some animals, show emotion, sometimes even through similar behaviours. For Darwin, the origin of emotions could be traced to circumstances as cultures (for humans) or species (for animals).

In the 1960's and 1970's several important theoretical contributions were made by Paul Ekman, Robert Plutchik, Carroll Izard and Silvan Tomkins, building on work made by Darwin and also William James.

In 1972, psychologist Paul Ekman proposed that there are six basic emotions, common to all human cultures[2]. Those emotions are fear, sadness, happiness, surprise, anger, and disgust. Afterwards, in 1980, Robert Plutchik[3] introduced the "wheel of emotions", which was a model that showed how different emotions can be combined, creating emotions that slightly differ from the basic ones.

In 1999, Paul Ekman added several other emotions to his list, including embarrassment, excitement, pride, satisfaction, shame, and amusement[4].

While Darwin was writing The Expression of the Emotions in Man and Animals[5], he contacted other researchers, including Guillaume-Benjamin-Amand Duchenne, a French physician who believed that there were at least 60 distinct emotions the human face could express, each emotion depending on a specific set of facial muscles. Dr. Duchenne is considered by many the father of electrotherapy. He studied emotions by applying electrical[6] current to different combinations of his patients' facial muscles, managing to replicate genuine emotions by doing so.

However, Darwin believed that not all the emotions that Duchenne stimulated were universal emotions so, in order to test his theory, he showed 11 of Duchenne's slides to guests he invited to the study, placing the slides in a random order without giving hints about them. After this study, according to written notes found, most guests had the same opinion on some of the emotions that were more general like happiness, sadness, or surprise, but firmly disagreed on other more ambiguous emotions. Darwin considered the emotions on which the guests most agreed on were the true universal emotions, and considered the rest of the emotions failed simulations made by Dr. Duchenne (emotions that caused disagreement) [7].

There are two most widely accepted theories when it comes to studies on emotions: the basic emotion theory (also known as the discrete emotion theory), and the dimensional theory of emotion.

2.1.1 Discrete Emotion Theory

Although the current basic theory of emotion was conceived by Charles Darwin in 1872, it has its roots, the first basic emotions theories, in ancient Greece and China. This theory suggests that human beings have a limited number of emotions biologically and psychologically innate and these emotions individually manifest through a specific array of behavioural factors. Carroll Izard, an American psychologist who worked with Ekman, claimed in 1977 that basic emotions are able to withstand unchanged due to their biological and social importance when it comes to evolution or adaptation to the environment. Basic emotions are capable of handling various circumstances, for example, fear and anger can aid in survival skills, in a fight or flight situation. Despite the fact that several psychologists have accepted the theory of basic emotions, the precise number of actual basic emotions still causes disagreement between psychologists. Robert Plutchik stated that there were eight primary emotions: anger, fear, sadness, disgust, surprise, anticipation, trust, and joy. He compiled them in the "wheel of emotions" mentioned above[3]. Paul Ekman (who can be seen in the picture presented below), in turn, proposed seven primary emotions: fear, anger, hapiness, sadness, disgust, and surprise[2].

A recent study actually found that anger and disgust share similar expressions[8], more specifically the wrinkled nose, and both fear and surprise usually share a raised eyebrow. The subtle differences between these pairs of emotions imply that these emotions may have developed after other types of emotions, destined to social functions instead of survival skills.

Other basic emotions theories[9] have stated that there are four distinct basic emotions: fear, anger, joy, and sadness, and these are able to explain, respectively, situations of flight in a fight-or-flight situation, frustration of a failed goal, pride of achievement and undergoing an unfortunate loss.

Description of the basic emotions

- **Happiness** - It is often described as the most pleasant emotion and can be described by feelings of joy, gladness, satisfaction, between others. This emotion can be expressed by facial expressions (such as smiling), body language (such as a relaxed stance) or tone of voice (such as a pleasing way of speaking). It is still unclear what are the key elements to achieve happiness, though it is conjectured that happiness and health share a strong connection (for example, researchers have proven that happiness plays a role in both physical and mental health).

- **Sadness** - It is an emotion characterized by feelings like disappointment, disinterest, or grief. In some cases, people that feel sadness during long periods

of time may start to feel depressed. This emotion can be expressed in various ways including crying, low mood, fatigue, and isolation from others. The type and intensity of sadness may depend on the cause or the way that the person that feels sadness deals with that emotion.

- **Fear** - It's seen as one of the most intense emotions and it's usually triggered when a person is faced with a threat. Fear can stimulate an increase of heart rate, arrhythmic breathing or trigger a fight-or-flight response. In fact, some people seek for the adrenaline rush that comes with the sense of fear by, for example, watching scary movies or riding a roller coaster.

- **Disgust** - This emotion is used to describe a natural response to something someone dislikes. For example, a person can feel disgust when smelling rotten food or seeing blood. It is considered a positive emotional state since it helps people safeguard themselves from situations they want to avoid. When feeling disgust, people might feel uncomfortable, nauseated, disturbed, or offended by something. Disgust can also be moral when a person sees someone doing something they consider unpleasant or immoral.

- **Anger** - It is considered one of the most powerful emotional states and it's usually triggered when someone feels victimized or experiences unrighteousness. When feeling anger, a person might feel frustrated, hostility or agitation. Despite being seen as an emotion with negative impact, anger might be helpful as a motivation to make life-changing decisions, or it might have a major role in a fight-or-flight response.

- **Surprise** - Usually occurs when someone is faced with an unexpected event, positive or negative. Surprise might also trigger a fight-or-flight response and can trigger substantial effects on human behaviour[4][10].

2.1.2 Dimensional Theory of Emotion

The dimensional theory of emotion was initially created by Wilhelm Wundt[11] (a German doctor, philosopher, and psychologist, considered one of the fathers of experimental psychology) in 1897 and further developed by Harold Schlosberg in 1954. According to this theory, emotions can be defined by three different dimensions, which are pleasant-unpleasant, tension-relaxation, and excitation-calm. Some stated that the tension-relaxation and excitation-calm dimensions are too similar to be considered independent from each other. One of such people was Paul Ekman who, in 1957, said that it was possible to describe all emotions with two dimensions: pleasant-unpleasant and active-passive.

In 1980, James A. Russell proposed that emotions could be organized in a circle organized in two dimensions[12]: pleasure-displeasure and rest-activated. These dimensions could also be called hedonic and arousal, respectively.

There is another popular dimensional model conceived by Professor Cynthia M. Whissell[13]: the activation-evaluation space model of emotions, also known as Whissell's wheel (shown below). This model consists of a simple circular depiction of different emotions simplified into two dimensions: activation and evaluation, as the name suggests.

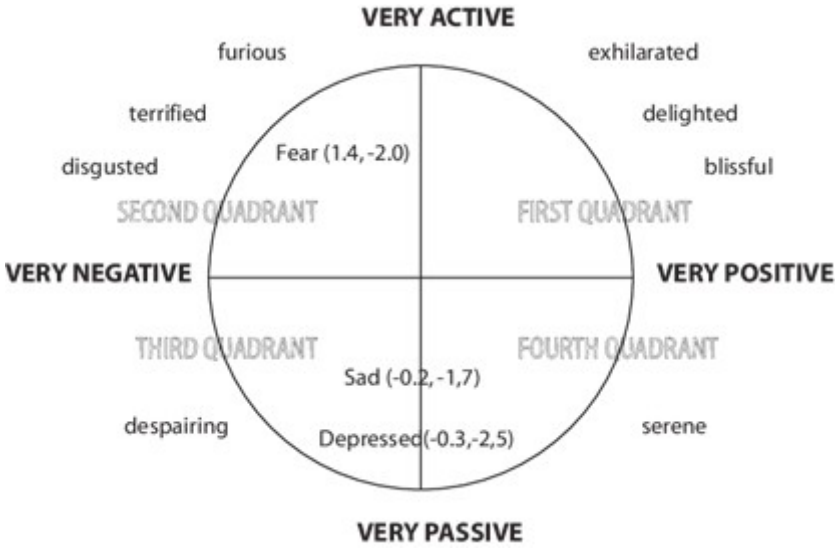


Figure 2.1: Whissel's Wheel of Emotion[14]

2.1.3 Emotions elements

It is mostly agreed amongst researchers that emotions have three key elements. Those elements are subjective experience, psychological response, and behavioural response.

Subjective Response

While it is believed that basic emotions are common to all individuals, whatever the culture or education, the events that trigger them can be particularly subjective. In other words, even though there might be the same broad label for emotions like anger or hapiness, an individual's experience with these emotions can be multi-dimensional and experienced differently depending on the individual.

Psychological Response

Researchers also believe that emotions are well capable of inducing psychological responses. For example, many people feel stomachache when anxious. As stated

before, there are emotions directly connected fight-or-flight reactions and, as such, they can prepare the body for a fight or an escape in the event of a threat. Brain scans have proved that the amygdala (region of the brain that belongs to the limbic system) plays a major role in emotion processing, especially fear. It has also been proven that individuals with damaged amygdalas have weakened fear responses.

Behavioural Response

The behavioural response refers to the actual manifestation of emotion, such as laughing, smiling, crying, between others. Although many expressions are universal, sociocultural customs may also influence the way in how we express or decipher emotions. For example, there are cultures in which it is more common to express negative emotions in public, while in others it is more common for individuals to express them more privately[4].

2.2 Paul Ekman and the creation of FACS

Paul Ekman, born in 1934, is a renowned American psychologist, one of the main pioneers in the study of emotions and facial expressions. He created the FACS (Facial Action Coding System), which categorized every expression, whether in the face or the rest of the body.

In the beginning of his professional career, Paul Ekman started to feel interest in understanding if the analysis of facial expressions could help diagnose mental disorders. For such, he recorded a number of psychological evaluation sessions, in which the patients were discharged. The Mary case was one of the most paradigmatic ones. Mary was a patient diagnosed with depression. During the evaluation session, she told she felt a lot better and asked to spend the weekend at home and so the doctor allowed her to return home for a weekend. However, before leaving the hospital, Mary confessed that she had lied: she felt horribly and was having suicidal tendencies. This case caught Ekman's attention, since Mary had lied about a very serious matter without ever manifesting suspicious behaviour, and so Ekman analyzed the video many times until he finally noticed a remarkably quick facial expression showing despair in Mary's face, almost invisible to the naked eye.

Ekman gave the name micro expression to these involuntary and unconscious facial muscle contractions that reveal what a person is truly thinking, independently from what the person might be saying. This case became a pillar for the rest of Paul Ekman's career, as he spent the next decades performing a number of scientific experiments and research that proved the importance of facial micro expressions in communication or trustworthiness assessment[15].

In his cross-cultural studies researching display rules (socially learned standards that regulate the expression of emotions), Paul Ekman developed FAST. Facial

Affect Scoring Technique, or FAST, is a tool used to measure facial movements. However, FAST was proved to have flaw, because one day an anthropologist named Wade Seaford showed Ekman a movement in his own face that was not included in FAST, so Ekman understood that, if he was going to develop an effective system for facial measurement, he would have to include every relevant aspect of the face, which meant he would possibly have to study the face's anatomy.

Paul Ekman realized the traditional anatomy manuals were not of much use to him since they were written based on dissected dead bodies and so he needed to create a new textbook that was based on functional muscles. For that, he read Duchenne de Boulogne's (also known as Guillaume Duchenne) book named The Mechanism of Human Facial Expression, published in 1862. Duchenne, as mentioned before, is considered the father of electrotherapy. While he was writing his book[6], he worked with patients that did not feel pain in their face so he could use electrical stimulation on their facial muscles and later document the outcome photographing the resulting facial muscle movement, as can be seen in the image shown below. For example, when he stimulated the patients' cheeks placing electrodes in the cheekbone area, he noticed that the stimulation resulted in a smiling appearance.

Duchenne's work proved useful in distinguishing true enjoyment smiles from other types of non-genuine ones and his conclusions have even bigger implications for distinguishing voluntary from involuntary expressions. There are muscular movements that are hard to perform voluntarily and, as such, the absence of those involuntary muscle movements implies that the facial expression is not triggered by an emotion. On the other hand, facial expressions that incorporate those hard-to-mimic muscular movements are what Ekman called "reliable facial expressions". In his book Emotions Revealed, Paul Ekman described what he believes are the reliable facial marks for emotions like anger, fear, surprise, and sadness.

Afterwards, as they were building upon Duchenne's descriptions, Paul Ekman and his colleague Wallace Friesen recorded a number of videos where, in each one, they attempted to make each muscle contract, one by one. For every muscle contraction, they wrote down a description of how the same muscle contraction influenced the facial appearance. They then identified over 40 Action Units (that are individual components of muscle movement that can be used in the FACS system, described below, to recognize emotions) and then they examined what happened when two Action Units were combined together. This way they identified more than 300 two-way combinations of Action Units and analyzed how many combinations there were for three, four, five and six-way combinations. In some cases, when a specific muscle could not be voluntarily forced to contract, they would occasionally resort to electrical stimulations.

Figure 2.2: Duchenne's Experiment[16]

Finally, the Facial Action Coding System[17] was published in 1978 and proved to be an essential tool for scientists and graduate students. It is also used by animation studios like, for example, Pixar and Disney, who explored FACS to assist in depicting human-like expressions in animated characters. The use of FACS has also been proposed for the analysis of depression and the measurement of pain levels in patients who struggle to express themselves verbally.

EMFACS

The Emotion Facial Action Coding System (EMFACS)[17] is an application of the FACS scoring. As each Action Unit represents a deformation, each emotion is deduced by the EMFACS which, in turn, classifies the emotions based on the sum of one or more AU's that constitute the emotion. FACS is a purely descriptive system. However, when combining the labels from the facial muscular movements and converting them with the EMFACS, it becomes possible to infer basic emotions' expressions like, for example, anger, joy, sadness, surprise, between others. For example, when detecting the simultaneous presence of the AU's 12 and 15 (lip corner puller and lip corner depressor, respectively) it is identified as "embarrassment", even though the AU 12, when alone, is used to detect the emotion of joy (a smile is formed when stretching the lips by raising the cheeks). When combining the AU 15 (or other AU's), this interpretation of emotion is considerably altered. Table 2.1 shows a number of 18 emotions (6 of which are basic emotions and the other 12 are compound emotions) and the respective Action Units. There were added percentages to some AU's because, in the study made[18], some of the participants used additional AU's to display the given emotions.

Table 2.1: Emotions and their respective Action Units[18]

Emotion	**Prototypical AU's**
Happiness	12, 25, 6
Sadness	4, 15 [1(60%),6(50%),11(26%),17(67%)]
Fear	1, 2, 20, 25 [2(57%),5(63%),26(33%)]
Angry	4,7,24 [10(26%,17(52%),23(29%)]
Surprise	1, 2, 25, 26 [5(66%)]
Disgust	9, 10, 17 [4(31%),24(26%)]
Happily surprised	1, 2, 12, 25 [5(64%),26(67%)]
Happily disgusted	10, 12, 25 [4(32%),6(61%),9(59%)]
Sadly fearful	1, 4, 20, 25 [2(46%),5(24%),6(34%),15(30%)]
Sadly angry	4, 15 [6(26%),7(48%),11(20%),17(50%)]
Sadly surprised	1, 4, 25, 26 [2(27%),6(31%)]
Sadly disgusted	4, 10, 25 [1(49%),6(61%),9(20%),11(35%),15(54%),17(47%)]
Fearfully angry	4, 20, 25 [5(40%),7(39%),10(30%)]
Fearfully surprised	1, 2, 5, 20, 25 [4(47%),26(51%)]
Fearfully disgusted	1, 4, 10, 20, 25 [2(64%),5(50%),9(28%),15(33%)]
Angrily surprised	4, 25, 26 [5(35%),7(50%),10(34%)]
Angrily disgusted	4, 10, 17 [7(60%),9(57%),24(36%)]
Disgustedly surprised	1, 2, 5, 10 [4(45%),9(37%),17(66%),24(33%)]

2.3 Wheels of Emotion

2.3.1 Plutchik's Wheel of Emotion

Robert Plutchik was a psychologist and professor at the University of South Florida. Throughout his research on emotions, Plutchik proposed the existence of eight primary emotions[3], as mencioned before.

The eight basic emotions of the wheel can be grouped in opposite pairs:

- Joy and sadness
- Acceptance and disgust
- Fear and anger
- Surprise and anticipation

Figure 2.3: Plutchik's Wheel of Emotion[19]

When analyzing the wheel of emotions, three main traits can be noticed: colors, layers and associations.

Colors - Basic emotions are located in the second circle, each having a different color. Emotions that have softer colors are a mix of two basic emotions

Layers - Emotions in the center of the wheel are the most intense, while emotions located in the outer layers are less intense. The saturation of the color also increases in the inner layers and the intensity also decreases in the outer layers of the wheel. For example, the red in the annoyance section in less intense than the red in the rage section

Associations - Opposite pairs of emotions are placed across from each other. The space between emotions demonstrates mixtures between different emotions. For example, love is considered a mix between joy and trust

According to Plutchik's Sequential Model, emotions are triggered by specific stimuli which can set off certain behaviours, which means that the activation of a certain emotion is meant to evoke one of the survival behaviours[20].

- Protection - Withdrawal (provoked by fear and terror)
- Destruction - Elimination of barriers to the satisfaction needs (provoked by anger and rage)

- Incorporation - Ingesting nourishment (provoked by acceptance)
- Rejection - Exclusion response to harmful material (provoked by disgust)
- Reproduction - Approach, genetic exchanges (provoked by joy and pleasure)
- Reintegration - Reaction to loss of nutrient product (provoked by sadness and grief)
- Exploration - Investigating an environment (provoked by curiosity)
- Orientation - Reaction to contact with an unfamiliar object or environment (provoked by surprise)

2.3.2 Geneva Emotion Wheel

The GEW, or Geneva Emotion Wheel, is a theoretically developed and empirically tested tool used to measure emotional response to events or situations. The Geneva Emotion Wheel was initially created by Klaus R. Scherer[21] and later developed by many other members of the Swiss Center for Affective Sciences.

The wheel has various levels of intensity for each emotion, as well as a blend of different emotions out of 20 different emotion families. These emotion families are arranged in a circular shape with the axes being defined by two dimensions of emotional experience[21].

There are five different levels of intensity, embodied by different circle sizes as seen in the image shown below.

