INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO

MESTRADO EM ENGENHARIA INFORMÁTICA

FullExpression - Emotion Recognition Software

RICARDO GOMES DA ROCHA

Outubro de 2019

POLITÉCNICO
DO PORTO

FullExpression

Emotion Recognition Software

Ricardo Gomes da Rocha

Master's Dissertation

Computer-Science

Software Engineering Specialization

Supervisor: Isabel Praça

Porto, October 2019

Instituto Superior de Engenharia do Porto

A ti Sofia e à tua sabedoria. Brindemos ao conhecimento!

"I believe that at the end of the century the use of words and general educated opinion will have altered so much that one will be able to speak of machines thinking without expecting to be contradicted."

Alan Turing, Computing machinery and intelligence

AGRADECIMENTOS

À Sofia Neves, não só pela revisão da tese, ajuda na análise estatística e debate de ideias, mas pelo apoio emocional.

À orientadora, Professora Doutora Isabel Praça, pela confiança, abertura no debate de ideias, confiança no meu conhecimento e na minha capacidade de organização, pela liberdade de pensamento e acima de tudo pela orientação. *Muito Obrigado!*

Ao Daniel Bento, por estar presente em todo o processo de conceção e escrita da tese, pelo debate de ideia e partilha de conhecimento.

À malta do ICVS, pela partilha de experiências sobre como “dar à luz uma tese”, pelo debate de temas académicos e de ciência em geral, pelo apoio durante todo o processo e por elevarem a qualidade deste trabalho.

A todos os participantes da experiência, por terem disponibilizado o seu tempo em prol deste trabalho.

À Família e amigos por todo o apoio.

Muito Obrigado!

ABSTRACT

During human evolution emotion expression became an important social tool that contributed to the complexification of societies. Human-computer interaction is commonly present in our daily life, and the industry is struggling for solutions that can analyze human emotions, in an attempt to provide better experiences. The purpose of this study was to understand if a software built using the transfer-learning technique on a deep learning model was capable of classifying human emotions, through facial expression analysis. A Convolutional Neuronal Network model was trained and used in a web application, which is available online. Several tools were created to facilitate the software development process, including the training and validation processes, and these are also available online. The data was collected after the combination of several facial expression emotion databases, such as KDEF_AKDEF, TFEID, Face_Place and jaffe. Software evaluation reveled an accuracy in identifying the correct emotions close to 80%. In addition, a comparison between the software and preliminary data from human's performance, on recognizing facial expressed emotions, suggested that the software performed better. This work can be useful in many different domains such as marketing (to understand the effect of marketing campaigns on people's emotional states), health (to help mental diseases diagnosis) and industry 4.0 (to create a better collaborating environment between humans and machines).

Keywords: Emotions, Facial Expressions, Artificial Intelligence, Deep Learning, Web Application

RESUMO

Durante a evolução da espécie humana, a expressões de emoções tornou-se uma ferramenta social importante, que permitiu a criação de sociedades cada vez mais complexas. A interação entre humanos e máquinas acontece regularmente, evidenciando a necessidade da indústria desenvolver soluções que possam analisar emoções, de modo a proporcionar melhores experiências aos utilizadores. O propósito deste trabalho foi perceber se soluções de software desenvolvidas a partir da técnica de *transfer-learning* são capazes de classificar emoções humanas, a partir da análise de expressões faciais. Um modelo que implementa a arquitetura *Convolutional Neuronal Network* foi escolhido para ser treinado e utilizado na aplicação web desenvolvida neste trabalho, que está disponível online. A par da aplicação web, diferentes ferramentas foram criadas de forma a facilitar o processo de criação e avaliação de modelos *Deep Learning*, e estas também estão disponíveis online. Os dados foram recolhidos após a combinação de várias bases de dados de expressões de emoções (KDEF_AKDEF, TFEID, Face_Place and jaffe). A avaliação do software demostrou uma precisão na classificação de emoções próxima dos 80%. Para além disso, uma comparação entre o software e dados preliminares relativos ao reconhecimento de emoções por pessoas sugere que o software é melhor a classificar emoções. Os resultados deste trabalho podem aplicados em diversas áreas, como a publicidade (de forma a perceber os efeitos das campanhas no estado emocional das pessoas), a saúde (para um melhor diagnóstico de doenças mentais) e na indústria 4.0 (de forma a criar um melhor ambiente de colaboração entre humanos e máquinas).

Palavras-chave: Emoções, Expressões Faciais; Inteligência Artificial; *Deep Learning*; Aplicações Web

INDEX

FIGURE INDEX

TABLE INDEX

ACRONYMS

AU	Action Units
AHP	Analytic Hierarchy Process
AI	Artificial Intelligence
ANN	Artificial Neuronal Network
CNN	Convolutional Neuronal Network
CPU	Central Processing Unit
DAN	Deep Alignment Network
DL	Deep Learning
DOM	Document Object Model
FACS	Facial Action Coding System
FAST	Function Analysis Systems Technique
FFE	Fuzzy Front End
GPU	Graphic Processing Unit
ISEP	Instituto Superior de Engenharia do Porto
ML	Machine Learning
NCD	New Concept Development model
NPD	New Product Development
OOP	Object-oriented programing
Scss/Sass	Syntactically Awesome Style Sheets
SPA	Single Page Application
TM	Tongue Movements
UI	User Interface

GLOSSARY

DOM	Stands for Document Object Model and is used on HTML to organize element in tree of objects
Batch size	Portion of the whole databased used to train or test the model
Deep learning model	Big and complex artificial neuronal network
Epoch	Cycle through the full training dataset
FFE	Fuzzy Front End
GPU	Graphic Processing Unit
ISEP	Instituto Superior de Engenharia do Porto
Learnable Layers	Number of deep learning model layers which the neuron weight will be changed through the training process
Learning Rate	Value used by optimizers to know how much the error should change the artificial neuronal network weight values
Loss Function	Function that calculates the difference between the predicted output and the actual output (error)
ML	Machine Learning

1. INTRODUCTION

During the human evolution, emotion expression became an important social tool that contributed to build more complex societies. Neuronal structures that support expression of emotions are present in the primitive brain, being most of the times expressed in an unconscious way [1]. Human-computer interactions are present on daily basis and it seems that computers are not an emotional wall, meaning that humans are able to express emotions through computers with the same frequency as they do face-to-face [2]. But are there software solutions able to recognize emotions and offer useful insights, improving human-computer interactions? Several studies suggest that possibility and point out different application areas, such as disease diagnostics and improvement of work productivity [3][4][5][6].

Therefore, emotion recognition techniques aim to provide mechanisms to detect and classify human emotions, expressed in different forms such as facial, verbal and behavior responses [7]. Recently, advancements on computer vision and Artificial Intelligence (AI) fields [8] make it possible to improve the accuracy and speed of facial emotion classification on the affective computing field [9]. Also, the new advancements on Artificial Neuronal Networks (ANNs) and the new computational resources available, improved the general classification accuracy rate on computer vision classification problems [10], being one of the most popular type of algorithms used nowadays [8][11][12].

1.1 UNDERSTANDING THE PROBLEM

The aim of FullExpression is to answer the question "Is it possible for a computer solution to recognize emotions, and offer useful insights to the society, with an accuracy and efficiency that makes it suitable to improve human-computer interactions?". To do so, it is important to separate the engineering part from the philosophic and psychological part and simplify the initial question in order to understand better the problem. This is possible by dividing it in small questions: "What is an emotion?", "What are the most distinct emotions?", "How do emotions affect society?", and "Is it possible for a computer program to be able to categorize emotions in order to improve and support interaction between humans and machine in the digital era?". Understanding the first and second question will give insight of what an emotion is, how many emotions exist, the most distinguishable ones and how they are expressed. The third question will allow to understand how important emotions are to society, how humans use emotions to communicate and the benefits of understanding emotions in different environments. The results of the first three questions will be used to answer the fourth question, by creating an engineering solution that will give rise to a computer program that can recognize emotions and interact with users.

1.2 BUSINESS PERSPECTIVE

This project called FullExpression aims to create a software solution to detect emotions, in order to improve human-computer interaction and human life. Using the most advanced AI algorithms, to obtain the best accuracy result for individuals and companies through software solutions and integrated services.

To do so, FullExpression proposes a software solution that enables the generation of emotional reports, the integration with other systems, the availability in different devices and the contribution to the scientific community regarding the new technologies and techniques used during this work.

First, a proof of concept should be created in order to understand if the business idea is reliable and feasible. The proof of concept software solution should be concerned first about reliability (emotion detection accuracy), then performance (velocity of emotion detection), extensibility (possibility of integration with other solutions), flexibility (easiness to extend functionalities) and, the technological base (technology and program languages used).

1.3 GOALS

Considering the problem understanding and the hypotheses exposed, this work aims to:

1. Identify the core emotions and the most distinguishable ones.
2. Understand the problem from a business perspective.
3. Understand how AI and Deep Learning (DL) techniques could be helpful for emotion recognition and classification.
4. Study different research works and commercial solutions and understand how these solutions are implemented and how well they performed on emotion classification.
5. Understand the impact of data and model fine-tuning, in order to achieve the best results for the domain problem.
6. Create software solutions to build and evaluate DL models.
7. Gather data from several emotions databases in order to have as much data as possible for training and evaluating DL models.
8. Run DL models exclusive on browser.
9. Create a web application capable of recognizing emotions from facial expressions by interacting with users using images (uploaded by the user or collected in real time using the webcam) with the goal of producing personal reports. In additional, the application should work in different devices and the model consumption should be fast enough to analyze images from the webcam.

10. All solutions should implement a modular architecture where some modules should be consumed as libraries, allowing their usage from internal and external application.
11. Conduct an experiment to gather preliminary data about the human capabilities of recognizing emotions from facial expressions and compare the data collected with the software capability.
12. Provide online access for the most of software created.

1.4 DOCUMENT STRUCTURE

The document is organized, besides this introductory chapter, in five other chapters: Context, State of the Art, Research Works and Commercial Solutions, Design and Implementation, and Evaluation.

In the Context chapter, it will be explored what emotions are, the different types of emotions and how emotions affect the society and the machine-human interaction. Then, a business value proposal will be presented.

In State of the Art chapter, AI field evolution, AI sub-fields, techniques/technologies and more in detail the DL field, will be addressed.

The Research Works and Commercial Solutions chapter will expose research works on emotion classification field, more specific for facial expression classification problems, and some commercial solutions available on the market.

In the Design chapter a solution design will be exposed as well as alternative designs.

Finally, the Evaluation chapter will expose the hypotheses and experiments done in order to test the solution.

2. CONTEXT

The present work aims to understand if a computer can detect and classify correctly different types of emotions. In order to do so, it is important to understand the human emotions world. The following chapters will give some level of understanding of what an emotion is, the different types of emotions, what are their purpose and the role of them in the society. Also, a business value proposition aligned with this work will be made.

2.1 EMOTIONS

The term “emotion” is wieldy used in society but a definition for emotion has generated discordance among the research community. In 1884, William James proposed that emotions are expressed as a result of physiological reaction events which are triggered by an environmental stimuli [13]. Since then, a countless number of definitions were proposed and nowadays many researches prefer to suggest that emotions are the result of physiological, behavior and subjective responses [7] [14] [15].

Figure 1 shows the three key elements of emotions, having the previous definition has a basis [7]: a physiological response and/or a behavior/expressive response relies on an experience. Also, three types of reactions can occur: physiological responses (e.g. faster breathing, dry mouth, changes on body temperature, nausea), behavior responses (e.g. aggressive body movements, start speaking loudly, changing daily routines) and expressive responses (e.g. facial expressions) [7].

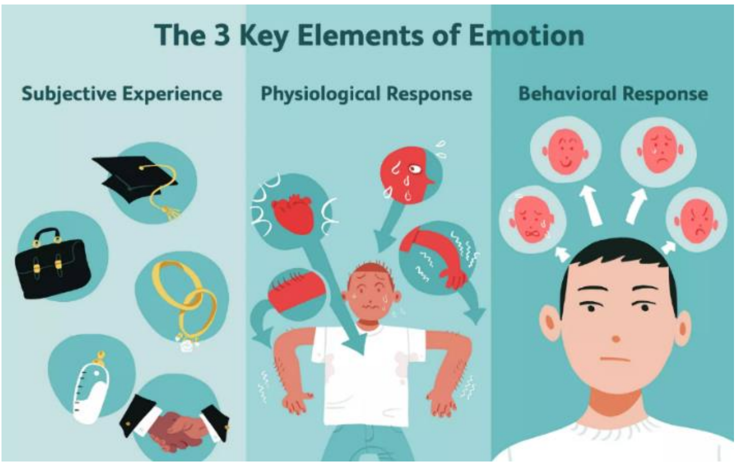


Figure 1 – The 3 key elements of emotion (Illustration by Emily Roberts, Verywell) [16]

2.1.1 EMOTION TYPES

There is a countless number of emotions, produced by micro-expressions that are difficult to detect. These can be influenced by several factors such as human body format or environment context (e.g. food and weather). Likewise, humans can feel different types of emotions at the same time and those emotions can have a cultural component too [14][17].

Trying to reduce the number of emotions to the most distinguishable ones, the following sections will expose the core emotions and which ones could lead to better classification results.

2.1.1.1 CORE EMOTIONS

The Plutchik's wheel [18] of emotions is a diagram that helps to understand how emotions are connected in terms of similarity and intensification (Figure 2). By combining emotions, secondary emotions are created (e.g. trust combined with fear creates submission). The core emotions are arranged by colors: the same color indicates similar emotions and the emotions in the middle of each flower petal are the combination of two different emotions. The intensity of emotions is arranged by layers: less color intensity means less emotion intensity. Opposite emotions are placed in opposite directions (e.g. anger is on the opposite side of fear).

By studying the diagram, it is possible to perceive the existence of eight core emotions: joy, sadness, trust, disgust, fear, anger, surprise and anticipation. Correspondingly, the opposite of joy is sadness, is the opposite of trust is disgust, the opposite of fear is anger and, the opposite of surprise is anticipation. Also, by combining the core emotions, eight additional ones can be considered: love, submission, awe, disapproval, remorse, contempt, aggressiveness and optimism.

Figure 2 – The Plutchik's wheel of emotions (Adapted from [19]**)**

2.1.1.2 EMOTIONS CLASSIFICATION

In order to evaluate types of emotions, Paul Ekman [20] created a classification model called Facial Action Coding System (FACS), which measures and evaluates facial muscles, eyes and head movements, and proposed seven universal emotional expressions: joy (happiness), sadness, surprise, fear, anger, disgust and contempt. Nevertheless, some emotions could use the same facial muscles and might lead to wrong categorization. By saying that, a research team at the University of Glasgow downsized the number of universal emotions to joy, sadness, anger and fear [21].

Based on the core emotions described, Figure 3 shows the emotions that have the highest probability of being well recognized at the center.

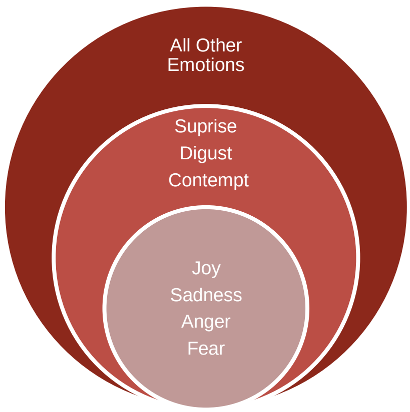


Figure 3 – Onion graph representing the emotions based on the probability of recognition

2.1.2 EMOTIONS AND THE SOCIETY

Several studies suggest that emotions are widely present in society and can affect it positively and negatively. Productivity, judgment, health, cognitive abilities, communication and resistance to change are some examples were emotions play an important role.

According to James B. Avey (2008) [22], employee's resistance to change is recognized as one of the biggest threats to organizations, and positive emotions could be used to minimize the negative effects usually associated with change. Similarly, positive and negative emotions like happiness, excitement, sadness, motivation and stress have an impact on productivity [23][22].

Previous studies suggested that people who made decisions influenced by anger were unlikely to stop and consider alternative options before acting, becoming less objective and rational [24]. On the other hand, in a work environment, people that are aware of their feelings can control their thoughts to become more confident, effective, efficient, optimistic and resilient to stress [22]. Moreover, a positive affect leads to quicker decision making, with less information and higher efficiency [25].

Research on positive affect suggest that positive thinking is associated with healthier people, presenting higher sleep quality and low stress levels [26]. Likewise, the broaden-and-build theory suggests that positive emotions are useful to health [27] and Tugade et al. research suggested that positive emotions speed-up recovery from cardiovascular diseases [28]. Regarding emotional contagion, i.e. the spread of emotions from one person to the people around them, it was shown that positive emotions are more contagious than negative ones [29].

To enable a better emotional management, especially in a crisis context, it is important to select the appropriate type of information to be delivered and a good platform of dissemination [30]. Particularly in the doctor-patient context, empathy "may help a patient experience improved psychological well-being", feeling less negative and more positive emotions [31].

2.1.3 EMOTIONS AND COMPUTERS

Emotions have an important role on human society but how can the analysis of emotional expressions by computers be helpful? The literature suggests that emotions should be considered in computer-human interactions and, by analyzing emotions, computer systems could adapt easily and serve more effectively human needs.

Emotions are not necessary only for human creativity and intelligence but also for rational thinking and decision-making. In order to became more natural, the emotions resulting from computer-human interaction should be analyzed and expressed in both sides. Intelligent decision-making systems should consider emotions as well. R. W. Picard publish the first book on affective computing area, describing it as "computing that relates to, arises from, or deliberately influences emotions" and gave an enormous contribution to the use of emotions on human-computer interactions [9]. The work identified four main areas to define the affective computing goals: recognize emotions, express emotions, having emotions and having emotional intelligence. Where the last two are more difficult to accomplished than the first ones [9].

People express emotions in computer-mediated communications similarly to face-to-face communications. Correspondingly, people accept the asynchronous computer-mediated communications nature and fill the conversational gaps with emotions, meaning that humans use rich emotion expressions when using technology to communicate and computers are not an emotional expression wall [2].

2.2 BUSINESS VALUE PROPOSAL

Based on the importance of emotions on the society, in this chapter a business model will be proposed. Firstly, the value proposition, value for the customer and perceived value will be exposed. Then, for the innovation process, Verna Alle and Potter Value Chain and the canvas model diagram will be exposed. And finally, the proof of concept main characteristics, ordered by its importance by using the Analytic Hierarchy Process method (AHP).

2.2.1 VALUE PROPOSITION

The FullExpression project aims to develop emotion detection solutions to improve human-computer interactions and human life in the digital era. It will use advanced AI algorithms to bring the best results possible for individuals or companies through software solutions and integrated services.

2.2.2 VALUE, VALUE FOR THE CUSTOMER AND PERCEIVED VALUE

The purpose of all businesses and their activities is to create value for the customer by exchanging tangible and/or intangible goods or services and be accepted by them or by the business collaboration network. Value, value for customer and perceived value are important concepts to organizations, but at the same time they are difficult to understand and to be expressed in a common language were all stakeholders comprehend [32] [33].

In order to create value, a business needs to understand which drivers influence the value creation, how they are related to their product and services and how they act on the business-customer relationships [33][34].

For FullExpression products and services value, as Table 1 shows, by organizing the drivers by their benefits and sacrifices for services and customer relationship it is possible to understand clearly how the drivers affect the value for the customer.

To create trust and solidarity with customers, the services must be technically competent (have technically competent teams), responsive (acting quickly to solve customers' problems and needs), flexible (quickly meet customer emergencies and expectations), reliable (have the best accuracy emotion detection reports) and have a high performance (quickly create emotion reports). This will result in using, installing and training costs for the companies, and time, effort and perceived costs for the relationship between the business and the customers.

Table 1 – Drivers that influence the creation and perception of value.

	Service	Relationship
Benefit	Technical competence Responsiveness Flexibility Reliability Performance	Trust Solidarity
Sacrifice	Cost of use Installation costs Training costs	Time Effort Perceived costs

The value for the customer could also be divided into four value temporal positions: pre-purchase (tries to predict how customers will perceive the business products and services), at the point of trade (value that the customer perceived at the point of trade), post-purchase (customer value obtained based on customers'/suppliers' choices) and after/use experience (customer value after sales/disposal) [34].

Since FullExpression is a proof of concept and not a business, it is only possible to predict how the customers will perceive value (fulfilling only the first state of value temporal positions) in two dimensions: how this project desires the value offer to be seen and how it will actual be seen. On the long term, we expect that clients see the services as reliable, flexible, having great performance and having responsiveness. However, in a first stage, it is expected that, at least, the customer recognizes the FullExpression services as reliable and having great performance.

2.2.3 INNOVATION PROCESS

Innovation is an opportunity for companies to introduce new advanced technologies to the markets, by incrementing their products or creating new ones, allowing it to be competitive and to bring the best products to the consumers. Likewise, companies that have a formal innovation process implemented and which interact with science have their innovation ability increased [35].

The innovation process may be divided into three different stages: the front of innovation or Fuzzy Front End (FEE), the process of incrementing or the development of a new product and its commercialization. [36].

This work is sustained by the methodologies, tools and techniques provided by Peter Koen [36], New Concept Development model (NCD), present on the FEE innovation stage, helping to increase the innovation process success.

2.2.3.1 NEW CONCEPT DEVELOPMENT MODEL

Peter Koen created the NCD providing a common language for the FEE. This model divides the FEE in three areas:

1. Engine: provides power to the FEE to drive five key activities. The power is the leadership, culture and business strategy of the company;
2. Inner Spoke Areas: consists of five key activities - Opportunity Identification, Opportunity Analysis, Idea Generation and Enrichment, Idea Selection and Concept Definition;
3. Influencing Factors: external environmental factors that influence the engine and activities such as government policy, economy, competitors and customers.

As Figure 4 shows, the model has a circular shape showing that ideas flow and iterate among five key activities. The circular shape is meant to represent how ideas iterate and flow among activities. Also, the big arrows represent the activities that are start/end points to the FEE. Product concept begins on Opportunity Identification or Idea Generation and Enrichment activity and leaves for incrementing or development of new products on Concept Definition activity.

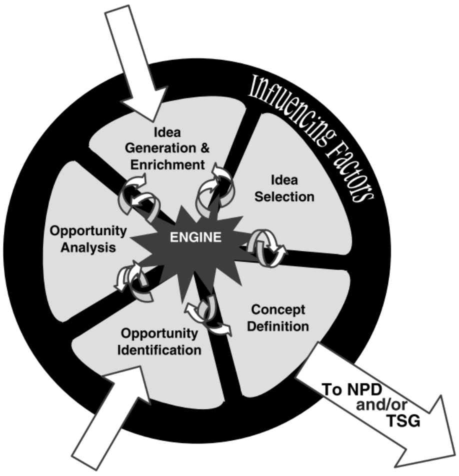


Figure 4 – The new concept development (NCD) created by Peter Koen providing a common language for Fuzzy Front End (FFE) [36]

2.2.3.2 METHODOLOGIES, TECHNOLOGIES AND TOOLS OF NCD

For the influencing factors, engine and five key activities of the NCD, Peter Koen described a set of most effective methods, tools and techniques that are summed up on this section.

2.2.3.2.1 INFLUENCING FACTORS

There are several external factors that could influence positively or negatively the organizations. Creating a strategy or plan to be executed when the company external environment changes seems to mitigate the negative and strengthen the positive effects of the environment transformation, allowing companies to adapt faster and be successful.

2.2.3.2.2 THE ENGINE

The companies should create strategies to promote a culture of innovation and creativity. Outlook Questionnaire and KEYS are instruments that can be used to encourage and measure the climate for innovation on organizations.

2.2.3.2.3 OPPORTUNITY IDENTIFICATION

Typically, Opportunity Identification is driven by the business goals. These opportunities should envision the future trough road mapping, technology trend analysis, customer trend analysis, competitive intelligence analysis, market research and scenario planning.

2.2.3.2.4 OPPORTUNITY ANALYSIS

Opportunity analysis uses the same tools as opportunity identification to confirm in detail the opportunity raised. For a large-scale opportunity, the analysis should include strategic framing, market segment assessment, competitor analysis and customer assessment.

2.2.3.2.5 IDEA GENERATION AND ENRICHMENT

Idea generation and enrichment is responsible for the birth, development and maturation of an idea. It is a complex activity that must be supported by a strong innovation culture that encourages employees to spend time testing and validating their own ideas as well as having the necessary resources to shape them. Customers, market and business needs should be identified, and technological solutions must be addressed. Likewise, a formal process of idea generation helps to create the same measurable goals or metrics for all companies and promoting the inclusion of people with different backgrounds in the same team helps to enrich the idea.

2.2.3.2.6 IDEA SELECTION

Usually, a common problem to companies is to downsize the number of ideas in order to achieve the ones with most business value. Anchored scales (ordinal measures scales) methodologies is a good option to select ideas and must use multiple factors such as technical and commercial success probability, reward, strategic fit and strategic leverage. Likewise, a formal idea selection process should be implemented, and a proper feedback must be given to the idea submitters in order to encourage them to propose new ideas.

2.2.3.2.7 CONCEPT DEFINITION

Concept definition is the final NCD activity and provides the exit to the New Product Development (NPD). Project concept creators must carefully define the project goals and outcomes and define criteria to describe the project in an attractive form (e.g. financials, market growth and market size). Also, they must address the aids and risks of the project and should present alternatives to mitigate the risks.

2.2.3.3 WORK CONTEXT

Transporting the NCD to the context of this work, it is possible to affirm that the engine is Instituto Superior de Engenharia do Porto (ISEP), more specifically the master thesis subject, and it is responsible to drive all the five activities; the student is responsible for exploring all activities in order to create the product concept; the supervisor professor is the innovation leader that helps the iterations of ideas over the five activities, and the thesis jury and scientific community act as the company that confirms the business value of the idea and product concept.

For the Concept Definition activity, the Function Analysis Systems Technique (FAST) was used to identify all required product functions. Fast methodology was introduced by Charles W. Bytheway in 1965 and it provides a systematic tool for developing, decomposing and understanding the products functions, process or services [37]. The main goal is to understand the problem by objectively identifying all required functions. Correspondingly, it helps the stakeholders to share the understanding of the problem, by identifying missed, misunderstood and duplicated functions, and to identify clearly the problem and share understanding between functions relations [38].

By applying the FAST principals, a diagram is produced exposing the product functions. This technique can be used to concept new products or to analyze existent ones.

Figure 5 exposes the diagram shaped using FAST for the emotion detection product concept: reading the diagram from the left to the right side, the question “how this function is achieved?” should be made; reading the diagram from the right to the left side, the question “why this function is needed?” should be made; reading the diagram from bottom to top, the question “when this function should exist?” should be made.

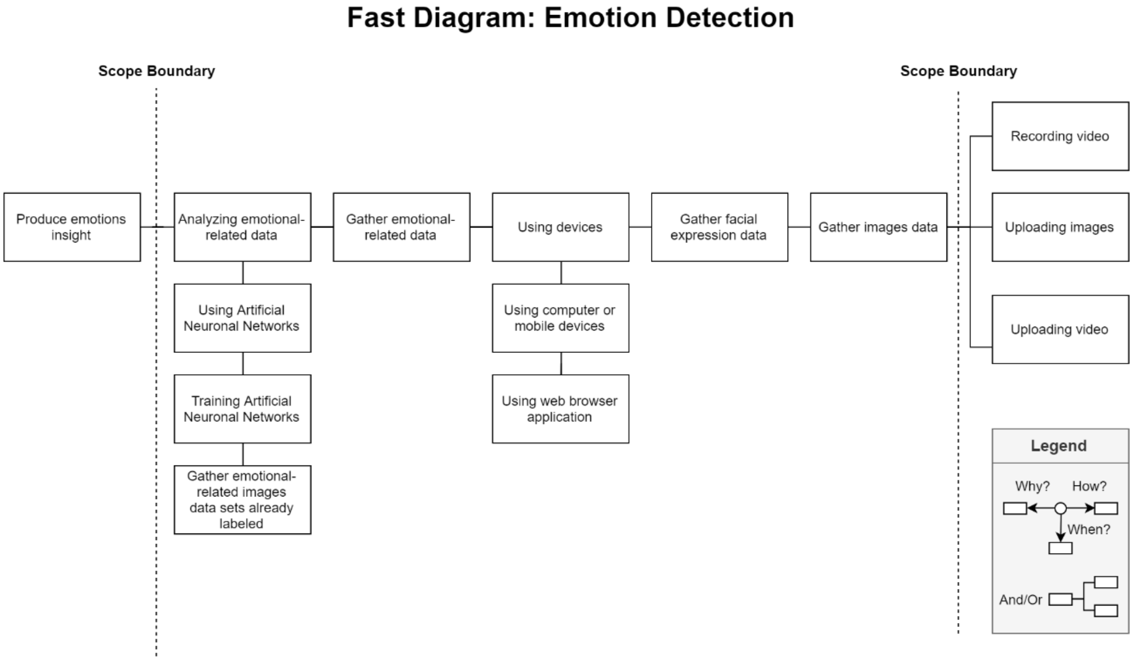


Figure 5 – Function Analysis System Technique (FAST) diagram exposing the emotion detection product concept

2.2.4 CANVAS MODEL

FullExpression proposes a software solution that allows to generate emotional reports, able to be integrated with other systems, working in as many devices as possible and provide research work for the scientific community about new technologies and techniques discovered during product creation. To accomplish that, FullExpression must be presented in the following customers segments:

1. Working places: helping companies to better understand the emotions expressed by their employees and anticipating strategies to improve their productivity;
2. Human profiling: helping individuals or companies creating their emotional intelligence profile;
3. Health care: providing emotional insights to physicians to improve treatments;
4. Research: providing emotional insights about study participants;
5. Media consumers companies: providing emotional state reports of their clients during media consumption.
6. Industry 4.0 changes: to better deal with the changes in workforce tasks, healthcare and researching

It is important to create a trustful customer relationship. The data gathered for the emotion report creation must be confidential and only authorized persons should have access. Similarly, the confidence about software results (if they are reliable or not), the constant improving, the possibility to be used in different devices and the easy integration with other solutions allows the enrichment of the customer relationship. And, to bring the FullExpression products and services to the customers, partner integrations and an e-commerce platform should be used.