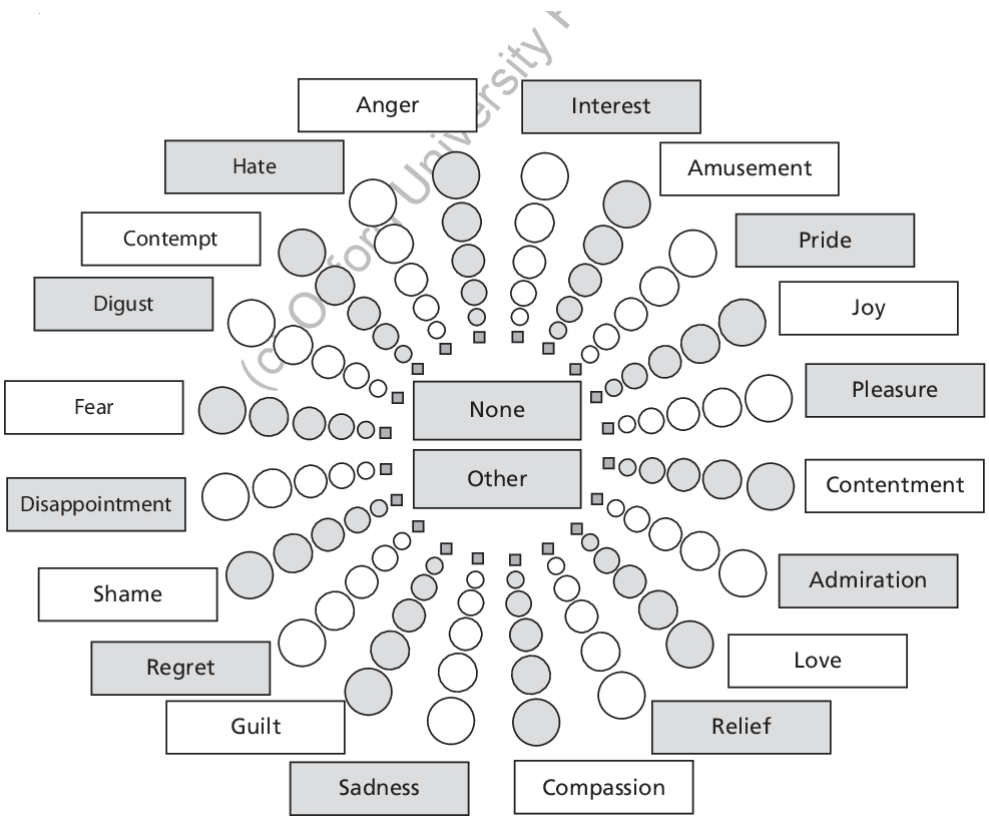


Figure 2.4: The Geneva Wheel of Emotion[21]

2.3.3 Difference between Plutchik's Wheel of Emotions and the GEW

While both wheels focus on emotions and their respective intensities, the Geneva Emotion Wheel utilizes a distinct approach. In the GEW, there are no primary emotions. Instead, there is a set of twenty emotions evaluated by two sets of polar parameters. These two parameters are valence and control. Valence is used to describe a situation as pleasant/unpleasant, and control is used to describe whether the individual has high or low control over the situation.

The evaluation of intensity is also different in both wheels. For example, in the GEW the larger circles represent stronger emotions, and these are reduced in size the closer they are to the center. One of the biggest differences between the wheels is actually the fact that the GEW allows individuals to choose options like "no emotions" or "other emotions" and, therefore, gives its respondents more freedom and flexibility to express themselves. However, a free response can be unfavorable since there might be a significant variation in how respondents express themselves in their own words and this variation might represent a be a problem in reducing measuring reliability.

Chapter 3

Facial Expression Recognition

3.1 Facial Expression Recognition Datasets

The dataset is one of the most important elements in Facial Expression Recognition (FER). The FER dataset is a set of images or videos of facial expressions representing different types of emotions. A dataset with well-annotated content is essential for the development of precise facial expression recognition systems. Most datasets have have their emotions labelled based on the basic emotion theory mentioned before (i.e., anger, fear, disgust, surprise, joy, sadness). However, some datasets might have their emotions labelled based on a continuous scale (i.e., the arousal-valence scale, previously explained).

Datasets may also be separated in two different types, which are posed expression datasets, and spontaneous expression datasets. In posed expression datasets, the participants are asked to recreate certain basic emotional expressions. However, in spontaneous expression datasets, as the name suggests, the expressions of emotion are natural. Natural expressions of emotion differ significantly from posed expressions of emotion, mainly in duration, intensity and configuration. Some of the most popular FER datasets are listed below.

Acted Facial Expressions In The Wild (AFEW)

This dataset consists of 1809 video segments extracted from movies. It is labeled with Ekman's discrete emotional model plus a neutral emotion class. The labeling process uses a recommendation system to suggest video clips to a human labeler

through their subtitles. The annotations contain the perceived emotions and information regarding the actors present in the clip, such as their name, head-pose and age.

AFEW-VA

The AFEW-VA is an extension of the AFEW dataset, from which 600 videos were selected and annotated for every frame using the dimensional emotion model (valence and arousal) for every facial region, which is described using 68 facial landmarks.

AffectNet

The AffectNet dataset contains more than 1 million facial images collected from the web by making queries using 1250 keywords related to emotions in six different languages. The entire database was annotated in the dimensional model (valence and arousal), and half of the database was manually annotated in both the categorical (with the eight labels: neutral, happy, sad, surprise, fear, disgust, anger, contempt, none, uncertain and non-face) and dimensional models.

Aff-Wild2

The extended Aff-Wild database, Aff-Wild2, contains 558 videos annotated in continuous emotions (dimensional model—valence and arousal), using different AU's, and a set of 18 discrete FER classes, which also contain the six basic emotions.

AM-FED+

The extended dataset of naturalistic and spontaneous facial expressions collected in Everyday Settings (AM-FED+) consists of 1044 facial videos recorded in real-world conditions. All the videos have automatically detected facial landmark locations for every frame and 545 of the videos were manually FACS coded. A self-report of "liking" and "familiarity" responses from the viewers is also provided.

CK+

The Extended Cohn-Kanade (CK+) is the most widely adopted laboratory controlled dataset. The database is composed of 593 FACS coded videos, 327 of which are labeled with the six basic expression labels (anger, disgust, fear, happiness, sadness and surprise) and contempt. CK+ does not provide specific training, validation and test sets.

EmotioNet

The EmotioNet database includes 950,000 images collected from the Web, annotated with AU, AU intensity, basic and compound emotion category, and WordNet

concept. The emotion category is a set of classes extended from the discrete emotion model.

FER2013

FER2013 was introduced in the ICML 2013 Challenges in Representation Learning, and consists of 48 × 48 pixel gray scale images of faces. The images were collected using Google's image search Application Programming Interface (API), in which the facial region is centered, resized, and cropped to roughly occupy the same amount of space in each image. The database is composed of 28,709 training, 3589 validation and 3589 test images with seven emotion labels: anger, disgust, fear, happiness, sadness, surprise and neutral. The FER2013 dataset was the one chose for the realization of this project.

However, FER2013 has a number of issues. It has an imbalance problem, which can be solved with data augmentation. Data augmentation consists in artificially increasing the amount of data creating new data points from data that already exists. This new data can be synthetic or augmented. Synthetic data is data that is generated artificially without using real images, while augmented data is derived from original images using geometric transformations, such as rotating an image or flipping it. FER2013 also has an intra-class variation of FER, which means that, inside the same class, there are pictures of human faces mixed with pictures of sketches or cartoons. It also has a problem with occlusion of parts of the faces, mostly done with hands (eyeglasses also constitute a problem of the same sort). FER2013 contains spontaneous and posed expressions.

JAFFE

The Japanese Female Facial Expression (JAFFE) is one of the first facial expression datasets. It contains seven facial expressions. The database is composed of 253 grayscale images with a resolution of 256 × 256 px. It is a posed dataset.

KDEF

The Karolinska Directed Emotional Faces (KDEF) is a set of 4900 pictures annotated using a model with six facial expression classes (happy, angry, afraid, disgusted, sad, surprised and neutral). The set of pictures registers 70 subjects (35 men and 35 women), viewed from five different angles. It is a posed dataset.

MMI

MMI Facial Expression is a laboratory-controlled dataset and has over 2900 videos of 75 subjects. Each video was annotated for the presence of AU's and the six basic expressions plus neutral. It contains recordings of the full temporal pattern

of a facial expression, from the neutral state to the peak expression, and back to neutral. It is a posed dataset.

OULU-CASIA

Contains 2880 videos categorized into six basic expressions: happiness, sadness, surprise, anger, fear, disgust. The videos were recorded in a laboratory environment, using two different cameras (near-infrared and visible light) under three different illumination conditions (normal, weak and dark conditions). The first eight frames of each video correspond to the neutral class, while the last frame contains the peak expression. It is a posed dataset.

RAF-DB

Real-world Affective Faces Database (RAF-DB) contains 29,672 facial images downloaded from the Internet. The dataset has a crowdsourcingbased annotation with the six basic emotions, a neutral label, and twelve compound emotions. For each image, facial landmarks, bounding box, race, age range and gender attributes are also available. Is contains spontaneous and posed expressions.

SFEW

Static Facial Expressions in the Wild (SFEW) contains frames selected from AFEW. The dataset was labeled using the discrete emotion model plus the neutral class. It contains 958 training, 372 testing and 436 validation samples. The authors also made available a pre-processed version of the dataset with the faces aligned in the image. SFEW was built following a Strictly Person Independent (SPI) protocol, therefore the train and test datasets contain different subjects. Is contains spontaneous and posed expressions.

Table 3.1: Datasets and their respective contents

Dataset	Sample	Subjects	Source	Annotation
AFEW	1809 videos (RGB)	330	Movies (Spontaneous and Posed)	6 BE's + Neutral
AFEW-VA	600 videos (RGB)	240	Movies (Spontaneous and Posed)	Valence, Arousal, FLs
AffectNet	450000 (RGB 425x25)	N/A	Web (Spontaneous and Posed)	6 BE's + Neutral
Aff-Wild2	558 videos (RGB 1454x890)	458	Web (Spontaneous)	Valence, Arousal
AM-FED+	1044 videos (RGB 320x240)	1044	Web (Spontaneous)	11 AU's, FL, Liking
CK+	593 videos (RGB 640x480)	123	Lab (Posed)	7 BE's + contempt, AU's, Facial Landmarks
EmotioNet	950000 images (RGB)	N/A	Web (Spontaneous and Posed)	12 AU's, 23 BE's, Compound Emotions (CE's)
FER2013	35887 images (G 48x48)	N/A	Web (Spontaneous and Posed)	6 BE's + Neutral
KDEF	4900 images (RGB 562x762)	70	Lab (Posed)	6 BE's + Neutral
JAFFE	231 images (G 256x256)	10	Lab (Posed)	6 BE's+ Neutral
MMI	2900 videos (RGB 720x576)	75	Lab (Posed)	6 BE's + Neutral, AU's
OULU-CASIA	2880 videos (RGB 320x240)	80	Lab (Posed)	6 BE's
RAF-DB	26672 images (RGB)	N/A	Web (Spontaneous and Posed)	6 BE's + Neutral, 42 FL's and 12 CE's
SFEW	1766 images (RGB)	95	Movies (Spontaneous and Posed)	6 BE's + Neutral

The FER2013 dataset was chosen for this project due mainly to the fact that it is one of the most easily accessible datasets and it has a high number of sample images for its classes. The main drawback is that it is considered a dataset with a certain lack of balance, that can be easily solved thanks to techniques like data augmentation.

3.2 Data Augmentation

Data augmentation is an important technique used for balancing a dataset. It consists of the development of techniques for constructing iterative optimization by implementing unobserved data (data for whom there are no measurements). There are a few effective data augmentation methods destined for image processing (which is the first step in FER) that increase image diversity. Some of the best data augmentation techniques for image processing are listed below.

- Adding noise ↪ There are three types of noise that can be added, which are Gaussian noise, Salt and Pepper noise, and Speckle noise[22]. Figure 3.1 exemplifies this approach.

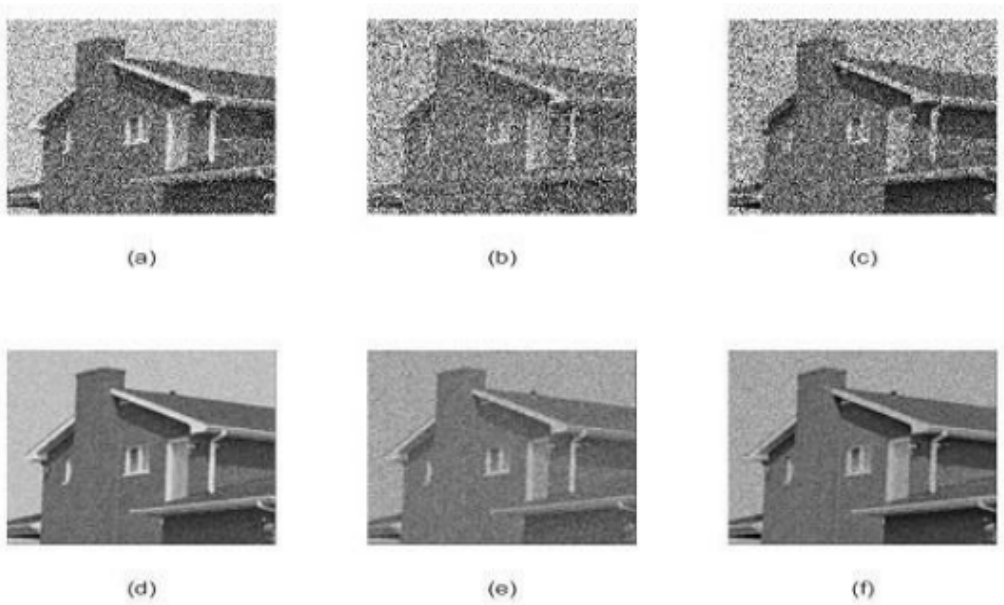


Figure 3.1: (a) Input image with Gaussian noise (b) Input image with Salt and Pepper noise (c) Input image with Speckle noise. (d), (e), and (f) Output denoised images[22]

- Cropping ↪ A portion of the image is cropped and resized to the image's original size. This can be seen in Figure 3.2.

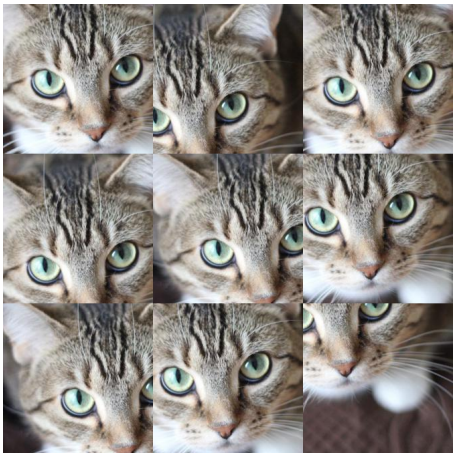


Figure 3.2: Cropping an image or portions of it[23]

- Flipping ↪ The image is flipped vertically and horizontally, thus rearranging the pixels that constitute the image while still maintaining the image's integrity. Figure 3.3 shows an image being flipped.

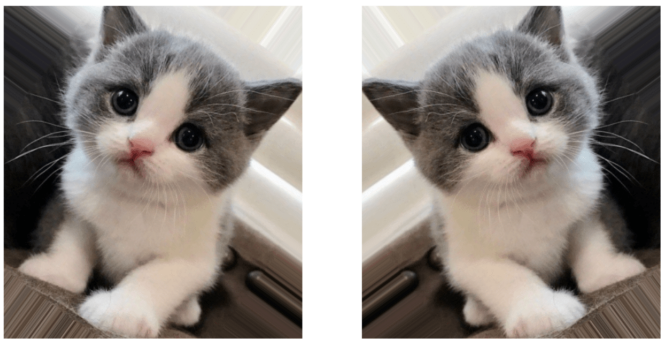


Figure 3.3: Flipping an image[23]

- Rotation ↪ The image is rotated between 0 and 360 degrees, each degree corresponding to an independent image. Figure 3.4 shows the use of this technique.

Figure 3.4: Rotating an image[23]

- Scaling ↪ The image is zoomed in and zoomed out, thus creating a new image with each scaling value. Figure 3.5 shows an example of this technique.

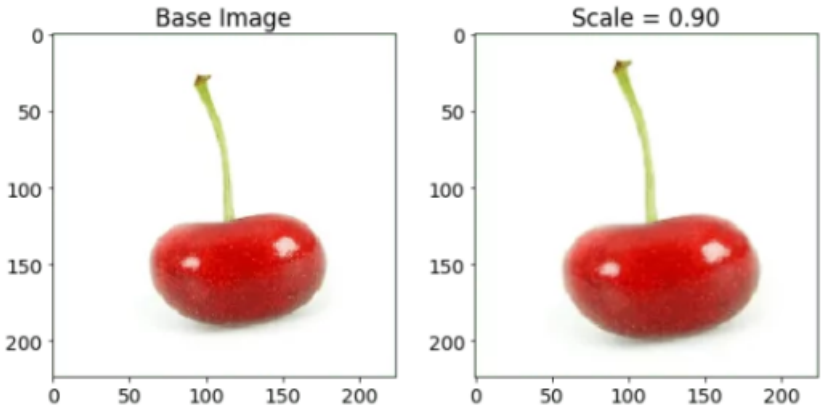


Figure 3.5: Different scales of the same image[23]

- Translation ↪ The image is shifted in various directions and lengths, creating a new image with each shift value and direction. Figure 3.6 exemplifies this approach.

Figure 3.6: Different scales of the same image[23]

- Brightness ↪ The brightness intensity of an image is changed, creating a new image for every brightness intensity value. Figure 3.7 displays change of brightness in an image.

Figure 3.7: Change of brightness intensity of an image[23]

- Contrast ↪ The contrast of an image is changed, thus creating new images with different contrast values. Figure 3.8 exemplifies a contrast change.

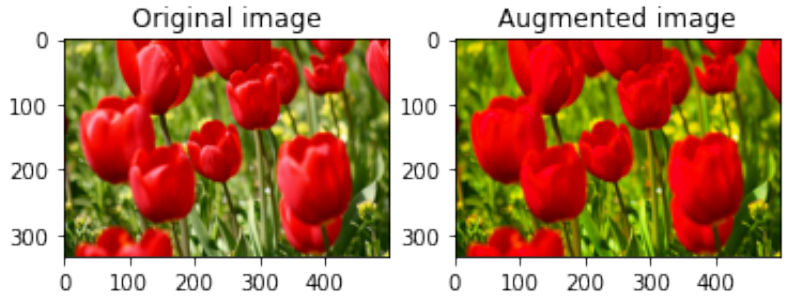


Figure 3.8: Changing the contrast of an image[23]

- Color Change ↪ The color of the image is changed. This approach can be seen in Figure 3.9.

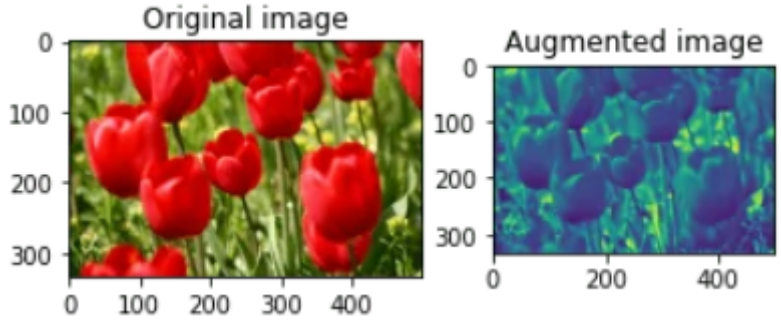


Figure 3.9: Changing an image to grayscale[23]

- Saturation ↪ Changing the saturation (color intensity) of the image also creates new images. Figure 3.10 shows an example this approach.

Figure 3.10: Changing the saturation of an image[23]

3.3 Neural Networks

Although the idea of a machine that is able to think can be traced to Ancient Greece, the history of neural networks is much more recent.

- 1943 - is published by Warren S. McCulloch and Walter Pitts. This research had the goal of understanding how the neurons in the human brain could produce such complex patterns. As a result, one of the main ideas was that the brain's neurons acted like binary thresholds to Boolean logic;
- 1949 - is published by Donald Hebb. This book pointed out that neural pathways are invigorated each time they are used;
- 1958 - The development of the perceptron (an algorithm for supervised learning of binary classifiers) is credited to Frank Rosenblatt, who documented it in study . He took McCulloch and Pitt's work a step further by introducing the concept of weights. He later managed to program a computer to learn how to differenciate cards with a mark on their left side from cards with a mark on their right side;
- 1959 - Bernard Widrow and Marcian Hoff developed ADALINE (Adaptive Linear Neuron), one of the earliest examples of a single-layer artificial neural network that suffered a multi-layer adaptation known as MADALINE (ManyADALINE), that became the first neural network to be applied to a real world problem;
- 1974 - The idea of backpropagation, even though many researchers contributed to it, was first applied in practise within Paul Werbos' PhD thesis;
- 1989 - Yann LeCun published a paper demonstrating how the use of restrictions in backpropagation and its subsequent integration into a neural network architecture can be used to train algorithms. His research proved successful when it managed to ameliorate a neural network, making it capable of recognizing hand-written zip codes provided by U.S. Postal Services[24].