For marketing, software development, integration, security and research, the main FullExpression activities would require high performance computers, and software developers and other human resources. Also, to have the best human resources and software test environments, partners such as universities, research labs, hospitals and partners would allow the constant feeding of the teams and solutions with new data, helping the improvement of software and services consumption experience. These entities would be partners and clients at the same time.

A monthly fee to use FullExpression products/services and private/public funds for research would be required to support the company’s costs (online services, human resources, high-performance computers, marketing, software research and development) and to make it profitable.

The canvas model is a great example on how the business characteristics could be organized in a simple board. Figure 6 shows the canvas model for the FullExpression business model, which divides the business characteristics into key partners, key activities, key resources, key propositions, customer relationships, channels, customer segments, cost structure and revenue streams.

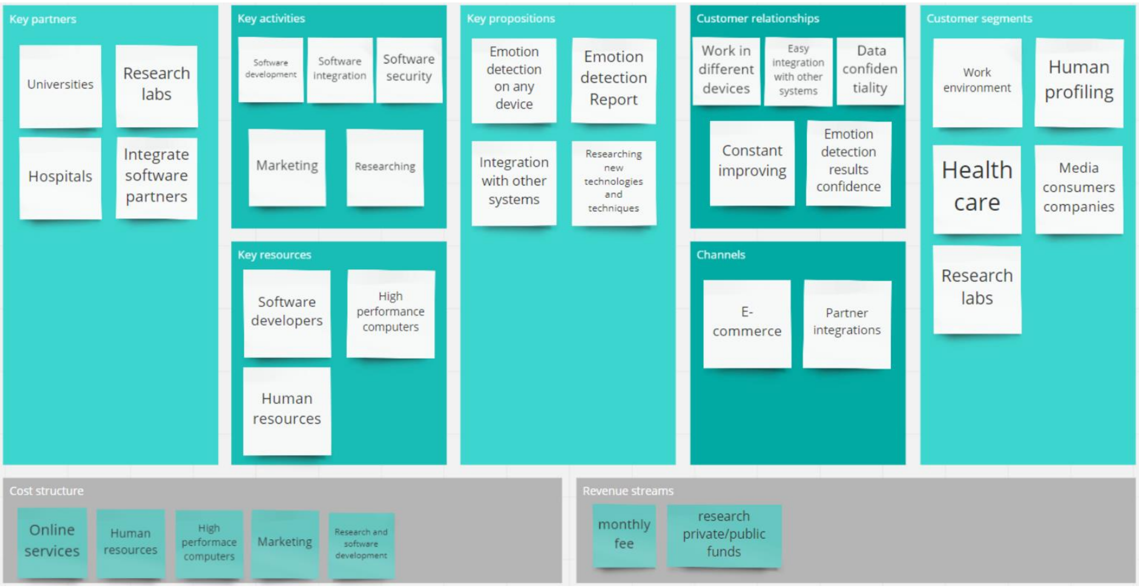


Figure 6 – Business canvas model

2.2.5 VERNA ALLE AND POTTER VALUE CHAIN

Every business has their own value chain representing how the organizations operate and manage their departments or units in order to create value for the customers. Porter suggested a general value chain which companies could use to analyze their activities and see how they are related, understanding where the value sources are. The value chain proposed works in two dimensions: primary or operation activities and support or managing activities[39].

People and companies tend to build networks in order to meet individual and group needs or desires. V. Allee proposed a value network analysis model for business that allows the analysis, evaluation and improvement of the network to “convert both tangible and intangible assets into other forms of negotiable value”. It is especially important to the understanding of value creation from intangibles, a reality much present on business today. When a scientist produces a paper or a professor teaches students, the knowledge (intangible) is converted into negotiable forms of value that can be money (students pay to professors to teach) or reputation/influence (work citation) [40].

Porter general value chain helps organizations to look inside and understand how they work in order to create value for the customers. The value network analysis model helps organizations to look outside and understand how they exchange and delivery value to the customers and the community.

The FullExpression aims to build a network of partners that allows to exchange knowledge, implement proof of concept technologies and sell mature services and products. Building a network analysis model and analyzing the organization value chain will help to understand how the network will work, the rules behind it, the participants and how the organization operates in order to deliver value. Since this is a prove of concept work and not a business, the network analysis will not be made at this stage.

2.2.6 FULLEXPRESSION SOLUTION MAIN CHARACTERISTICS USING ANALYTIC HIERARCHY PROCESS

AHP is a multi-criteria decision mathematical method that helps decision-makers to prioritize criteria according to their calculated weight. It was created by Professor Thomas L. Saaty in 1980 and since then it has been helping businesses making rational decisions [41][42].

To construct the FullExpression proof of concept prototype there are five characteristics that should be considered: reliability, performance, flexibility, technological base and extensibility. A priority list should be made, ordered by the characteristic importance, to favor the comparison of characteristics in order to simplify the decision-making process. To do so, a classic AHP mathematical method was used.

First, as Figure 7 shows, a decision-making tree was created, decomposing the problem in two parts: goal and criteria (characteristics).

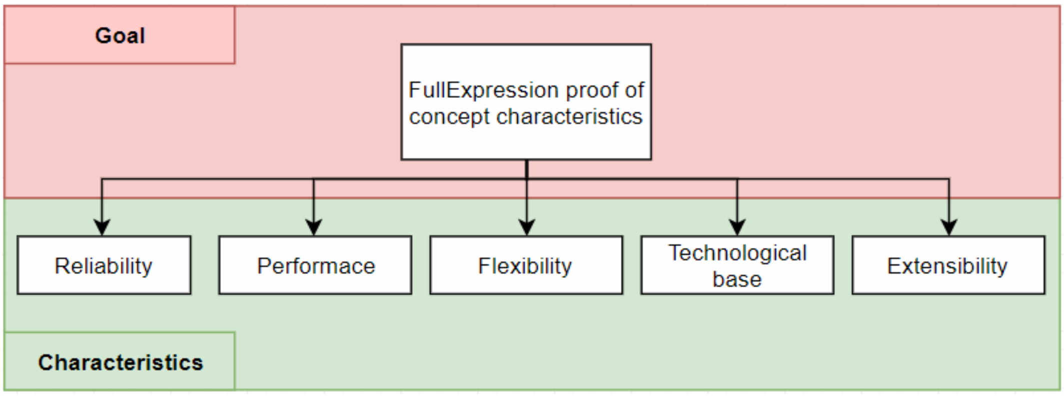


Figure 7 – AHP decision-making tree

Second, a criteria comparison matrix [C] was created in order to compare the importance of each criteria based on the fundamental scale of absolute numbers created by T. L. Saaty (2008) [42]. Table 2 shows all values for the comparison matrix created based on the decision-making tree characteristics.

Third, as Table 3 shows, a normalized matrix was created, defined by the formula (1), where each column element must be summed and again divided for the resultant summed value.

$$A' = [a'_{ij}] = \frac{a_{ij}}{\sum_{k=1}^{n} a_{ik}} \; for \; 1 \leq i \; \leq n \; and \; 1 \leq j \; \leq n \qquad (1)$$

Table 2 – AHP criteria comparation matrix

	Reliability	Performance	Flexibility	Technological base	Extensibility
Reliability	1	3	7	9	7
Performance	1/3	1	5	9	7
Flexibility	1/7	1/5	1	9	1/5
Technological base	1/9	1/9	1/9	1	1/5
Extensibility	1/7	1/7	5	5	1

Table 3 – AHP normalized criteria comparation matrix

	Reliability	Performance	Flexibility	Technological base	Extensibility
Reliability	0.58	0.67	0.39	0.27	0.45
Performance	0.19	0.22	0.28	0.27	0.45
Flexibility	0.08	0.04	0.06	0.27	0.01
Technological base	0.06	0.02	0.01	0.03	0.01
Extensibility	0.08	0.03	0.28	0.15	0.06

Finally, as the Table 4 shows, the criteria weights were created, defined by the formula (2), were nt is the number of terminal nodes for each alternative d, nl the number of tree levels, t the "leaf" node related to d and $t, nl - 1, nl - 2, \ldots$ the tree path from d to the tree root.

$$C = [c_d], for\ 1 \leq d \leq n\ where\ c_d = \sum_{t=1}^{nt} W_t \times \prod_{l=1}^{nl-1} W_1 \qquad (2)$$

Table 4 – AHP criteria weights

	Reliability	Performance	Extensibility	Flexibility	Technological base
Criteria Weights	0.47	0.28	0.12	0.09	0.03

By evaluating Figure 8 columns bar graphic it is possible to infer that reliability is the most important characteristic, followed by performance, extensibility, flexibility and technological base.

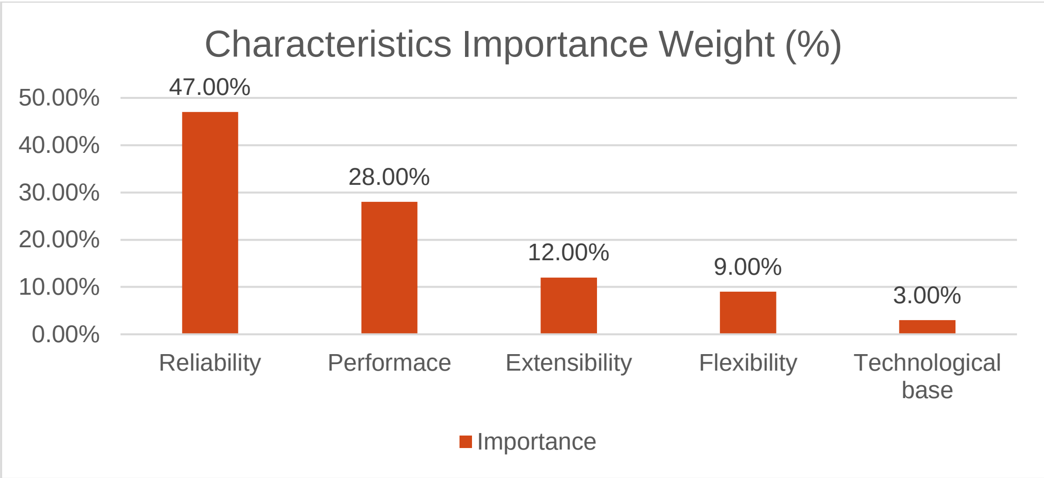


Figure 8 – FullExpression proof of concept characteristics importance weight

2.3 CONCLUSIONS

Emotions seem to be the result of physiological (e.g faster breathing), behavior (e.g body movements) and expressive responses (facial expressions). Because of the huge diversity of emotions, Paul Ekman created the FACS which measures and evaluates facial muscles, eye and head movements and proposed seven universal emotions (also known as core emotions): happiness, sadness, surprise, fear, anger, disgust and contempt. Several studies suggest that emotions are widely present in society and can affect it positively or negatively. In addition, emotions in human-computer interactions are relevant and should be considered during machine designs. Contrary to the common sense, technology is not an emotional expression wall and studies suggested that people express emotions with same frequency as face-to-face.

In a business perspective, the FullExpression project aims to develop emotion solutions to improve human-computer interactions and human life in the digital era, and it should target the working place, human profiling, heath care, research, media consumers companies and the industry 4.0 customer segments. The solutions should be reliable and flexible, have good performance and be easy to be integrated with other systems. Partners such as Universities, Research labs and Hospitals will be important to collect resources and data to improve the FullExpression products and services quality.

The next chapter describes how AI could help in solving emotion classification problems.

3. STATE OF THE ART

This chapter will present the state of the art of AI field. It will explain what AI is, the AI evolution, problem solving capabilities, sub-fields and key concerns of the DL subfield. Since this work aims at using DL models to classify facial expression emotions, most of techniques exposed are from or can be applied to the DL field. Moreover, a broad explanation will be provided for AI and Machine Learning (ML).

3.1 ARTIFICIAL INTELLIGENCE

Stuart Russel and Petter Norvig defined AI as “the designing and building of intelligent agents that receive percepts from the environment and take actions that affect that environment” [43]. AI can also be described as a computer science field that tries to reproduce human cognitive capabilities to solve complex engineering problems when other computer science fields could not [44]. Thanks to complex problem solving capabilities, companies are investing more and more resources into AI [45]. Daily, subset fields like ML are getting more attention and being applied in different domains such as large scale ML systems (systems that improve their performance with experience), DL (subset field of ML that uses ANNs to solve complex problems), robotics (making possible robot navigation in dynamic environments), computer vision (concerned with how the computer visually perceives the world), natural language processing (systems able of perceiving and understanding spoken human language), recommendation systems (make suggestions based on human behavior), algorithmic game theory and multi-agent (fields that considers multiple agents from different perspectives such as social or economic) and internet of things and biometry (helping machine became more humanized) [46].

3.1.1 EVOLUTION

Humans became one of the most successful species in Nature, mostly caused by our big and complex brain, associated with vast learning and cognitive capabilities [1]. As human species, we always asked how our brain works and how it manages solving complex problems in a small amount of time. Nevertheless, we never had the resources needed to truly understand what is behind this organ until the start of the XX century, when progresses in computation helped us achieve huge advancements in the neuroscience research field [47], which has created the foundations of AI.

The first reference to AI was raised by John McCarthy in 1956 when he gave the first conference about this subject. However, the first work now generally recognized was made by Warren McCulloch and Walter Pitts in 1943 in a paper called "A logical calculus of the ideas immanent in nervous activity". The authors started a set of neuronal network theories, which resulted in the proposal of a new model where each artificial neuron as an "on" or "off" state that is activated by its neighbor. They showed that this model could be applied in the computation world and suggested that machines could learn similarly to humans. Base on this, Donald Hebb (1949) proposed a new model: if the neuron A constantly and persistently stimulates the neuron B, the connection is reinforced. Hebbian Learning is how this model is known nowadays [43].

In 1950, in an Alan Turin paper, the idea of simulating the human behavior and intelligence on machines has emerged. He proposed the discussion around the question "Can machines think?" and introduced "The Imitation Game" (also known as Turing Test) as a possible answer to this problem. This question is difficult to answer because "thinking" and "machine" are concepts hard to define in an universal way. Then, other questions were proposed based on "The imitation game" to introduce new ways of understanding and solving the problem. The game suggested creating an environment where a human, a machine and a judge are in the same place. The judge interacts with the human and the machine, trying to understand who is who. If the judge could not distinguish the human from the machine, the machine wins. We can conclude that Alan Turin replaced questions like "What is thinking?", "What is intelligence?", "What is conscience?", subsets of the question "Can machines think?", for the question "Can machines imitate human behavior?" in a way that allows to separate the philosophic and engineering part of the problem [48][49]. It was an important work for the AI community allowing theoretical concepts to be applied on the real world. "The imitation game" took 65 years to be overtaken by a program called Eugene Goostam, which simulated an Ukrainian 13 years old teenager that was mistaken for a human more than 30% of the time, during a series of five-minute keyboard conversations [50].

Still, in 1950, Marvin Minsky and Dean Edmonds built the first Artificial Neuronal Network (ANN) in a computer called SNARC, which used 3000 valves of an automatic pilot system to simulate a neuronal network with 40 neurons [43].

John McCarthy convinced Minsky, Claude Shannon and Nathaniel Rochester in 1956 to organize a two months' workshop, where researches from all around the USA could work deeper on the AI field and describe well the problems in a way they could be solved by machines. This workshop was an important step for creating the AI field and to associate it to computer science [43].

In 1958, psychologist Frank Rosenblatt, inspired by the work of Donald Hebb in1949, created the concept *perceptron* and explained it in a mathematical model which describes the way that a neuronal network works. This model, illustrated in Figure 9, takes a set of binary inputs from its neighbor neurons, multiply each input by a weight (representing the synapse weight between neurons), which is different for each input, sums the result of all multiplications and returns 1, if the sum is higher than a given threshold (known as activation function), or 0 if not [51]. We can assume that this model is an evolution of the model created by Warren McCulloch and Walter Pitts, with the introduction of weights between synapses bringing a new learning mechanism. This mechanism has the following steps: perceptron creation with a set of weights between synapses and a random training data set; an input data set insertion and the output verification; depending on the expected output, the weights must be updated (if the expected output is 0, but the received one is 1, the weight of all responsible synapses for this output should be decreased; if the expected output is 1, but the received one is 0, the weight of all responsible synapses for this output should be increased). Frank Rosenblatt also implemented this model in a computer created exclusively for this work and proved that it is possible to use it to classify forms by using input images with 20x20 pixels [52].

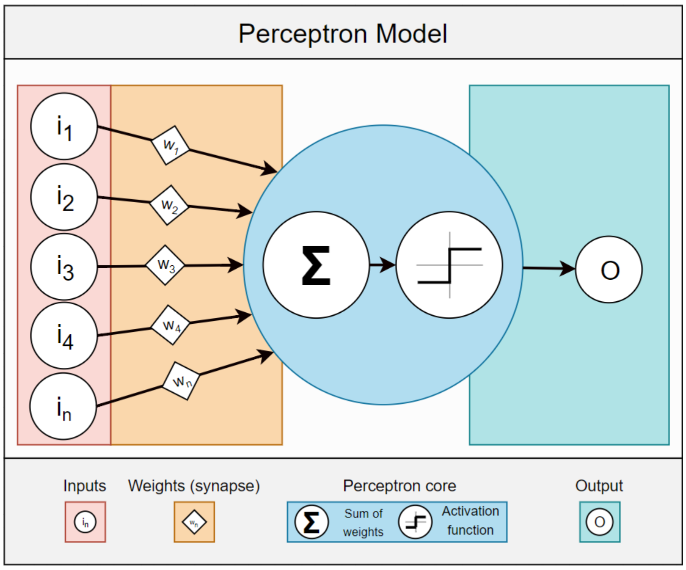


Figure 9 – Visual representation of the perceptron [51]

Thanks to Donald Hebb and his mathematical learning model, the ML area gives its first steps but is only recognized in 1959 when Arthur Samuel defined ML as the area that gives to computers the capability to learn without being explicitly programmed [53].

Despite the term DL was only introduced in 1986 by Rina Deschter [54], a previous work by Alexey Ivakhnenko and Lapa [54] introduced algorithms like supervised learning, feedforward and multilayer perceptrons, which are nowadays widely used in the DL field.

The AI research field faced a setback in 1969, after the work by Minsky and Papert (1969) [55] exposed the limitations of the Rosenblatt model. More specifically, they reported that perceptron was too basic and simplistic to be applied to more general problems [52], even if organized in several layers, instead of just one.

David E. Rumelhart et. al. [56] brought AI to the top again, when they proposed the concept of back-propagation, being the neuronal optimization strategy most popular nowadays. This later is based on the Gradient Descent mathematic algorithm to find the minimum value of a given function. The concept proposes the steps: through the process of front-propagation, the input data is inserted into the input layer and crosses all hidden layers of a neuronal network, being the result showed on the output layer (changing the synapses weight); with this result, the error is identified; then, the process of back-propagation starts where the synapses weight are changed based on the error previously identified. The adjustment flow of the weight follows the opposite direction of the front-propagation, starting on the output layer and finishing on the input layer [57][43].

The cognitron that was proposed in 1975, evolved to neocognitron in 1980 and was described in detail in 1988 by Kunihiko Fukushima as a mechanism of visual pattern recognition [58]. The neocognitron is a hierarchical multilayered neural network that was used to solve the handwritten character recognition problem as a prove of concept. Fukushima suggested that this model works in two different levels at the same time: learning with and without a teacher. Furthermore, he proved that the acquired ability to recognize patterns was not affected by their positions nor by small distortions of their shapes [59][58].

In 1991, Lorien Pratt came up with the idea of transferring the information from a trained ANN to another and in 1993 she created the concept of *discriminability-based transfer* algorithm, establishing the foundations for the Transfer-Learning technique [60]. These works were very important to the Machine and DL field, stimulating the usage of large ANN capable of solving large classification problems to be re-train and applied on more specific problems, with the advantage of taking less time and resources to do so [61][60].

The evolution of neocognitron to solve image recognition problems was proposed by Yann Lecun, Léon Bottou, Yoshua Bengio and Patrick Haffner, with the first Convolutional Neuronal Network (CNN), created in 1998, to solve the handwritten numbers problems using the MNIST dataset [62]. CNNs were inspired by the brain, and D.H Hubel and T.N Wiesel suggested in their research, on the 1950s and 1960s, a new model about how mammals perceive visually the world, and showed that cats and monkeys visual cortexes included neurons that exclusively respond to neurons in their direct environment [63][64]. This CNN nowadays is known as LeNet-5 and was applied by several banks to recognize handwritten numbers on checks in a 32x32 pixels grayscale image format.

In 2006, Fei-Fei Li suggested the idea of mapping out the entire world of objects in order to have data that represented the real world. The resulting dataset was called ImageNet and was originally published in 2009 [65]. This project grew annually from 2010 until 2017, and allowed to challenge the community to create better, faster and more accurate CNN architectures. In just 7 years, the accuracy in classifying objects increased from 71.8% to 97.3% [10].

3.1.2 FIELDS AND TECHNIQUES

Some techniques used in AI are applied in classification problems. AI problem solving approaches are driven by symbolic representations or data. Therefore, there are two big AI subfields called Symbolic Learning and ML, driven by symbolic representations or data, respectively. The content of Symbolic Learning field is represented in a form that is human readable, on the other hand, ML uses algorithms and statistical models to solve problems based on data. Nevertheless, in some cases, ML uses symbolic representations to extract knowledge from data [66].

Computer Vision (extraction and processing of information from images and videos in order to infer conclusions and solve problems) and Robotics (allowing robot navigation in a dynamic environment) are examples of fields in which symbolic representation is used.

ML uses pattern recognition algorithms to find patterns on data, providing knowledge to users and systems. The knowledge can be extracted using statistical algorithms called Statistical Learning, or using neural networks, that imitate how the human brain works. Statistical Learning typically is applied to speech recognition and natural language processing problems.

ML uses three techniques to train algorithms:

1. Supervised Learning: uses labeled data as an input, while already having the answer the algorithms should accomplish;
2. Unsupervised Learning: uses algorithms to find patterns for a given data;
3. Reinforcing Learning uses goals to orient the algorithms to learn and solve complex problems.

ANN became much more complex and a new field called DL was created. These complex and well-organized ANNs are helping to solve complex problems that were not possible to solve before. There are more than 25 types of ANN structures. Two structures that are popular nowadays are CNN and Recurrent Neuronal networks (RNN). Typically, CNN solves classification problems and RNN temporal or prediction problems [67].

The AI subfields hierarchy, the main problem-solving categories (classification and prediction) and ML techniques (supervised, unsupervised and reinforcing learning) are represented in Figure 10.

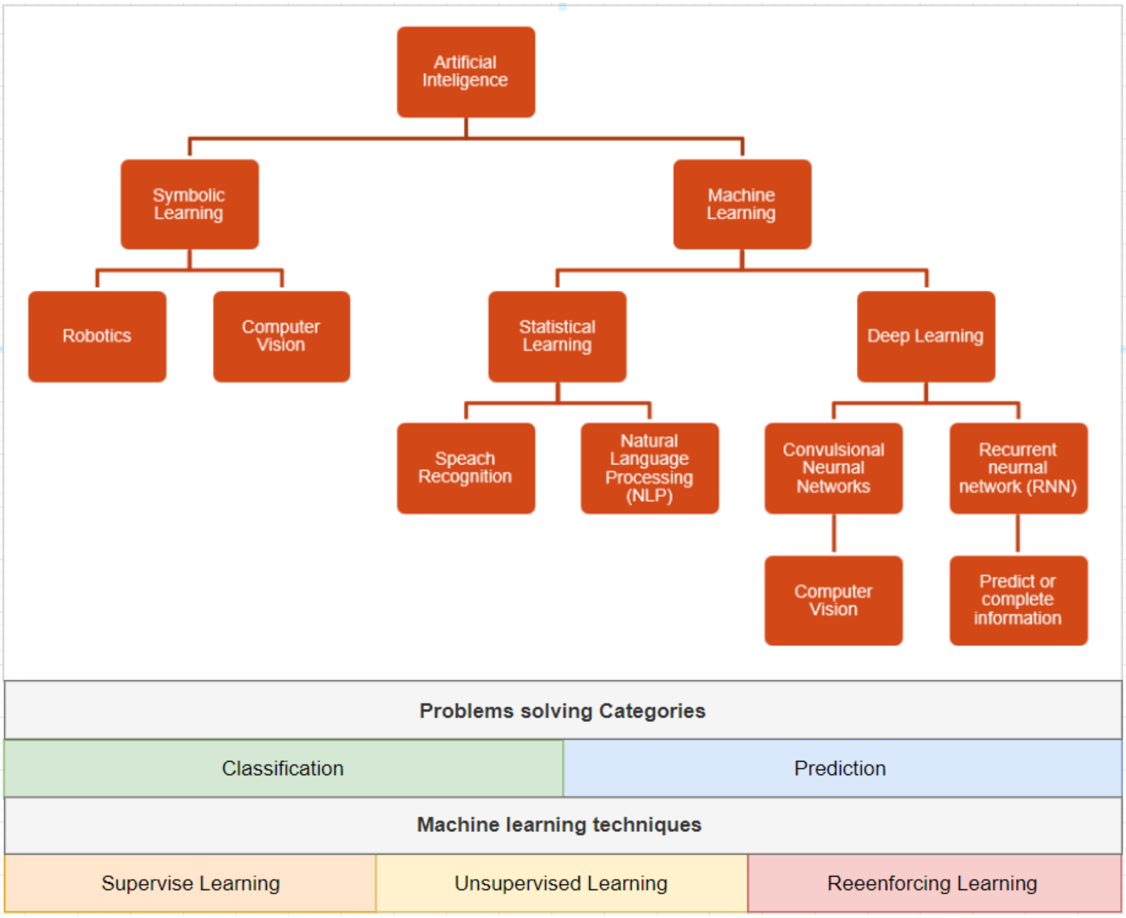


Figure 10 – AI fields

3.2 DEEP LEARNING FIELD

DL is a subfield of ML, that uses deep ANNs in order to solve complex problems. During the years, DL showed tremendous improvements on accuracy, robustness, cross-technology and language usage over traditional approaches, mainly because of the large amount of data available and computation power [68]. For computer vision, DL offers a specialized type of ANN (CNN) to solve image and video pattern recognition.

In the following sections the key factors to succeed in DL will be presented, and the main available models will be compared.

3.2.1 ARTIFICIAL NEURONAL NETWORK

Haykin (1998) describes ANN as “a massively parallel distributed processor made up of simple processing units that has a natural propensity for storing experiential knowledge and making it available for use” [69]. In other words, is a software structure capable of finding patterns in data to provide answers for a given domain problem, without the need for explicit programming of all rules. ANNs are inspired by the human brain, more specifically by how neurons communicate with each other and how they are organized to form a network [70].

An ANN is composed by artificial neurons (also known as Perceptrons), connection and weights (to connect each neuron with the next ones and to provide different levels of importance for each connection) and a propagation function (function responsible to transport values through the network). In addition, they are organized by several layers, in which the first layer is called input layer (first neurons that will receive data), zero or more hidden layers (intermediate layers which will store the learning patterns from data) and the output layer (final layer which will produce the answers), as can be seen on Figure 11 [71].

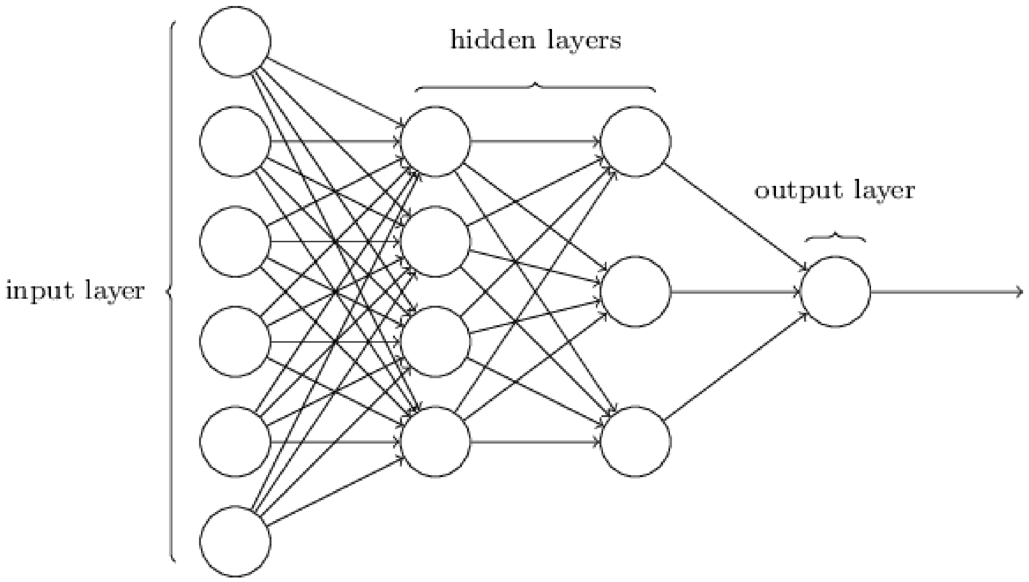


Figure 11 – Example of ANN [72]

If the neurons from one layer are connected with all neurons from the next layers, it is called fully connected layer; if specific groups of neurons only connect with some groups of neurons from the next layer this is called pooling or convolution layer. The purpose of pooling layers is to reduce the data size in order to decrease the general computation power needed to run complex ANNs; the convolution layer is responsible for finding patterns on data [73].

During DL evolution, dozens of ANN architectures were created and used alone or in mixed forms in order to solve different AI problems. In Attachment VII is exposed a chart of neuronal networks called “The neural network zoo” created by Fjodor van Veen in 2016 [67]. The Deep Convolution Network (also known as CNN) exposed is an ANN architecture specialized on image classification problems [74]. The hidden layers are composed by groups of convolutional and pooling layers as can been seen in detail on Figure 12.

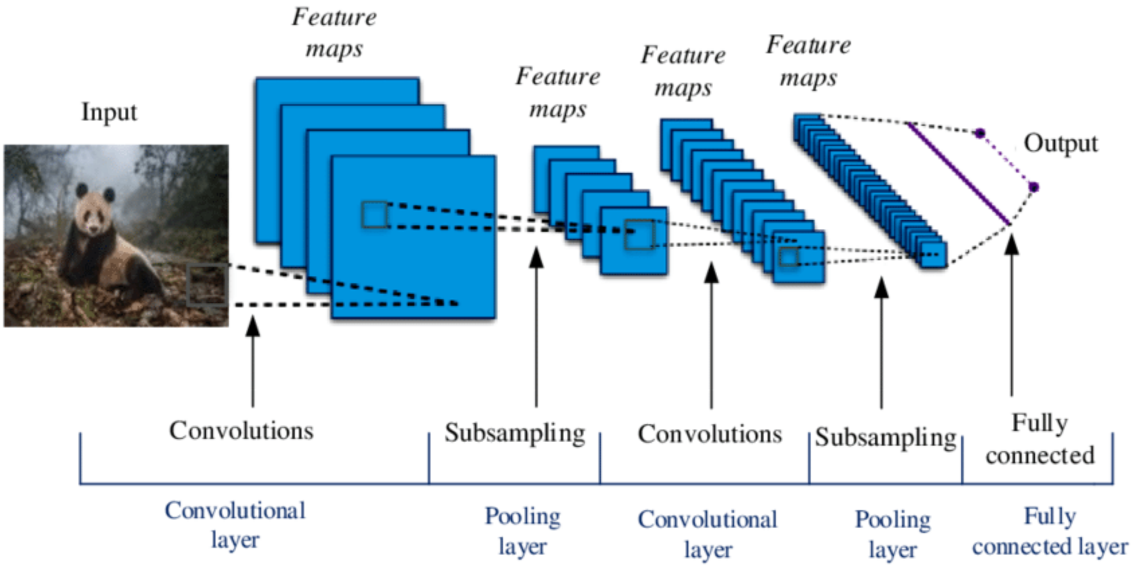


Figure 12 – CNN architecture [75]

Different CNN architectures, such as AlexNet [74], GoogleLeNet (known as Inception) [76], VGG [77] and ResNet [78] were build because of the need for CNN’s to became more deep and complex, in the search for better accuracy results. They present impressive accuracy results but with the cost of being computational heavy and not suitable to work in mobile devices.

Then, CNN focused on mobile optimization, and DL models appeared such as MobileNet [79], SqueezeNext [80], ShuffleNet [81], CondenseNet [82], NASNet-A [83], AmoebaNet [84], DARTS [85] and ANTNet [86], which reduce significantly the CNN size without compromising a lot the accuracy values. The size reduction is accomplished by removing neurons and layers with less impact on the final classification results.

Most of these models were validate using the ImageNet database [65], which contains more than 14 million images, divided in more than 20.000 categories. Because of that, they are good candidates to be used in the transfer learning process [87], since they are good at classifying thousands of different categories.

Table 5 and Table 6 expose a comparison between the number of parameters, top-1 and top-5 accuracy for tests made against the Imagenet database. The number of parameters influences the size of the models, the Top 1 accuracy is the percentage of times that a CNN predicts the correct label classification and Top 5 accuracy is the percentage of times that a CNN predicts the correct label classification in a range of five possible choices.

Table 5 – CNN model comparation

Model	Parameters	Top-1 Accuracy (%)	Top-5 Accuracy (%)
AlexNet [74]	~60 million	62.5	83
GoogleLeNet (Inception – V4) [88]	~4 million	81.3	95.8
VGG [77]	~143 million	76.3	93.2
NASNet-A [83]	~88.9 million	82.7	96.2

Table 6 – CNN models optimized for mobile devices usage comparation

Model	Parameters	Top-1 Accuracy (%)	Top-5 Accuracy (%)
Mobilenet (V1) [79]	~4.2 million	70.9	89.9
MobileNet (V2) [89]	~6.9 million	74.7	92.5
SqueezeNext [80]	~3.2 million	67.5	88.2
ShuffleNet [81]	~5.4 million	73.7	-----
CondenseNet [82]	~4.8 million	73.8	91.7
NASNet-A [83]	~5.3 million	74.0	91.3
AmoebaNet [84]	~5.1 million	74.5	92.0
DARTS [85]	~4.9 million	73.1	91
ANTNet [86]	~6.8 million	75.0	92.3

Figure 13 shows the comparison between 23 CNN pre-trained models, on TensorFlow technology, based on Top 1 accuracy (percentage of times that a CNN predicts the correct label classification) and Top 5 accuracy (percentage of times that a CNN predicts the correct label classification in a range of five possible choices) [90].

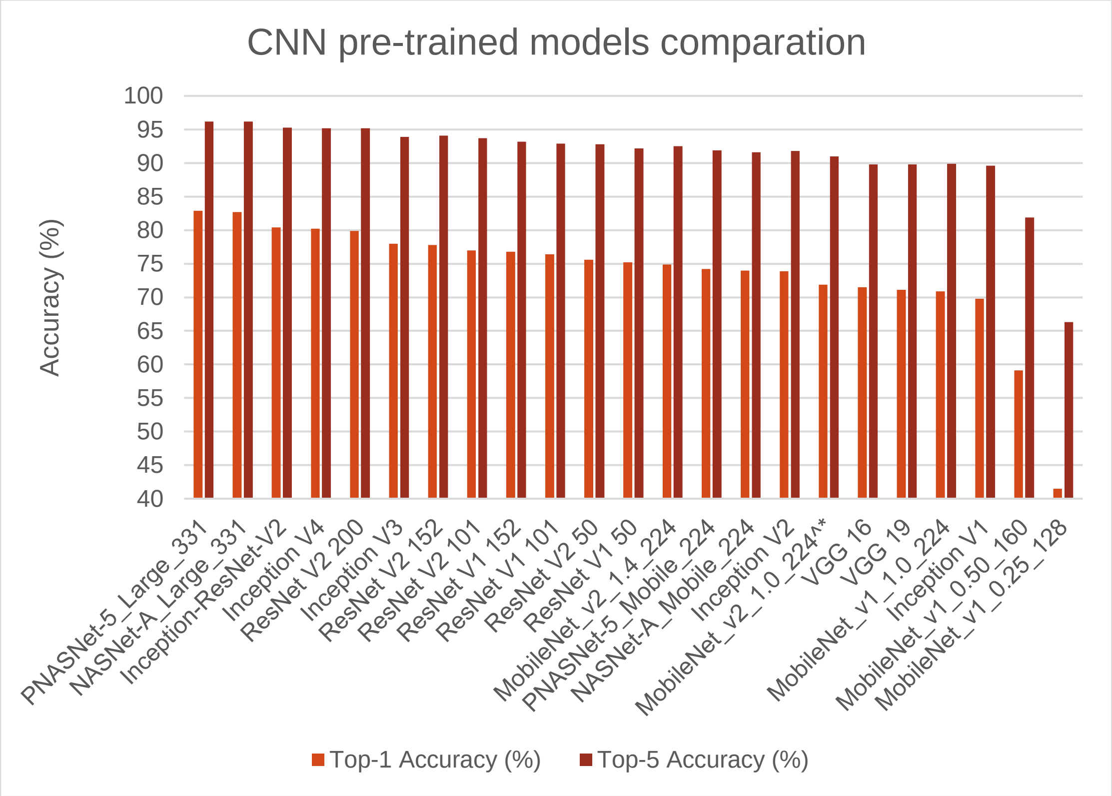


Figure 13 – The best 23 CNN's accuracy values for Top-1 and Top-5 scores

3.2.2 DEVELOPMENT PROCESS

Essentially, the DL development process can be resumed in four main phases: data gathering, exploration and normalization; training; testing and inference (also known as prediction or classification phase). Each one of them will help fine-tune models in order to achieve the best results possible.

Data gathering, exploration and normalization has an important role in the ML and DL field and influences directly how well models solve real world problems. It comprises the raw material that is used to train and validate the ANN factory and it should be carefully gathered and labeled. Also, the amount and diversification of data seems to lead to better accuracy results [91], being the task which data scientists spend most of their time on [92]. In addition, the data collected should be divided into two parts, training and validation data, which will be used to train and validate the model, respectively. This allows the usage of real-world data in the validation processes which was never used to train the model.

In the training phase, the models use the data previous collected in order to find patterns. For classifying problems, the Supervised Learning technique is applied, and data should be labeled and divided by categories. Then, models can correlate the characteristics of the data with the respective classification.

After training the model, tests should be made to understand how well the model is performing. If the model is not performing as expected, changes to dataset or to the parameters used to train the model should be made until the model behaves as expected.

If all previous phases are completed, the model is ready to be consumed, also known as inference phase.

3.2.3 MODEL EVALUATION

It is important to choose the correct metrics and techniques in order to evaluate properly the software and minimize the variance and bias of the results. To do so, this chapter will present some metrics and techniques that can be used in the evaluation of the engineering solution presented here.

Based on the confusion matrix [93][94], metrics like accuracy, precision and recall can be calculated with the CNN results. Also, techniques like cross validation could be used in order to reduce bias and variance of the results obtained. This section will explain the importance of this metrics and techniques and how they can be calculated and used.

Statistical analysis was performed using GraphPad Prism Version 6.01 (USA).

3.2.3.1 CONFUSION MATRIX

A confusion matrix (or error matrix) is a $n \times n$ matrix, where n is the number of classes which the model predicts, often used in the AI field to describe the performance of classification models [93][94]. The matrix enables the comparison between the number of correct and incorrect predictions by exposing the predicted and observed results side-by-side. Correspondingly, the comparison can be divided in four groups:

1. True Positives (TP): the number of correct positive predictions;
2. True Negatives (TN): the number of correct negative predictions;
3. False Positives (FP): the number of incorrect positive predictions;
4. False Negatives (FN): the number of incorrect negative predictions.

Taking as an example a model which classifies data between “Has Emotion” or “Does Not Have Emotion”, it is possible to create a confusion matrix 2×2 where each column/row combination represents one group (Table 7) and infer the accuracy of the model by applying formula (3):

$$Accuracy = \frac{Number\ of\ correct\ predictions}{Number\ of\ total\ predictions} = \frac{TP + TN}{TP + TN + FP + FN} \tag{3}$$

Table 7 - Example of a confusion matrix

<table>
<tr><td colspan="3" rowspan="2"></td><td colspan="2">Observed Result</td></tr>
<tr><td>Positive
Has emotion</td><td>Negative
Does Not Have Emotion</td></tr>
<tr><td rowspan="2">Predicted Result</td><td>Positive</td><td>Has Emotion</td><td>TP</td><td>FP</td></tr>
<tr><td>Negative</td><td>Does Not Have Emotion</td><td>FN</td><td>TN</td></tr>
</table>

Several reasons, such as unbalanced data, makes accuracy a bad metric to be used alone, especially when generalization is required (known as accuracy paradox [95][96]). But combining accuracy with precision and recall, improves the reliability of the testing results by seeing them in different perspectives [96][97]. Precision (formula 4) is the proposition of correct positive classifications (TP) from predicted results that are positive ($TP + FP$). Recall (formula 5) is the proposition of correct positive classifications (TP) from observed results that are positive ($TP + FN$).

$$Precision = \frac{TP}{TP + FP} \quad (4)$$

$$Recall = \frac{TP}{TP + FN} \quad (5)$$

In order to reduce the number of metrics used, the F_{score} (formula 6) joins in a single metric both precision and recall, defined as the harmonic mean of Precision and Recall. The closest the F_{score} values are to 1, the better is the model performance.

$$F_{score} = \frac{2}{\frac{1}{Recall} + \frac{1}{Precision}} \equiv F_{score} = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (6)$$

3.2.3.2 ROC AND AUC

In ML, Receiver Operating Characteristic (ROC) and Area Under the Curve (AUC) are metrics that are well known and widely applied to evaluate classification model results. When combined, they can form a graph with a curve (ROC) and area (AUC), with the y-axis representing True Positive Rate (TPR or Recall; formula 7) values and the x-axis False Positive Rate (FPR; formula 8) values. This type of graphs (e.g. Figure 14) helps the visualization of the model performance. Similarly to F_{score}, the higher the AUC value the better is the model performance [98][99].

$$TPR = Recall = \frac{TP}{TP + FN} \quad (7)$$

$$FPR = \frac{FP}{TN + FP} \quad (8)$$

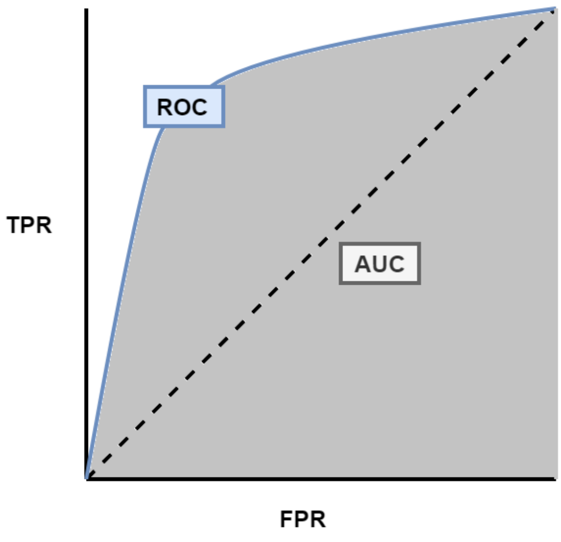


Figure 14 – ROC-AUC Graph

3.2.3.3 CROSS VALIDATION

Cross validation techniques are used to improve the generalization of accuracy results (how well the model predicts or classifies data unseen before) by splitting the available data into training and testing data. The first will be used to train the model and the second to test the generalization accuracy results.

The simplest type of cross validation is called “Hold out” and simply divides the dataset into training and testing datasets randomly. Due to its simplicity it requires less computational resources, but its evaluation can have a high variance.

In order to reduce the high variance associated with Hold out, the k-fold cross validation technique solves this variance problem by splitting the data in *k* equal parts. Then *k* training-test cycles (rounds) are performed, where each cycle uses different training-test data and an average of all test accuracy results is made. This technique is represented on Figure 15 as a 10-round example. On each round, the yellow rectangle represents the test (validation) dataset, where the accuracy result is inferred, and the blue rectangles the training datasets. Then, a final accuracy value will be accomplished by averaging all test accuracy results.

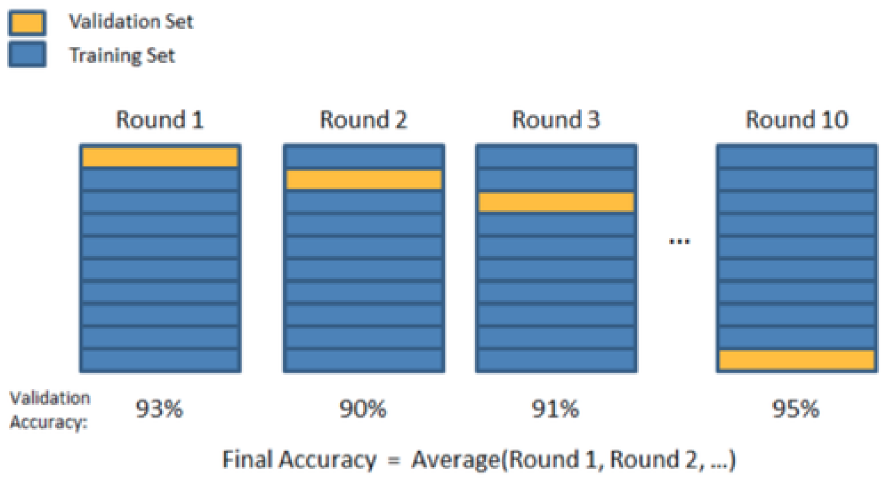


Figure 15 – K-fold cross validation technique (adapted from [100]**)**

The k-fold cross validation technique leads to less biased accuracy results, but also takes *k* times more resources to train the model. This issue can be particularly problematic when large datasets or complex models are used.

Likewise, the Leave-one-out technique splits the data into *k* equal parts, where *k* rounds are performed, but *k* is equal to the number of dataset points. Taking as an example an image dataset that contains 20 images, the Leave-one-out technique will execute 20 rounds, where in each round the dataset will be divided in 19 training images and 1 testing image, allowing that each dataset image can be used both in the training and testing phases. This technique is a variance of the k-fold taken to the extreme, were the datasets are divided into as many parts as possible, leading to lowest bias accuracy results, but at the cost of high computational resources.

The k-fold cross validation technique was used in all training processes performed on this work, where metrics in real time were calculated in order to help fine-tuning the model and achieve less biased accuracy results without compromising too much the training times.

3.2.4 KEY CONCERNS TO SUCCEED ON THE DEEP LEARNING FIELD

In the pipeline of creating, training and using ANN, there are a set of characteristics that should be studied to maximize the chances of success. Key DL success factors like data, loss functions and optimizers, transfer-Learning and pruning or dropout techniques will be addressed in this section.

3.2.4.1 ABSENT DATA

When big amounts of data are absent, it is possible to create more by using data augmentation. This involves using the existing data set and changing small features in order to create more data without changing the labeling. Taking as example an image data set, changing the size, color, brightness, contrast, blur or even rotating or flipping the images could lead to an extra amount of images without changing the labeling [101].

3.2.4.2 IMBALANCED DATA

Most of the real-world data is imbalanced. This means that it is more likely to have increased amounts of a certain type of data than another. Taking people faces dataset as an example, it is more likely to have more adults than children. There are some tactics that can be used to diminish the impact of this problem:

1. Class weight: using class weight according to the representation of the data type will help to balance the underrepresented types. This means that the higher the data type representation is, the lowest is the impact on the ANN training phase.
2. Over sampling: like in data absent, data augmentation strategy is used to increase the representation of one or many data types and balance the data type representation.
3. Under sampling: instead of using data augmentation to balance the data type representation, skipping some data in the training phase could also balance the data set [101].

3.2.4.3 LOSS FUNCTIONS AND OPTIMIZER

A loss function is a method of evaluating how well models are performing based on a given dataset. The higher the number returned by the loss functions, the higher is the error associated with the predictions [102]. They could be divided in three major categories:

1. Regressive Loss Functions: used in regressive problems were a quantity should be outputted based on a given input. It is used to solve problems like “how much will a car cost” or “how many people are in this picture”. The most common loss function in this category is the Mean Square Error;
2. Classification Loss Functions: used in the classification of problems when it requires labelling or categorization; the output is given in a confidence score format that could be binary (true or false) or in percentage. Problems with image classification often use this strategy. Binary Cross Entropy, Negative Log Likelihood, Margin Classifier and soft margin classifier are the most common loss functions under this category;
3. Embedding Loss Functions: capable of dealing with problems in which two inputs have to be compared by its similarity [103].

The optimizer changes the weight of ANN taking into account the number returned by the loss function, trying to minimize the error associated with the prediction [104]. It can be divided in two major categories:

1. Constant Learning Rate Algorithms: algorithms that have a parameter called learning rate that define how much the result of the loss function impacts on changing the synapse weight of an ANN. The lowest the learning rate, the lowest the learning velocity to achieve the optimal synapse weight and the highest the possibility to find the optimal synapse weight value.
2. Adaptive Learning Algorithms: algorithms that can adapt the learning rate during the training phase. They produce better predictions but they are computationally heavier compared to the Constant Learning Rate Algorithms [103].

3.2.4.4 TRANSFER-LEARNING

In cases where there is a lack of data, lower computational resources or little time available for the training phase, Transfer-Learning presents advantages. Transfer-learning is a technique that uses already trained ANNs for a large domain problem as a starting point for training in a specific domain. Taking as an example a CNN capable of predicting if an image contains a person face, it is possible to retrain this CNN to detect facial expressions. The closest the CNN domain problem is to the new domain problem, the highest is the probability to succeed on transfer-learning [87] [105].

3.2.4.5 PRUNING OR DROPOUT

It is known that more layers on ANN result in better accuracy but takes more resources and are more time consuming. The idea is to prune or dropout some parameters (perceptrons) in order to compact the overfitting while compromising the prediction accuracy to get faster predictions. This technique needs to be used with caution since it could lead to a bad prediction accuracy result [101][106].

As Figure 16 shows, the pruning cycle as three major steps: evaluating the importance of a neuron; removing the neuron; evaluating the impact of the removal neuron on the ANN performance. This cycle is performed until the optimal ANN state is achieved. The optimal state is accomplished when there is a good balance between the accuracy and the time taken for prediction.

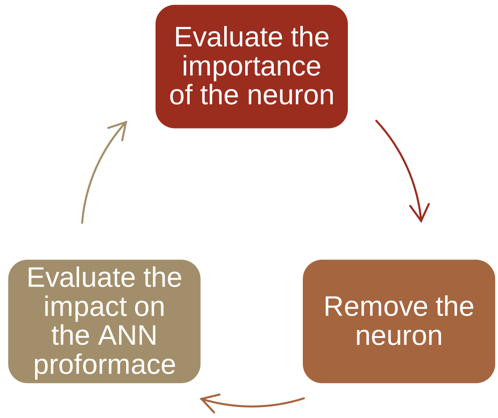


Figure 16 – Pruning cycle to find the optimal performance point on an ANN

3.2.5 ARTIFICIAL NEURONAL NETWORK PIPELINE

In AI, ensemble learning methods are the state-of-the art for many complex solutions. This concept breaks down the problem in several parts to reduce their complexity and difficulty to solve it [107]. Then, several models are applied to provide answers for each individual part. The advantages are: improving on general model performance; requires less complexity on models; the solutions are more readable and maintainable; different teams can work in different models.

The ANN pipeline is inspired in those methods by combining several ANNs that implement different models and use them as a pipeline to get better results. There are two different approaches to implement pipelines: specialize the ANNs for a specific subset of the domain problem or use many different ANNs to choose which one gives better results. Also, these could be used separately or mixed.

3.2.5.1 MODULAR ARTIFICIAL NEURONAL NETWORK PIPELINE

Based on divide and conquer algorithms [108] and on Bart L.M Happel's work [109], this model proposes dividing the domain problem into several parts and specialize each ANN to predict the results for each individual part. Then, the output of all specialized ANNs is used as input of a new ANN that will predict the final result.

Taking as an example the problem of detecting if a person is having a heart attack, it is possible to divide the domain problem in the following small concerns: heart rate, face expression, body posture, brain activity, body temperature and vocal sounds. Each domain has specific characteristics and produces different types of data. For example, the heart rate could produce a graphic, in which the x-axis could be the date-time and the y-axis the heart rate measure, and the face expression could produce images. For each concern and data collected a specialized ANN is created that predicts how confident is the result given. Then, the predictions of all specialized ANNs are used as input data to a final ANN that predicts the final result. As Figure 17 shows, there are four phases in this process:

1. Data separation: separation of the data in individual parts, each one representing part of the domain;
2. Specialized prediction: each data set, from the previous phase, is used in this phase as input to a specific ANN;
3. Specialized Data: the prediction from each specialize ANN made in during the previous phase is stored;
4. Final Prediction: collects all predictions from the specialized ANNs and uses them as input data to a final ANN (or a simple classifier) that will give the final prediction.

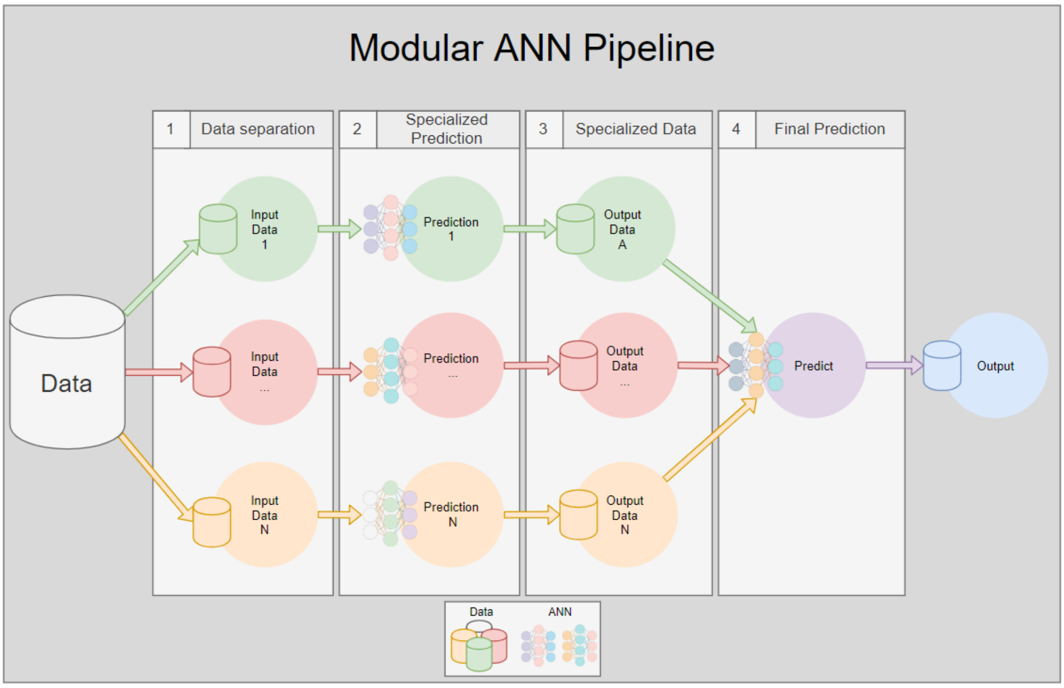


Figure 17 – Modular ANN pipeline model

There are several advantages of using this model:

1. Specialized teams for each concern: each concern can be handled by a small team specialized on getting the right data to feed the ANN. Also, they can create new ANN models to obtain better prediction results;
2. Reduced ANN complexity: instead of having an ANN with several hidden layers, for each specialized ANN the number of layers is reduced making each model less complex and easy to understand;
3. Modularity: as any software solution, modularity is a crucial point when the system becomes large. This allows software solutions to be manageable, scalable, controllable, monitored, composable, expandable and autonomous. Also, it allows the use of well-known and defined traditional software strategies in the ANN context.

3.2.5.2 BETTER ARTIFICIAL NEURONAL NETWORK PIPELINE

The concept is to train, using the same data, different ANN's that implement different models and, in real time, choose the ANN result that provides more confidence rates. This last classifier could be a simple function or a new ANN [101].

As Figure 18 shows, this pipeline has three main phases:

1. ANN prediction: all different ANNs will be fed with the same data and make their own prediction;
2. ANN result: all predictions given on phase 1 will be stored;
3. Choose better: the predictions stored will be used by an algorithm that will choose the best prediction. This algorithm could be another ANN or not.

This model increases the confidence of the prediction but needs more resources comparing to the Modular ANN Pipeline architecture.