3.3.1 Elements and Architectures

A neural network (NN), also know as Artificial Neural Network (ANN), consists in a series of algorithms that try to recognize inherent correlations in a set of data, mimicking the thought process of a human brain. The name itself refers to a network of neurons, in this case, artificial ones. Neural networks are considered the heart of deep learning and a machine learning subgroup.

Artifical Neural Networks are made up of node layers that contain an input layer, one (or more) hidden layers, and finally an output layer. Each node is connected

to another while having an associated weight and threshold. When the output of a certain node is above the threshold value, the node is activated, sending data to the next layer of the NN. If the output value is bellow the threshold, there is no transmission of data to the next layer[25].

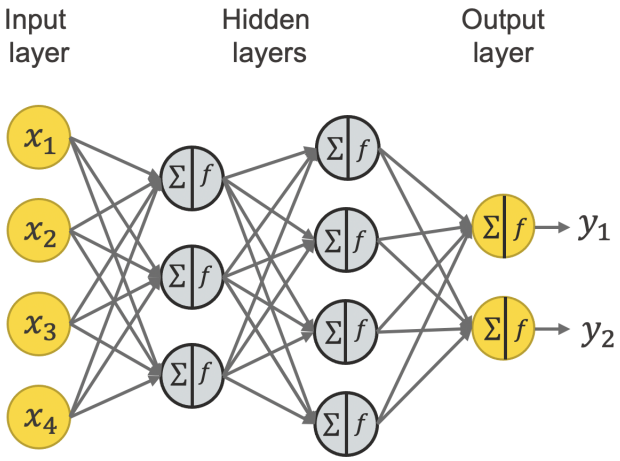


Figure 3.11: Example of a neural network[26]

Neural networks depend on training data to learn and improve their accuracy. After achieving a high level of accuracy, neural networks become very useful and powerful tools in, not only artificial intelligence, but a number of other fields as well. A neural network's smallest block is the artificial neuron, as can be seen below.

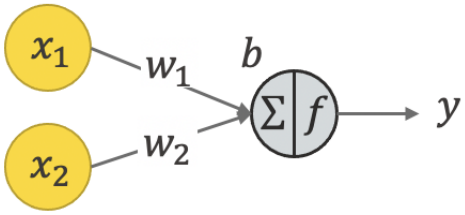


Figure 3.12: Example of an Artifical Neuron[26]

The artificial neuron possesses two inputs, x_1 and x_2, which are used to calculate the output (y) also using fixed weights (w_1 and w_2), a bias (b) and a defined activation function (f). The output is calculated after two steps.

The first step is multiplying each input by the associated weight.

$$x_1 * w_1 + x_2 * w_2 + b$$

In the second step the activation function converts the result into the output of the neuron.

$$o = f(x_1 * w_1 + x_2 * w_2 + x_3 * w_3 + b)$$

The activation function can have value of 1 or 0 (1 for when the previous output is greater than or equal to the threshold value, and 0 for when the same output is lower than the threshold value)

$$\text{output} = f(x) = \begin{cases} 1 \text{ if } \sum_{w_1 x_1} + b \geq 0 \\ 0 \text{ if } \sum w_1 x_1 + b < 0 \end{cases}$$

There is also a weight factor assigned to the variables, which determines their importance (some variables have more importance to the output, so they have more weight compared to others less relevant). After the weight assignment, the inputs are multiplied by their respective weights and posteriorly summed. Afterwards, the current output passes through an activation function, which determines the subsequent output. If the resulting output exceeds a specific threshold, the node is activated, transmitting the data to the next layer. Thus, it can be said that most of the time the output of one layer constitutes the input of the next layer.

A good and simple representation of what a single node would be, deciding if it is worth driving to a certain place depending on outside factors[24]. For example, a person is trying to figure out if it is worth driving to the shopping mall when there are three different factors that can influence the decision-making.

1. Is it raining? (Yes=1 ; No=0)

2. Is the trip urgent? (Yes=1 ; No=0)

3. Does the car have gas? (Yes=1 ; No=0)

Let's assume the following inputs:

-> $X_1 = 1$ (since it is raining outside)

-> $X_2 = 0$ (since the trip is not urgent)

-> $X_3 = 1$ (since the car does have gas)

Now let's assume weights to determine the importance of each input:

-> $W_1 = 3$ (since the rain is not a major problem, although it is very uncomfortable

driving in the rain)
-> $W_2 = 5$ (since the urgency of the trip is directly proportional to its importance)

-> $W_3 = 1$ (since the gas station is near)

Finally, we'll also assume a threshold value of 3, so the bias of the formula has a value of -3.

$$y = (1 * 3) + (0 * 5) + (1 * 1) - 3 = 1$$

Applying the activation function shown before, we can determine that this node's output would have the value of 1 (since $y \geq 0$), therefore the person should decide to drive to the shopping mall. Nevertheless, the result depends on the person, since each person can assign different importance to the various deciding factors (i.e., a person that is very afraid of driving in the rain should assign a weight much greater than 3 to X_1), so the output could change depending on the person, and, consequently, possibly changing the activation function's result.

As the model is trained, its accuracy needs to be measured with the help of a loss function (or cost function), presented below. The represents de index of the outcome, the is the predicted outcome, the is the real value, and represents the number of samples.

$$\text{Loss Function} = \text{MSE} = \frac{1}{2m} \sum_{m}^{i=1} (\hat{y} - y)^2$$

The main goal is, of course, to minimize the loss function to improve accuracy of fit of the observations. The model determines the path to take to minimize the loss function (reducing errors) by adjusting its weights in a training process. The values of the weights are adjusted with every training sample, converging at the minimum.

3.3.2 Types of neural networks

There are various types of neural networks that can have different applications or process different types of data.

Perceptron

The perceptron, mentioned before, consists of the simplest and oldest form of neural networks. It is made up of a single neutron that applies the activation function to the input (with the respective weights), thus producing a binary output.

Feed Forward Network

The FFN, or Feed Forward Network, is made up of multiple neurons and hidden layers connected to each other. The name "feed-forward" derives from the fact that these neural networks only transmit data forwards (there is no backward spread of data). This neural network is better at learning than the simple perceptron, since more layers mean more weight customization and more information processed. Feed Forward Netwworks are usually used in classification and recognition of speech, faces and patterns[27].

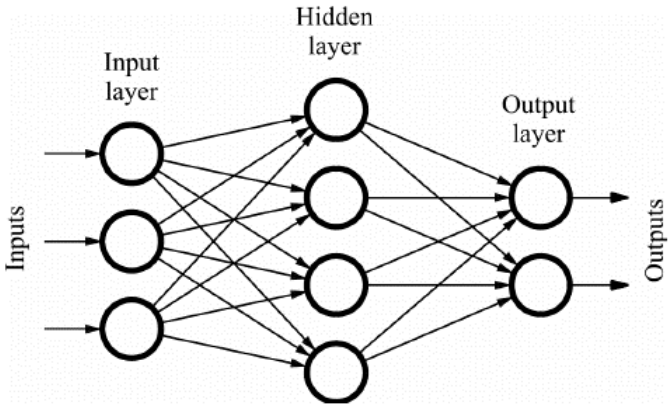


Figure 3.13: Example of a Feed Forward Network[27]

Multi-Layer Perceptron

As its name suggests, the main problem with the Feed Forward Network is the fact that it is not able to learn with backward propagation of data. For such, there are Multi-Layer Perceptrons, which are neural networks with multiple hidden layers and activation functions that learn in a supervised manner (use of labeled datasets) where the weights are updated through Gradient Descent (or GD, an iterative first-order optimisation algorithm used to find a local minimum or maximum of a given function).

This type of neural network is bi-directional, this is, the inputs are transmitted forwards while the weight updates, on the other side are transmitted backwards. Multi-Layer Perceptrons are usually used in Deep Learning, although they are relatively slow due to their complex nature[27].

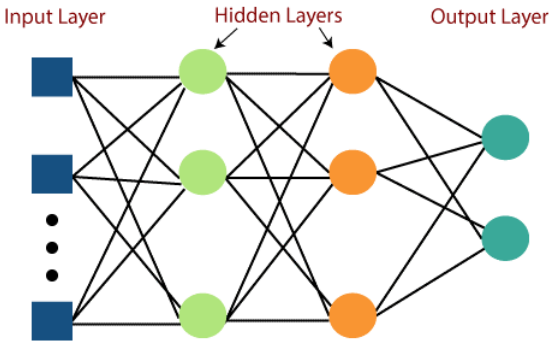


Figure 3.14: Example of a Multi-Layer Perceptron[27]

Radial Basis Network

The Radial Basis Network (or RBN) consists of an input layer, a layer with RBF (Radial Basis Function) neurons and an output layer. The RBF neurons are used to store the classes for each of the data training steps. When data is fed to this neural network, the RBF neurons compare the Euclidian distance (length of a line segment between the two points in the Euclidian space) of the feature values with the classes stored in the neurons, and the class where the distance in the lowest is defined as the predicted class[27]. The Radial Basis Network is usually used in Power Restoration systems.

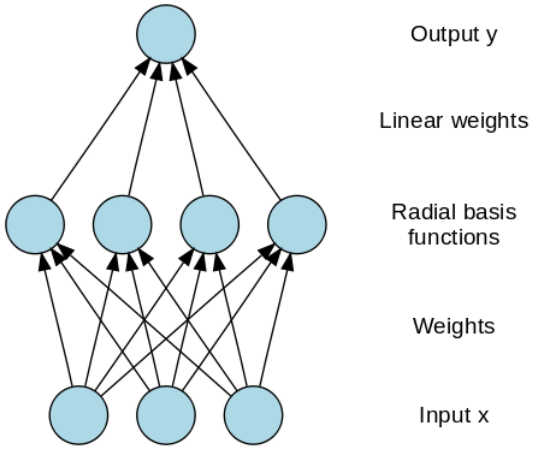


Figure 3.15: Example of a Radial Basis Network[27]

Convolutional Neural Networks

The CNN, or Convolutional Neural Network, is the most commonly used neural network for image classification. It possesses multiple convolution layers (as the name suggests) that analyze importance features in images. A convolution is integral that

expresses the amount of overlap of one function as it is shifted over another function. The convolutional operation of CNN's utilizes a custom matrix (also known as filters) to convolute over the input image and draw maps. These filters are initiated arbitrarily and updated through backward propagation. After the convolution layer, the maps produced in that layer are aggregated in a pooling (aggregation) layer. It is also possible to add dropout layers to a CNN that are capable of disabling certain neurons so as to reduce overfitting. In the hidden layers, ReLU (Rectified Linear Unit) is used as an activation function, which is a linear function that will output the input directly if it is positive, otherwise, it will output zero[27].

$$\text{ReLU formula: } f(x) = max(0, x)$$

TheCNN's have, as the last layer, a fully connected dense layer, whose activation function is usually Softmax (used to assign decimal probabilities to each class in a multi-class problem) for classification, and ReLU for regression.

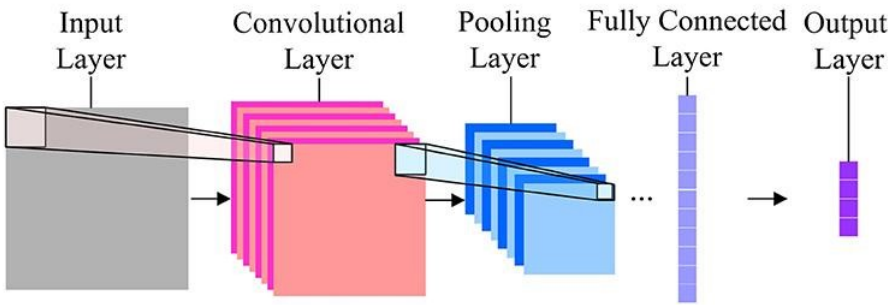


Figure 3.16: Example of a Convolutional Neural Network[27]

Recurrent Neural Networks

Recurrent Neural Networks, or RNN's, are used for predictions when using sequential data, data that can consist of images, words, among others. Their structure is similar to Feed-Forward Networks', with the difference that RNN's layers also receive a delayed input of the previous instance prediction. This instance prediction is kept in the RNN cell, which is a second input for every prediction.

However, the main disadvantage of Recurrent Neural Networks is the Vanishing Gradient Problem. When more layers, that use certain activation funtions, are added, the gradients of the loss function start to approach zero, making the network hard to train[27].

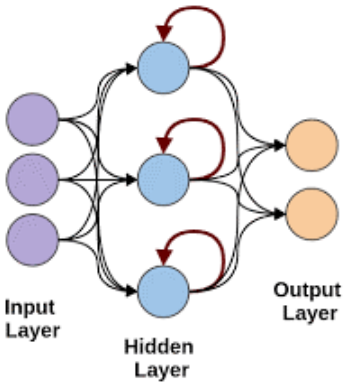


Figure 3.17: Example of a Recurrent Neural Network[27]

Long Short-Term Memory Networks

Long Short-Term Memory Networks, or LSTM neural networks, overcome the Vanishing Gradient Problem. They manage this by adding a special memory cell capable of storing information during long periods of time. They also use gates to determine which output should be used or discarded. There are three gates, which are the input gate, the output gate, and the forget gate. The input gate determines what data should be kept, the output gate manages the data transmitted to the next layer, and the forget gate determines when to dump or forget the data that is not needed. These types of neural networks have applications in fields such as gesture recognition, speech recognition, and text prediction[27].

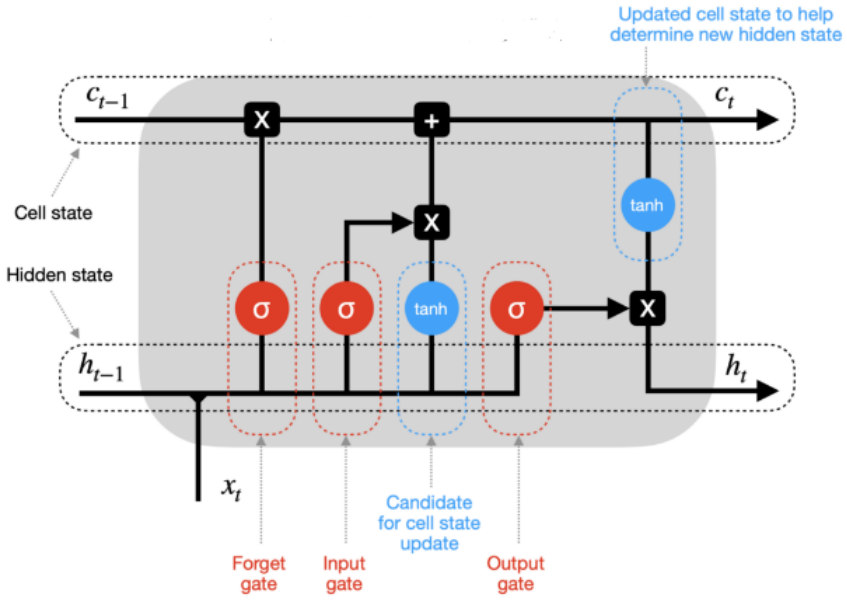


Figure 3.18: Example of a Long Short-Term Memory Network[27]

The image shown above shows an example of a LSTM neural network. The h_{t-1} variable represents the hidden state at the previous timestep $t-1$ (the short-term memory). The c_{t-1} variable represents the cell state at the previous timestep $t-1$ (long-term memory). The x_t variable represents the input vector at the current timestep t. The h_t variable represents the hidden stat at the current timestep t. The c_t variable represents the cell state at the current timestep t. The "X" squares represent vector pointwise multiplications. The "+" squares represent vector pointwise additions. The "tanh" circles represent hyperbolic tangent activation functions. Finally, the "σ" circles represent sigmoid activation functions.

3.3.3 Supervised and Unsupervised Learning

Within artificial intelligence and machine learning, there are two basic approaches, which are supervised learning and unsupervised learning. The main difference is the fact that one uses labeled data to help predict outcomes, while the other does not. However, there are still other differences between both approaches, as well as areas where one outperforms the other.

Supervised Learning - In supervised learning it is required that both the output and input are labelled during the training phase of the ML lifecycle[28]. The data is usually labelled during the preparation phase before being used in the training phase and once the model learns how to relate the input and output data, it can classify new unknown datasets and predict new results or outcomes. As its name suggests, supervised learning is named that way due to the fact that it needs human intervention, at least a small part of it. Human intervention is important to accurately label data destined to be used in supervised learning, but this can prove to be a very impractical process, especially when large arrays of labelled data are needed.

Supervised learning if most used for:

- Classifying different types of files, such as images, documents or written words;
- Predicting future trends or outcomes through learning patters in training data.

Unsupervised Learning - Unlike supervised learning, unsupervised machine learning constitutes the training of models on unlabelled training data. It's usually used to identify patterns or trends in raw datasets, or to group analogous data into arrays. As its name also implies, unsupervised learning requires almost no human intervention when compared to supervised machine learning. There are some parameters that can still be defined by a human, such as the number of cluster points, but the model will be able to process large amounts of data without any

human intervention. Therefore, unsupervised learning is more valuable when used to predict never-seen trends or unknown connections within its own data.

Unsupervised learning is most used for:

- Cluster datasets or connections within their data;
- Understanding relationships between different data points;
- Performing initial data assessment.

3.3.4 Train vs Validation vs Test

When building a deep learning model there are three different steps when it comes to data partitioning: training, validation and testing.

- Training - The training phase is where the model first learns the features or patterns in the data and, which each epoch of training, the data destined for training is fed to the model's architecture.
- Validation - The validation phase is where the model's performance is validated. This process provides information useful for tuning the model's configuration for the maximization of eficiency.
- Testing - The testing phase is the final phase, where the model is tested with a separate section of data and it is in this phase where the final accuracy (among others like precision or f1-score) is calculated.

A study made by Quang Hung Nguyen et al.[29] proved the 70-30 (70% training and 30% validation) training dataset ratio to be the best in maximizing the model's performance. In their article they explained how they experimented with other ratios (for example, 10-90, 20-80, 30-70, and so on), and with the help of statistical indicators such as Root Mean Squared Error, Correlation Coefficient, among others, they evaluated the model's ability to correctly predict under different dataset division ratios.

3.3.5 Performance Metrics

A metric is a function that is used to judge the performance of deep learning models. Some of the main metrics that are used to evaluate that performance (and the ones used to evaluate the models trained in this project) are accuracy, loss, precision, recall, and f1-score.

- **Accuracy** is metric that represents the model's performance throughout all classes and it is of high relevancy when all classes have similar importance. The accuracy of a model is given by the following expression:

  $Accuracy = \frac{True_{positive}+True_{negative}}{True_{positive}+True_{negative}+False_{positive}+False_{negative}}$

- **Loss** is used to represent how incorrect are the model's predictions

- **Precision** of a model represents its accuracy in classifying an image as positive. A model with high precision is more likely to classify more images as positive samples. The equation for calculating the precision is as follows:

  $Precision = \frac{True_{positive}}{True_{positive}+False_{positive}}$

- **Recall** is used to represent the model's ability to detect positive images. For example, if a model classifies all images as positive, then the model's recall will be of 1.0 (or 100%). The recall can be calculated using the following expression:

  $Recall = \frac{True_{positive}}{True_{positive}+False_{negative}}$

- **F1-score** is a combination of the precision and recall values. In other words, the F1-score is used to compare those other two metrics. It can be calculated by the following expression:

  $F1 = \frac{2\times(Precision+Recall)}{Precision+Recall}$

3.4 Related Work

Although CNN's have been around since the 1990's, the creation of more powerful computers, algorithms and softwares allowed for the exponential growth of FER. Some of the most popular FER architectures are listed below (to be noted that some of these architectures were not initially designed specifically for Facial Expression Recognition, although they can be used for that purpose while also having a very good performance).