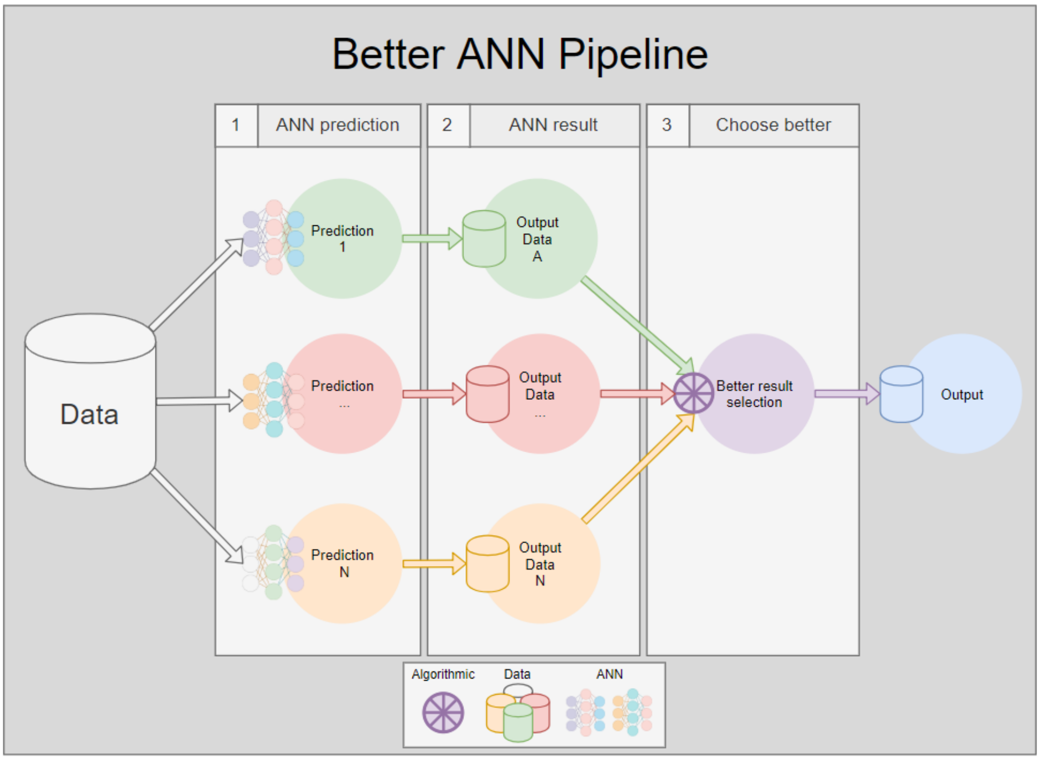


Figure 18 – Best ANN pipeline model

3.2.5.3 MIXING APPROACHES

The two models presented could be mixed to take advantage of both approaches. For instance, it is possible to use the Modular ANN Pipeline architecture and instead of having one specialized ANN for each concern, several ANN models could be used and the output of the best one forward to the next phase. Also, several modular ANN pipelines could be used in the same software solution in order to solve complex problems. Each module is responsible for one part of the software and gives an input to the core module. Taking a humanoid robot as an example, there is a need to interact with a complex environment. The data collected from the environment could be used to feed different modules. Each module will give insights to a core module that will decide what action the robot should take. For instance, if the robot is in a supermarket and a client makes a question, the robot could take the data from the environment and send it to the emotion detection and natural language modules, and use the information given to feed the last module, which decides the proper answer to be given to the client.

3.3 TECHNOLOGIES

Choosing the right technologies can have a significant impact on how well the FullExpression Ecosystem will answer to the challenge of building a software solution for emotion classification. Here, it will be described the main technologies from the DL and Web ecosystems and what fits the most on this work goals.

3.3.1 DEEP LEARNING

There are several DL learning frameworks and libraries to handle ANNs. They offer support for different languages, platforms and devices. Table 8 describes some of them, including Scikit-Learn [110], Microsoft Cognitive [111], Caffe [112], Torch [113], Accord.Net [114], Brain.js [115] and Tensorflow [116].

Table 8 – DL technologies

Technology	Description
Scikit-Learn	Scikit-learn is an open source library written for Python and provides several ML tools for classification, regression, clustering, dimensionality, model selection and preprocessing problems [117]. Is designed to work with scientific libraries such as NumPy (matrix calculation) [118], SciPy (“mathematics, science and engineering”) [119], MatplotLib (provides several interactive figures specially used for data visualization) [120], Sympy (symbolic mathematics) [121] and Pandas (provides high-performance, easy to use data structures and data analysis) [110].
Microsoft Cognitive Toolkit	Microsoft cognitive Toolkit (also known as CNTK) is an open-source framework capable of training DL models and can be included in Python, c#, c++ and BarinScript. It is easily integrated with Microsoft Azure, providing an ecosystem for facilitating the process of continuous integrations and delivery [111].
Caffe	Caffe is a DL framework optimized for speed and modularity. It is written in c++ and can be integrated with Python. Claims to be present on “academic research projects, startup prototypes and even large-scale industrial applications in vision, speed and multimedia” [112].

Torch	Torch is an open source library which offers several algorithms for deep learning. Optimized for speed, since it uses the GPU. Written in Lua and can be use in c language. Some key features include N-dimensional arrays, linear algebra routines, numeric optimization routines and GPU, Android and iOS support [113].
Accord.Net	Accord.Net is an open source framework written entirely in c#. It is a framework for building computer vision, computer audition, signal processing and statics application and has dedicated tools for the DL field too. It has an extensive documentation and a group of sample applications for fast project set ups [114].
Brain.js	Brain.js is a javascript open source library focused on reducing the complexity of using ML and DL algorithms. Prepared to run in node.js and browser. Uses WebGL technology to access the GPU device and improve the calculations speed. It is possible to import and export DL models [115].
Tensorflow/Keras	For manipulating ANN, Tensorflow and Keras are widelly used. Open-source library that can be used in several languages suchs as Python, c# and java and even in javascript. It is a symbolic math library which facilitates the development of ANN by creating an abastraction layer between the math calculations and computing resources needed to build and run models. Also handles the memory and CPU/GPU resource usage efficiently, regardless of the device being used. The models can be store in several formats such as Keras or tensorflow.js in order to be used in different scenarios [116]. In addiction, the Tensorflow community provides several pre-build models that can be consumed easly, facilitating the process of transfer learning which saves time and resources.

Only brain.js and TensorFlow offers the option to run DL models exclusive on browser, which is one of the goals of this work. Since training CNN models is computationally heavy, training models using directly the GPU is faster comparing to training models directly on the browser, which has several abstraction layers before reaching the GPU. From all technologies presented on Table 8, the Tensorflow seems to fit better the requirements of this work: is fast, since it can use directly the GPU to build and train DL models; facilitates the process of transfer learning, by providing several pre-trained well known models ready to be re-trained; and the models can be exported and consumed directly on the browser.

3.3.1 WEB

There are several technologies to build and maintain front-end web applications. The most popular are Vue [122], React [123] and Angular [124]. The Table 9 describes both advantages and disadvantages of using such technologies.

Table 9 – Vue vs React vs Angular

	PROS	CONS
VUE *RELEASED IN 2014 IS THE YOUNGEST. IT WAS DEVELOPED BY AN EX-GOOLGE EMPLOYEE (EVAN YOU) AND IS SUPPORTED ENTIRELY BY THE OPEN SOURCE COMMUNITY* [125][126][122]	Uses virtual DOM which helps on efficiency and code debugging. Is the most lightweight from all web technologies studied.	Is new, meaning an immature ecosystem. Has a small community.
	Flexibility which speed up the development process and learning curve, but best practices can be left out in the process, which untimely could lead to unreadable and unmaintainable code.	
	Is entirely driven by the open source community, which can be a pro or con depending on the perspective view.	
	Is growing in popularity and community size. Is easy to learn. Has an extensive and detailed documentation. Uses Component-based architecture, driving the structure to be reusable, readable, testable and maintainable.	
REACT *RELEASED IN 2013 BY FACEBOOK IS ONE OF THE MOST POPULAR WEB TECHNOLOGIES, BEING USED IN FACEBOOK, INSTAGRAM AND WHATSAPP* [125][126] [123]	As an extensive detailed documentation.	Is not a complete framework and requires using third-party libraries.
	Flexible which speeds up the development process and learning curve, but best practices can be left out in the process, that could untimely lead to unreadable and unmaintainable code.	
	Component-based architecture, driving the structure to be reusable, readable, testable and maintainable. Well documented and easy to setup a new project. Used and approved by the industry.	
ANGULAR *RELEASED IN 2010 BY GOOGLE BEING THE OLDEST. IN 2016 GOOGLE REWROTE THE FRAMEWORK MAKING SUBSTANTIAL IMPROVEMENTS ON THE TECHNOLOGY. IS USED BY BIG COMMERCIAL COMPANIES SUCH AS GOOGLE AND WIX*[125][126] [127][128]	Google Long-Term support, ensuring regular updates both in the framework and documentations. As an extensive detailed documentation. Used and approved by the industry. Used Component-based architecture, driving the structure to be reusable, readable, testable and maintainable. Uses RxJS library to handle asynchronous events easily, a well-known requisite in front end web application. Compressing the code transpiled to javascript, html and css, providing a level of abstraction to DOM manipulation and by lazy loading code only when it is needed.	Angular is verbose and complex leading to a steep learning curve; Migration legacy systems from AngularJS is difficult. CLI documentation could be better. Requires learning typescript which is complex and OOP.

Vue and React offer better performance and flexibility than Angular. For that reason, they are more suited for light-weight applications. But angular offers a solid modular architecture and several tools that makes the best choice for large enterprise applications. Even if Angular is more difficult to learn, has a more mature community and regular updates. Moreover, the FullExpression team as experience with angular, which mitigates the learning curve problem. And, both Tensorflow and Angular are supported by the same community (Google), which means less compatibly issues between them. For those reasons, Angular seems the wise choice to be used to create the FullExpression Ecosystem.

3.4 CONCLUSIONS

AI is a computer science field that tries to reproduce human cognitive capabilities to solve complex engineering problems, when other computer science files could not. Subsets like ML and DL are getting more attention, not only because of model capabilities, but also due to the current computer power and data available. Simple ML ANN evolved to complex CNN architectures capable of solving general image classifications with high reliability but at the cost of being computationally demanding. Architectures such as MobileNet were created to be used in mobile devices, or devices with less resources available, without compromising a lot the accuracy results. Applying the transfer learning technique, which uses information stored on one model and uses it to a different but related problem, is a good choice when time and computation resources are limited.

There are different DL models [90] which provide different accuracies, file sizes and latency ratings. The accuracies are the most important aspect to be considered when choosing the best model for FullExpression Ecosystem, but the tradeoff between accuracy, model size and latency values should not be high, since the FullExpression Application runs on browsers and for real-time experiences, such as classifying emotions from the webcam, heavy loading and prediction times could lead to a bad user experience. Furthermore, not all models are compatible with tensorflow.js framework (the best framework available to manipulate models on javascript). Thus, it seems that MobileNet V1 is the best model to use in our case, since it does not compromise a lot the accuracy values and has a reduced size and latency values, which fits in the FullExpression Application needs (less than 20 megabytes in tensorflow.js format).

DL development process has four main steps: data gathering, exploration and normalization; training; testing and inference. Most of time spent by data scientists seems to be related with collection and exploration of data, suggesting this is a key concern to be successful on DL field. Amount and data quality have a huge impact on the final accuracy results but is not the only factor: choosing the right loss and optimizer function as well as removing or adding the right neurons and layers seems to also have an impact.

Moreover, two ANN pipelines were described, exposing how several specialized ANNs could be organized in order to solve a much general problem with the benefits of reducing the complexity of building ANN, creating a specialized team for each concern and obtaining a scalable, manageable, controllable, monitored, composable, expandable and autonomous software.

The stack technologies to fulfil all FullExpression Ecosystem requirements can be divided into DL and Web. From the several frameworks and libraries studied, it seems that both Tensorflow and Angular are the best choice for building DL models and consume them on front-end web applications.

The next chapter will study and compare different research works and commercial solutions, describing the main techniques and target markets if applied.

4. RESEARCH WORKS AND COMMERCIAL SOLUTIONS

There are several research works and commercial software that propose solutions to automatically detect emotions from facial expressions. This chapter describes and compares some of them, as well as the techniques and methodologies used. Additionally, an engineering process for building large emotion recognition systems will be proposed.

4.1 RESEARCH WORKS

FACS is a coding language that maps all facial expressions, due to facial muscle movements, into codes. These codes are divided in Action Units (AU) and Tongue Movements (TM) and each code has a number and a name. FACS is widely used on emotion studies and software since it provides a systematical way of categorizing facial expression emotions. Each emotion can be expressed by combining AU and TM. The original version was created by Carl-Hearman (1970) with 23 AU [129], and it was later developed by Paul Ekman and Wallace Friesen, publishing a first version in 1978 and a second version in 2002 [130]. Figure 19 shows some AU codes and their descriptions divided by Upper Face and Lower Face facial patches.

Core emotions can be composed by one or many AU, and FACS based software try to recognize the face land marking (AU points) present on images or videos in order to correctly identify emotions. By using FACS it is possible to eliminate the noise information about differences in age, sex, skin color or ethnicity since it only looks for differences between facial muscles. For that reason, FACS seems to be the most popular framework in emotion recognition systems [131] [132] [133] [134] [135] [136] even when DL techniques are used [137].

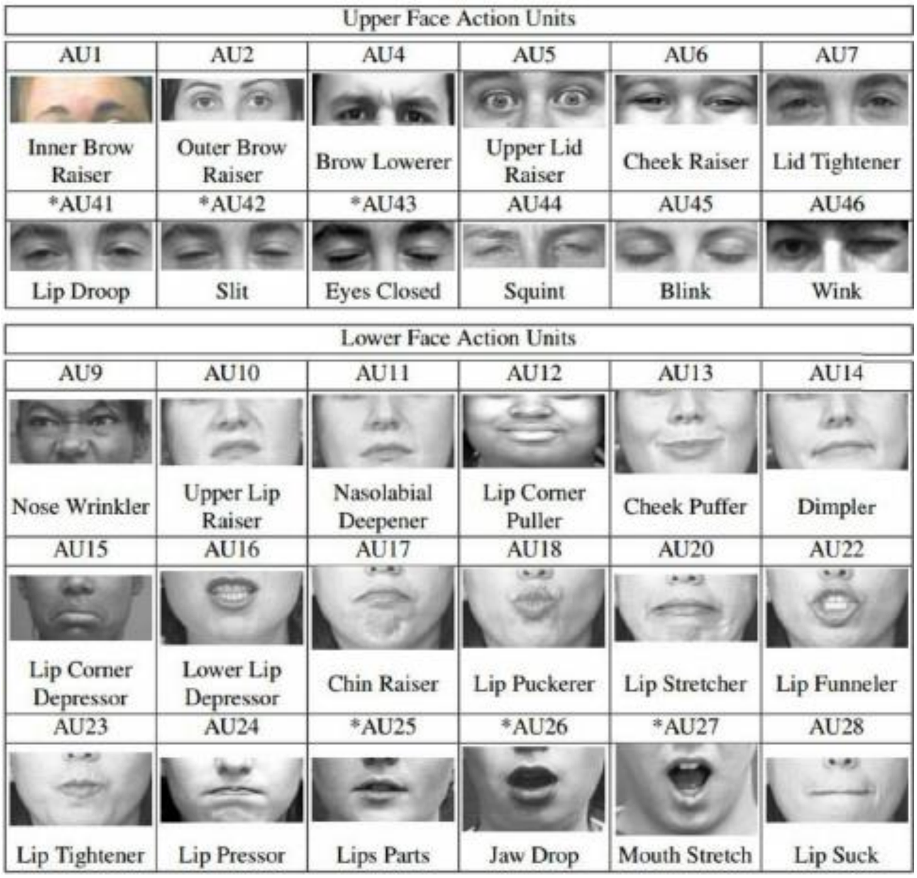


Figure 19 – Facial action code system: action units examples (*Adapted from* [138]*)*

There are a set of steps, common to several research works, in order to get the best results on facial emotion recognition:

1. Facial Recognition: extraction of people's faces from images in order to avoid useless information. The Viola-Jones face detection algorithm [139] is an example that can be used to extract faces from images and can be found on several works [135][140][141][142]. Other works use the "Multi-pose Face Detection Based on Adaptive Skin Color and Structure Model" [143] technique [144] and their own algorithms to extract faces [131];
2. Facial Alignment or Face Land marking: detection of the edges of the face, eyes, nose, mouth and eyebrows. To do so it is possible to use Local Binary Patterns (LBP) [136][142], Active Shape Model (ASM) [135], Active Appearance Models (AAM) [133], Fiducial Facial Point Detector [141], MATLAB tools [134], or even a hybrid approach of AAM and LBP [145]. Other authors have also proposed their own methods [132][140]. Some works skip the first step as the algorithm used does not need the initial facial recognition step (maybe due the highly controlled test environment provided). Figure 20 shows, as an example, faces after the Facial Alignment or Face Landmarking process;
3. Dividing Landmarking by Facial Patches: division of the face in different parts (eyes, nose, mouth and eyebrows). The work by Sadeghi et al. (2013) divided some facial patches to remove the cheeks and face around, achieving a faster detection result on the classification phase, and reducing computational costs [136];
4. Emotion classification: based on the facial alignment step, an algorithm will be used in order to detect the associated emotion. Different AI classification algorithms are used such as Lean Mean Square [144], Support Vector Machine [142], [140] and [146], ANN

[145], [141] and Adaboost [140]. Also, the work by Velusamy (2011) mapped Facial Landmarks into AU in order to create their own algorithm that classified emotions based on different AU combinations [131].

Figure 20 – Example of facial alignment/face landmarking (adapted from [90] and [91])

Table 10 compares emotion detection accuracy values of several FACS-based research works that used traditional algorithms to extract Facial Alignment or Face Landmarking to classify the emotions present in an image, or group of images. The information about accuracy values was extracted from each research work meaning the same conditions were not always guaranteed. Likewise, not all works detected the same amount of emotions.

Table 10 – FACS-based research works comparison

Reference	Accuracy Value	Database Used	Number of Emotions
Human-computer interaction using emotion recognition from facial expression [140] [1]	98.8%	CK+ [147][148]	**6** (Anger, Disgust, Fear, Sad, Sad, Surprise)
A method to infer emotions from facial action units [131]	97.0%	CK+ [147][148]	**6** (Anger, Disgust, Fear, Happy, Sad, Surprise)
Facial expression recognition using geometric normalization and appearance representation [136]	96.5%	CK+ [147][148]	**6** (Angry, Disgust, Fear, Happy, Sad, Surprise)
Real-Time Emotion Recognition from Facial Images using Raspberry Pi II [135]	94.0%	Database locally created	**5** (Anger, Disgust, Happy, Neutral, Surprise)
Fast facial expression recognition based on local binary patterns [142]	86.7%	JAFFE [149]	**5** (Anger, Disgust, Happy, Sad, Surprise)
Facial emotion recognition in modern distant education systems using SVM [133] [1]	83.9%	CK+[147][148]	**6** (Anger, Disgust, Fear, Happy, Sad, Sad)

[1] The work presents accuracy values for each individual emotion, that were averaged to obtain the value presented.

Facial Emotion Recognition Using Fuzzy Systems [134]	78.8%	JAFFE [149]	**6** (Anger, Disgust, Fear, Happy, Sad, Surprise)
Emotion Recognition from 3D Videos using Optical Flow Method [132]	75.3%	BU3DFE [150]	**5** (Angry, Disgust, Happy, Sad, Surprise)

Few works [151][152][153][137] use CNNs to detect emotions from images, however it seems that EmotionalDan [137] is the CNN solution that achieves a higher accuracy value (75%). EmotionalDan uses Deep Alignment Network (DAN) architecture (originally created to solve face alignment problems) to detect emotions. EmotionalDan detects seven emotions, including neutral. Table 11 compares the accuracy results obtained by Tautkute et al. (2018) using different emotion detection CNN' architectures (the CNN (2) and CNN (5) are ANNs fully connected with two and five hidden layers; Inception-V3 is the third version of the Google Inception model). Also, the authors suggest that "contrary to the competing methods, DAN can therefore handle entire face images and not patches which leads to a significant reduction in head pose variance and improves its performance on a landmark recognition task" [137].

Table 11 – Accuracy results: Inception-V3 vs EmotionNet 2 vs EmotionalDAN *(adapted from* [137]*)*

	CK+	JAFFE	ISED
CNN (2)	62.8%	48.4%	51.6%
CNN (5)	72.8%	**50.2%**	59.3%
Inception-V3	30.4%	26.8%	47.9%
EmotionalDAN	**73.6%**	46.5%	**62.0%**

By analyzing Table 10 and Table 11 it is possible to infer that CNN architectures are less accurate than traditional approaches, even when the CNN is FACS-based (like EmotionalDan). The reason could be the lack of research works on DL-based emotion detection techniques. Nevertheless, for some databases, EmotionalDAN presented good accuracy results [137].

4.2 COMMERCIAL SOLUTIONS

There are more than twenty emotion detection commercial software [154]. In this section we will cover five of the most known, showing their main characteristics and what they have in common.

Affectiva is an emotion AI platform created for brands, advertisers and market researchers in other to understand the emotional engagement about their digital content. Affectiva aims to help organizations creating the best ads and media content. The *affdex for market research* is a cloud-based solution that uses a webcam to record facial emotions that media content consumers are expressing. The results are aggregated and displayed in dashboards [155].

Eyeris developed a multi-modal AI engine to interpret complex visual behavior patterns for occupants inside autonomous vehicles. Among other things, it can read 7 facial expression emotions based on a large facial dataset that represents different races, ages, body sizes and activities [156].

Nvisio uses DL technologies to, among other features, recognize facial expression emotions. Their products are present on finance, automotive, healthcare and media markets [157].

Project Oxford is promoted by Microsoft and uses AI to solve problems on computer vision, speech and language analysis. For emotion detection, they provide a RESTfull service that uses an image to obtain a JSON that detects faces in images and the related emotions [158].

FaceReader is a professional software for facial expression emotion analysis and is delivery by Noldus company. Noldus aims to create software for human behavior research, collaborating closely with the scientific community [159].

For the commercial software solutions presented, Table 12 compares them regarding market target, number of emotions detected, techniques used (for those with information available), integration with other systems and if the products are paid.

Table 12 – Commercial Software comparison

Software	Market targeted	Number of emotions	Techniques used	Possibility to integrate with other systems	Payed products
Affectivia	Ads and media content	7 (anger, contempt, disgust, fear, joy, sadness and surprise)	Computer vision algorithms for facial landmarking and DL algorithms to classify expressions	Yes	Yes
Emovu	Autonomous vehicles	7 (anger, contempt, disgust, fear, joy, sadness and surprise)	Usage of DL algorithms (lacks information about how it is implemented)	Yes	Yes
Nviso	Finance, automotive, healthcare and media markets	6 (anger, disgust, fear, happiness, sadness, surprise)	Facial Imaging (using ML techniques and FACS framework)	Yes	Yes
Project Oxford	High tech companies	7 (anger, contempt, disgust, fear, happiness, sadness and surprise)	*(lacks information)*	Yes	Yes (depending on software usage)
FaceReader	Scientific community	6 (boredom, confusion, contempt, interest neutral)	Facial Imaging (using DL techniques and FACS framework)	Yes	Yes

By analyzing all commercial solutions, it is possible to infer that, on average, seven is the number of emotions detected, all solutions are paid, there is the possibility of integration with other systems, different techniques are used to detect emotions on images and the target market is different from solution to solution.

4.3 CONCLUSIONS

After analyzing research works and commercial solutions it is possible to conclude that AI algorithms can effectively provide solutions to emotion recognition from face expressions. Likewise, most solutions have great accuracy results (more than 70% accuracy). However, information about how well they behave in real situations is poor, which can lead to bad generalization results. Accordingly, some of the works tested their solutions on highly controlled environments, which can lead to biased results.

Nevertheless, traditional AI and ML algorithms were used in the majority of the works to classify facial expression emotions. Few works seem to use CNN to solve this kind of classification problems, even when CNNs seems an interesting alternative to traditional techniques, showing promising results ([137]). In order to explore more the facial expression classification using CNNs, and to better understand the capabilities of these techniques on the emotion recognition field, this work will use this technique.

The next chapter will describe the solution implemented in five different points of view: logical, process, physical, development and use case views.

5. DESIGN AND IMPLEMENTATION

FullExpression Application is the product developed in this work and contains all functionalities that allow users to generate, see and interact with emotion recognition. In order to predict emotions from facial expressions, a total of three web applications and a DL training script was created. Each one plays a specific function: the FullExpression Web Tools Application has a set of functionalities which helps to build and evaluate the FullExpression Application; the FullExpression Experiment Application is a web application created to collect data about the human accuracy on recognizing emotions from facial expressions and the FullExpression Training Script is a Python program used to train, fine-tune and generate DL models.

All solutions will be described according to a 4+1 view model of software architecture proposed by Philippe Kruchten [160], which allows the observation of the system from five different points of view: logical, process, physical, development and use case views.

The Design Alternatives chapter will expose different design options, justifying the decisions made regarding software architecture. Lastly, a software engineering process for emotion recognition solutions will be exposed, based on the state of art and this work.

5.1 USE CASE VIEW

The use case view shows the system functionalities from the viewpoint of the outside world, capturing user goals and scenarios. Thus, it is helpful for defining and explaining the structures and functionalities present in all viewpoints of the system.

An UML use case diagram will be presented for each application, as well as an explanation about all functionalities. Moreover, print screens from some applications will be presented, illustrating how some features are shown in the software and how the user can interact with them.

5.1.1 FULLEXPRESSION APPLICATION

The two main functionalities of the system are: emotion analysis in real time and emotion analysis from images. The first, uses the webcam for capturing real time images and detect potential faces and emotions expressed; the second allows users to generate emotional reports from their own images. Additionally, the user can interact with the report by searching images, downloading a version of the report in excel format or downloading the images organized by the corresponding emotion classification.

All functionalities are represented in an UML use case diagram format (Figure 21).

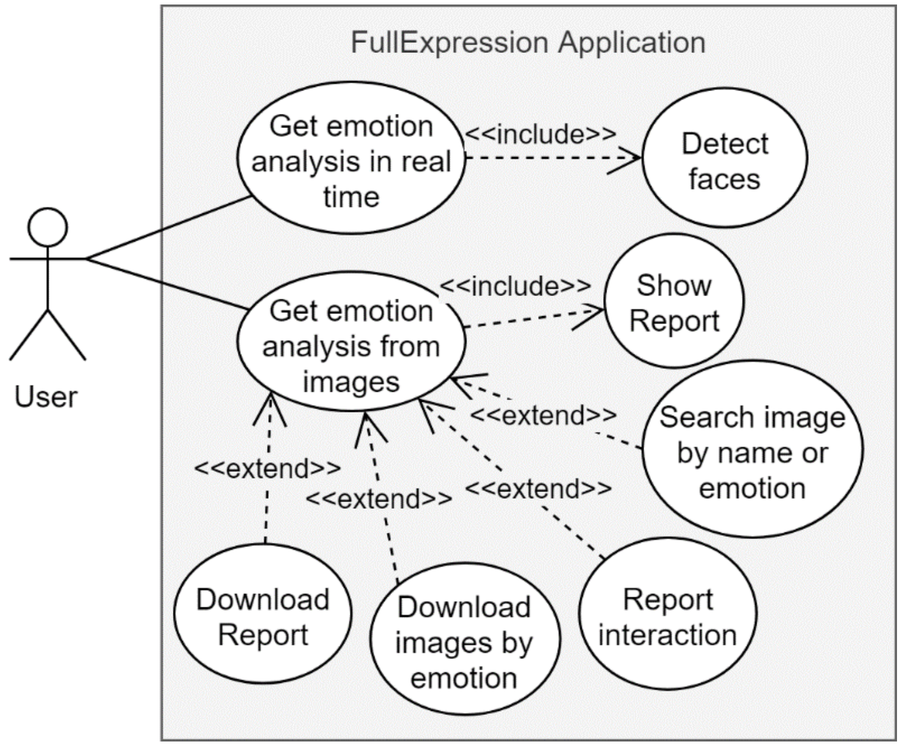


Figure 21 – FullExpression Application UML use case diagram

As Figure 22 demonstrates, to access the main functionalities, the software as a main menu with two options: “Check emotions from your webcam!” and “Check emotions from your photos!”.

By choosing the first option, the user is redirected to the screen shown in Figure 23. In the top left corner, he can change the application to fullscreen mode. In the center of the screen the video captured from the user webcam is presented, and it is visible the software is finding, in real time, faces and emotions. In the middle bottom of the screen, the left button activates the “find face” feature and the right button the “emotion recognition” feature. Additionally, in the bottom right corner of the screen, the user can be redirected to the main area.

Figure 22 – PrintScreen of FullExpression Application Main Area

Figure 23 – Print Screen of FullExpression Application analysis in real time feature

The “Check Emotions from your Photos” can also be accessed in the main area, where the users can upload their own images (Figure 24). Then, the software will classify the emotions expressed in those images producing a dynamic report (Figure 25). The report is divided by graphics showing statistical data and individual images. For each image, the associated name and emotion is presented. Images can also be searched by emotions or name, by writing in the area positioned on the right side of the magnifier icon. Furthermore, the context menu in the bottom right of the screen, allows the user to import new images, download images organized by emotions or download an excel report.

Finally, the application is web responsive, meaning the software will work in different devices and screen sizes. An example for desktop, tablet and mobile screen sizes can be found on Attachment I.