- AlexNet - In 2012, Alex Krizhevesky et al.[30] proposed AlexNet, which was an architecture that, at the time, achieved state-of-the-art recognition accuracy compared to the traditional approaches. Its architecture consists of eight layers (five convolutional layers and three fully-connected layers). However, the most important aspects of this architecture were its use of ReLU Nonlinearity, Multiple GPU's and Overlapping Pooling.

- VGG16 - VGG stands for Visual Geometry Group. VGG16 is a deep learning architecture[31] that possesses 16 layers, like the name suggests, and it was first proposed by A. Zisserman and K. Simonyan. This pre-trained network

can classify images into more than 1000 different object classes and it has an input size of 224x224.

- VGG19 - VGG19 is a variant of VGG16, with the main difference being the VGG19 architecture consisting of 19 layers instead of 16. Since VGG19 has more layers than VGG16, it is able to extract more deeper features than VGG16, proved in various studies[32].

- ResNet-50 (or ResidualNetwork50) is a CNN that consists of 50 layers (48 convolutional layers, 1 MaxPool layer and 1 Average Pool layer). This architecture proved revolutionary since it introduced for the first time the concept of skip connection, which is, as the name suggests, the method of being able to skip some of the layers and feed the output to other later layers. The ResNet-50 architecture won the top position at the ILSVRC (ImageNet Large Scale Visual Recognition Challenge) in 2015 with an error of only 3.57%[33].

- InceptionV3 - The Inception architecture was first introduced by Szegedy et al. in 2014[34]. It is trained on the ImageNet datasets and it has a top-5 error rate of 3.5% and a top-1 error rate of 17.3% (top-1 and top-5 error rates are, respectively, the probability of error in predicting the most accurate class and one of the 5 most accurate classes). The model is made up of symmetric and asymmetric building blocks including convolutions, average pooling, max pooling, concatenations, among others. It is an architecture where Batch Normalization is used extensively throughout the model. InceptionV3 is a neural network with 48 layers and it has an image input size of 299x299.

- MobileNetV2 - The MobileNetV2 is a 53 layer convolutional neural network that, as the name suggests, is mainly destined for use in mobile devices[35]. It consists of 53 layers and its architecture is based on an inverted residual structure where both the input and output of the residual block are thin bottleneck layers. MobileNetV2 also uses lightweight in-depth convolutions to filter features in the intermediate expansion layer.

Facial Expression Recognition can be divided in three different phases:

- Phase 1 - Face Detection;

- Phase 2 - Facial Expression Detection;

- Phase 3 - Classify and assign the Facial Expression to an emotional state.

Since each one of these phases involves a wide stretch of theoretical/practical areas (i.e., emotion classification can be based in various emotion theories, such as the Discrete or Dimensional emotion theories, which can vary the way that the images

are classified in the dataset), there is a large variety of methodologies presented for the detection of emotion in the human face.

Francisco Luque Sánchez et al.[36] developed a model that was based on the HAPPEI dataset (a dataset made up of 4886 images imported from Facebook and Flickr, each image being classified from 1 to 10 depending on the humor intensity). This model was designated GEM, or Group Expression Model, and it allows ascertaining the general mood of a crowd through each individual's average happiness intensity. Each face is individually analyzed using the HOG, or Histogram of Gradients (feature used in computer vision and image processing for the purpose of object detection. This technique counts on occurrences of gradient orientation in the localized portion of an image), and then classified in terms of intensity by an SVM classifier, or Support Vector Machines classifier (SVM's are a set of supervised learning methods used for classification, regression and outliers detection). The contribution that each face makes in shaping the overall mood of the group is also measured depending on several attributes, such as group attributes like age or beauty, and contextual attributes such as the person's position within the group or the distance between that person and the camera.

In *A Survey on Human Face Expression Recognition Techniques*[37], I.Michael Revina and W.R. Sam Emmanuel developed a method based on various datasets inluding CK+, JAFFE and KDEF, among others, that consisted of the combination of the ROI (Region of Interest) segmentation method with the GF (Geometric Facial) feature extraction method and SVM classifiers that made possible a 99% accuracy while also maintaining a lower level of complexity. The performance comparison of the survey was based on complexity rates, feature extraction approaches, recognition accuracy on different datasets, among others. Figure 3.19[37] represents the complexity rates of various FER techniques and Figure 3.20[37] shows the accuracy rates of different FER approaches.

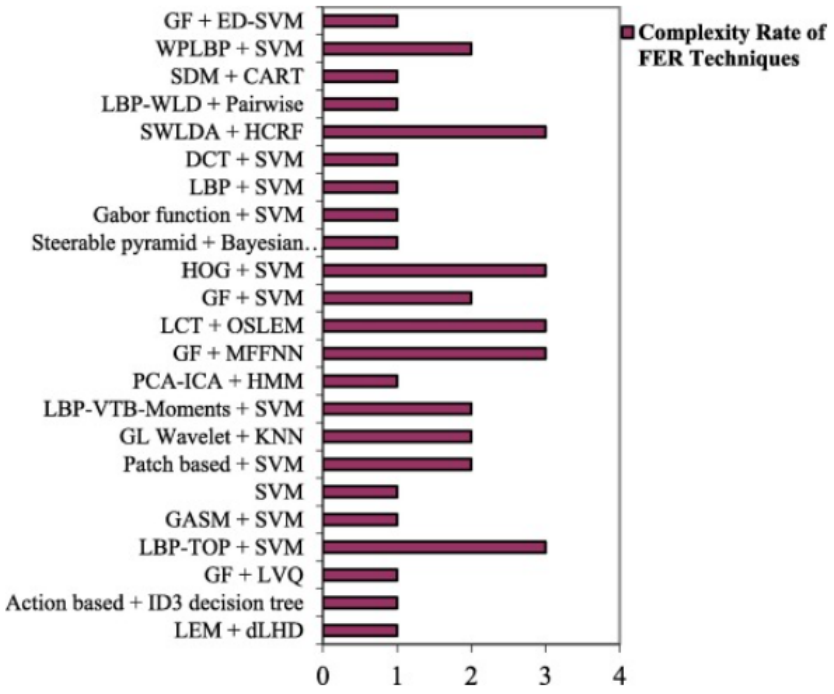


Figure 3.19: Complexity Rate of FER Techniques[37]

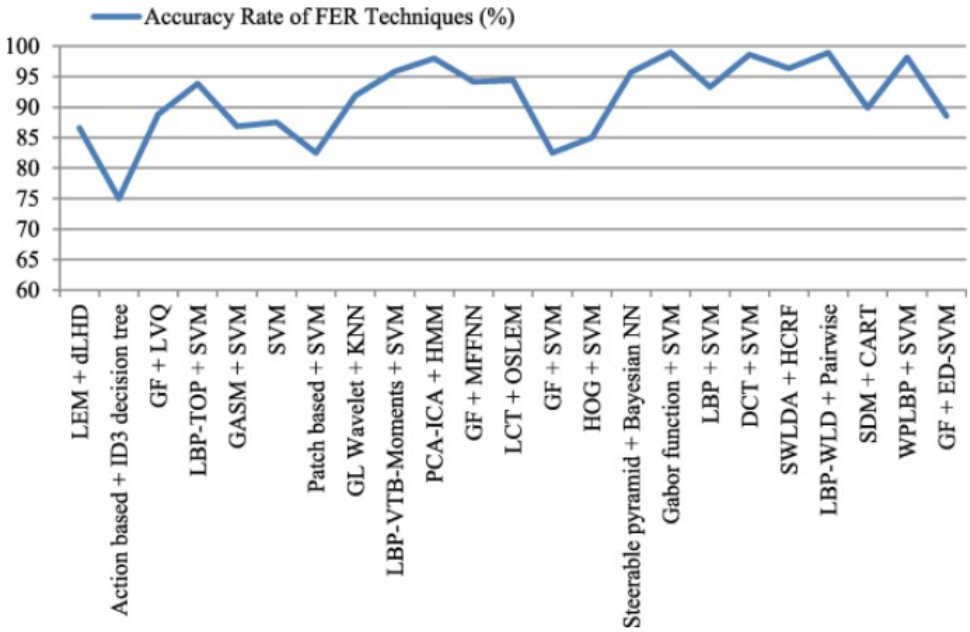


Figure 3.20: Accuracy Rates of FER Techniques[37]

Chapter 4

Implementation and Results

4.1 Definition of the Problem

Facial Expression Recognition is a very comprehensive field of deep learning and, as such, it can have numerous approaches, as mentioned in 3.4. There are a number of different ways that we can create and train a model to identify facial expressions. However, its accuracy can also significantly vary depending on many factors, such as the type of the model used, the number of layers, dataset balancing, among others.

In this project, the same dataset was used to train different types of models and a study was made to compare the values of each accuracy, as well as generating the respective confusion matrices.

4.2 Implementation

This project was developed on Windows 10 using Visual Studio Code. Visual Studio Code is a lightweight yet quite versatile source code editor[38] avaiable for Windows, macOS, Linux, and Raspberry Pi OS. It has an incorporated support for JavaScript, TypeScript, and Node.js and also possesses extensions meant for other coding languages such as C++, C#, Java, Python, PHP, and Go. It also has extensions for runtimes, such as .NET and Unity, and environments, such as Docker and Kubernetes. In this project's case the coding language used was Python (both in Text-Files and Notebook). The computer used to run the programs had a Intel(R) Core(TM)

i7-7700HQ CPU @ 2.80GHz with 16 GB of RAM and an NVIDIA 1050 Ti. The *Pandas* library was used for data analysis and manipulation[39]. The *Numpy* library was used to create n-dimensional arrays and perform mathematical operations on these arrays[40]. In some instances, *OpenCV* was used for image processing operations[41].

Tensorflow is an end-to-end platform[42] used for building machine learning models and *Keras* is a Python neural network library[43] and was used to perform experimentation with the neural networks. CUDA®, by NVIDIA, is a parallel computing platform[44] and programming model. It was used to significantly increase computer performance by utilizing the GPU's (Graphics Processing Unit) power.

Matplotlib[45] is a Python library used for creating different types of data visualizations. It was used in this project to help display relevant data like, for example, graphs showing accuracy and loss values.

The versions of the software used are shown below.

Table 4.1: Versions of software used

Software	**Version**
Tensorflow	2.10.0
Keras	2.10.0
keras-utils	1.0.13
Keras-Preprocessing	1.1.2
Python	3.10.2
Numpy	1.23.2
OpenCV	4.6.0.66
Matplotlib	3.5.3
CUDA	11.7.1
Pandas	1.4.4
Visual Studio Code	1.22.2

4.2.1 Dataset analysis

The chosen FER dataset for this project was the FER2013, as mentioned before. This dataset was originally created using Google's image search API with synonyms related to different emotions. It is composed of 7 different folders, each one containing images corresponding to the each folder's respective emotion (angry, disgust, fear, happy, neutral, sad and surprised). In the image shown below it can be seen the FER2013's class distribution.

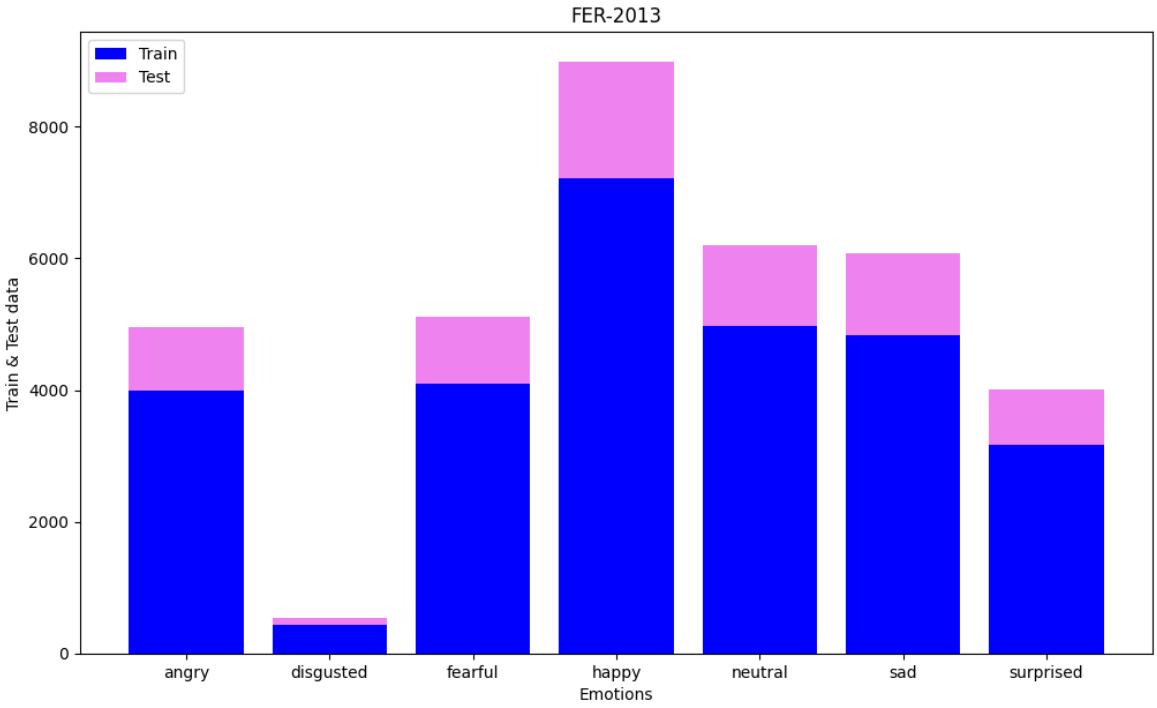


Figure 4.1: FER2013 class distribution

As can be seen in the figure before displayed, it is quite evident that there is a balance problem within this dataset, since there are so few samples in the "disgust" class. This is due to the fact that, as mentioned before[8], it has been proven that, for example, anger is very difficult to distinguish from disgust due to the fact that they share a lot of similarities in the way they are expressed. This problem, however, can be solved thanks to techniques like data augmentation (explained in 3.2).

4.2.2 Data Augmentation

The deep learning Keras library grants the ability to fit models with data augmentation thanks to its ImageDataGenerator class[46]. The six main types of data augmentation that the ImageDataGenerator provides are called through the following arguments:

- *width_shift_range* or *height_shift_range* for image shifting;
- *horizontal_flip* or *verical_flip* for image flipping;
- *rotation_range* for rotating the images;
- *brightness_range* for varying the brightness of the images;
- *zoom_range* for zooming in or out the images;
- *rescale* for image rescaling.

The image shown below displays an example of how this class should be called with the data augmentation methods before mentioned.

```
from keras.preprocessing.image import ImageDataGenerator


train_datagen = ImageDataGenerator(rescale=1./255,
                                   zoom_range=0.5,
                                   horizontal_flip=True
                                   vertical_flip=True
                                   width_shift_range=0.5
                                   height_shift_range=0.5
                                   rotation_range=30
                                   brightness_range=1)
```

Figure 4.2: Example of an ImageDataGenerator configuration

4.2.3 CNN architecture

The next step after augmenting the data was deciding the structure of the CNN. First the Keras model API was chosen.

There are two Keras model API's when it comes to CNN architectures: Sequential and Functional[43]. When using the Sequential model all the layers in the CNN are connected sequentially, this is, each layer has exactly one input and one output while all the layers are positioned consecutively. When using the Functional the layers can have multiple outputs or inputs, so this model can be used for more powerful and complex CNN's. In this case the Sequential model was chosen due to the fact that it is simpler and easier to use.

After defining the model as Sequential, the CNN layers were added. Firstly, two consecutive Conv2D layers were added. They were added consecutively due to the fact that, since the second Conv2D layer's input is the previous Conv2D layer's extracted features, its layer's features are going to be of higher-level than the previous one's, allowing for the extraction of more complex features. Afterwards, it was added a Batch Normalization layer, which is a layer that is used for normalizing the contributions to a layer for every mini-batch (process where a subset of the data is taken during one iteration). Afterwards a Max Pooling layer was added, which is used to selecting the maximum element from the region of the feature map covered by the filter, meaning it selects the most prominent features from the ones filtered. A Dropout layer is added next to randomly ignore certain neurons during the training process, making the network capable of better generalization and less likely to overfit the training data. A better representation of this architecture can be seen in Figure 4.3.

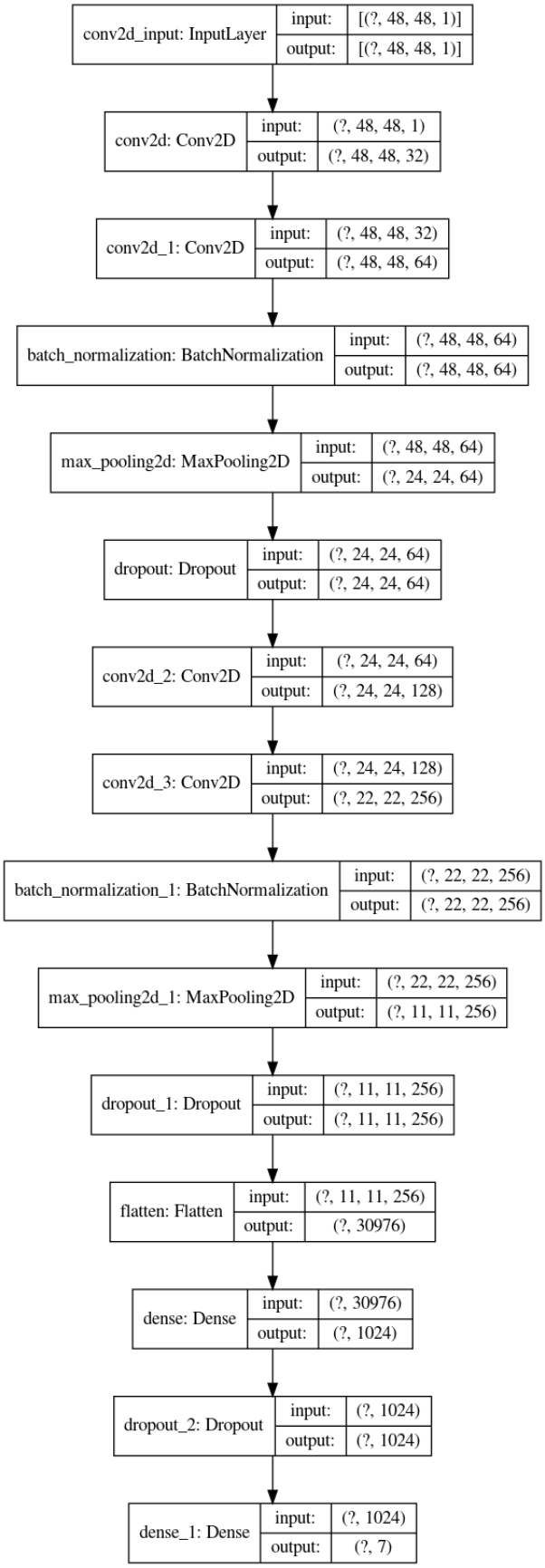


Figure 4.3: architecture

After building the network it was necessary to decide which optimizer would be used in this project. Optimizers are algorithms that alter certain parameters of the neural network (such as Learning Rate or weights) to minimize loss values. Firstly, the optimizer chosen was Adam, which is an extension of SGD, or Stochastic Gradient Descent. While SGD is an optimizer that keeps the same Learning Rate through training, Adam is capable of updating the Learning Rate for each weight individually. The image shown below shows the configuration used for the Adam optimizer.

```
model.compile(optimizer=Adam(learning_rate=0.0001, decay=1e-6),
              loss='categorical_crossentropy',
              metrics=['accuracy'])
```

Figure 4.4: Adam model's configuration

The model was later trained for 60 epochs with the initial Learning Rate being 0.0001 and the Batch Size being 64. Figure 4.5 show the evolution of the Adam model's accuracy, which is the number of classification that a model makes correctly divided by the total number of predictions made. Figure 4.6 shows the evolution of the Adam model's loss, which is the penalty for a bad prediction. Figure 4.7 shows the Adam model's Confusion Matrix, which is a matrix used to compare the model's predictions (True Positives and Negatives, and False Positives and Negatives). Figure 4.8 shows the Adam model's Classification Report, which utilizes different metrics to measure the model's performance. These metrics are Precision, Recall, and F1-Score (they are explained in 3.3.5).