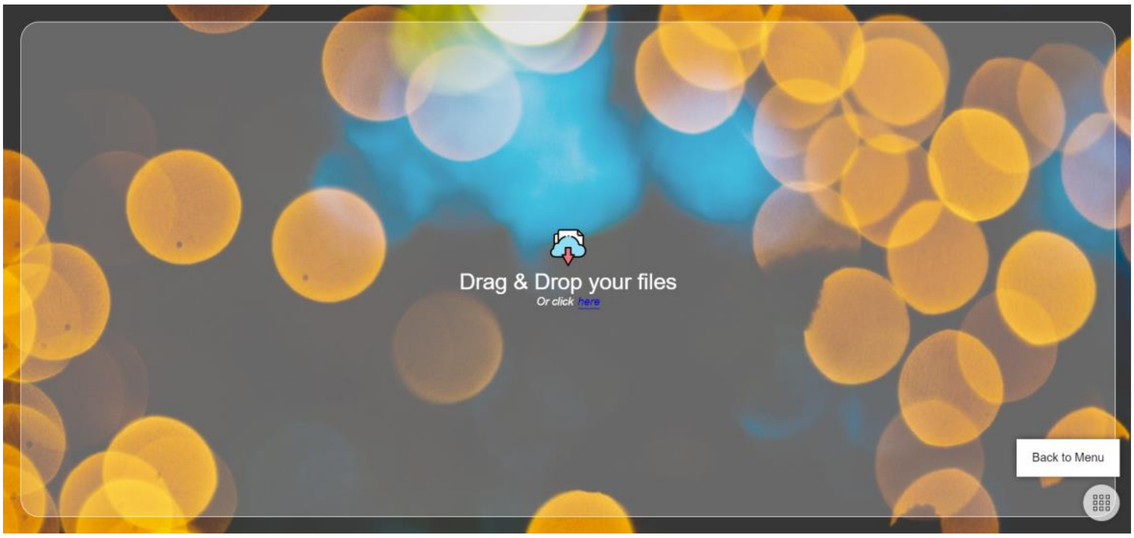


Figure 24 – Print screen of FullExpression Application showing the upload images area

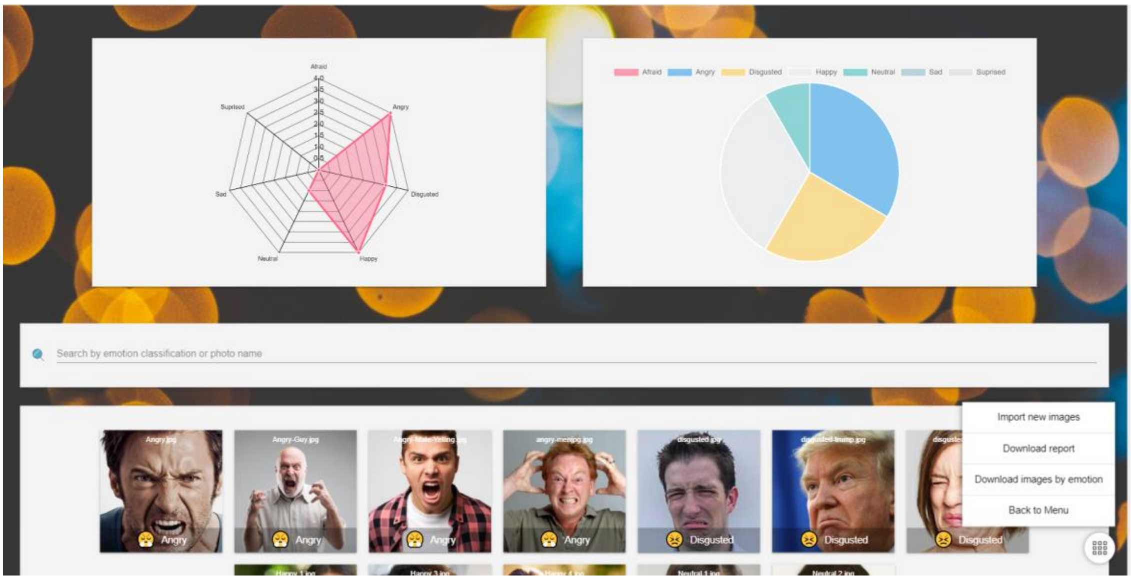


Figure 25 – Print screen of FullExpression Application showing the report generated after image classification

5.1.2 FULLEXPRESSION WEB TOOLS APPLICATION

The FullExpression Web Tools Application provides three main functionalities: normalize images, augment images and test models. The normalize image shapes a set of pictures for the same format in terms of color (gray scale) and size, where the size can be adjusted by the user, but not the color scale. Consequently, this feature is mainly used to normalize the facial expression databases for the same format before being used to train the DL model. The “augment images functionality” expands the number of images present on a given database, by copying randomly some images and then performing image transformations such as rotation or flip (horizontal and vertical). This feature is mainly used to balance databases, where the number of images present are in different proportions for different emotions. The “test model functionality” tests a trained DL model against a given image database and shows a report with the testing results, including the confusion matrix, model accuracy, precision, recall, F_{score} and the accuracy metrics for each core emotion.

All these functionalities are represented in Figure 26 in an UML use case diagram.

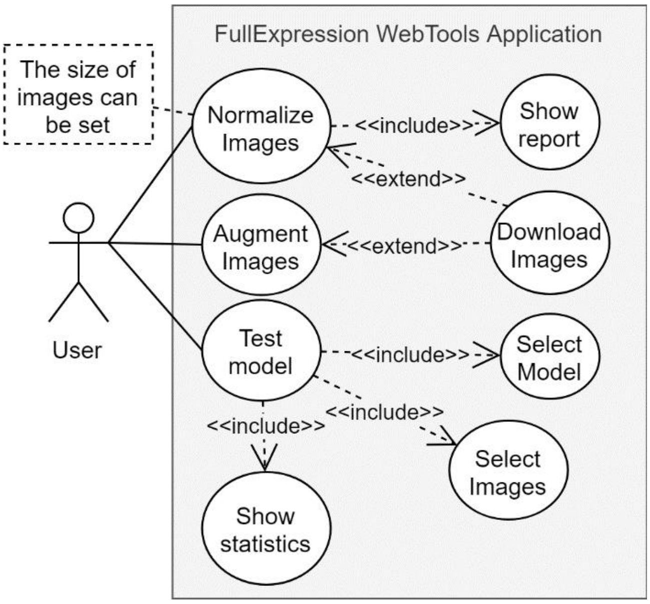


Figure 26 – FullExpression Web Tools Application UML use case diagram

5.1.3 FULLEXPRESSION EXPERIMENT APPLICATION

The FullExpression Experiment Application has two main players: the administrator, responsible to start a new experiment, and the participant. Only the administrator has access to the system, after authentication. He can start a new experiment and download the data collected from the experiments. To start a new experiment, the administrator also needs to choose the database with images, to be used during the experiment. The participant will start by reading the experiment instructions (Figure 27) and providing personal data (Figure 28). Then, he will classify the emotions expressed on each image provided (Figure 29) by clicking on the bottom buttons. In the end, individual accuracy, average accuracy scores of all participants, and the FullExpression Application score are presented (Figure 30).

All functionalities are represented on Figure 31 in a UML use case diagram format.

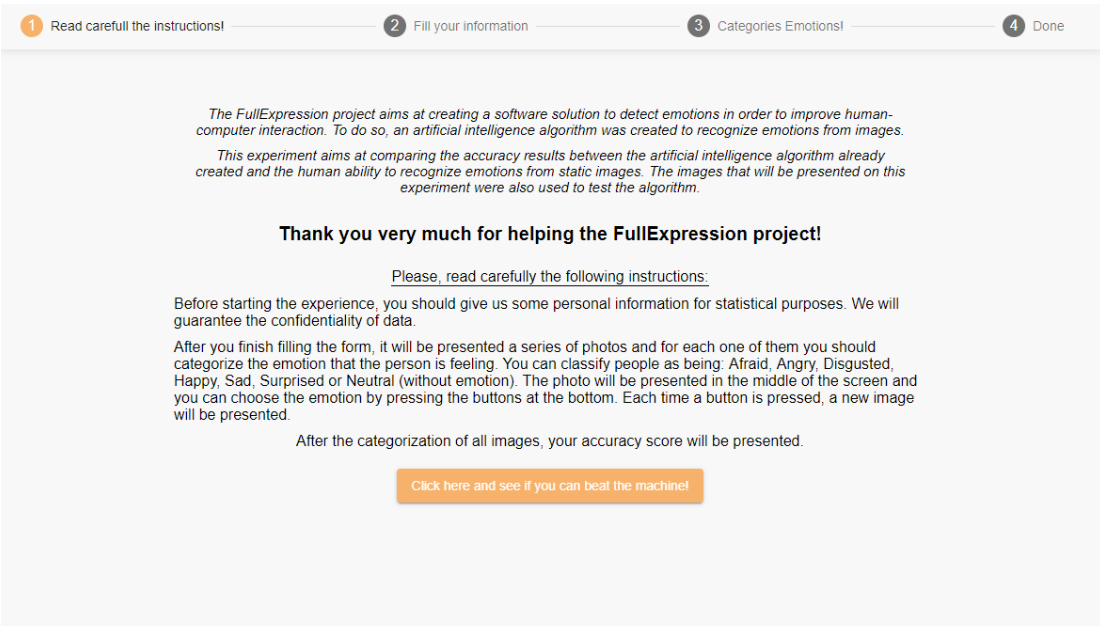


Figure 27 – FullExpression Experiment instructions screen

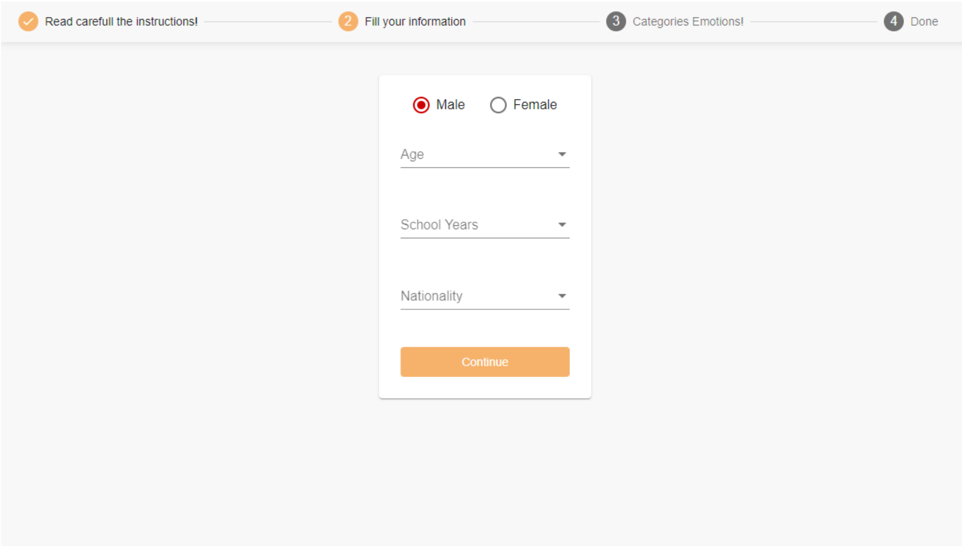


Figure 28 – FullExpression Experiment personal information screen

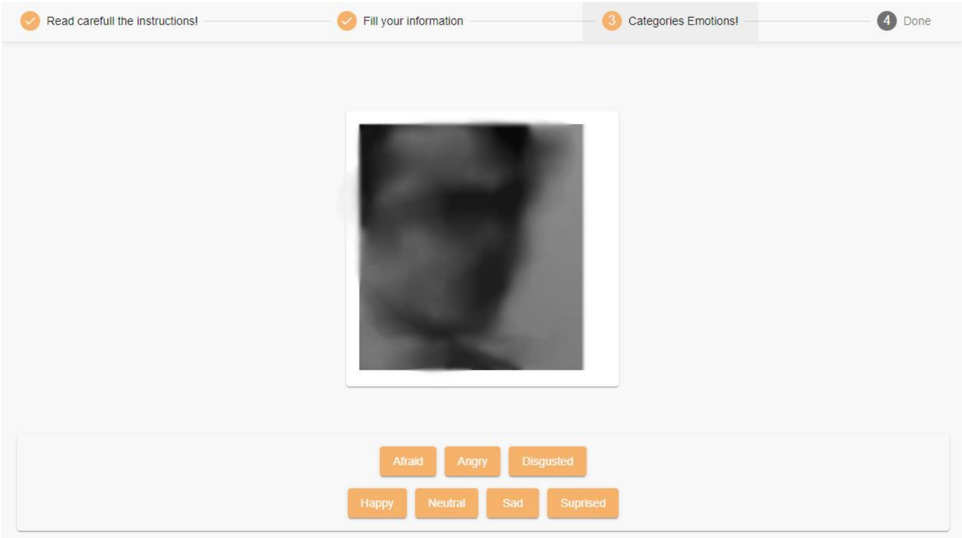


Figure 29 – FullExpression Experiment main area

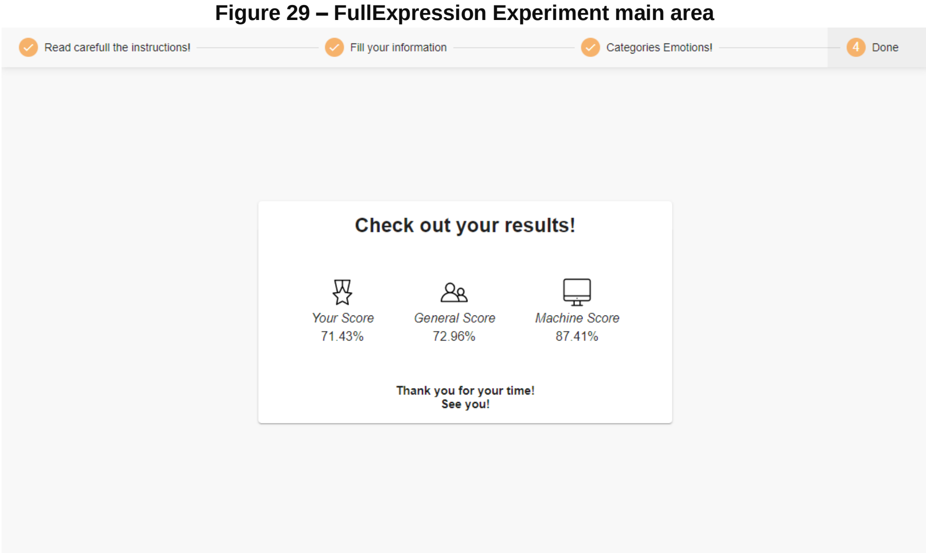


Figure 30 – FullExpression Experiment scores board screen

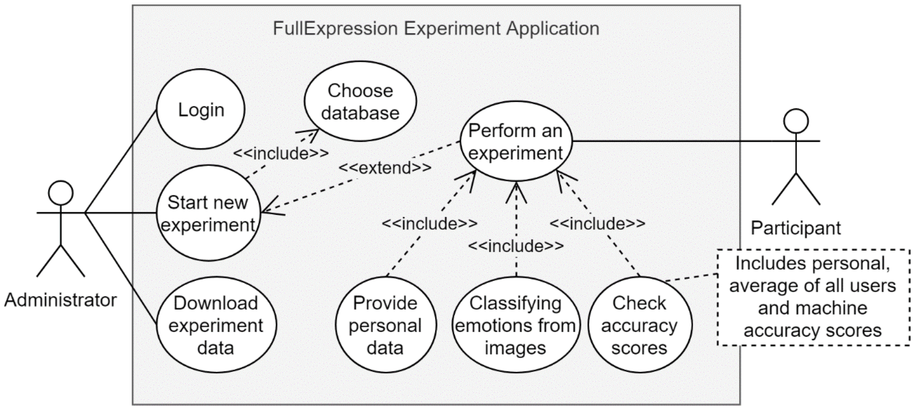


Figure 31 – FullExpression Experiment Application UML use case diagram

5.1.4 FULLEXPRESSION TRAINING SCRIPT

The FullExpression Training Script allows the user to select the model to be trained and the images used to train it, to validate and to test the model (section 6.1.3 Cross Validation). The configurations are used to train, fine-tune the model and run the script in order to train the model and generate artifacts (e.g. metrics in text format, the model in Keras and tensorflow.js formats and the confusion matrix). Modifying configurations includes changing the training, validation and test database batch sizes, number of epochs, the optimizer and loss functions, the learning rate value and the number of trainable layers (the number of ANN layers which the neuron weights will change during the training process). Also, after training the model, the script produces the training model artifact in two different formats, Keras and tensorflow.js, and shows a confusion matrix of the test results, giving the possibility to save it on the disk.

All functionalities are represented on Figure 32 in a UML use case diagram format.

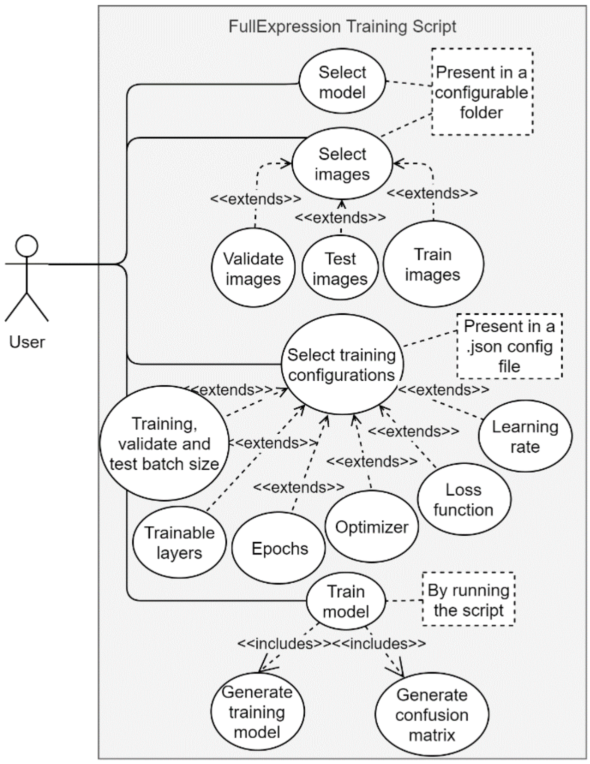


Figure 32 – FullExpression Training Script UML use case diagram

5.2 LOGICAL VIEW

The logical view breaks the system down in its computational, communicational and behavioral responsibilities, detailing the system logical parts as well as their interactions. Using UML components and sequence diagrams, this chapter will decompose and explain the logical parts of the FullExpression ecosystem.

To implement all functionalities described before, a total of three web applications (FullExpression Application, FullExpression Web Tools Application and FullExpression Experiment Application), one training script (Training Script) and three libraries (Core, Emotion Classification and Confusion Matrix) were created. The Main Application is responsible for providing all functionalities described in Figure 21, which depend on Core, Emotion Classification and Confusion Matrix libraries.

The Core library provides the core functionalities for all web application such as UI animations, loaders, interaction with charts, uploading images, statistical boards, access to user webcam, browser canvas manipulation and excel/general files manipulation and download. To do so, dependencies to external libraries were needed including xlsx [161] (handling of excel files), chartJS [162] (visualization and interaction with charts), angular materials (building and organizing the UI), jszip [163] (creating .zip files) and file saver [164] (downloading files).

The Emotion Classification library provides functionalities to detect and classify emotions. It is composed by services that allow face detection (implementation of the viola-jones algorithm [165]), image normalization and interaction with the model (CNN), in order to classify the emotions present in the images. Also, the software uses tensorflow.js [166] to handle the model and the Confusion Matrix library, then providing functionalities to visualize and calculate statistical data from a given confusion matrix.

As the FullExpression Application depends on Core, Emotion Classification and Confusion Matrix libraries, so does the Web Tools Application and Experiment Application.

The Training Script has a key role on building the FullExpression Application since it provides the functionalities necessary to train the model through the transfer learning technique. The script uses models in Keras format and self-sets-up by reading a configuration json file, which can be changed before running the script. Similarly, it reads images from the disk organized by training, validation and testing folders, whose location can be set through the configuration file.

Moreover, the software architecture is driven by the separation of concerns (also known as modular architecture), as can be observed easily on Figure 33 and Figure 34, where each web application, library and module have a specific purpose in creating a chain of abstraction and ultimately promoting code readability, maintainability and reusage. With this architecture, other web applications outside the FullExpression ecosystem can easily install the Emotion Classification library and be ready to analyze emotions on images.

For better understanding of how the applications interact with the libraries several UML sequence diagrams were created for the Classify Emotion Process, Emotion Classification From Webcam, Download Experiment Data, Test Model and Training Process functionalities. The diagrams are available under Attachment II, Attachment III, Attachment IV, Attachment V and Attachment VI, respectively.

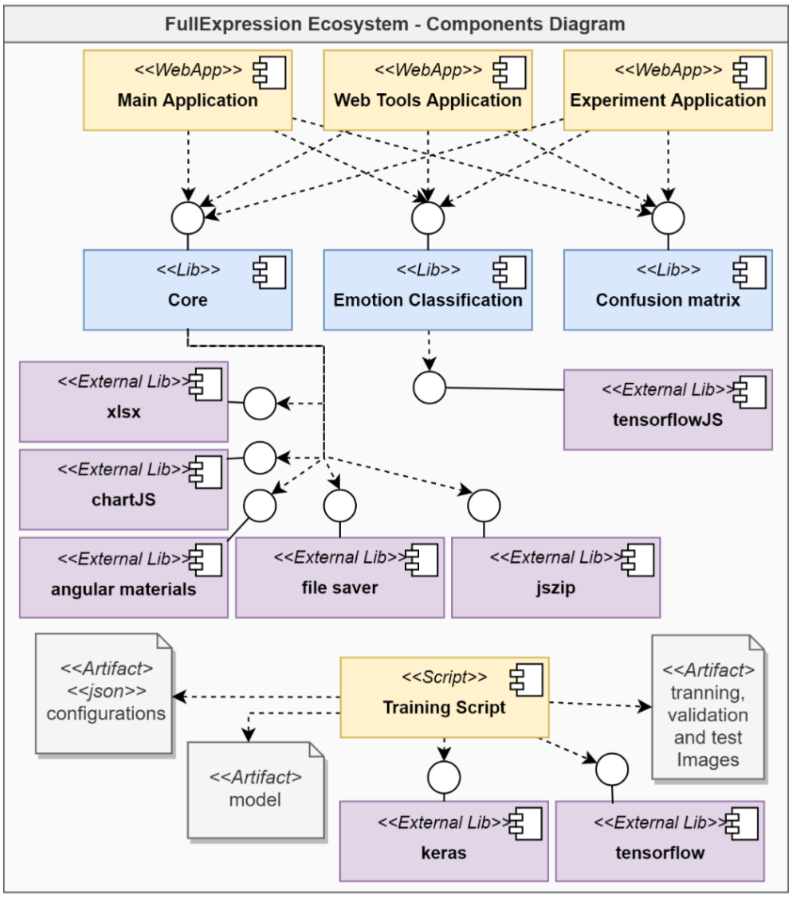


Figure 33 – UML components diagram of the FullExpression ecosystem

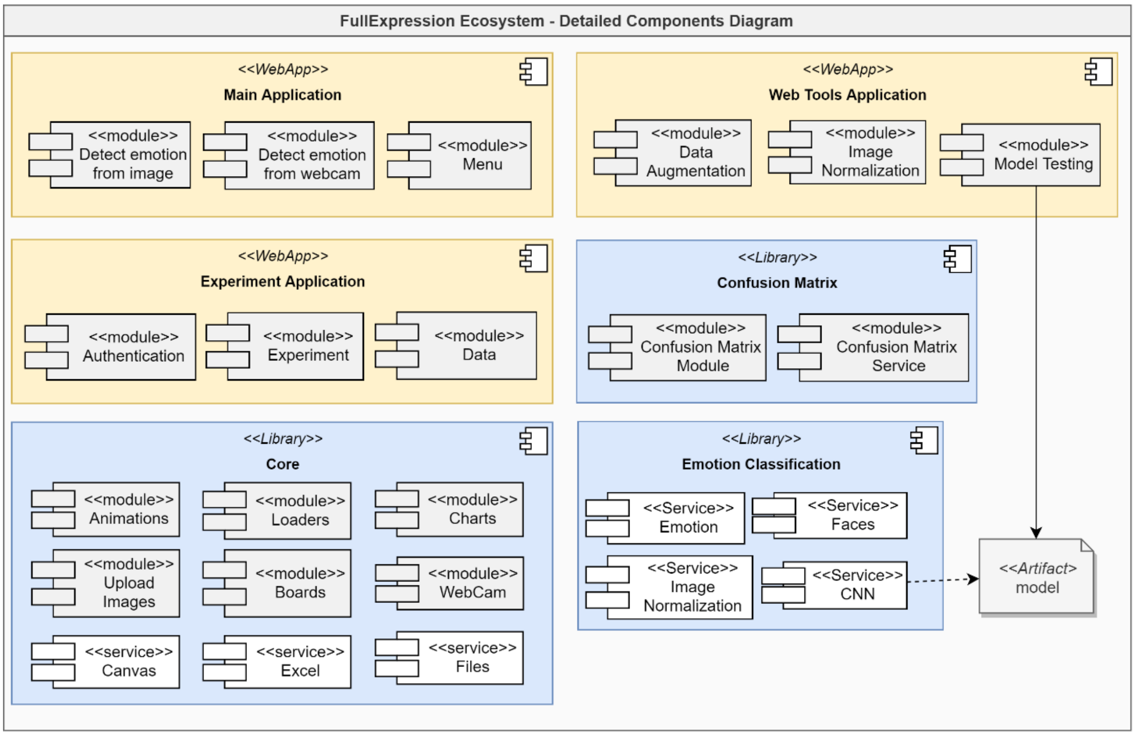


Figure 34 – UML Components diagram of the FullExpression Ecosystem

5.3 PROCESS VIEW

The process view describes the system processes and how they communicate. Additionally, it explores what needs to happen inside the system to obtain a specific result.

For all three software solutions and Training Script, this chapter will describe their process using UML activity and sequence diagrams.

5.3.1 FULLEXPRESSION APPLICATION

When the user accesses the FullExpression Application, the main application menu is presented having the possibility to access both the emotion analysis in real time, or the emotion analysis from images.

By choosing the first option, the system will load the model and show images obtained from the webcam. After that, the user can either access the detect faces or detect emotions functionalities, and for both features images are collected in 50 milliseconds time intervals. Also, at any time, the user can activate or deactivate both functionalities.

On the other hand, if the user chooses the emotion analysis from images, a set of images should be uploaded by the user, and the system will classify the emotions expressed on them and show the results in a report format. The user can interact with the report (in which the system will show more information), search information by image name or emotion, download a report (the system will create an excel file and download it) and download images organized by emotion (the system will divided the images into seven folders, representing the seven core emotions). Lastly, after the user has accessed to the emotion analysis in real time or the emotion analysis images, he can access to the main application menu, without the need of loading again the browser page.

The model is cachable, meaning each time the system tries to access it, the model can either be downloaded to the browser or be loaded from the browser cache. Moreover, the model is stored into the browser memory only one time per page load and, consequently, further model accesses will be made using the browser memory. Finally, the process of loading the model is not made on the page loaded, but just each time the model is needed.

The process of emotion classification is divided into loading the model, normalizing the images and classifying the emotion. Normalizing images means extracting faces from images, converting all faces to gray scale and resizing the images to 300x300 pixels in an effort of reducing noise information for improving the accuracy of results.

The FullExpression Application processes are also described on Figure 35.

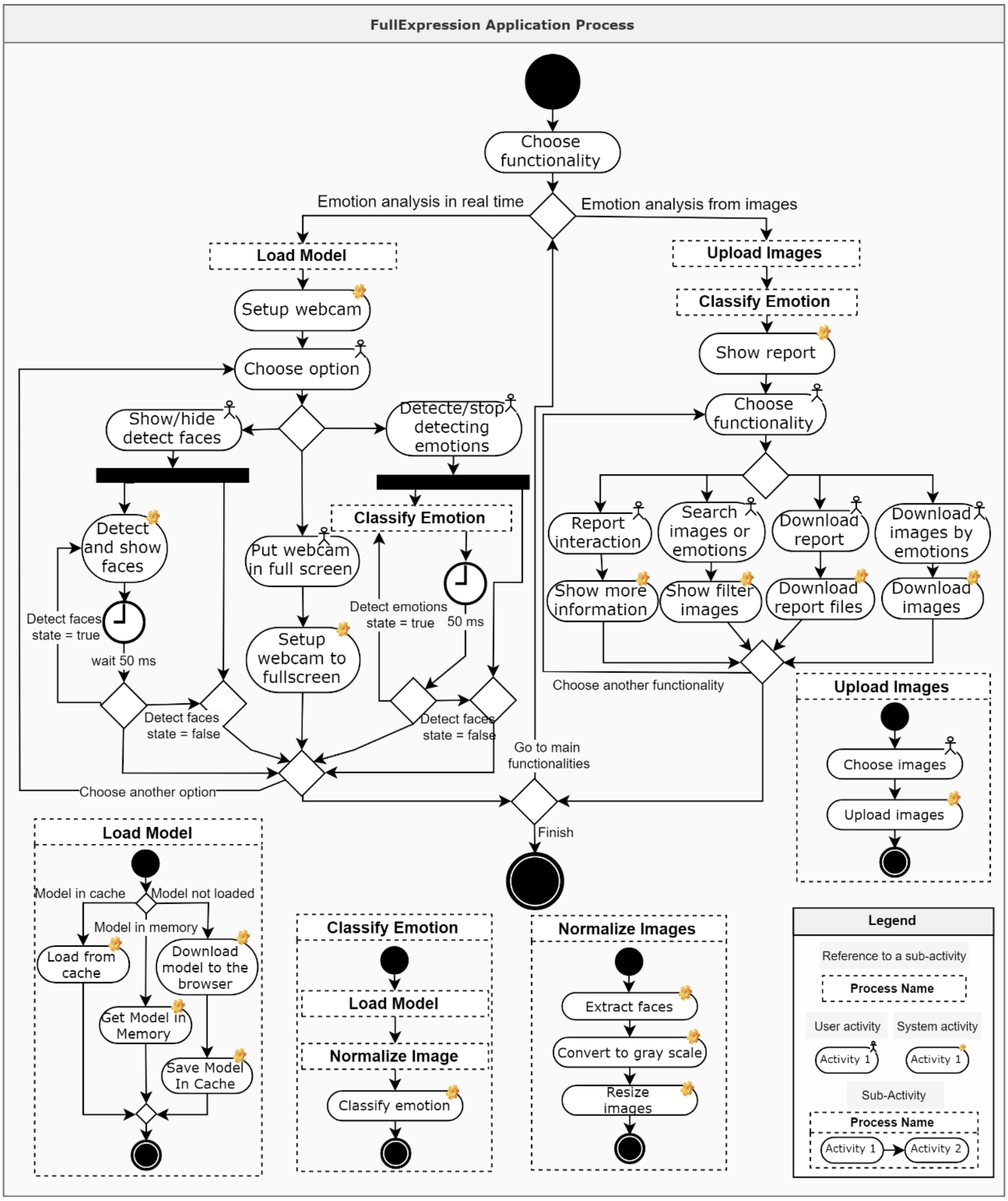


Figure 35 – FullExpression Application process represented with UML activity diagram

5.3.1 FULLEXPRESSION WEB TOOLS APPLICATION

The FullExpression Web Tools Application provides tools for helping the development process of training and testing DL models. The user can choose between normalizing images, test trained models or augment images.

By choosing the normalizing images feature, the user should set the final image size and upload the images to be used. The system will normalize and show the original images, the faces extracted, the images normalized, the number of uploaded images, the number of faces found and the number of faces per image. It can then download, not only the original images but also the faces and the normalized images. This feature guarantees the usage of the same image format on the training and predictive phases.

The user can also test the model by uploading images divided by the seven-core emotions. The system will then classify the emotion expressed for each image, build the confusion matrix, calculate and show statistics including precision, recall, F_{score}, average of accuracy for core emotions alone or altogether.

Additionally, the user can augment an image dataset by setting the total number of images, which should be greater than the number of uploaded images, and the system will copy, rotate, flip randomly some images and download the uploaded and copied.

The FullExpression Application processes are described on Figure 36.

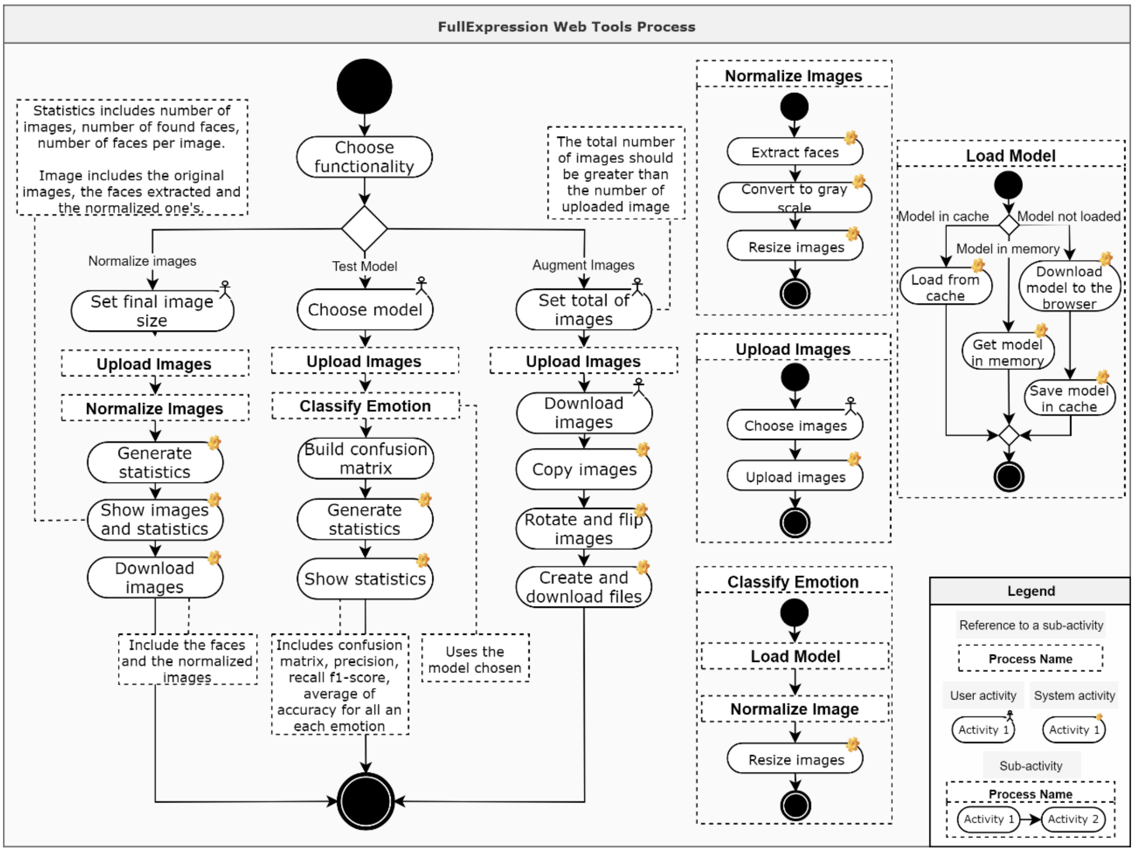


Figure 36 – FullExpression Web Tools process represented with UML activity diagram

5.3.1 FULLEXPRESSION EXPERIMENT APPLICATION

The FullExpression Experiment Application is responsible for the collection of emotion classification accuracy data of facial expression by human participants. The administrator can login in the system by inserting his credentials. Then the system will grant access to the application and either start a new experiment (by choosing an image database to be used) or download the participants' data. After the database selection, the participant should read the experiment instructions, and next, fill some personal information. Next, images will be shown to the participant, which will have to classify them in seven possible options (Afraid, Angry, Disgusted, Happy, Neutral, Sad and Surprised). After showing all images, a confusion matrix is built and saved in the database together with the time taken to complete the experiment. Lastly, the accuracy score of the test is calculated, the average score of all participants and the machine score are shown.

The FullExpression Application processes are described on Figure 37.

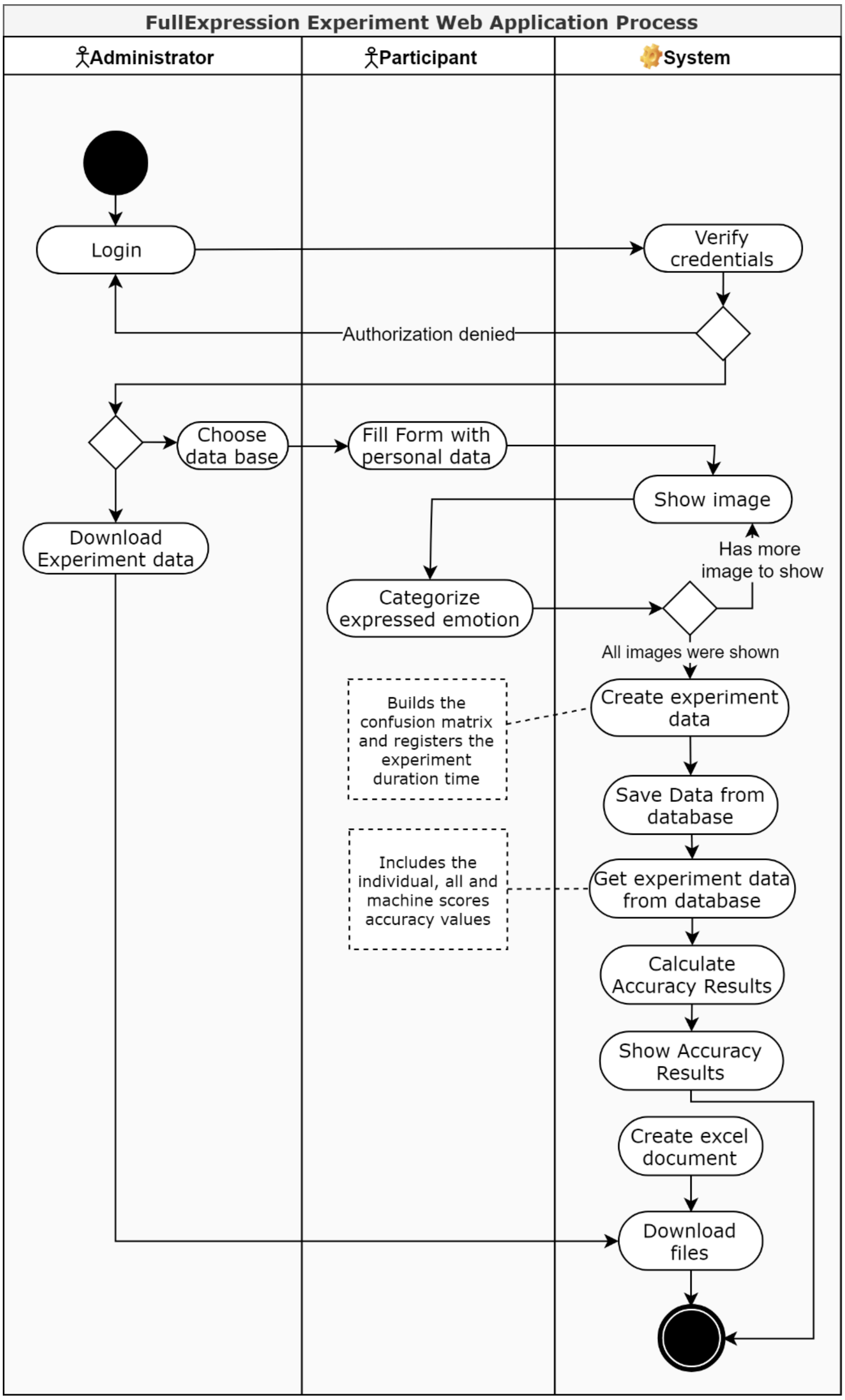


Figure 37 – FullExpression Experiment process represented with UML activity diagram

5.3.1 FULLEXPRESSION TRAINING SCRIPT

The FullExpression Training script is responsible for the implementation of an algorithm that automates the transfer learning technique process in order to train models. The training process is represented on Figure 38 and starts by setting the configurations (Figure 39), such as the training, validation and batch sizes, learning layers, epochs, learning rate, loss and optimizer functions. After that, the script will load the model and configurations, train the model according to the configurations loaded, producing the trained model in Keras and tensorflow.js formats, and show the confusion matrix generated by testing the model. Finally, the user can save in the disk the confusion matrix shown.

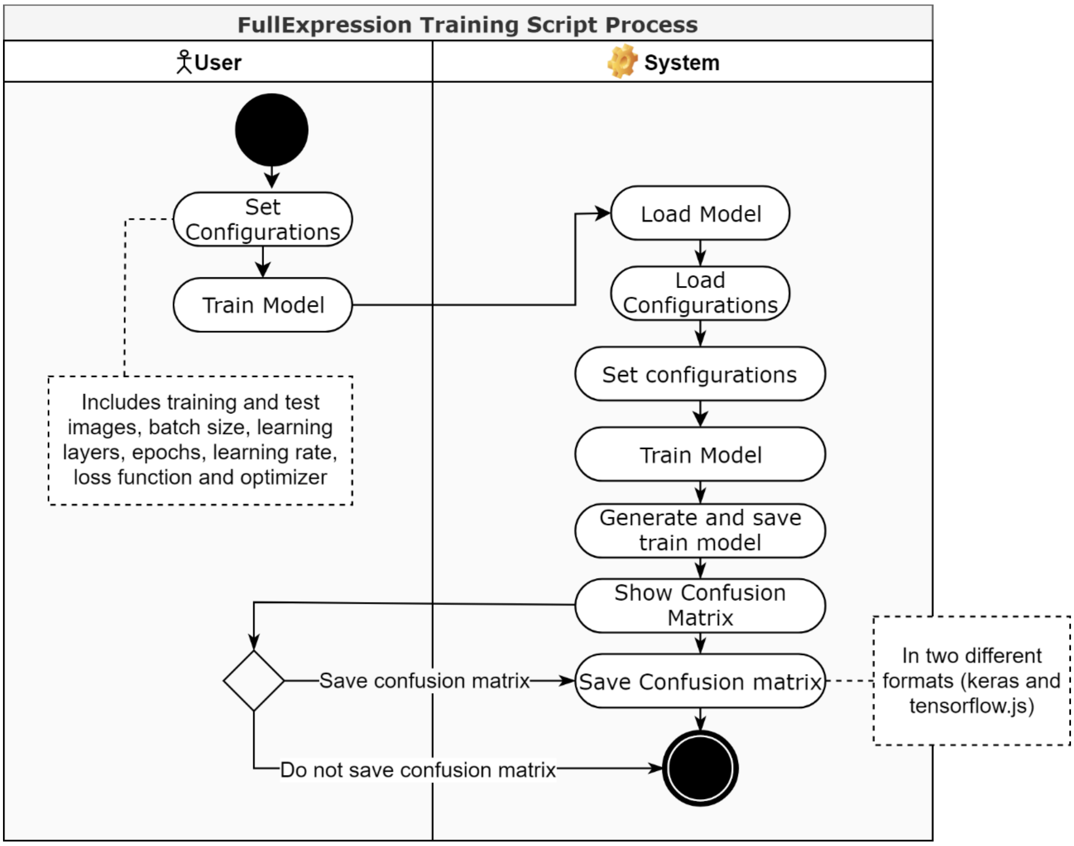


Figure 38 – FullExpression Training script process represented with UML activity diagram

```
{
    "categories": ["Afraid", "Angry", "Disgusted", "Happy", "Neutral", "Sad", "Suprised"],
    "train_path": "C:\\Theses_Repo\\train-cnn\\src\\images\\train",
    "valid_path": "C:\\Theses_Repo\\train-cnn\\src\\images\\valid",
    "test_path": "C:\\Theses_Repo\\train-cnn\\src\\images\\test",
    "train_batch_size": 10,
    "valid_batch_size": 10,
    "test_batch_size": 10,
    "learnable_layers": 1,
    "epochs": 20,
    "learning_rate": 0.0001,
    "loss_function": "categorical_crossentropy",
    "optimizer": "Adam",
    "model": "MobileNet_V1"
}
```

Figure 39 – Extract of code from a training configuration file

5.4 PHYSICAL VIEW

The physical view models the system's execution environment and maps software artifact into the hardware that hosts them.

As can be observed on Figure 40, the FullExpression Ecosystem has three web servers, one for each web application that hosts all artifacts necessary to run the application on the user browser. Moreover, the Experiment Application component communicates with a No SQL database in order to save the data collected from the experiments. Lastly, a device with a Python environment runs the Training Script.

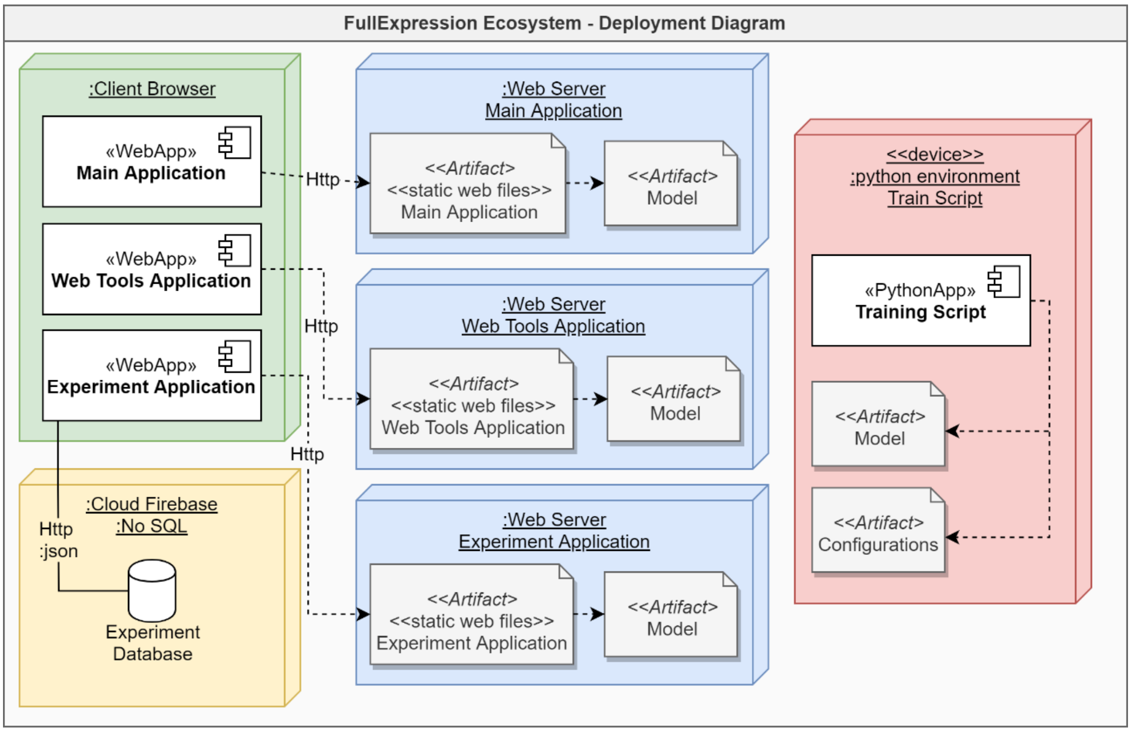


Figure 40 – FullExpression Ecosystem represented with UML deployment diagram

5.5 DEPLOYMENT VIEW

Deployment view describes the system's modules or components including packages, sub-systems and class libraries which are helpful to understand the system building-blocks. Also, we will explain how the storage of the source code is organized and the languages and technologies used to build the solutions.

For all four solutions, the code was stored on GitHub [167] using the Bitbucket platform [168], having seven different GitHub projects being wrapped in one big GitHub project, providing version control for each individual FullExpression component. The three web applications uses the Firebase [169] platform to deploy and host the website[2], manage databases and centralize the user authentication process.

[2] The FullExpression Application is hosted in https://fullexpressionwebtools.firebaseapp.com/, the FullExpression Web Tools Application in https://fullexpressionwebtools.firebaseapp.com/ and the FullExpression Experiment Application in https://full-expression-experiment.firebaseapp.com/

All web applications are single page applications (SPA), meaning once the user enters in the application, the page will not be reloaded again. This leads to better software performance and user experience. Moreover, the web applications are written in Angular 8 [124], Typescript [170] and SCSS [171], and uses NPM [172] for package management. On the other hand, the Training Script is written in Python [173] and uses Pip [174] for package management and virtualenv [175] to create a Python virtual environment (also known as container).

The three web applications, three libraries and script are stored in their own folder, inside the *FullExpression Workspace* folder. Also, each library and web application is divided in *src* (the application/library source code) and *dist* (the application/library compile code). Moreover, *src* folders can be divided into *assets* (contain artifacts such as images, fonts and DL models), *code* (contain artifacts such as typescript, html and scss files) and *images*. The Train Script has a different folder structure, where the *original models* folder contains the models used to be trained, *artifacts* generated the models, saved after the training process, and the source code, including the images used in the training process, the configuration file and the training script.

The FullExpression Ecosystem folder structure is described on Figure 41.

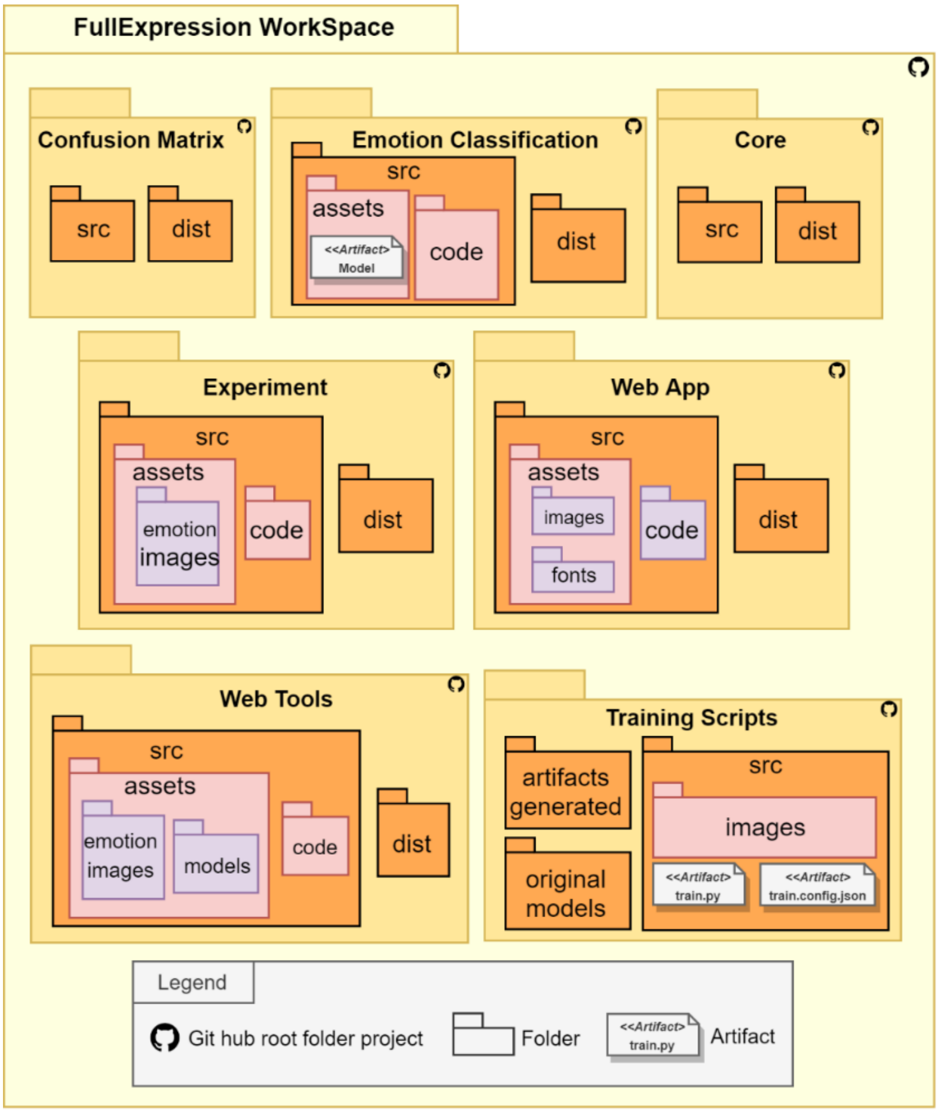


Figure 41 – FullExpression Ecosystem package diagram

5.6 DESIGN DECISIONS AND ALTERNATIVES

After understanding the software architecture in the 4+1 point of view, this chapter will be dedicated to expose some design decisions and alternatives, explaining the advantages and disadvantages of the main architecture.

By changing the library FullExpression Emotion Classification (responsible to classify emotions) to an http restfull service, an abstraction layer for the model will be provided, which will protect it from malicious reproduction, since on javascript libraries all artifacts would be downloaded to the browser and anyone could access them. The restfull API's also presents advantages, such as platform and language independence, separation of concerns, visibility, reliability and scalability. However, it requires more resources (such as hardware for hosting), code versioning is much harder and there is a communication latency problem. The decision was the creation of an Angular Library due to the lack of resources and the will of proving DL models that perform well on browsers and not only on servers or desktop solutions.

For the case of FullExpression Training Script, it could have been developed using javascript and then be integrated in the FullExpression Web Tools Application, opening the possibility to integrate the image normalization into the training model process, since the image normalization process is performed on the Web Tools Application and then the normalized images are used to train the model on the Training Script. Moreover, it would reduce the number of languages and technologies used, promoting code maintainability. Nevertheless, the Tensorflow and Keras for Python are more mature, with a big community of users, documentation, less bugs, frequent releases and better performance rating.

Most FullExpression Ecosystem applications are written in Angular and typescript, which are frameworks and languages popular for building web solutions. All code generates html, css and javascript files through a process called transpile code (similar to compile code). These files are static and can be executed offline by the browser, as all pure html, css and javascript applications do; or can be downloaded and executed each time the user accesses a given URL. Also, it does not have the typical client-side application architecture, since the business logic is implemented on the client-side code and the firebase database does not require a middle server for the communication between the application and data. This means less languages and frameworks in the system and less resources are needed, improving the ecosystem maintainability. Finally, all web applications are SPA, meaning the user interactions does not reload new pages improving the user experience.

5.7 EMOTIONS RECOGNITION SOFTWARE ENGINEERING PROCESS PROPOSAL

Based on previous research works, commercial solutions and this work, we can conclude that data is the most important aspect to achieve robust emotion classification. Therefore, it is important to use the correct technology to obtain data, according to the type of information we intend to obtain - physiological, behavior or expressive responses. Then, data can be organized by response and emotion type, datasets could be created, analyzed and stored. Also, software solutions could be made using the datasets created.

Figure 42 proposes an engineering process called "Emotion Software Creation Process", in which the goal is to provide a systematic process for building and maintaining big emotion classification solutions. Is composed by four phases in order to select and gather data for emotion types and create software solutions.

This process has four main steps, and the majority of them will be used in this work:

1. Emotion Selection: the domain of emotions has several emotion types. This is the first step of the process which is responsible to identify and choose emotions from the emotion domain in order to be used in the next step of the process;
2. Emotion Responses: each emotion as one or many types of human responses. They can be divided into physiological, expression and behavior. This step attributes each type of response to the corresponding emotion;
3. Gather Data: after studying the human responses, there are information that needs to be gathered, validated, analyzed and organized. In this step data will be gathered from each response and normalized in a way that could be added to a database. This allows the continuous feed of the emotion database with new data;
4. Create Software solution: with a dataset of emotions created and constantly fed, the software solutions can be made and improved.

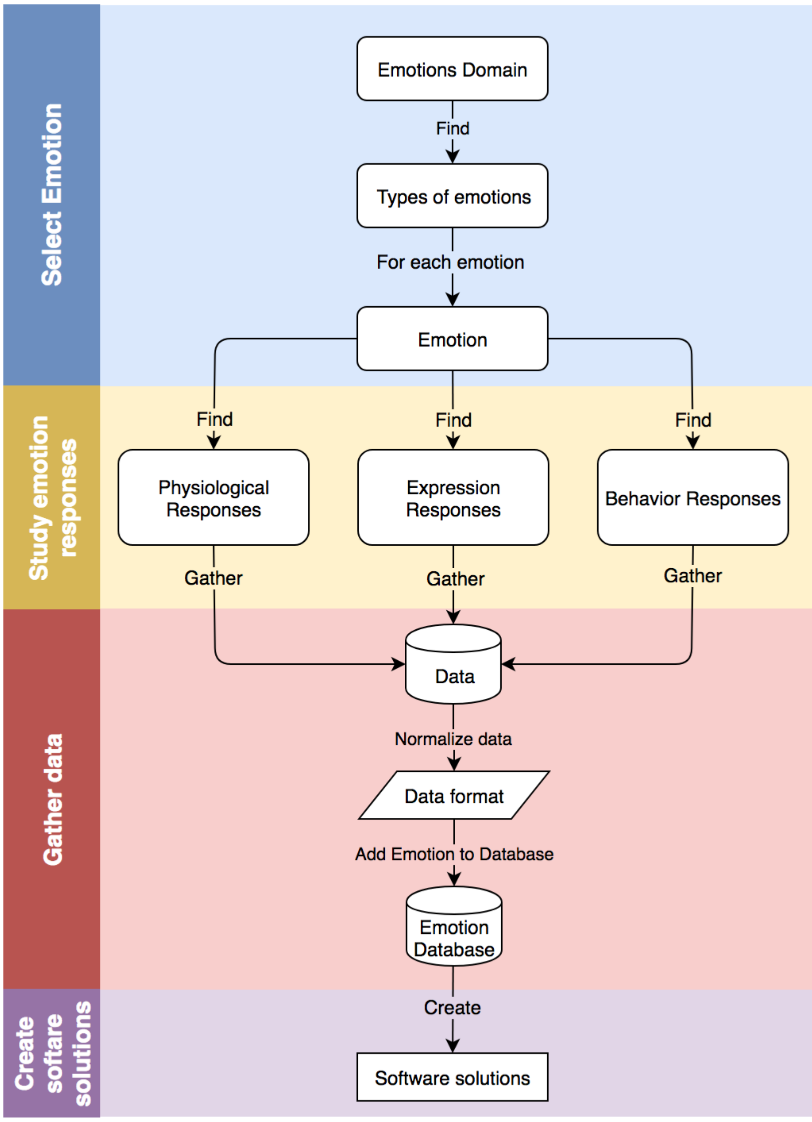


Figure 42 – Emotion Software Creation Process

5.8 CONCLUSIONS

FullExpression Application was the product developed in this work and contains all functionalities that allow users to generate, see and interact with emotions recognition either from uploaded images or from the webcam. In addition, two more web applications and a DL training script were added to the FullExpression ecosystem, in order to build and evaluate models as well as to collect preliminary data about human accuracy on recognizing emotions from facial expressions.

All web applications are driven by the separation of concerns (also know as modular architecture) promoting readability, maintainability, stability and reusage. They share a group of libraries, created for sharing functionalities across applications, and are prepared to be used by applications outside the FullExpression Ecosystem. They were designed to adapt the UI layout to different screen sizes and devices and obey to the single page application principal.

The FullExpression Ecosystem used a lot of different technologies: Angular 8, Typescript, Scss and Python as programing languages; NPM and Pip for package management; Github for control version purposes; Firebase to deploy and serve both databases and applications; nodeJS to run the applications and virtualenv to create Python containers.

Based on the experience acquired on developing those applications, an emotion recognition software engineering process was proposed which is divided in choosing the emotion, collection of data for each type of responses and using that data to build software solutions.