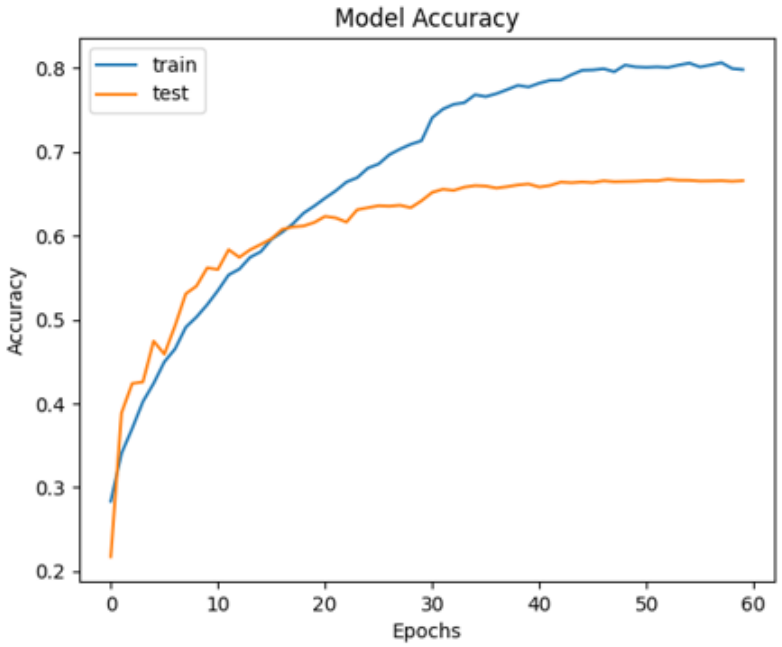


Figure 4.5: Adam model's Accuracy Graph

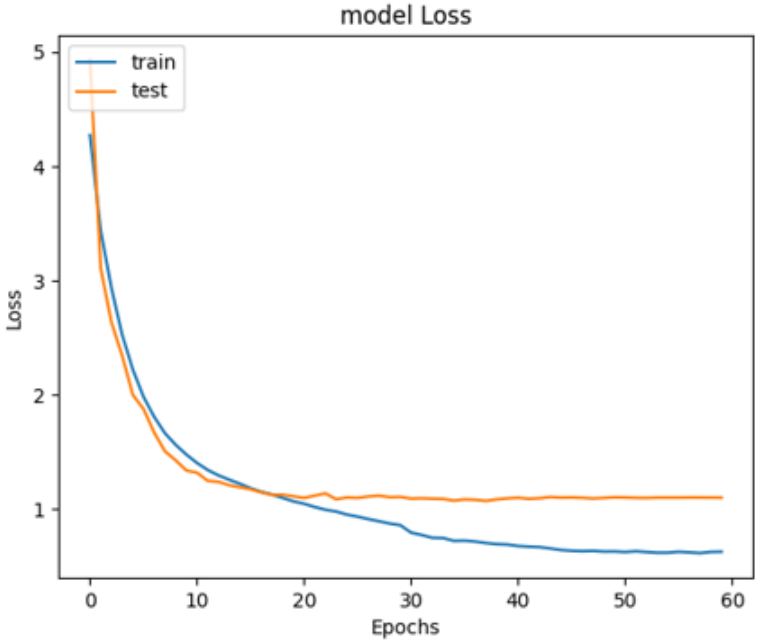


Figure 4.6: Adam model's Loss Graph

```
Confusion Matrix
[[ 488   54  516 1020  731  730  456]
 [  54    8   48  122   84   76   44]
 [ 533   58  508 1049  797  665  487]
 [1002   94  943 1846 1365 1194  771]
 [ 691   54  616 1267  901  886  550]
 [ 645   70  632 1245  840  871  527]
 [ 450   37  436  852  550  496  350]]
```

Figure 4.7: Adam model's Confusion Matrix

Classification Report	precision	recall	f1-score	support
angry	0.13	0.12	0.12	3995
disgusted	0.02	0.02	0.02	436
neutral	0.17	0.18	0.18	4965
sad	0.18	0.18	0.18	4830
surprised	0.11	0.11	0.11	3171
accuracy			0.17	28709
macro avg	0.14	0.14	0.14	28709
weighted avg	0.17	0.17	0.17	28709

Figure 4.8: Adam model's Classification Report

MobileNetV2

MobileNetV2, as mencioned in 3.4, is a relatively light architecture that is mainly destined for mobile devices. Shown below are the training results of the model using the MobileNetV2 architecture. Figure 4.9 show the evolution of the MobileNetV2 model's accuracy and loss. Figure 4.10 shows the history of AUC (also known as Area Under Curve, it is another way to measure the model's performance. The closer the curve is to the upper left corner, the hight the accuracy), Precision and F1-Score evolution. These graphs are another way of showing the metrics displayed in the Classification Report (with the difference being that the graphs show the overall evolution while the Classification Report provides us the metrics for each emotion as well). Figure 4.11 shows the model's Confusion Matrix.

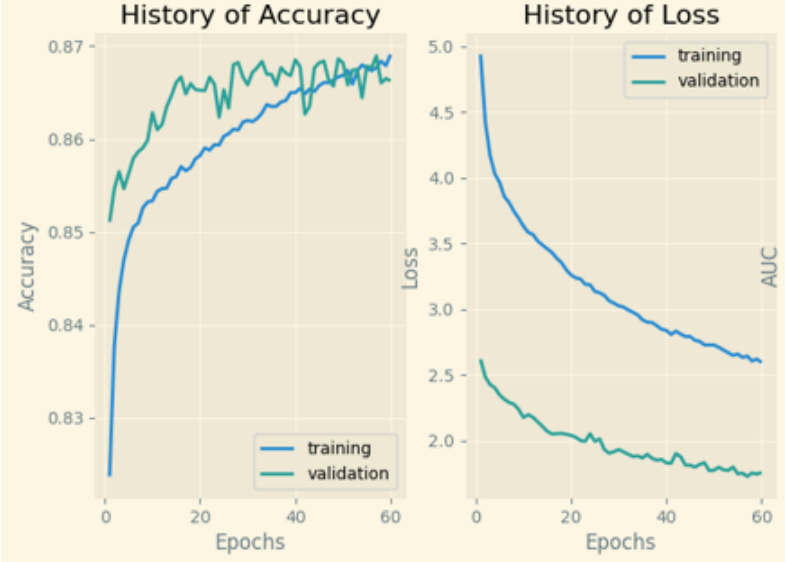


Figure 4.9: MobileNetV2 Model's Accuracy and Loss Graphs

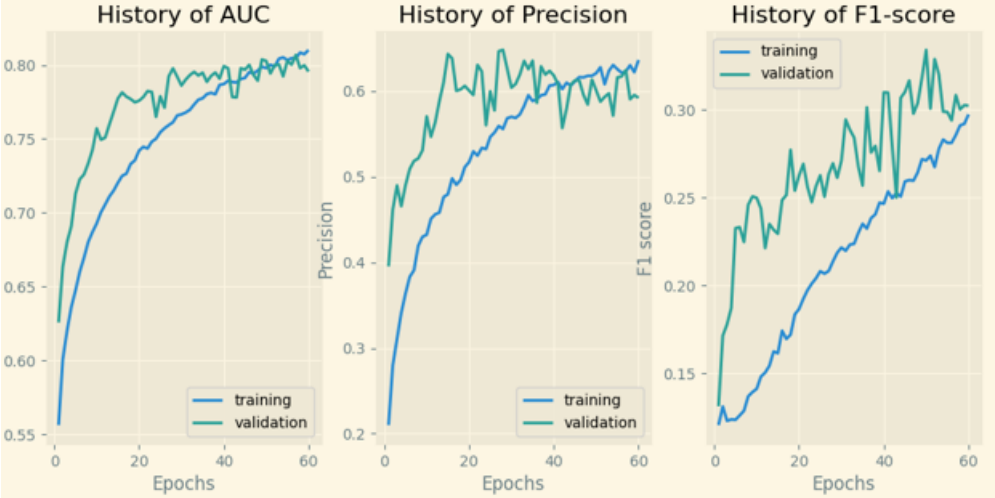


Figure 4.10: MobileNetV2 Model's AUC, Precision and F1-Score Plots

```
Confusion Matrix
[[ 492  293  430 1044  650  511  575]
 [  53   30   42  124   70   61   56]
 [ 519  286  474 1055  667  481  615]
 [ 954  506  815 1818 1153  946 1023]
 [ 665  353  602 1236  812  615  682]
 [ 630  374  553 1195  793  580  705]
 [ 390  242  365  779  495  422  478]]
```

Figure 4.11: MobileNetV2 Model's Confusion Matrix

Xception

After MobileNetV2, a model using the Xception architecture was trained. The results can be seen below. Figure 4.12 shows the Xception model's accuracy and loss graphs. Figure 4.13 shows the evolution of AUC, Precision and F1-Score. Figure 4.14 shows the model's Confusion Matrix.

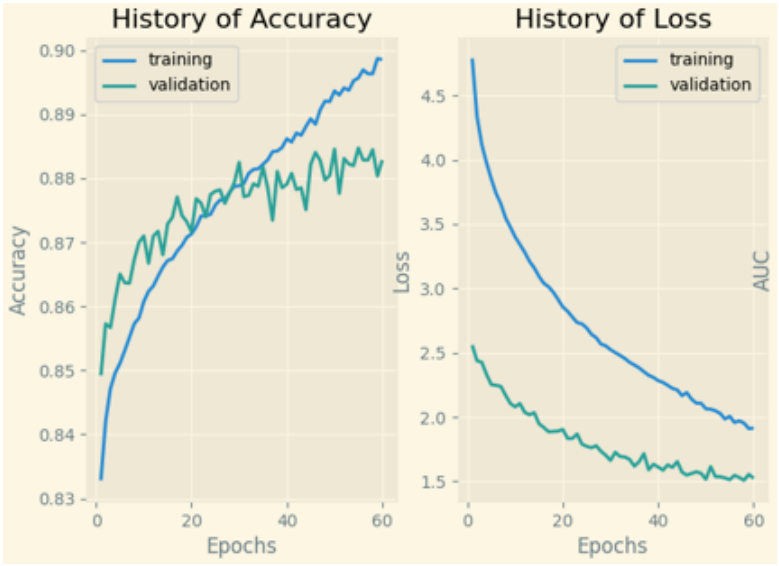


Figure 4.12: Xception Model's Accuracy and Loss Graphs

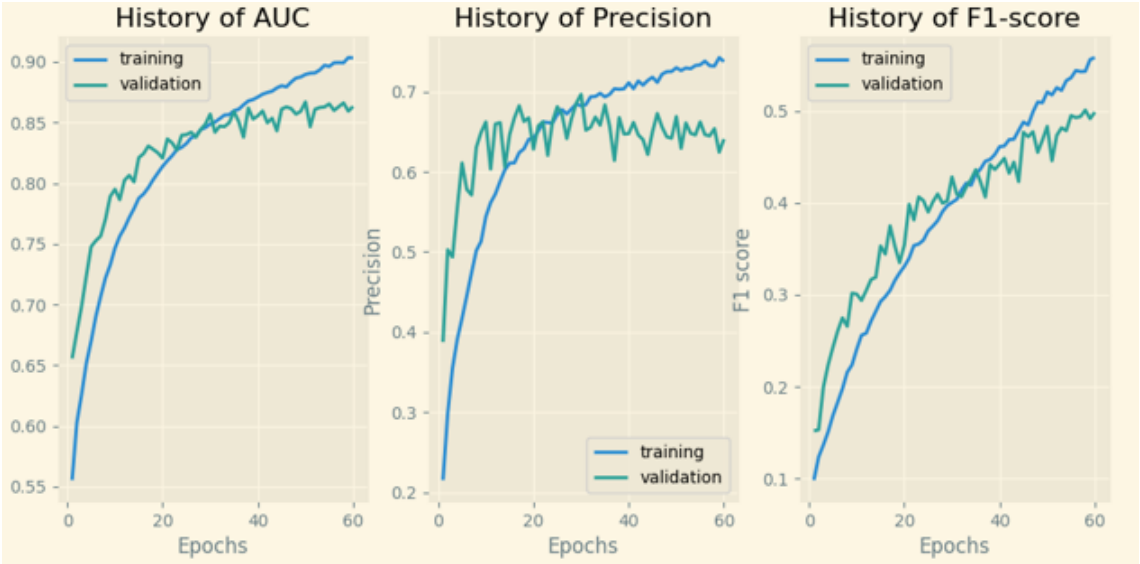


Figure 4.13: Xception Model's AUC, Precision and F1-Score Graphs

```
Confusion Matrix
[[ 571   66  566  968  662  714  448]
 [  68   10   63  103   62   75   55]
 [ 576   63  583  988  685  759  443]
 [1055  103 1002 1788 1314 1177  776]
 [ 703   69  717 1256  834  794  592]
 [ 709   69  703 1152  838  818  541]
 [ 417   63  485  817  523  528  338]]
```

Figure 4.14: Xception model's Confusion Matrix

```
Classification Report
              precision    recall  f1-score   support

       angry       0.14      0.14      0.14      3995
     disgust       0.02      0.02      0.02       436
        fear       0.14      0.14      0.14      4097
       happy       0.25      0.25      0.25      7215
     neutral       0.17      0.17      0.17      4965
         sad       0.17      0.17      0.17      4830
   surprised       0.11      0.11      0.11      3171

    accuracy                           0.17     28709
   macro avg       0.14      0.14      0.14     28709
weighted avg       0.17      0.17      0.17     28709
```

Figure 4.15: Xception model's Classification Report

ResNet-50

After testing the Xception architecture, a model with ResNet-50 architecture was tested, with the following results. Figures 4.16 and 4.17 shows the ResNet-50 model's accuracy and loss evolution, respectively. Figure 4.18 shows the model's ROC Curves (ROC is the curve that defines the AUC). Figure 4.14 shows the model's Confusion Matrix (for the ResNet-50 and VGG19 models another design was experimented while plotting the confusion matrix that uses different tones of blue so that the highest values stand out, although this design was not used for the rest of the models).

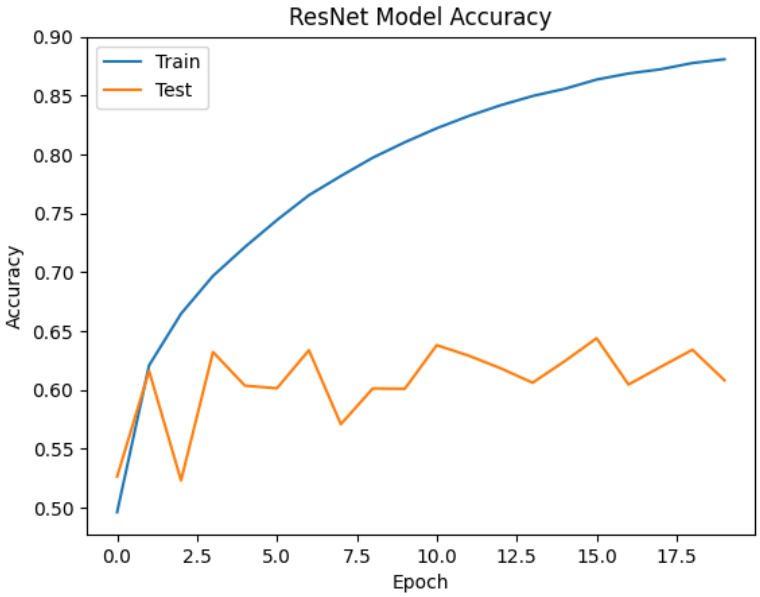


Figure 4.16: ResNet-50 model's Accuracy Graph

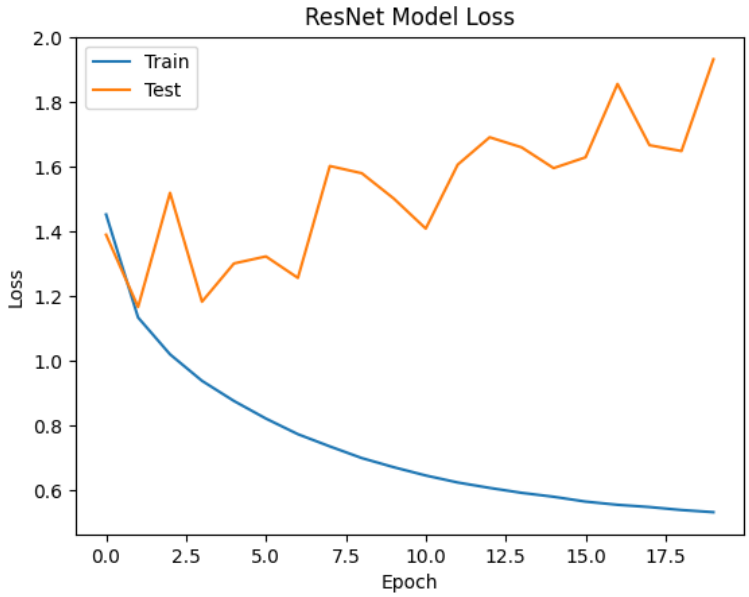


Figure 4.17: ResNet-50 model's Loss Graph

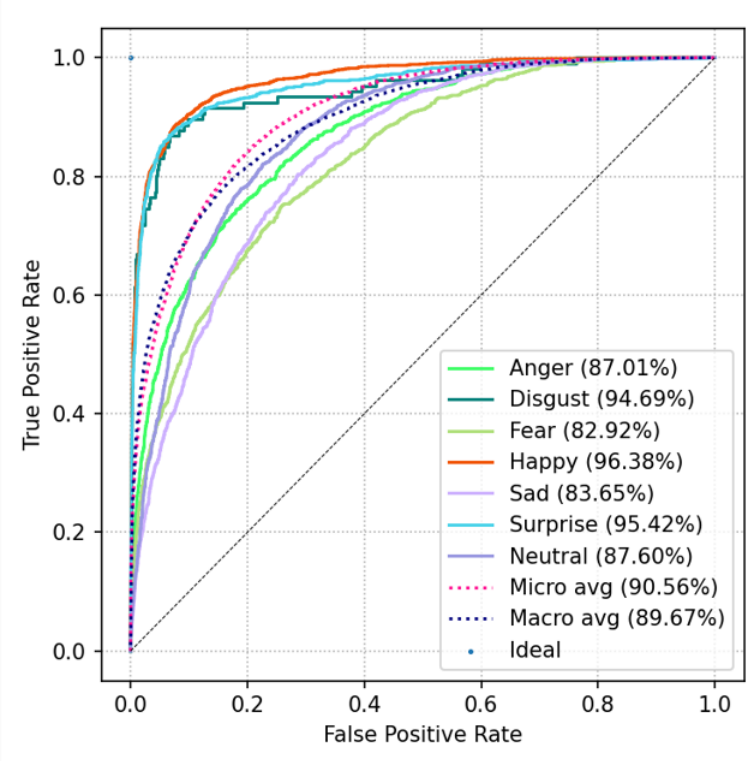


Figure 4.18: ResNet-50 model's ROC Curves

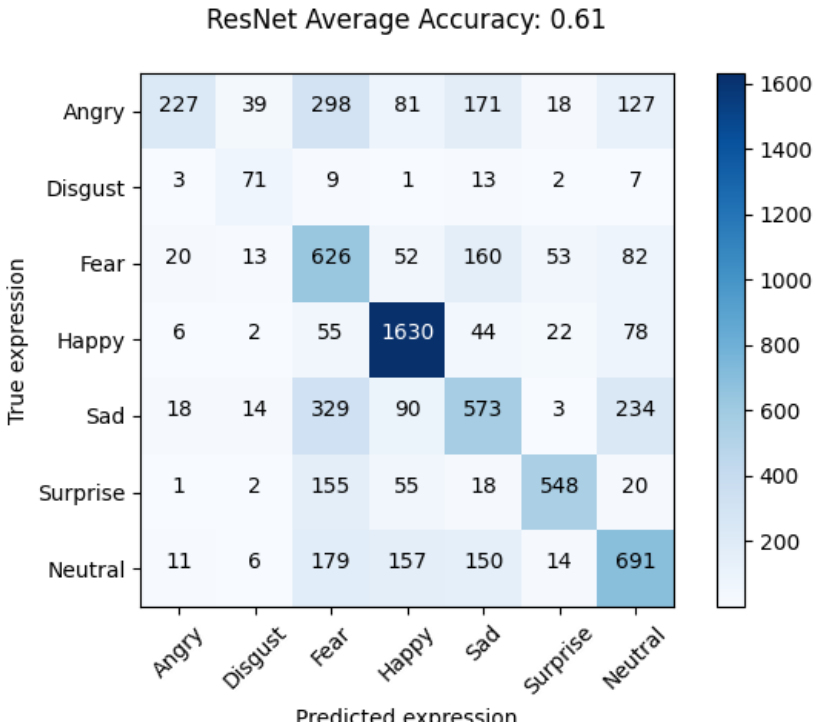


Figure 4.19: ResNet-50 model's Confusion Matrix

VGG19

The following graphs show the results of the trained model using the VGG19 architecture. Figures 4.20 and 4.21 show the VGG19 model's accuracy and loss evolution, respectively. Figure 4.22 shows the model's ROC curves and Figure 4.23 shows the Confusion Matrix.