The next chapter will present the software evaluation and a comparation between preliminary data from human capabilities of recognizing emotions in images and the FullExpression Application.

$$H_0: \mu < 0.70$$

6. EVALUATION

This chapter will cover the software evaluation results, exposing the metrics and techniques used, the software accuracy and the results of the experiments made.

6.1 SOFTWARE ACCURACY

The accuracy evaluation will be expressed in a hypothesis form. The null hypothesis (H_0, formula 9) is “A computer program has an accuracy inferior to 70% on detecting emotions in facial expressions” and the alternative hypothesis (H_a, formula 10) is “A computer program can have an accuracy of 70% or more, on detecting emotions in facial expressions” defined by:

$$H_0: \mu < 0.70 \quad (9)$$

$$H_a: \mu \geq 0.70 \quad (10)$$

A database containing 4699 images was created, from which 3769 (~80%) were used for training and 930 (~20%) to evaluate the model. This database combines images from KDEF & AKDEF [176] (~62%), TFEID [177] (~11%), Face_Place [178] (~22%) and jaffe [149] (~5%) databases.

The software evaluation was performed against images from each individual database that was not used during the DL model training process. The Web Tools Application was responsible not only for creating a confusion matrix, but also for calculating metrics such as accuracy, precision, recall and F_{score}.

As can be observed in Figure 43, on average, the software classified correctly 78.44% of the total images contained facial expressed emotions, which confirmed the alternative hypothesis. Happy (97.80%), disgusted (88.60%), surprised (86.30%) and neutral (81.70%) had highest accuracy ratings comparing to sad (73.10%), angry (69.10%) and afraid (62.50%). In addition, the accuracy results obtained for each core emotion varied from database to database (Figure 44). For KDEF & AKDEF, afraid emotions presented the lowest accuracy rating value, while the same emotion for TFEID database is positioned in the top 3 of highest accuracy ratings (Figure 44). Nevertheless, common to all databases, happy and disgusted are the top 2 emotions with better accuracy ratings. Overall, the software classified correctly ~85% of the total images from KDEF & AKDEF database, ~79% for TFEID database, ~74% for jaffe database and 43% for Face_Place database (Figure 44).

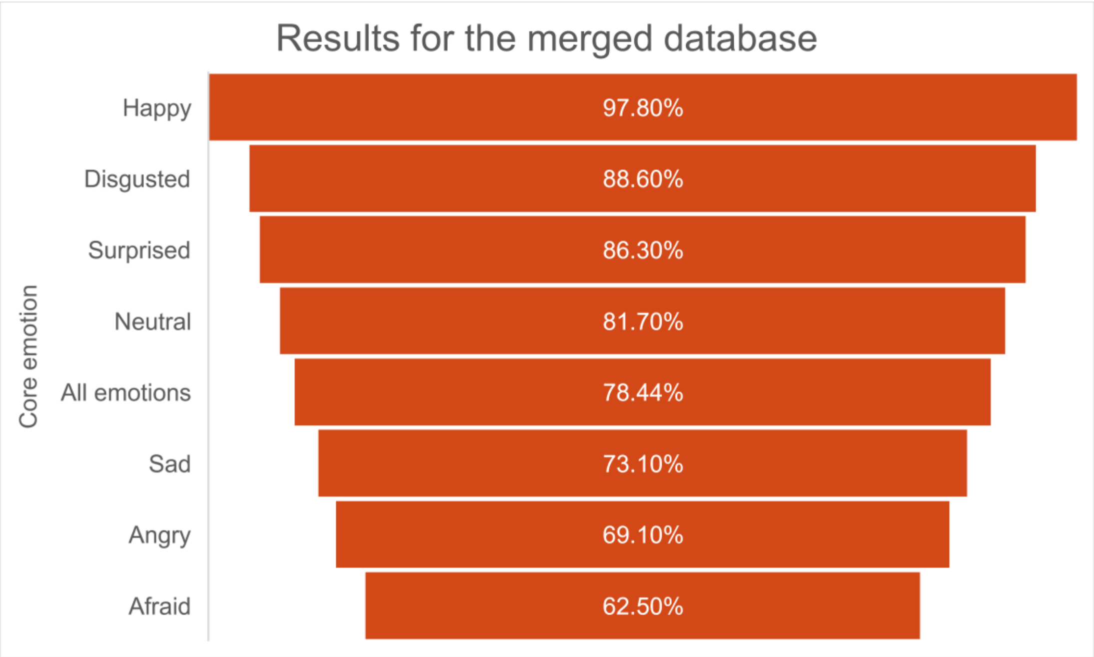


Figure 43 – Software accuracy results

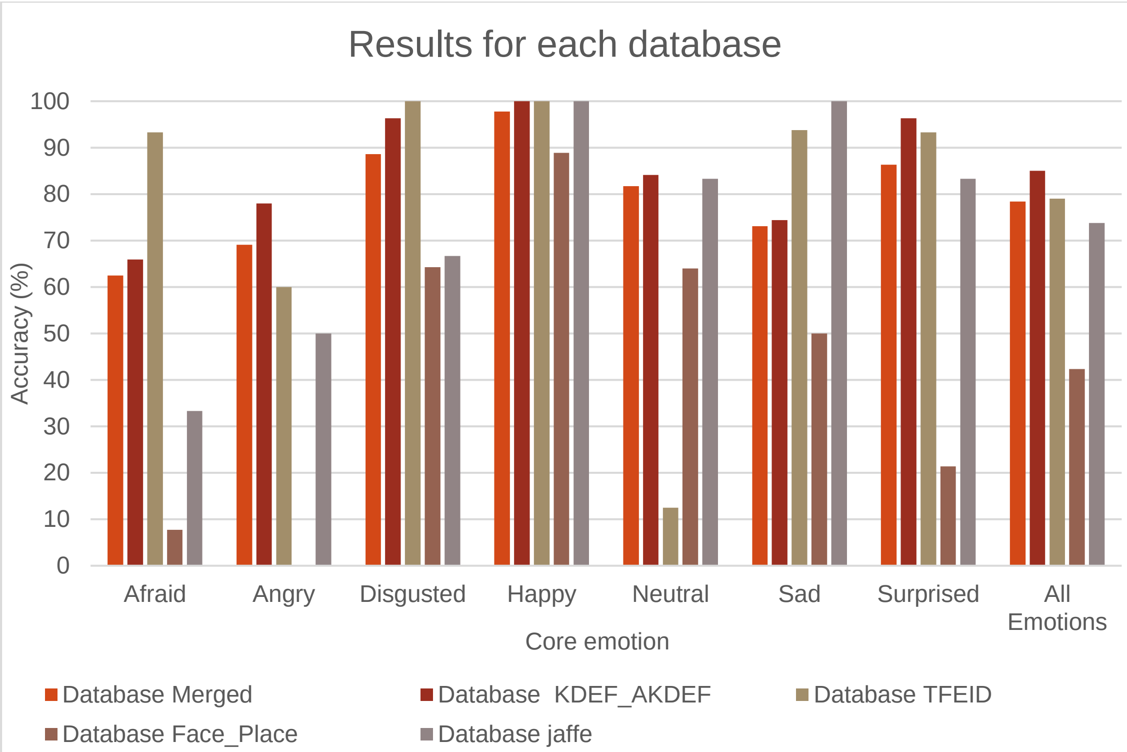


Figure 44 – Software accuracy for different emotion databases

6.2 EXPERIMENTS AND RESULTS

In total, three experiments were made to assess how to train and evaluate a DL model, as well as to compare the software accuracy results with human facial expression emotion recognition capabilities. The experimental processes and results will be exposed in the next sections.

6.2.1 FINE-TUNING

The purpose of the Fine-Tuning experiment was to understand the training configuration combination that obtained the best accuracy rates. The learnable layers, learning rate, epochs and training batch size configurations were evaluated by training MobileNet V1 against the KDEF & AKDEF database (80% for training and 20% for testing).

All training models shared the same configurations (Table 13), changing only the parameter that was tested. The same computer was used in all experiments to ensure the same conditions.

Table 13 – Default configuration values for Fine-Tuning Experiment

Learnable Layers	Learning Rate	Epochs	Train Batch Size
60%	0.0001	4	10
Optimizer	**Model**	**Database**	**Loss function**
Adam	MobileNet V1	KDEF & AKDEF [176]	category_crossentropy

For testing the Learnable Layers parameters, ten trainings tests were performed between [10%, 100%] interval of values, representing the percentage of learnable layers. The learnable layers are set from the last layers to the first layers of the model.

As can be observed on Figure 45 and Figure 46, more trainable layers are associated with increased training time, yet better accuracy results are also obtained. Moreover, the best accuracy value was provided by training all model layers.

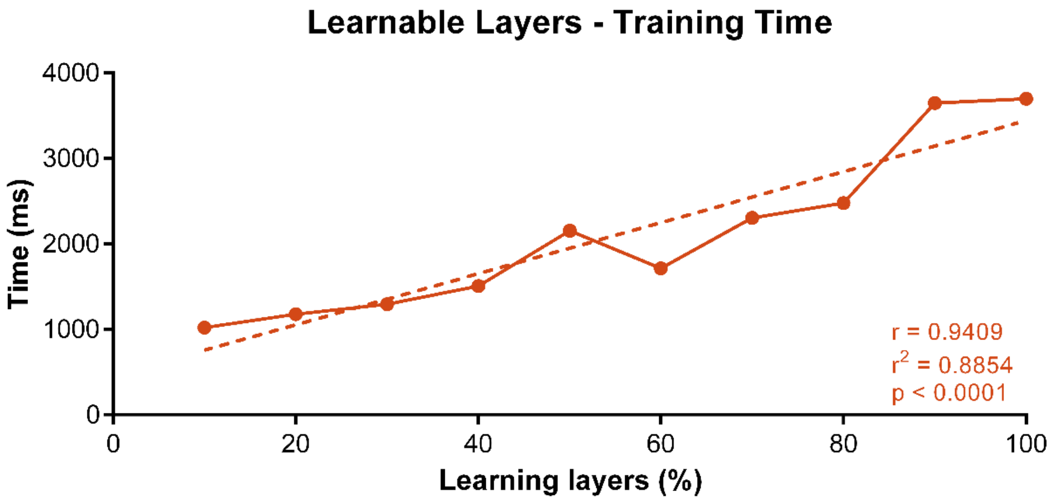


Figure 45 – Time taken to train a model versus the number of learnable layers

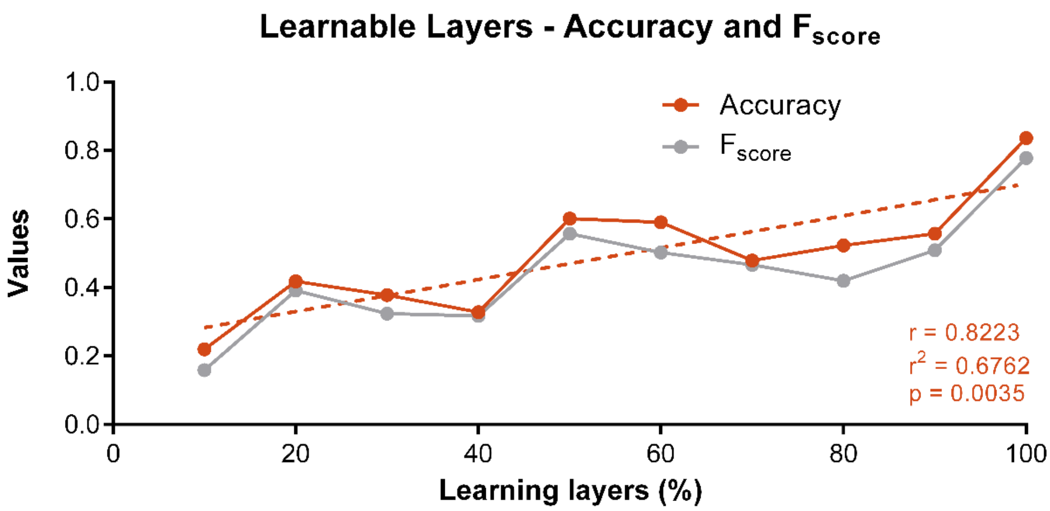


Figure 46 – Line chart which correlates the trained model accuracy and F_{score} values with the number of learnable layers

All learning rate values tested presented similar training times, with the exception of “0.00001” (Figure 47). For accuracy (Figure 48), 0.0001 seems to be the best learning rate value, since the accuracy values decrease for higher or lower learning rate values.

Figure 47 – Line chart which correlates the time taken to train a model with the Learning Rate Value

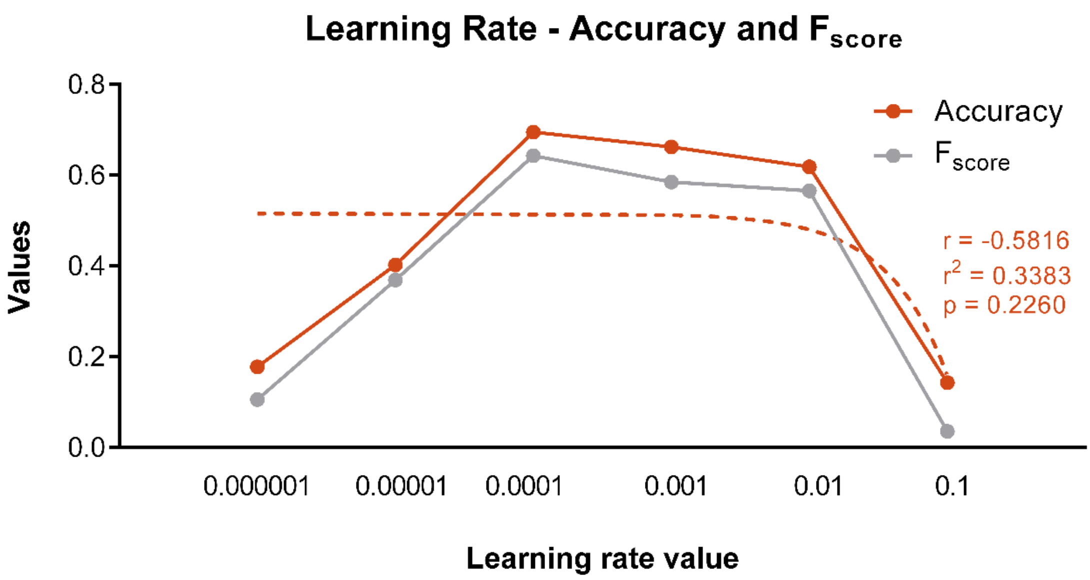


Figure 48 – Line chart which correlates the trained model accuracy and F_{score} values with the Learning Rate Value

For the epochs experiments (Figure 49 and Figure 50), more epochs are associated with increased training time, but after 2 epochs the accuracy value does not improve at all; it even decreases after 8 epochs. Thus, it seems the best epoch value is between 2 and 4.

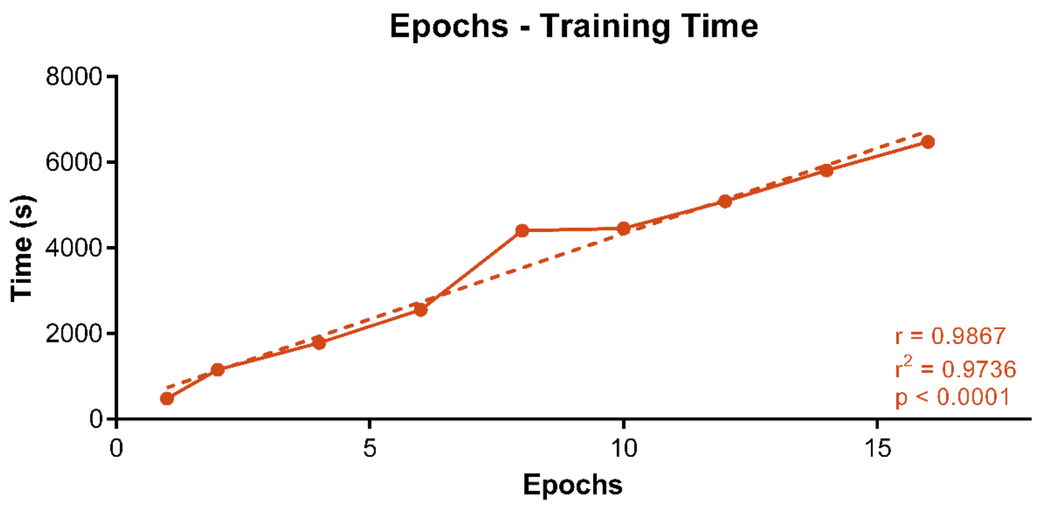


Figure 49 – Correlates the time taken to train a model with the number of epochs

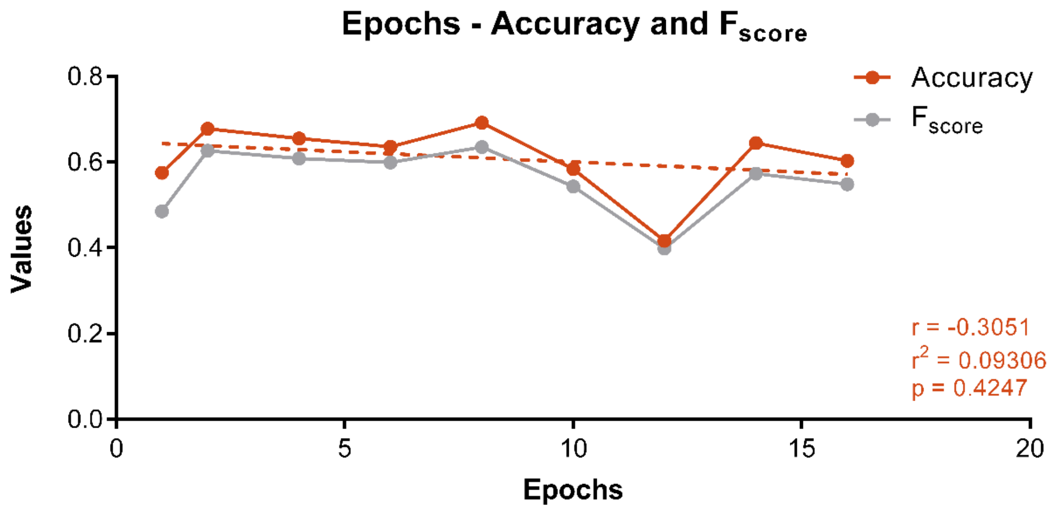


Figure 50 – Line chart which correlates the trained model accuracy and F_{score} values with the number of epochs

Evaluating the training batch size experimental results (Figure 51 and Figure 52), the higher is the training batch size the lower are the accuracy results. Evaluating the training batch size, does not seem to influence the training time, nor the accuracy and F_{score} values.

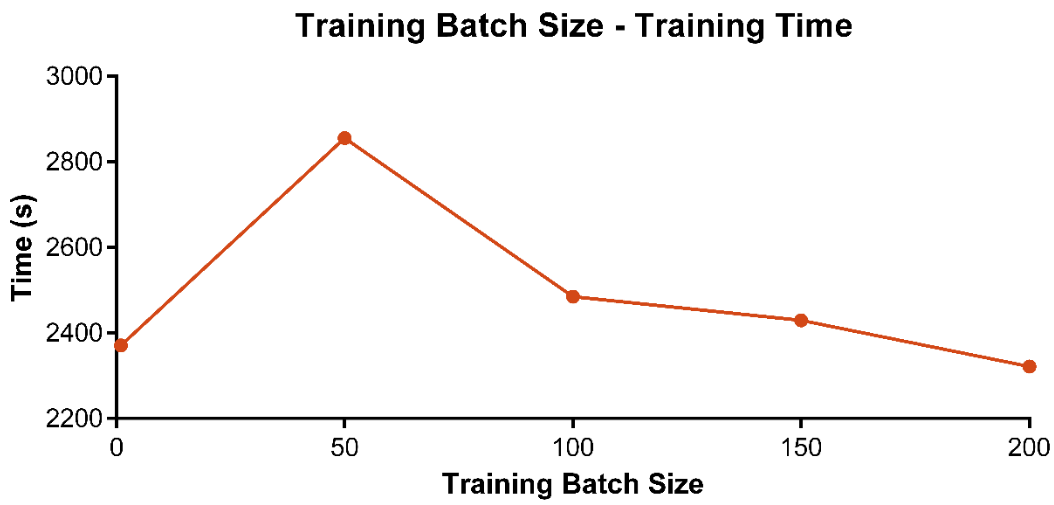


Figure 51 – Line chart which correlates the time taken to train a model with the Training Batch Size

Subsequently, to have the best training model configurations, the values chosen were: 100% for learnable layers, 0.0001 for learning rate, a value between 2 and 4 for the number of epochs and a value between 1 and 50 for the training batch size (Table 14).

Table 14 - Best configuration values for training the DL model

Learnable Layers	Learning Rate	Epochs	Training Batch Size
100%	*0.0001*	*[2-4]*	*[1-50]*

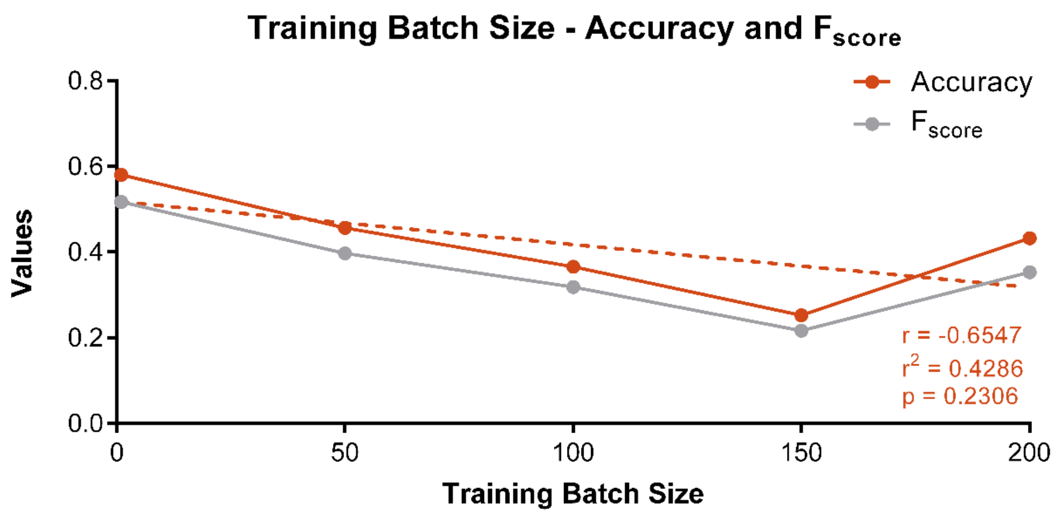


Figure 52 – Correlates the trained model accuracy and F_{score} values with the Training Batch Size

6.2.2 IMPORTANCE OF DATA

In order to understand the importance of data, an experiment was conducted: three models were trained against first the KDEF & AKDEF database (2295 images for training and 574 for testing), next with TFEID (404 images for training and 101 images for testing) and finally with both of them.

Then, for each model the accuracy results were tested against both databases, and the results can be observed on Figure 53. All experiments were trained using the same computer and with the same training model configurations (Table 15) in order to maintain the same conditions. Moreover, the images used to train and test the models were different, assuring the tests could be made with images which the model had not seen before.

Table 15 – Importance of data experiment configuration settings

Learnable Layers	Learning Rate	Epochs	Train Batch Size
100%	0.0001	15	10
Optimizer	**Model**		**Loss function**
Adam	*MobileNet V1*		*category_crossentropy*

After the experiment analyses, the results showed that:

1. The larger the training database was, the better the model would perform against images significantly different for those used to train the model, since the KDEF & AKDEF is significantly bigger then TFEID (approximately 5 times more in number of images) and the model trained with KDEF & AKDEF performed 18.44% better when tested against TFEID compared to the model trained with TFEID when tested against KDEF & AKDEF;
2. The bigger the difference between the training and testing images, the worst the model would perform, since the images from KDEF & AKDEF and TFEID databases are significantly different and when used separately performed approximately 41% worst than used together for training the model.

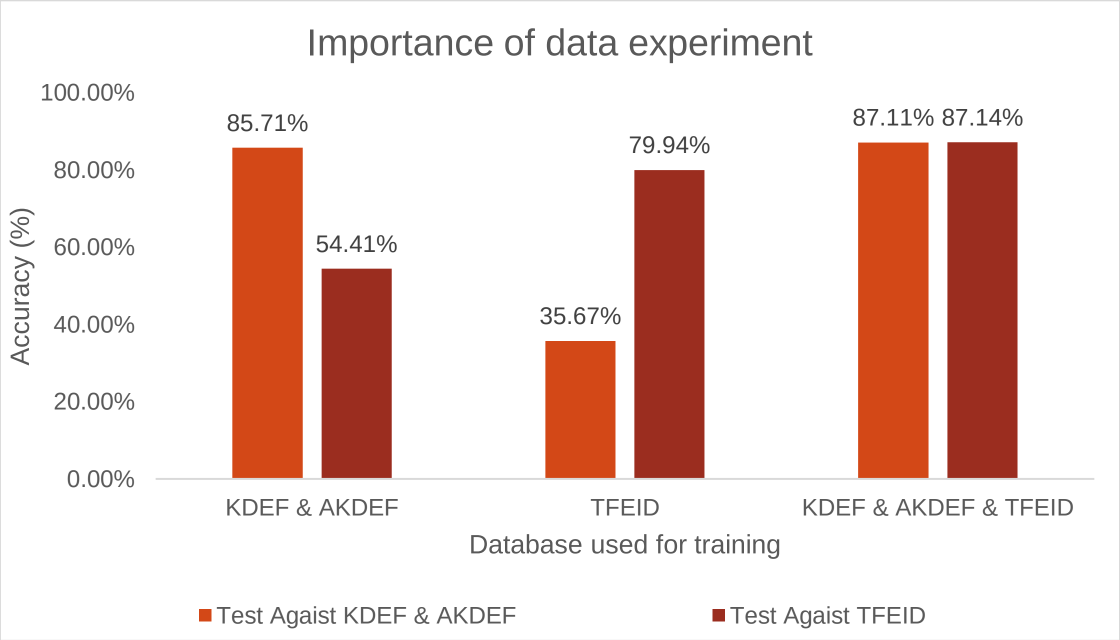


Figure 53 – Importance of data experiment results

6.2.3 COMPARISON BETWEEN SOFTWARE AND HUMAN PERFORMANCE

In order to compare the software and human capability to classify emotions from facial expressions, an experiment was performed using the FullExpression Experiment Application. In this task, images containing faces expressing emotions were shown to different people (participants). For each image, 7 options to classify emotions were given (Afraid, Angry, Disgusted, Happy, Neutral, Sad and Surprised), and the participant had to choose which emotion he thought was presented (for more information regarding software workflow see chapter 5.1.3 - FullExpression Experiment Application).

A total of 24 participants (Table 16) were included in the experiment, and all of them classified 2 different image databases, namely KDEF & AKDEF and TFEID databases. On average, people took 7.25 minutes to conclude the classification of KDEF & AKDEF images and 7.21 minutes to classify TFEID images.

Table 16 – Description of participant sample

Women:Men ratio	16:31
Age average (age interval)	29.7 (21-46)
Average number of years of formal education (years interval)	16.9 (12-22)

The average accuracy results for each core emotion, and the average of all emotions, are presented in Figure 54 (for KDEF_AKEF database) and Figure 55 (for TFEID database), with each circle representing the accuracy value of one person, the black and gray lines the participant average accuracy and the green line the software accuracy.

Using KDEF & AKDEF images, the software classified afraid, angry and sad emotions with approximately the same accuracy as the participants, being better for the remaining emotions. On average, the software was 12.47% more accurate than the participants. Using the TFEID images, the software classified better all emotions, expect the neutral emotion, being, on average, 6.42% more accurate then participates.