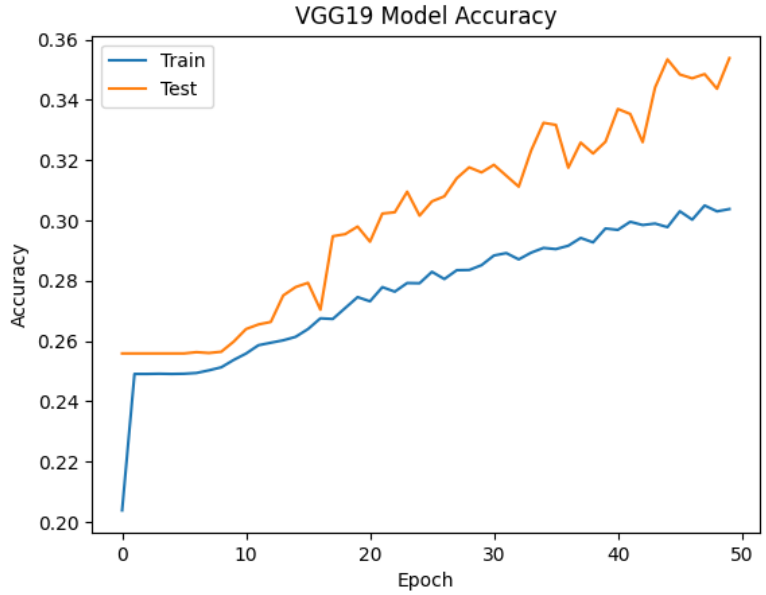


Figure 4.20: VGG19 model's Accuracy Graph

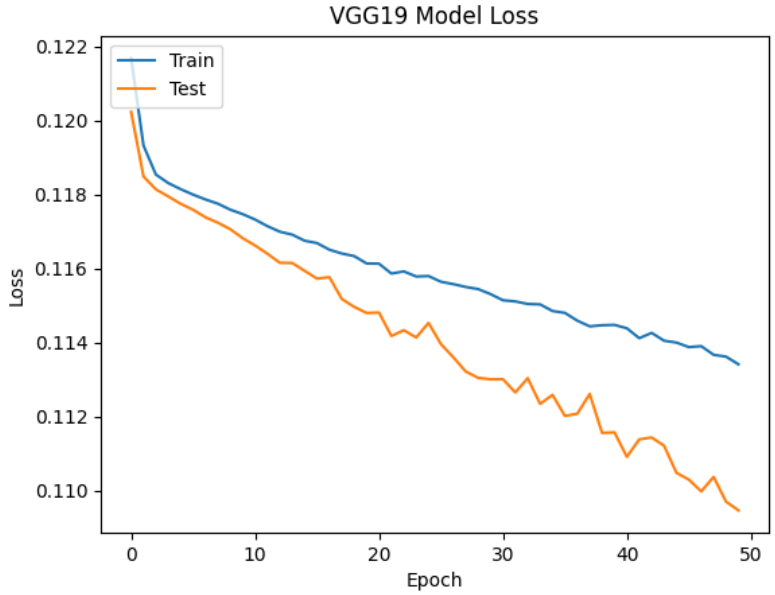


Figure 4.21: VGG19 model's Loss Graph

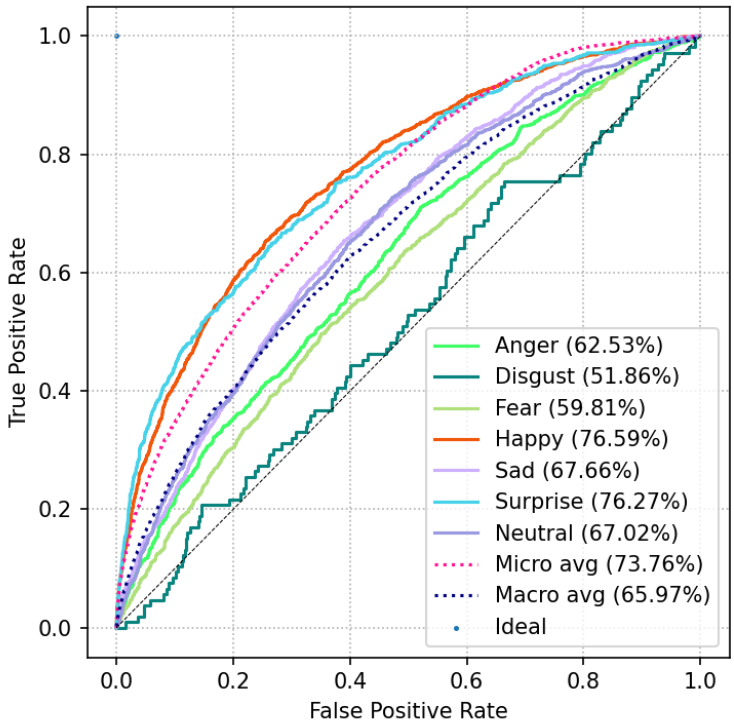


Figure 4.22: VGG19 model's ROC Curves

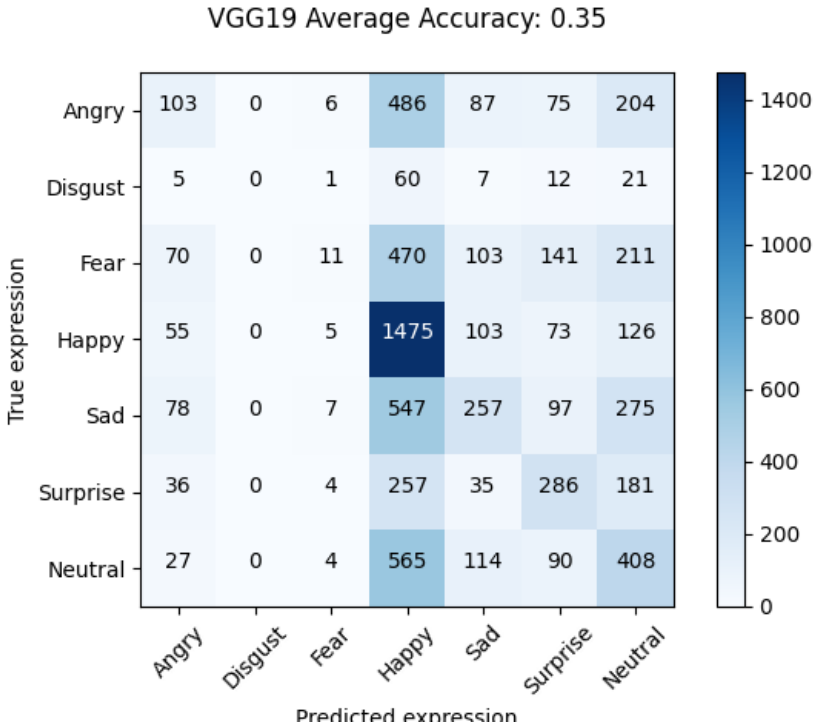


Figure 4.23: VGG19 model's Confusion Matrix

Architecture Comparison

When comparing the accuracy evolution when training the models with pre-trained architectures, it can be noticed that they all had high accuracy values except for VGG19, which only achieved around 35.7% testing accuracy and around 30.0% training accuracy. This can be due to the fact that the model might not have the correct amount of augmented data or due to the fact that, to achieve higher accuracy, the model needed more epochs to train so that the accuracy value continued to increase. The model with the best F1-Score values was the one trained with the Xception architecture, achieving an F1-Score value of around 0.54 in training and 0.5 in validation.

As for the confusion matrices, the model using the MobileNetV2 architecture achieved the best results, since the diagonal elements (this is, the elements that represent the True Positives and True Negatives) are highest than in the rest of the models, proving that the MobileNetV2 architecture had better performance in making correct predictions.

4.2.4 Optimizer Training

After training the models with different pre-trained architectures, it was time to start training the same model (in this case the model with the structure defined in 4.3 that was optimized originally with Adam) with different optimizers, as to compare results to identify the optimizer that maximizes the model's performance. For this step, 4 different optimizers were used, two of them (Adam and SGD) being used twice with a slightly different configuration. The optimizers (and specific configurations) used are listed below.

- Adam
- Adam with AMSGrad
- Adagrad
- Adadelta
- SGD
- SGD with Nesterov momentum

Adam

The Adam optimizer[47], name derived from Adaptive Moment Estimation, consists of a method for stochastic optimization, an extension of stochastic gradient descent (SGD). Since it requires first-order gradients, it requires a less amount of

memory. This method is best known for its adaptive learning rates for different parameters from estimates of first and second moments of the gradients (first moment being the mean and second moment is uncentered variance, which is the exponential moving average of the square of the gradients) and it has several benefits: its parameters' magnitudes updates do not vary with the rescaling of the gradient, it works with sparse gradients, it has faster running time, it requires less tuning than most optimizer algorithms, among others. The equation shown below represents how the Adam optimizer works (β_1 and β_2 represent the decay rates of the gradients' average, m and v are moving averages and $\frac{\delta L}{\delta w_t}$ are gradients on current mini-batch t).

$$m_t = \beta_1 m_{t-1} + (1-\beta_1)[\tfrac{\delta L}{\delta w_t}] v_t = \beta_2 v_{t-1} + (1-\beta_2)[\tfrac{\delta L}{\delta w_t}]^2$$

Since m and v are estimates of the first and second moments, the following properties are expected:

$$E[m_t] = E[g_t] \text{ and } E[v_t] = E[{g_t}^2]$$

The averages are initialized with zeros so the first gradient's formula is $m_0 = 0$. However, as the value of t (in m_t) increases, the first values of gradients start to contribute less to the overall value, since they are multiplied by smaller values of β. This can be summarized in the following equation:

$$m_t = (1-\beta_1) \sum_{i=0}^{t} \beta_1^{t-i} g_i$$

For some of the first values of t the pattern should be as it follows:

$$m_1 = \beta_1 m_0 + (1-\beta_1) g_1 = (1-\beta_1) g_1$$
$$m_2 = \beta_1 m_1 + (1-\beta_1) g_2 = \beta_1 (1-\beta_1) g_1 + (1-\beta_1) g_2$$
$$m_3 = \beta_1 m_2 + (1-\beta_1) g_3 = {\beta_1}^2 (1-\beta_1) g_1 + \beta_1 (1-\beta_1) g_2 + (1-\beta_1) g_3$$

This being said, the simplified expression that finally provides the expected value of m is as follows ($\zeta = 0$ if the true second moment $E[{g_i}^2]$ is stationary, otherwise is can be kept at a small value since the exponential decay rate β_1 can and should be chosen so that the exponential moving average assigns small weights to gradients that are in the distant past):

$$E[m_t] = E[g_i](1-{\beta_1}^t) + \zeta$$

It is also necessary to correct the estimator, also known as bias correction.

$$\hat{m}_t = \tfrac{m_t}{1-{\beta_1}^t}$$

$$\hat{v}_t = \tfrac{v_t}{1-{\beta_2}^t}$$

Adam's weight update can be summed in the following expression (w represents the weights and η is the step size:

$$w_t = w_t - 1 - \eta \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon}$$

In 4.2.3 the Adam model's results were shown.

Adam with AMSGrad

AMSGrad is an extension to Adam's version of gradient descent. It consists of an attempt to improve its convergence properties, while avoiding major oscillations in the learning rate for the input variables. It utilizes the maximum value of past squared gradient v_t instead of its exponential average for parameter updates, as can be seen in the expression below (the m_t and v_t expressions are the same as before):

$$\hat{v}_t = \max(\hat{v}_{t-1}, v_t)$$

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{v}_t} + \epsilon}$$

In Sashank J. Reddi et al.[48], it was proven that there are simple convex optimization settings where the Adam optimizer is unable to converge into the optimal solution. Additionally, Adam updates its gradients separately and these updates may change their values depending on the calculations of the gradients' exponential moving average. However, in some situations these updates may be too big for the occurrence of convergence. In sum, the main difference between Adam and AMSGrad resides in the calculation of the second moment vector (used to updated the parameters).

Figure 4.24 shows the initialization of the AMSGrad algorithm.

```
model.compile(optimizer=Adam(learning_rate=0.0001, decay=1e-6,amsgrad=True),
              loss='categorical_crossentropy',
              metrics=['accuracy'])
```

Figure 4.24: AMSGrad Initialization

The results of this model were as follows:

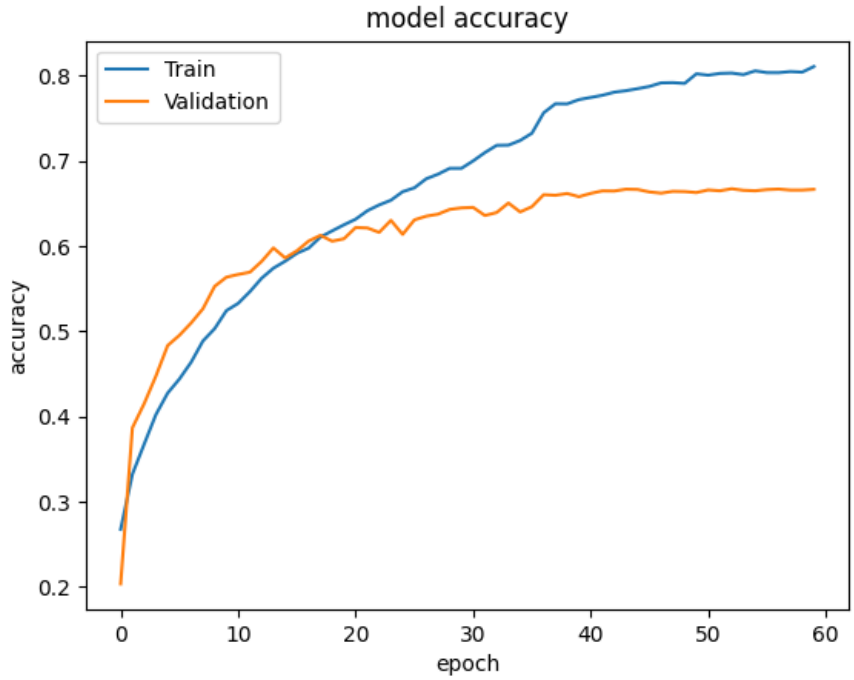


Figure 4.25: AMSGrad model's Accuracy Graph

8

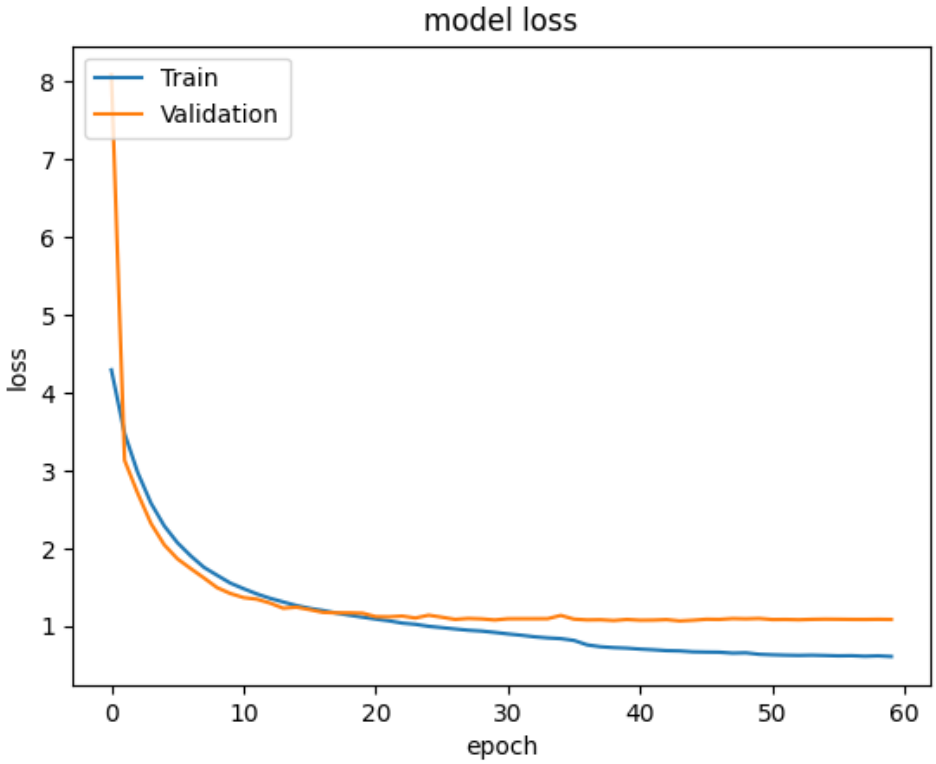


Figure 4.26: AMSGrad model's Loss Graph

```
Confusion Matrix
[[ 542   57  545  976  778  658  439]
 [  70    5   52  112   79   62   56]
 [ 541   54  520 1095  749  671  467]
 [ 942   93  937 1917 1328 1244  754]
 [ 660   68  643 1271  925  806  592]
 [ 665   70  650 1202  871  811  561]
 [ 415   42  426  850  518  568  352]]
```

Figure 4.27: AMSGrad model's Confusion Matrix

Classification Report

	precision	recall	f1-score	support
angry	0.14	0.14	0.14	3995
disgust	0.01	0.01	0.01	436
fear	0.14	0.13	0.13	4097
happy	0.26	0.27	0.26	7215
neutral	0.18	0.19	0.18	4965
sad	0.17	0.17	0.17	4830
surprised	0.11	0.11	0.11	3171
accuracy			0.18	28709
macro avg	0.14	0.14	0.14	28709
weighted avg	0.18	0.18	0.18	28709

Figure 4.28: AMSGrad model's Classification Report

Adagrad

Adagrad[49] consists of a gradient-based optimizer that adapts the learning rate to the parameters. Since this optimizer performs bigger updates for less common parameters and smaller updates for more common parameters, it is a good option for dealing with sparse data. The following equation provides the parameter update value (the learning rate η is modified at each time step t for every parameter θ_i based on the past gradients for θ_i):

$$\theta_{t+1,i} = \theta_{t,i} - \frac{\eta}{\sqrt{G_{t,ii}+\epsilon}} \cdot g_{t,i}$$

The G_t represents a diagonal matrix where each diagonal elemtent i,i is the sum of the squares of the gradients θ_i up to time step t and ϵ consists of a term that impedes the division by zero (has usually a value of about $1e$-8, thus making it always possible.

One advantage of the Adagrad optimizer is that it removes the need of manually tuning the learning rate. However, it struggles with the accumulation of squared gradients in the denominator, since the new added terms are positive, the sum grows and the final value starts to decrease rapidly to minuscule values.

```
model.compile(optimizer=Adagrad(learning_rate=0.0001, decay=1e-6),
              loss='categorical_crossentropy',
              metrics=['accuracy'])
```

Figure 4.29: Adagrad Initialization

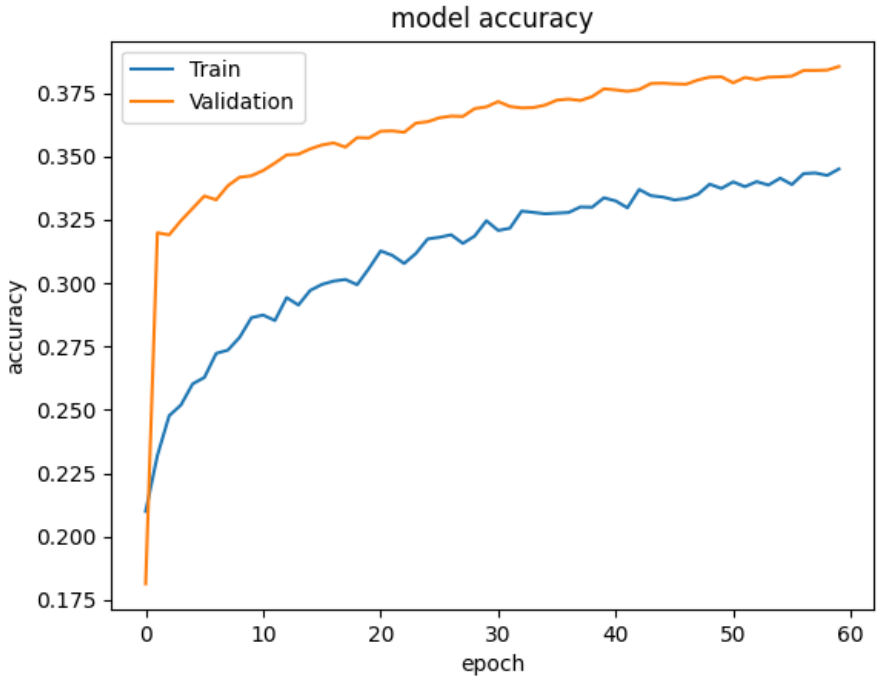


Figure 4.30: Adagrad model's Accuracy Graph

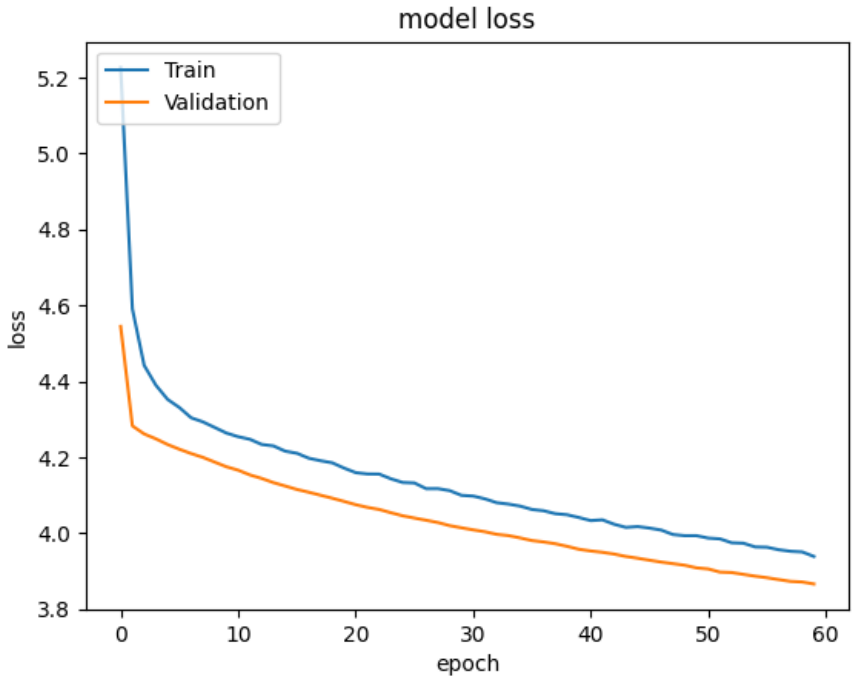


Figure 4.31: Adagrad model's Loss Graph

```
Confusion Matrix
[[ 171    0  104 1707  874  362  777]
 [  11    0   17  178  102   34   94]
 [ 171    0  122 1729  949  371  755]
 [ 299    0  222 3036 1676  630 1352]
 [ 213    0  138 2058 1159  454  943]
 [ 219    0  152 2020 1113  448  878]
 [ 141    0   83 1316  751  300  580]]
```

Figure 4.32: Adadelta model's Confusion Matrix

	precision	recall	f1-score	support
angry	0.14	0.04	0.07	3995
disgust	0.00	0.00	0.00	436
fear	0.15	0.03	0.05	4097
happy	0.25	0.42	0.32	7215
neutral	0.17	0.23	0.20	4965
sad	0.17	0.09	0.12	4830
surprised	0.11	0.18	0.14	3171
accuracy			0.19	28709
macro avg	0.14	0.14	0.13	28709
weighted avg	0.17	0.19	0.17	28709

Figure 4.33: Adagrad model's Classification Report

Adadelta

Adadelta is another stochastic optimization approach that consists of a per-dimension learning rate method for gradient descent. This approach is adaptive using only first order information and it is considered to be lighter in terms of computational power use. It is an extension of Adagrad, with the intent of improving its flaws. This is, instead storing all past squared gradients, Adadelta has a fixed value w of amassed past squared gradients.

In the Adadelta approach, the sum of gradients is defined as a decaying average of the past squared gradients. The following equation gives us the values of the running average (γ is usually set to 0.9[49]):

$$E[g^2]_t = \gamma E[g^2]_{t-1} + (1-\gamma){g_t}^2$$

The parameter update vector is given by the following expression:

$$\Delta\theta_t = -\eta \cdot g_{t,i}$$

$$\theta_{t+1} = \theta_t + \Delta\theta_t$$

$$\Delta\theta_t = -\frac{\eta}{\sqrt{E[g^2]_t + \epsilon}} g_t$$

Since the denominator of the fraction is the expression for RMS (Root Mean Squared) error criterion of the gradient, the equation can be simplified as follows:

$$\Delta\theta_t = -\frac{\eta}{RMS[g]_t} g_t$$

One of the main advantages of the Adadelta optimizer is that, when using it, there is no need to specify a learning rate, although one can be set if needed.

The results for the model trained with Adadelta can be seen below.

```
model.compile(optimizer=Adadelta(learning_rate=0.0001, decay=1e-6),
              loss='categorical_crossentropy',
              metrics=['accuracy'])
```

Figure 4.34: Adadelta Initialization

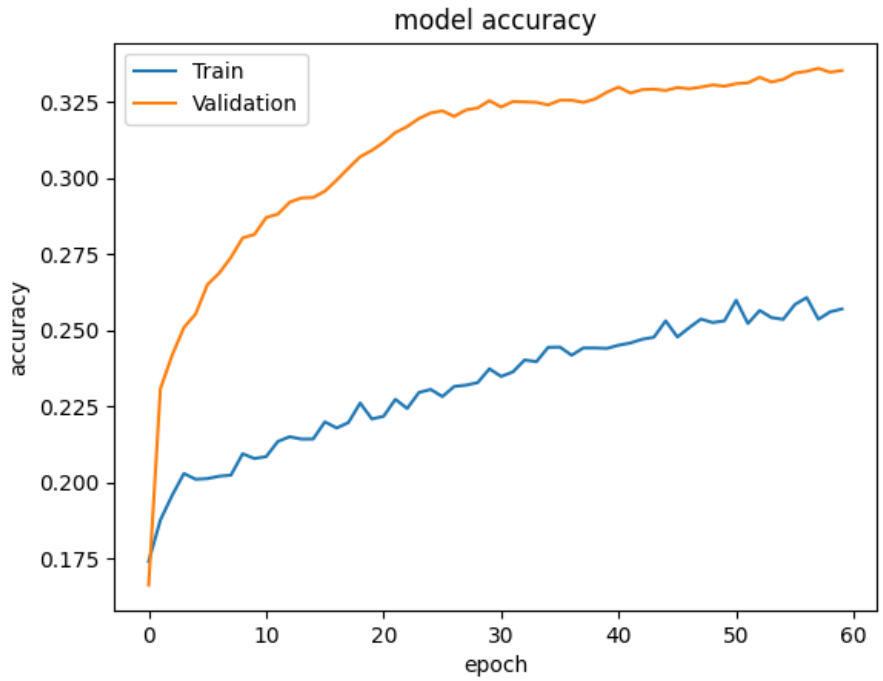


Figure 4.35: Adadelta model's Accuracy Graph

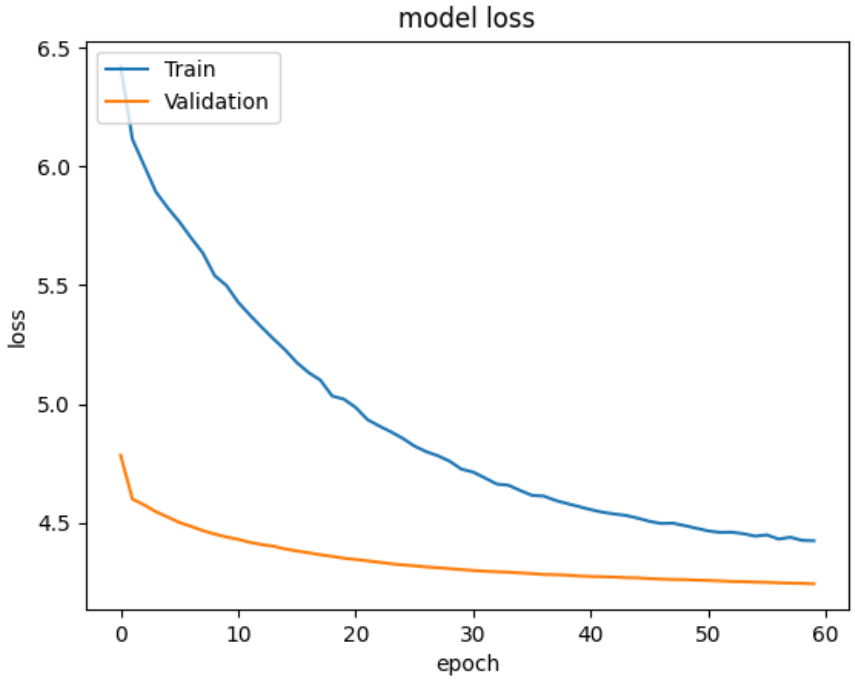


Figure 4.36: Adadelta model's Loss Graph

```
Confusion Matrix
[[ 151    0  214 1745 1034  354  497]
 [  16    0   18  180  139   29   54]
 [ 143    0  210 1763 1086  409  486]
 [ 263    2  372 3121 1930  572  955]
 [ 176    3  267 2204 1233  424  658]
 [ 172    1  260 2088 1269  412  628]
 [ 113    0  147 1372  865  250  424]]
```

Figure 4.37: Adadelta model's Confusion Matrix

```
Classification Report
              precision    recall  f1-score   support

       angry       0.15      0.04      0.06      3995
     disgust       0.00      0.00      0.00       436
        fear       0.14      0.05      0.08      4097
       happy       0.25      0.43      0.32      7215
     neutral       0.16      0.25      0.20      4965
         sad       0.17      0.09      0.11      4830
   surprised       0.11      0.13      0.12      3171

    accuracy                           0.19     28709
   macro avg       0.14      0.14      0.13     28709
weighted avg       0.17      0.19      0.17     28709
```

Figure 4.38: Adadelta model's Classification Report

SGD

Stochastic Gradient Descent, also known as SGD[49], is an optimizing algorithm that performs parameter refresh for each training sample $x^{(i)}$ and label $y^{(i)}$.

$$\theta = \theta - \eta \cdot \nabla_\theta J\left(\theta; x^{(i)}; y^{(i)}\right)$$

Unlike its predecessor (Batch Gradient Descent, which recomputes gradients for similar examples before refreshing each parameters), SGD performs one update at a time, thus avoiding redundancy. However, SGD performs high variance updates that cause the function to oscillate aggressively. This is due to the fact that only one sample from the dataset is chosen randomly for each iteration, thus making the path to reach the minima usually noisier than other types of Gradient Descent algorithms, although such fact is not very important since its importance is overcome by the fact that SGD still reaches the minima with a significantly faster training time, as can be seen in Table 4.2.

Using SGD, the gradient of the cost function of a single example is found at each iteration instead of the sum of the gradient of the cost function of all the examples (that happens in Batch Gradient Descent, where the batch is considered to be the size of the whole dataset, the batch being the total number of samples from a dataset used for calculating the gradient for each iteration).

The results of the model trained with SGD are shown below:

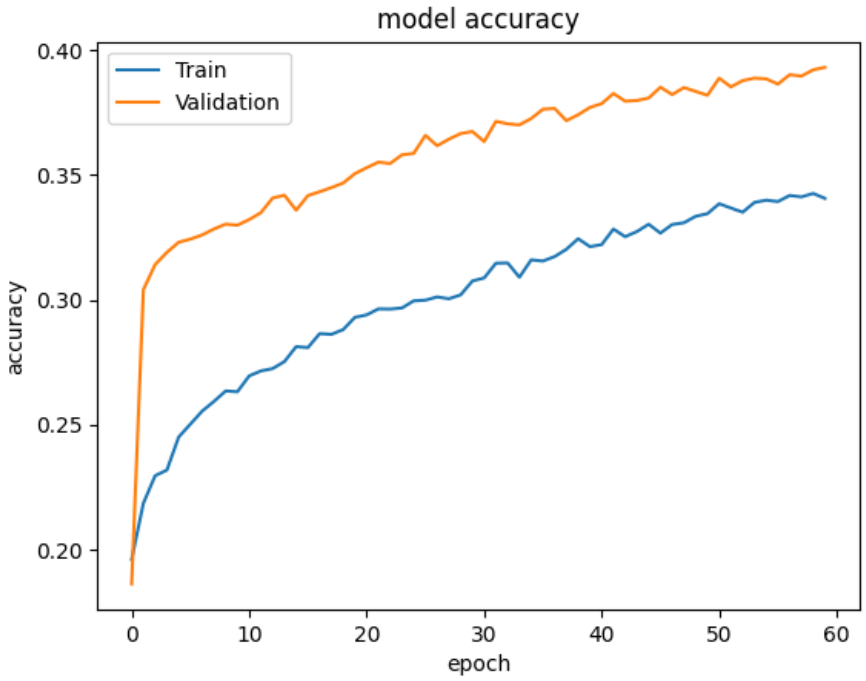


Figure 4.39: SGD model's Accuracy Graph

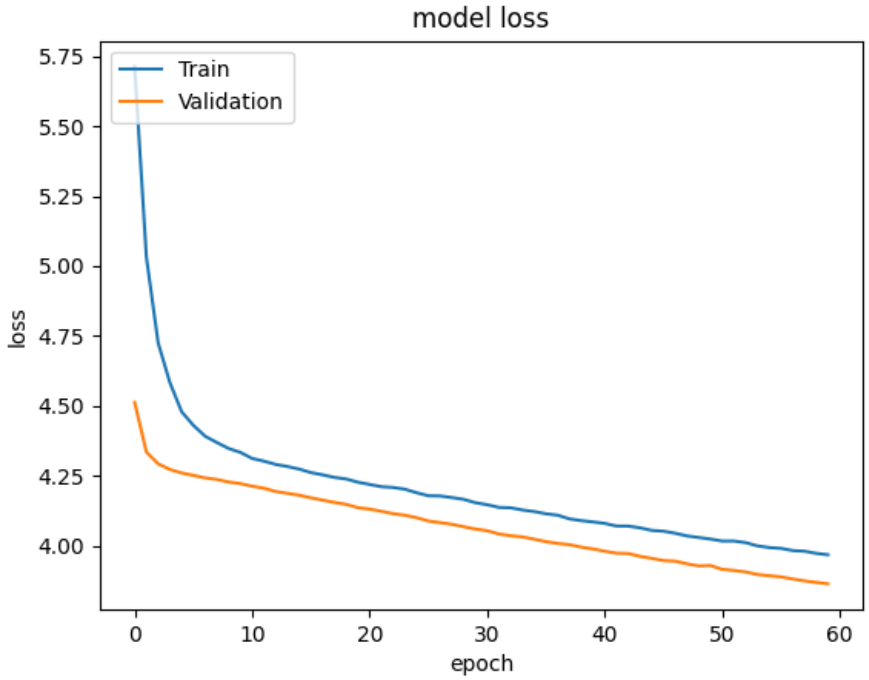


Figure 4.40: SGD model's Loss Graph

```
Confusion Matrix
[[ 125    0   81 1939  852  359  639]
 [  16    0   10  224   83   40   63]
 [ 119    0   93 1971  864  418  632]
 [ 223    0  169 3405 1515  711 1192]
 [ 158    0  102 2411 1045  474  775]
 [ 139    0  109 2297 1051  467  767]
 [  84    0   63 1498  721  296  509]]
```

Figure 4.41: SGD model's Confusion Matrix

	precision	recall	f1-score	support
angry	0.14	0.03	0.05	3995
disgust	0.00	0.00	0.00	436
fear	0.15	0.02	0.04	4097
happy	0.25	0.47	0.32	7215
neutral	0.17	0.21	0.19	4965
sad	0.17	0.10	0.12	4830
surprised	0.11	0.16	0.13	3171
accuracy			0.20	28709
macro avg	0.14	0.14	0.12	28709
weighted avg	0.17	0.20	0.16	28709

Figure 4.42: SGD model's Classification Report

SGD with Nesterov Momentum

Nesterov Momentum, also known as Nesterov Accelerated Gradient[49], is a type of momentum, a method that helps accelerate SGD and also softens oscillations. Is consists of a gradient descent step, followed by a momentum step. The following equations represent the gradient descent and the momentum steps, respectively (θ representing the parameters after they've been updated between both stages, and Φ_{t+1} represents the state of the parameters after the gradient descent stage):

$$\Phi_{t+1} = \theta_t - \epsilon_t \nabla f(\theta_t)$$

$$\theta_{t+1} = \Phi_{t+1} + \mu_t(\Phi_{t+1} - \Phi_t)$$

The main hindrance when using Nesterov Momentum is identifying what is the best learning rate and momentum values so that this method can achieve the best convergence.

```
model.compile(optimizer=SGD(learning_rate=0.0001,decay=1e-6,momentum=0.9,nesterov=True),
              loss='categorical_crossentropy',
              metrics=['accuracy'])
```

Figure 4.43: SGD with Nesterov Momentum model's Initialization

Figure 4.44: SGD with Nesterov Momentum model's Accuracy Graph

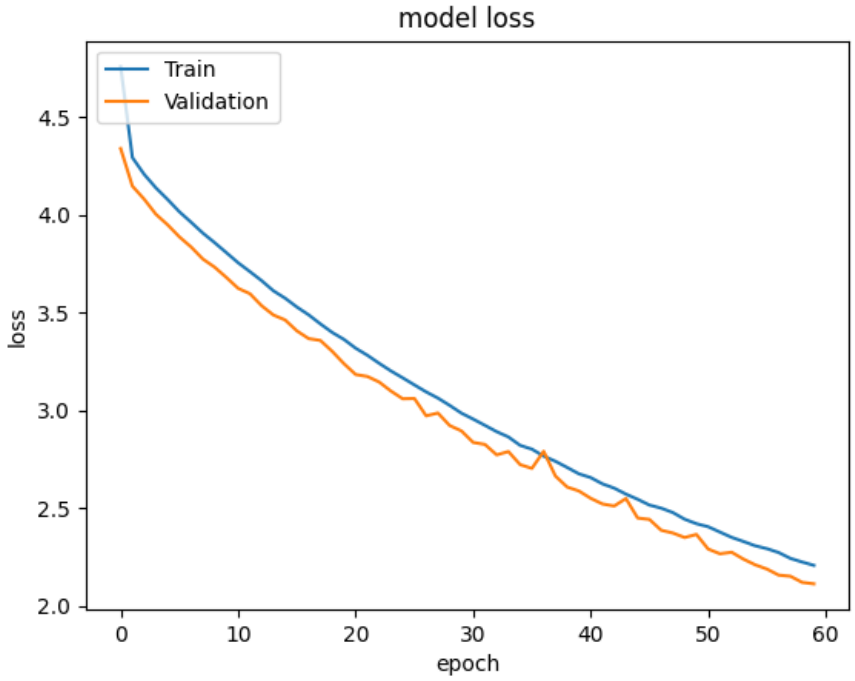


Figure 4.45: SGD with Nesterov Momentum model's Loss Graph

```
Confusion Matrix
[[ 450   10  412 1272  659  659  533]
 [  48    3   52  140   68   75   50]
 [ 494   12  382 1242  682  709  576]
 [ 863   30  776 2184 1206 1216  940]
 [ 572   18  503 1647  802  763  660]
 [ 536   15  492 1491  821  806  669]
 [ 382   13  317 1012  507  521  419]]
```

Figure 4.46: SGD with Nesterov Momentum model's Confusion Matrix

Classification Report

	precision	recall	f1-score	support
angry	0.13	0.11	0.12	3995
disgust	0.03	0.01	0.01	436
fear	0.13	0.09	0.11	4097
happy	0.24	0.30	0.27	7215
neutral	0.17	0.16	0.17	4965
sad	0.17	0.17	0.17	4830
surprised	0.11	0.13	0.12	3171
accuracy			0.18	28709
macro avg	0.14	0.14	0.14	28709
weighted avg	0.17	0.18	0.17	28709

Figure 4.47: SGD with Nesterov Momentum model's Classification Report

4.2.5 Results Comparison

When analysing the respective accuracy graphs, it could be observed that:

- The models with the highest train and validation accuracy were the ones that used Adam and Adam with AMSGrad with around 80.0% training accuracy and around 65.0% validation accuracy, both of them having very similar accuracy values throughout the training process
- The models with the lowest train and validation accuracy were the ones that used Adadelta and Adagrad, the first one having around 26.0% training accuracy and 33.0% validation accuracy. The model using the Adagrad optimizer had around 33.0% training accuracy and 38.0% validation accuracy.

As for the classification reports, the results were quite similar in terms of precision, recall and f1-score. The models trained with Adam, Adam with AMSGrad and SGD with Nesterov momentum achieved a slightly higher f1-score macro average (14%) although Adam with AMSGrad had the higher weighted average (taking in consideration the proportion of the emotions in the dataset to calculate this average).

Adam	890 ms
Adam with AMSGrad	884 ms
Adagrad	1000ms
Adadelta	884 ms
SGD	178 ms
SGD with Nesterov Momentum	946 ms

Table 4.2: Times per step

Chapter 5

Conclusions

Artificial Intelligence and in particular Machine Learning have been impacting several areas with results that, sometimes, surpass the human performance. Sentiment and emotion analysis is a field that has been exploring the use of ML to identify emotions reveled by several aspects of the human behaviour. In particular, Facial Expression Recognition, an interdisciplinary field that puts together the scientific areas of Computer Sciences and Psychology, has been calling the attention of the research community. It become clear that there are still a few hindrances to this field, such as the gathering of enough data to create large and balanced datasets with quality information that can greatly boost the performance of AI models, and discovering new and more efficient NN architectures that can better extract higher level features. The work developed in the scope of this thesis aims at understanding the impact that some aspects that can have on the performance of FER system.

Regarding the development of the project itself, a base model architecture was defined and trained, as well as other pre-trained architectures so as to perceive the influence a CNN architecture can have in the performance of a Facial Expression Recognition model. This work was not just limited to testing various architectures, as it was mainly focused in testing the same architecture with different optimizing algorithms and observing the influence they have in the performance of the model.

There were a certain amount of difficulties and obstacles in carrying out this project, since all the concepts present in it were new, whether is was the theories that explain the existence of emotions and the differences between them, the concept of Artificial Intelligence and the development of an AI model, or the fact that this

project was developed in a previously unknown programming language and, despite having encountered difficulties in many phases of this project, it is safe to say those were partially or fully overcome. Although some of the model performances were not ideal, we were able to develop different types of AI models successfully.

Although some conclusions could already be drawn based on the experiments, some improvement and future developments can still be identified:

Improve dataset quality

Although FER2013 is an easily accessible dataset with a high amount of data, it is a dataset with a certain lack of balance. As such, other more balanced datasets can be used in the future to develop models with better performance and ability to correctly predict emotions.

Explore the MobileNetV2 architecture

As mentioned in 3.4, MobileNetV2 is a CNN architecture that is mainly destined for use in mobile devices and, as such, this option can be explored. In the future, the optimizers can be tested with the MobileNetV2 architecture, along with the tuning of other hyperparameters so as to maximize the performance of the model for later testing in an actual mobile device, such as a smartphone or a smartwatch.

Combining Facial Expression Recognition with other fields

One of the main advantages of Artificial Intelligence is that it is very versatile. This means that it is possible to combine the field of emotion identification with other fields such as music (i.e., play a sad melody when sadness is detected or a joyful waltz when detecting happiness) or psychiatry (i.e., detect a person's inability to feel joy, detect depression or bipolar disorder).

References

[1] D. Konstan, "Affect and Emotion in Greek Literature," in *Oxford Handbook Topics in Classical Studies*, Oxford University Press. [Cited on page 3]

[2] P. Ekman, *Universal and Cultural Differences in Facial Expression of Emotions*. J. Cole, 1972. [Cited on pages 4 and 5]

[3] E. Tromp and M. Pechenizkiy, "Rule-based emotion detection on social media: Putting tweets on plutchik's wheel," *CoRR*, vol. abs/1412.4682, 2014. [Cited on pages 4, 5, and 11]

[4] K. Cherry, "Emotions and types of emotional responses." Available at `https://www.verywellmind.com/what-are-emotions-2795178`, Feb. 2022. [Cited on pages 4, 6, and 8]

[5] C. Darwin, *The Expression Of The Emotions In Man And Animals*. John Murray, 1872. [Cited on page 4]

[6] G.-B.-A. Duchenne, *The Mechanism of Human Facial Expression*. Cambridge University Press, 1990. [Cited on pages 4 and 9]

[7] P. J. Snyder, R. Kaufman, J. Harrison, and P. Maruff, "Charles darwin's emotional expression "experiment" and his contribution to modern neuropharmacology," *Journal of the History of the Neurosciences*, vol. 19, no. 2, pp. 158–170, 2010. PMID: 20446159. [Cited on page 4]

[8] J. R. C. G. J. S. A. A. M. M. S. B. Hillel Aviezer, Ran R. Hassin, "Angry, disgusted, or afraid?: Studies on the malleability of emotion perception," vol. 19, no. 7, p. 729, 2008. [Cited on pages 5 and 41]

[9] N. P. P. J. A. B. J. H. H. Simeng Gu, Fushun Wang, "A model for basic emotions using observations of behavior in drosophila," p. 1, 2019. [Cited on page 5]

[10] W. M. H. Association, "Emotions." Available at `https://mind.help/topic/emotions/`, June 2022. [Cited on page 6]

[11] J. D. S. . C. U. M. W. Balzer, "Structuralist knowledge representation: Paradigmatic examples (poznan studies in the philosophy of the sciences and the humanities," vol. 75, pp. 219–250, 2000. [Cited on page 6]

[12] J. A. Russel, "A circumplex model of affect," *Journal of Personality and Social Psychology*, vol. 39, no. 6, p. 1161–1178, 1980. [Cited on page 7]

[13] C. M. Whissell, "Emotion: A classification of current literature," vol. 59, no. 2, 1984. [Cited on page 7]

[14] A. Garcia Rojas M., F. Vexo, D. Thalmann, A. Raouzaiou, K. Karpouzis, and S. Kollias, "Emotional body expression parameters in virtual human ontology," 01 2006. [Cited on pages vii and 7]

[15] T. Baêna, "Paul ekman e a descoberta das microexpressões faciais." Avaiable at `https://inbodylanguage.com/paul-ekman-e-a-descoberta-das-microexpressoes-faciais/`, June 2022. [Cited on page 8]

[16] Alchetron, "Duchenne de boulogne." Avaiable at `https://alchetron.com/Duchenne-de-Boulogne`, June 2022. [Cited on pages vii and 10]

[17] P. E. Group, "Facial action coding system." Avaiable at `https://www.paulekman.com/facial-action-coding-system/`, June 2022. [Cited on page 10]

[18] J. X. Yue Zhao, "A convolutional neural network for compound micro-expression recognition," *Sensors*, vol. 19, no. 5553, p. 5, 2019. [Cited on pages xi, 10, and 11]

[19] Martec, "Using plutchik's wheel of emotions in market research." Avaiable at `https://martecgroup.com/using-plutchiks-wheel-of-emotions-in-market-research/`, June 2022. [Cited on pages vii and 12]

[20] H. Karimova, "The emotion wheel: What it is and how to use it." Avaiable at `https://positivepsychology.com/emotion-wheel/`, June 2022. [Cited on page 12]

[21] K. R. Scherer, "What are emotions? and how can they be measured?," *Social Science Information*, vol. 44, no. 4, pp. 693–727, 2005. [Cited on pages vii, 13, and 14]

[22] S. V. S. K. Srikanth Murali, K. S. Gokul Krishnan, "Image denoising based on weighted regularized least square method," pp. 1–5, 2017. [Cited on pages vii and 20]

[23] Avaiable at `https://www.tensorflow.org/tutorials/images/data_augmentation`. [Cited on pages vii, 21, 22, and 23]

[24] IBM, "What are neural networks." Avaiable at `https://www.ibm.com/cloud/learn/neural-networks`, Sept. 2022. [Cited on pages 24 and 26]

[25] J. Chen, "Neural network definition." Avaiable at `https://www.investopedia.com/terms/n/neuralnetwork.asp`, Sept. 2022. [Cited on page 25]

[26] K. Melcher, "A friendly introduction to [deep] neural networks." Avaiable at `https://www.knime.com/blog/a-friendly-introduction-to-deep-neural-networks`, Sept. 2022. [Cited on pages vii and 25]

[27] P. Vadapalli, "7 types of neural networks in artificial intelligence explained." Avaiable at `https://www.upgrad.com/blog/types-of-neural-networks/`, Sept. 2022. [Cited on pages vii, 28, 29, 30, and 31]

[28] J. Delua, "Supervised vs. unsupervised learning: What's the difference?." Avaiable at `https://www.ibm.com/cloud/blog/supervised-vs-unsupervised-learning`, Mar. 2021. [Cited on page 32]

[29] L. S. H. N. A.-A. H. V. L. V. Q. T. I. P. B. T. P. Quang Hung Nguyen, Hai-Bang Ly, "Influence of data splitting on performance of machine learning models in prediction of shear strength of soil," *Mathematical Problems in Engineering*, vol. 2021, p. 1, 2021. [Cited on page 33]

[30] M. Z. Alom, T. M. Taha, C. Yakopcic, S. Westberg, P. Sidike, M. S. Nasrin, B. C. Van Esesn, A. A. S. Awwal, and V. K. Asari, "The history began from alexnet: A comprehensive survey on deep learning approaches," *arXiv preprint arXiv:1803.01164*, 2018. [Cited on page 34]

[31] H. S. J. K. S. S. S. M. Z. H. A. K. A. K. M. B. A. C. B. Olga Russakovsky, Jia Deng and L. Fei-Fei, "Imagenet large scale visual recognition challenge," *International Journal of Computer Vision*, vol. 115, no. 3, p. 200–270, 2015. [Cited on page 34]

[32] W. Setiawan and F. Damayanti, "Layers modification of convolutional neural network for pneumonia detection," *Journal of Physics: Conference Series*, vol. 1477, p. 052055, 03 2020. [Cited on page 35]

[33] Avaiable at `https://viso.ai/deep-learning/resnet-residual-neural-network/`. [Cited on page 35]

[34] Y. J. P. S. S. E. R. D. A. D. E. V. V. A. R. Christian Szegedy, Wei Liu, "Going deeper with convolutions," *CoRR*, vol. abs/1409.4842, 2014. [Cited on page 35]

[35] M. Sandler, A. G. Howard, M. Zhu, A. Zhmoginov, and L. Chen, "Inverted residuals and linear bottlenecks: Mobile networks for classification, detection and segmentation," *CoRR*, vol. abs/1801.04381, 2018. [Cited on page 35]

[36] S. T. F. H. Francisco Luque Sánchez, Isabelle Hupont, ": Revisiting crowd behaviour analysis through deep learning: Taxonomy, anomaly detection, crowd emotions, datasets, opportunities and prospects," *Information Fusion*, vol. 64, pp. 318–335, 2020. [Cited on page 36]

[37] W. S. E. I. Michael Revina, "A survey on human face expression recognition techniques," *Journal of King Saud University - Computer and Information Sciences*, vol. 33, no. 6, pp. 619–628, 2021. [Cited on pages vii, 36, and 37]

[38] Avaiable at `https://code.visualstudio.com/`. [Cited on page 39]

[39] Avaiable at `https://pandas.pydata.org/`. [Cited on page 40]

[40] Avaiable at `https://numpy.org/`. [Cited on page 40]

[41] Avaiable at `https://opencv.org/`. [Cited on page 40]

[42] Avaiable at `https://www.tensorflow.org/`. [Cited on page 40]

[43] Avaiable at `https://keras.io/`. [Cited on pages 40 and 42]

[44] Avaiable at `https://blogs.nvidia.com/blog/2012/09/10/what-is-cuda-2/`. [Cited on page 40]

[45] Avaiable at `https://matplotlib.org/`. [Cited on page 40]

[46] Avaiable at `https://www.tensorflow.org/api_docs/python/tf/keras/preprocessing/image/ImageDataGenerator`. [Cited on page 41]

[47] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," pp. 1–6, 2014. [Cited on page 54]

[48] S. J. Reddi, S. Kale, and S. Kumar, "On the convergence of adam and beyond," *CoRR*, vol. abs/1904.09237, 2019. [Cited on page 56]

[49] S. Ruder, "An overview of gradient descent optimization algorithms," *CoRR*, vol. abs/1609.04747, 2016. [Cited on pages 59, 61, 64, and 66]

Appendix A

Action Units

Table A.1: Main Action Units

AU number	FACS name	Muscles used
1	Inner Brow Raiser	Frontalis, pars medialis
2	Outer Brow Raiser	Frontalis, pars lateralis
4	Brow Lowerer	Depressor Glabellae, Depressor Supercilli, Currugator
5	Upper Lid Raiser	Levator palpebrae superioris
6	Cheek Raiser	Orbicularis oculi, pars orbitalis
7	Lid Tightener	Orbicularis oculi, pars palpebralis
9	Nose Wrinkler	Levator labii superioris alaquae nasi
10	Upper Lip Raiser	Levator Labii Superioris, Caput infraorbitalis
11	Nasolabial Deepener	Zygomatic Minor
12	Lip Corner Puller	Zygomatic Major
13	Cheek Puffer	Levator anguli oris (Caninus)
14	Dimpler	Buccinator
15	Lip Corner Depressor	Depressor anguli oris (Triangularis)
16	Lower Lip Depressor	Depressor labii inferioris
17	Chin Raiser	Mentalis
18	Lip Puckerer	Incisivii labii superioris and Incisivii labii inferioris
20	Lip Stretcher	Risorius
22	Lip Funneler	Orbicularis oris
23	Lip Tightener	Orbicularis oris
24	Lip Pressor	Orbicularis oris
25	Lips Part	• Depressor Labii • Relaxation of Mentalis • Orbicularis Oris
26	Jaw Drop	Masetter; Temporal and Internal Pterygoid relaxed
27	Mouth Stretch	Pterygoids, Digastric
28	Lip Suck	Orbicularis oris
41	Lid Droop	Relaxation of Levator Palpebrae Superioris
42	Slit	Orbicularis oculi
43	Eyes Closed	Relaxation of Levator Palpebrae Superioris
44	Squint	• Orbicularis oculi • Pars palpebralis
45	Blink	• Relaxation of Levator Palpebrae and Contraction Orbicularis • Oculi, Pars Palpebralis.
46	Wink	• Levator palpebrae superioris • Orbicularis oculi • Pars palpebralis

Table A.2: Head Movement Action Units

AU number	Description
51	Head Turn Left
52	Head Turn Right
53	Head Up
54	Head Down
55	Head Tilt Left
56	Head Tilt Right
57	Head Forward
58	Head Back

Table A.3: Eye Movement Action Units

AU number	Description
61	Eyes Turn Left
62	Eyes Turn Right
63	Eyes Up
64	Eyes Down

Table A.4: Emotions and respective description

Emotion	AU number	Description
Happiness or Joy	6 and 12	• Cheek Raiser • Lip Corner Puller
Sadness	1, 4 and 15	• Inner Brow Raiser • Brow Lowerer • Lip Corner Depressor
Surprise	1, 2, 5 and 26	• Inner Brow Raiser • Outer Brow Raiser • Upper Lid Raiser • Jaw Drop
Fear	1, 2, 4, 5, 7, 20 and 26	• Inner Brow Raiser • Outer Brow Raiser • Brow Lowerer • Upper Lid Raiser • Lid Tightener • Lip Stretcher • Jaw Drop
Anger	4, 5, 7 and 23	• Brow Lowerer • Upper Lid Raiser • Lid Tightener • Lip Tightener
Disgust	9, 15 and 16	• Nose Wrinkler • Lip Corner Depressor , , • Lower Lip Depressor
Contempt	12, 14 (on one side of the face)	• Lip Corner Puller • Dimpler