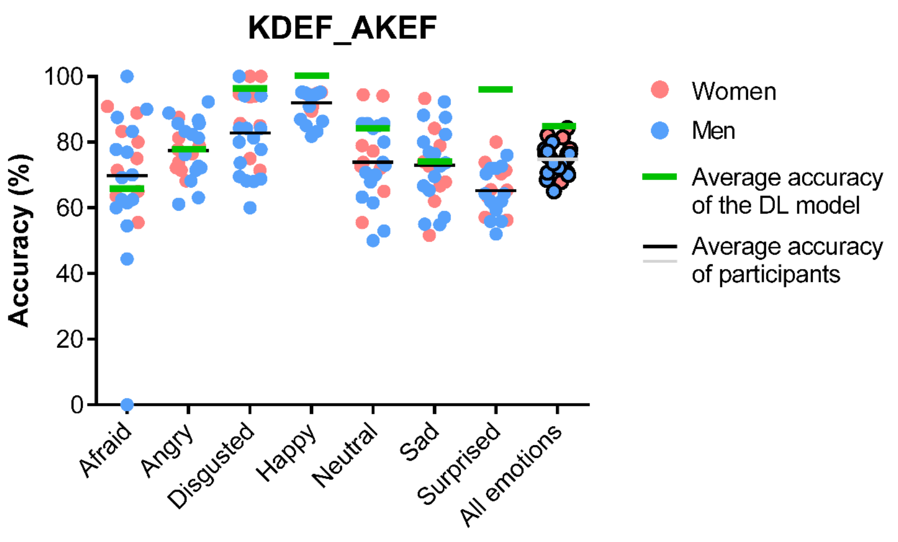


Figure 54 – Participant accuracy for KDED_AKEF database

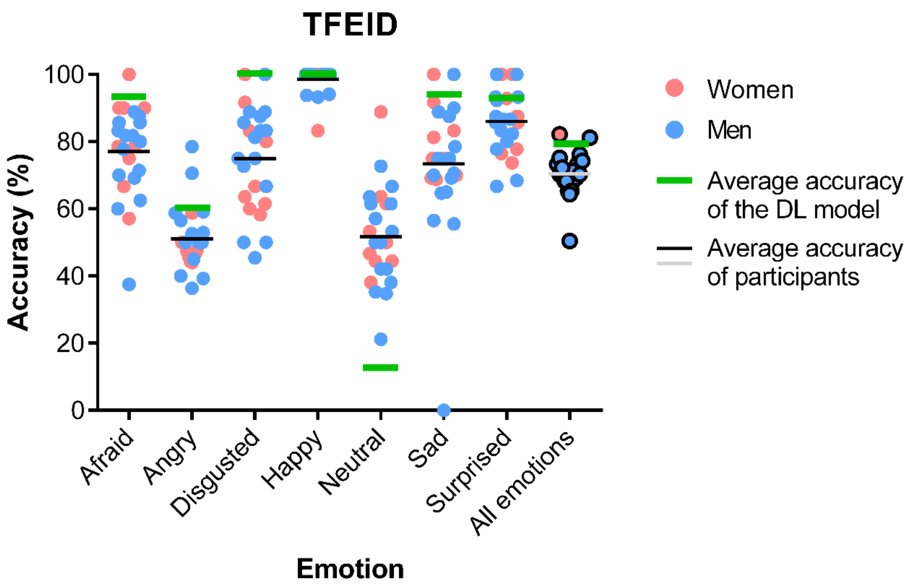


Figure 55 – Participant accuracy for TFEID database

6.3 CONCLUSIONS

The hypothesis formulated was confirmed: a software program can have an accuracy of 70%, or more, detecting emotions in facial expressions. To do so, 4699 images were used to train (80%) and evaluate (20%) the DL model consumed by the software. The software had an accuracy value of 78.44% with highest values for happy, disgusted, surprised and lowest values for sad, angry and afraid.

The transfer-learning technique was applied on MobileNet_V1. The ideal configuration variances depended on the model and data used in the training process. In order to find best training parameters, the KDEF & AKDEF database was used and by analyzing the experimental data it was possible to conclude that for MobileNet_V1 and KDEF & AKDEF database all layers of the model should be trained, the learning rate value should be 0.0001, the number of epochs should be a number between 2 to 4 and the training batch size should be between 1 and 10.

With the intention of understanding the importance of data in the DL models, an experiment was made, which consisted of training the DL model against 2 different databases and evaluating if the model performer better when trained against 2 image datasets instead of just one. The results were clear: the model performed 41% better when training with both databases, instead of trained against only one. This means that the more data is provided to the model, the better it will perform.

In order to compare how well the software classifies facial emotions compared to people an experiment was designed: a set of images containing facial expressions were presented to participants, which had to classify them in one of seven core emotions. This preliminary data suggests that the software is approximately 6% better than the participants on facial emotion classification.

7. DISCUSSION

Emotions have an important role on building complex societies by providing an effective channel of communication. They can be shown through expressive, physiological and behavioral responses and, frequently, in an unconscious way. Industry is struggling for solutions that can analyze human emotions, in an attempt to provide better user experiences. Since it is extremely difficult to classify emotions, it is imperative to understand which emotions are easier to categorize. With the purpose of describing the most distinguishable emotions, Paul Ekman proposed a division in seven core emotions (Afraid, Angry, Disgusted, Happy, Neutral, Sad and Surprised), and those are the emotions which FullExpression Application classifies.

Previous works mixed ML algorithms with FACS extraction algorithms in order to classify facial expressions with the Paul Ekman core emotions. They presented good accuracy results, but few use DL solutions, which proved to perform well on image classification. Moreover, none of the works analyzed used ML and DL models on browser. In this work, besides using DL solutions, we also developed our software for browser usage.

With the purpose of using DL models to classify emotion on images, a web application was created. Thus, The FullExpression project demonstrated not only the possibility to run DL models on browsers, which opens new possibilities to the ML community, but also the possibility of creating emotion classification software based on DL models.

To create a DL model specialized in detecting emotions, a technique called transfer-learning was used. This technique allowed the training of models much faster than building and training a new DL model from scratch. Since the FullExpression Application provides emotion image classification in real time, a model which combines high accuracy values with low classification speed times was required, leading to the choice of MobileNet V1. And, with the goal of training and evaluation the DL model, a training script and a web application were created.

Data has an important role on DL models performance. Several emotion databases were combined, organized and normalized (by resizing images and removing color) using the same algorithm that is used by the FullExpression Application.

DL model fine tuning experiments, using a transfer learning technique, showed that, for MobileNetV1, the number of trainable layers and the amount of data available had a big impact on the model accuracy but consumed more time in the training process.

The initial hypothesis was that a computer program could have an accuracy of 70% or more on detecting facial emotion expressions, which was verified by achieving an accuracy of 78.44%. From the seven core emotions, the software was better at classifying happy, disgusted, surprised and neutral facial expressions. Moreover, preliminary data from an experiment with 24 participants concluded that the software was between 5.78% and 5.89% more accurate on detecting emotions from images than the humans.

In this work, it was possible accomplish the purposed goals, namely to identify the core emotions and the most distinguishable ones; analyze the FullExpression project from a business perspective; understand how AI and DL techniques could be helpful for emotion recognition and classification; study different research works and commercial solutions; study the impact of data and model fine-tuning, in order to answer AI problems; gather, explore and use several emotion databases to train and evaluate DL models; create a web application which classifies emotions from facial expressions and works in different screen sizes and devices; create an ecosystem of tools to build and evaluate DL models; run DL models exclusively on browser; implement a modular architecture which allows code sharing not only between FullExpression applications but also with external applications; create a web application, which allows to collect preliminary data about human capabilities of recognizing emotions from facial expressions; and provide online access for most applications created.

Moreover, the work presented can be useful in many different domains, as the perception of the emotional state of users can be determinant in:

- Marketing, to understand the effect of marketing campaigns;
- Health: in elderly people monitoring applications, mental diseases diagnosis, interactive applications to monitor patients' behavior in hospitals environments, among others;
- Industry 4.0: to monitor human collaborating with robots (cobots), making it possible to detect safety problems, such as fatigue;
- Critical infrastructures operators: to monitor their attention levels, to support them in tiredness situations, among others.

7.1 FUTURE WORK

More work could be done to improve the FullExpression ecosystem. The web application only classifies emotions from facial expressions, but humans also produce emotions through physiological and behavioral responses. Data from these different ways of expressing emotions, could be collected and used to improve accuracy results. Nonetheless, analyzing different human responses are not the only option to improve the machines accuracy: as seen before, data quantity, quality and diversity has an important role. Thus, software solutions could be created to collect data not only from facial emotion expressions but also from other forms of emotional responses. Those solutions should be appealing in order to convince people to use them. This way, games seems to be an effective way of exchanging data and time for fun, similarly to what is being performed by The Cognition Project [179], a web tool composed by several mini games responsible for the collection of data for cognitive training researches. After the creation of those solutions, a software pipeline could be created which would automate the process of training models: the data collected should be stored and, from time to time, used to produce normalized data and to train ML models.

Also, there are some features that could be implemented to improve the FullExpression Application usability: the web application does not generate reports when detecting emotions on real time, and does not allow users to upload or interact with saved reports in the same way as the system does when generating the reports for the first time. Moreover, a functionality to collect images uploaded by users (with proper authorization) could be made which would help to increase the number of images used to train the models.

In order to detect faces, the Viola-Jones algorithm was used which is well known as one of the fastest and accurate face detection algorithms. Nevertheless, DL models could also perform well on detecting faces (e.g. VGG-Face [180]) and further research should be made with the intention of comparing performance of DL models with the Viola-Jones algorithm, particularly in cases where the people are using glasses, hat or any other objects.

In the design alternatives it was discussed the creation of a service which could be consumed by external applications, allowing them to analyze emotions from images. For future work, this service could also be created to provide an API that is language and framework independent and allows devices with low performance resources to run the software (since running the model is computationally heavy and some devices could have issues running it). In this case, the model would run in a server machine, with better hardware characteristics, and would allow to run heavy and more accurate models without compromising the loading times.

8. REFERENCES

[1] D. S. Massey, “A Brief History of Human Society: The Origin and Role of Emotion in Social Life: 2001 Presidential Address,” *Am. Sociol. Rev.*, vol. 67, no. 1, pp. 1–29, Feb. 2002.

[2] D. Derks, A. H. Fischer, and A. E. R. Bos, “The role of emotion in computer-mediated communication: A review,” *Comput. Human Behav.*, vol. 24, no. 3, pp. 766–785, 2008.

[3] L. T. Lam and S. L. Kirby, “Is Emotional Intelligence an Advantage? An Exploration of the Impact of Emotional and General Intelligence on Individual Performance,” *J. Soc. Psychol.*, vol. 142, no. 1, pp. 133–143, Feb. 2002.

[4] S. Amarakeerthi, R. Ranaweera, and M. Cohen, “Speech-based emotion characterization using postures and gestures in CVEs,” *Proc. - 2010 Int. Conf. Cyberworlds, CW 2010*, pp. 72–76, 2010.

[5] N. S. Consedine and J. T. Moskowitz, “The role of discrete emotions in health outcomes: A critical review,” *Appl. Prev. Psychol.*, vol. 12, no. 2, pp. 59–75, Nov. 2007.

[6] G. Venture, “Human characterization and emotion characterization from gait,” *2010 Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. EMBC’10*, pp. 1292–1295, 2010.

[7] D. . Hockenbury and S. . Hockenbury, *Discovering psychology*. New York: Worth Publishers, 2007.

[8] B. C. Ko, “A brief review of facial emotion recognition based on visual information,” *Sensors (Switzerland)*, vol. 18, no. 2, 2018.

[9] R. W. Picard, “Affective Computing,” Cambridge, 1995.

[10] D. Gershgorn, “The data that transformed AI research—and possibly the world,” 2017. [Online]. Available: https://qz.com/1034972/the-data-that-changed-the-direction-of-ai-research-and-possibly-the-world/. [Accessed: 11-Nov-2018].

[11] G. Seif, “I’ll tell you why Deep Learning is so popular and in demand,” *Medium: The Startup*, 2018. [Online]. Available: https://medium.com/swlh/ill-tell-you-why-deep-

learning-is-so-popular-and-in-demand-5aca72628780.

[12] S. Mahapatra, “Why Deep Learning over Traditional Machine Learning?,” *Towards Data Science*, 2018. [Online]. Available: https://towardsdatascience.com/why-deep-learning-is-needed-over-traditional-machine-learning-1b6a99177063.

[13] W. James, “What is an emotion?,” *Mind*, vol. 9, no. April, pp. 188–205, 1884.

[14] K. Cherry, “The James-Lange Theory of Emotion,” 2018. [Online]. Available: https://www.verywellmind.com/what-is-the-james-lange-theory-of-emotion-2795305. [Accessed: 11-Nov-2018].

[15] K. R. Scherer, “What are emotions? and how can they be measured?,” *Soc. Sci. Inf.*, vol. 44, no. 4, pp. 695–729, 2005.

[16] K. Cherry, “Emotion sand Types of Emotional Responses,” 2019. [Online]. Available: https://www.verywellmind.com/what-are-emotions-2795178. [Accessed: 10-Jul-2019].

[17] C. Bluhm, *Handbook on facial expression of emotion*, no. November. 2013.

[18] M. Donaldson, “A Plutchik ’ s Wheel of Emotions – 2017 Update,” 2018. [Online]. Available: https://www.6seconds.org/2017/04/27/plutchiks-model-of-emotions/. [Accessed: 11-Nov-2018].

[19] Hokuma, “The Emotion Wheel: What is It and How to Use it?,” 2017. [Online]. Available: https://positivepsychologyprogram.com/emotion-wheel/. [Accessed: 02-Aug-2019].

[20] P. Ekman and W. V. Friesen, “Measuring facial movement,” *Environ. Psychol. nonverbal Behav.*, vol. 1, no. 1, pp. 55–75, 1976.

[21] K. Cherry, “How Many Human Emotions Are There?,” 2018. [Online]. Available: https://www.verywellmind.com/how-many-emotions-are-there-2795179. [Accessed: 18-Nov-2018].

[22] J. B. Avey, T. S. Wernsing, and F. Luthans, “Can positive employees help positive organizational change?: Impact of psychological capital and emotions on relevant attitudes and behaviors,” *J. Appl. Behav. Sci.*, vol. 44, no. 1, pp. 48–70, 2008.

[23] N. Mendes, “How do emotions affect productivity?,” 2017. [Online]. Available: https://www.atlassian.com/blog/software-teams/new-research-emotional-intelligence-in-the-workplace. [Accessed: 12-Aug-2018].

[24] J. S. Lerner and L. Z. Tiedens, “Portrait of the angry decision maker: how appraisal tendencies shape anger’s influence on cognition,” *J. Behav. Decis. Mak.*, vol. 19, no. 2, pp. 115–137, Apr. 2006.

[25] A. M. Isen and B. Means, “The Influence of Positive Affect on Decision-Making Strategy,” *Soc. Cogn.*, vol. 2, no. 1, pp. 18–31, Mar. 1983.

[26] S. & P. S. Cohen, “Positive Affect and Health,” *J. Psychol. Sci.*, vol. 15, no. 3, pp. 122–125, 2006.

[27] P. Trans, R. S. Lond, and B. L. Fredrickson, “The broaden − and − build theory of positive emotions The broaden-and-build theory of positive emotions,” no. September, 2004.

[28] M. M. Tugade, B. L. Fredrickson, and L. Feldman Barrett, “Psychological Resilience and Positive Emotional Granularity: Examining the Benefits of Positive Emotions on Coping and Health,” *J. Pers.*, vol. 72, no. 6, pp. 1161–1190, Dec. 2004.

[29] S. G. Barsade, “The Ripple Effect: Emotional Contagion and Its Influence on Group Behavior,” *Adm. Sci. Q.*, vol. 47, no. 4, p. 644, 2002.

[30] B. F. Liu, L. Austin, and Y. Jin, “How publics respond to crisis communication strategies: The interplay of information form and source,” *Public Relat. Rev.*, vol. 37, pp. 345–353, 2011.

[31] R. L. Street, G. Makoul, N. K. Arora, and R. M. Epstein, “How does communication heal? Pathways linking clinician-patient communication to health outcomes,” *Patient Educ. Couns.*, vol. 74, no. 3, pp. 295–301, 2009.

[32] S. Nicola, E. P. Ferreira, and J. . P. Ferreira, “A NOVEL FRAMEWORK FOR MODELING VALUE FOR THE CUSTOMER, AN ESSAY ON NEGOTIATION,” *Int. J. Inf. Technol. Decis. Mak.*, vol. 11, no. 3, pp. 661–703, 2012.

[33] J. Lapierre, “Customer-perceived value in industrial contexts,” *J. Bus. Ind. Mark.*, vol. 15, no. 2, pp. 122–145, 2000.

[34] T. Woodall, “Conceptualising ‘Value for the Customer’: An Attributional, Structural and Dispositional Analysis,” *Acad. Mark. Sci. Rev.*, no. 12, pp. 1–44, 2003.

[35] A. Kaufmann and F. Tödtling, “Science – industry interaction in the process of innovation : the importance of boundary-crossing between systems,” vol. 30, pp. 791–804, 2001.

[36] P. A. Koen *et al.*, “Fuzzy Front End : Effective Methods, Tools and Techniques,” in *The PDMA Toolbook for new Product Development*, P. Belliveau, A. Griffin, and S. Stephen, Eds. New York: John Wiley & Sons, Inc., 2002, pp. 5–34.

[37] J. R. Wixson, “Function Analysis and Decomposistion using Function Analysis Systems Technique,” in *INCOSE International Symposium*, 1999, pp. 800–805.

[38] Canadian Society of Value Analysis, “FUNCTION ANALYSIS SYSTEM TECHNIQUE (FAST),” 2018. [Online]. Available: http://www.valueanalysis.ca/fast.php. [Accessed: 26-Dec-2018].

[39] M. E. Porter, *The Competitive Advantage: Creating and Sustaining Superior Performanceo Title*. New York: Free Press, 1985.

[40] V. Allee, “Value network analysis and value conversion of tangible and intangible assets,” *J. Intellect. Cap.*, vol. 9, no. 1, pp. 5–24, 2008.

[41] T. L. Satty, *The analytic hierarchy process : planning, priority setting, resource allocation*, 2nd ed. New York: McGraw-Hill, 1980.

[42] T. L. Saaty, “Decision making with the analytic hierarchy process,” *Int. J. Serv. Sci.*, vol. 1, no. 1, 2008.

[43] Stuart Russell and P. Norvig, *Artificial intelligence—a modern approach by Stuart Russell and Peter Norvig, Prentice Hall. Series in Artificial Intelligence, Englewood Cliffs, NJ.*, 3rd ed. Upper Saddle River, New Jersey 07458: Prentice Hall Press, 2009.

[44] M. A. Shahin, “State-of-the-art review of some artificial intelligence applications in pile foundations,” *Geosci. Front.*, vol. 7, no. 1, pp. 33–44, Jan. 2016.

[45] T. Simon, “The State-of-the-Art of AI,” *The State-of-the-Art of AI*, 2017. [Online]. Available: https://chatbotnewsdaily.com/the-state-of-the-art-of-ai-d50512deb70a. [Accessed: 13-Oct-2018].

[46] Abhishek Parbhakar, “AI Tale – Medium,” *Hot subtopics in AI research*, 2018. [Online]. Available: https://medium.com/ai-tale/. [Accessed: 04-Nov-2018].

[47] P. Golland *et al.*, “A New Age of Computing and the Brain,” *CCC Brain Work.*, 2015.

[48] A. M. TURING, “I.—COMPUTING MACHINERY AND INTELLIGENCE,” *Mind*, vol. LIX, no. 236, pp. 433–460, 1950.

[49] C. Smith, B. McGuire, T. Huang, and G. Yang, “The History of Artificial Intelligence,” 2006. [Online]. Available: https://courses.cs.washington.edu/courses/csep590/06au/projects/history-ai.pdf. [Accessed: 11-Apr-2018].

[50] D. Lee, “Computer AI passes Turing test in ‘world first,’” *BBC News*, 09-Jun-2014.

[51] F. Rosenblatt, “The perceptron: A probabilistic model for information storage and organization in the brain.,” *Psychol. Rev.*, vol. 65, no. 6, pp. 386–408, 1958.

[52] K. Andrey, “A ‘Brief’ History of Neural Nets and Deep Learning,” 2015. [Online]. Available: http://www.andreykurenkov.com/writing/ai/a-brief-history-of-neural-nets-and-deep-learning/. [Accessed: 11-Nov-2018].

[53] P. J. Francois, “What Is Machine Learning?,” 2016. [Online]. Available: https://www.ibm.com/developerworks/community/blogs/jfp/entry/What_Is_Machine_Learning?lang=en. [Accessed: 11-Nov-2018].

[54] R. Dechter, “Learning While Searching in Constraint-Satisfaction-Problems.,” *Aaai*, pp. 178–185, 1986.

[55] M. Minsky and P. Seymour A., *Perceptrons*, Reissue ed. MIT Press, 1969.

[56] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, “Learning representations by back-propagating errors,” *Nature*, vol. 323, no. 6088, pp. 533–536, Oct. 1986.

[57] A. Moawad, “Neural networks and back-propagation explained in a simple way,” 2018. [Online]. Available: https://medium.com/datathings/neural-networks-and-backpropagation-explained-in-a-simple-way-f540a3611f5e. [Accessed: 11-Nov-2018].

[58] K. Fukushima, “Neocognitron: A Hierarchical Neural Network Capable of Visual Pattern Recognition,” vol. 1, pp. 119–130, 1988.

[59] K. Fukushima, “Neocognition: a self organizing neural network model for a mechanism of pattern regonition unaffected by shift in position,” vol. 202, 1980.

[60] L. Y. Pratt, “Discriminability-Based Transfer between Neural Networks,” *Adv. Neural Inf. Process. Syst.*, pp. 204–211, 1993.

[61] L. Y. Pratt, J. Mostow, and C. A. Kamm, “Direct transfer of learned information among neural networks,” *Proc. Ninth AAAI Conf. Artif. Intell.*, pp. 584–589, 1991.

[62] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[63] D. Cornelisse, “An intuitive guide to Convolutional Neural Networks,” 2018. [Online]. Available: https://medium.freecodecamp.org/an-intuitive-guide-to-convolutional-neural-networks-260c2de0a050. [Accessed: 11-Nov-2018].

[64] L. Blade, “Demystifying Convolutional Neural Networks,” 2018. [Online]. Available: https://medium.com/@eternalzer0dayx/demystifying-convolutional-neural-networks-ca17bdc75559. [Accessed: 11-Nov-2018].

[65] J. Deng, W. Dong, R. Socher, L.-J. Li, Kai Li, and Li Fei-Fei, “ImageNet: A large-scale hierarchical image database,” in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 248–255.

[66] H. Chen, “Machine learning for information retrieval: Neural networks, symbolic learning, and genetic algorithms,” *J. Am. Soc. Inf. Sci.*, vol. 46, no. 3, pp. 194–216, 1995.

[67] F. Veen, “The neural network zoo,” *The Asimov Institute*, 2016. [Online]. Available: https://www.asimovinstitute.org/author/fjodorvanveen/.

[68] L. Deng, “Deep Learning: Methods and Applications,” *Found. Trends® Signal Process. Signal Process.*, vol. 7, no. 3–4, pp. 197–387, 2014.

[69] S. Haykin, *Neural Networks: A Comprehensive Foundation*, 2nd ed. NJ, USA: Prentice Hall PTR Upper Saddle River, 1998.

[70] E. Guresen and G. Kayakutlu, “Definition of artificial neural networks with comparison to other networks,” *Procedia Comput. Sci.*, vol. 3, pp. 426–433, 2011.

[71] Wikipedia, “Artificial neural network.” [Online]. Available: https://en.wikipedia.org/wiki/Artificial_neural_network. [Accessed: 10-Sep-2019].

[72] M. Nielsen, “Using neural nets to recognize handwritten digits,” 2019. [Online]. Available: http://neuralnetworksanddeeplearning.com/chap1.html. [Accessed: 10-

Oct-2019].

[73] J. Brownlee, "A Gentle Introduction to Pooling Layers for Convolutional Neural Networks," 2019. [Online]. Available: https://machinelearningmastery.com/pooling-layers-for-convolutional-neural-networks/. [Accessed: 10-Sep-2019].

[74] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Commun. ACM*, vol. 60, no. 6, pp. 84–90, May 2017.

[75] Ç. Kaymak and A. Uçar, "A Brief Survey and an Application of Semantic Image Segmentation for Autonomous Driving," Aug. 2018.

[76] C. Szegedy *et al.*, "Going Deeper with Convolutions," Sep. 2014.

[77] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," Sep. 2014.

[78] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," Dec. 2015.

[79] A. G. Howard *et al.*, "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications," Apr. 2017.

[80] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size," Feb. 2016.

[81] X. Zhang, X. Zhou, M. Lin, and J. Sun, "ShuffleNet: An Extremely Efficient Convolutional Neural Network for Mobile Devices," Jul. 2017.

[82] G. Huang, S. Liu, L. van der Maaten, and K. Q. Weinberger, "CondenseNet: An Efficient DenseNet using Learned Group Convolutions," Nov. 2017.

[83] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le, "Learning Transferable Architectures for Scalable Image Recognition," Jul. 2017.

[84] E. Real, A. Aggarwal, Y. Huang, and Q. V Le, "Regularized Evolution for Image Classifier Architecture Search," Feb. 2018.

[85] H. Liu, K. Simonyan, and Y. Yang, "DARTS: Differentiable Architecture Search," Jun. 2018.

[86] Y. Xiong, H. J. Kim, and V. Hedau, "ANTNets: Mobile Convolutional Neural Networks for Resource Efficient Image Classification," Apr. 2019.

[87] L. Yang, S. Hanneke, and J. Carbonell, "A theory of transfer learning with applications to active learning," *Mach. Learn.*, vol. 90, no. 2, pp. 161–189, 2013.

[88] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi, "Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning," Feb. 2016.

[89] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L. C. Chen, "MobileNetV2: Inverted Residuals and Linear Bottlenecks," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, pp. 4510–4520, 2018.

[90] N. Silberman and S. Guadarrama, "TensorFlow-Slim image classification model library," 2016. [Online]. Available: https://github.com/tensorflow/models/tree/master/research/slim.

[91] C. Sun, A. Shrivastava, S. Singh, and A. Gupta, "Revisiting Unreasonable Effectiveness of Data in Deep Learning Era," *arXiv1707.02968 [cs]*, 2017.

[92] Kaggle Inc., "Kaggle Machine Learning & Data Science Survey 2017," 2017.

[93] Wikipedia, "Confusion matrix," 2019. [Online]. Available: https://en.wikipedia.org/wiki/Confusion_matrix.

[94] S. Narkhede, "Understanding Confusion Matrix," 2019. .

[95] Wikipedia, "Accuracy paradox," 2019. [Online]. Available: https://en.wikipedia.org/wiki/Accuracy_paradox.

[96] J. Brownlee, “Classification Accuracy is Not Enough : More Performance Measures You Can Use Your Start in Machine Your Start in Machine,” 2014. .

[97] K. P. Shung, “Accuracy , Precision , Recall or F1 ?,” 2018. [Online]. Available: https://towardsdatascience.com/accuracy-precision-recall-or-f1-331fb37c5cb9.

[98] Sarang Narkhede, “Understanding AUC - ROC Curve,” 2018. [Online]. Available: https://towardsdatascience.com/understanding-auc-roc-curve-68b2303cc9c5.

[99] Google, “Classification : ROC Curve and AUC,” 2019. [Online]. Available: https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc.

[100] I. Shah, “For K-fold cross validation , what k should be selected ?,” 2018. [Online]. Available: https://www.quora.com/For-K-fold-cross-validation-what-k-should-be-selected.

[101] G. Seif, “7 Pratical Deep Learning Tips,” 2018. [Online]. Available: https://towardsdatascience.com/7-practical-deep-learning-tips-97a9f514100e. [Accessed: 11-Nov-2018].

[102] J. Gage, “Introduction to Loss Functions,” 2018. [Online]. Available: https://blog.algorithmia.com/introduction-to-loss-functions/. [Accessed: 11-Nov-2018].

[103] A. Agrawal, “Loss Functions and Optimization Algorithms. Demystified.,” 2017. [Online]. Available: https://medium.com/data-science-group-iitr/loss-functions-and-optimization-algorithms-demystified-bb92daff331c. [Accessed: 17-Nov-2018].

[104] J. Gage, “Introduction to Optimizers,” 2018. [Online]. Available: https://blog.algorithmia.com/introduction-to-optimizers/. [Accessed: 11-Nov-2018].

[105] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, “How transferable are features in deep neural networks ?,” vol. 27, 2014.

[106] J. Le, “The 10 Deep Learning Methods AI Practioners Need to Apply,” 2017. [Online]. Available: https://medium.com/cracking-the-data-science-interview/the-10-deep-learning-methods-ai-practitioners-need-to-apply-885259f402c1. [Accessed: 11-Nov-2018].

[107] O. Sagi and L. Rokach, “Ensemble learning: A survey,” *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.*, vol. 8, no. 4, pp. 1–18, 2018.

[108] SkerrittBrandon, “Divide and Conquer Algorithms,” 2018. [Online]. Available: https://medium.com/brandons-computer-science-notes/divide-and-conquer-algorithms-4e83d9999ffa. [Accessed: 11-Nov-2018].

[109] B. L. M. Happel and J. M. J. Murre, “Design and evolution of modular neural network architectures,” *Neural Networks*, vol. 7, no. 6–7, pp. 985–1004, 1994.

[110] Pandas, “Python Data Analysis Library.” [Online]. Available: https://pandas.pydata.org/. [Accessed: 10-Nov-2019].

[111] Microsoft, “The Microsoft Cognitive Toolkit,” 2017. [Online]. Available: https://docs.microsoft.com/en-us/cognitive-toolkit/. [Accessed: 10-Nov-2019].

[112] BAIR, “Caffe.” [Online]. Available: http://caffe.berkeleyvision.org/. [Accessed: 10-Nov-2019].

[113] Ronan, Clément, Koray, and Soumith, “Torch.” [Online]. Available: http://torch.ch/. [Accessed: 10-Nov-2019].

[114] Accord.Net, “Accord.Net.” [Online]. Available: http://accord-framework.net/. [Accessed: 10-Nov-2019].

[115] Brain.js, “Brain.js.” [Online]. Available: https://brain.js.org/#/. [Accessed: 10-Nov-2019].

[116] Google, “TensorflowJS.” [Online]. Available: https://www.tensorflow.org/js.

[Accessed: 14-Aug-2019].

[117] Scikit-learn, “scikit-learn.” [Online]. Available: https://scikit-learn.org/stable/. [Accessed: 10-Nov-2019].

[118] NumPy.org, “NumPy.” [Online]. Available: https://numpy.org/. [Accessed: 10-Nov-2019].

[119] SciPy.org, “SciPy.” [Online]. Available: https://www.scipy.org/. [Accessed: 10-Nov-2019].

[120] Matplotlib, “matplotlib.”

[121] Sympy, “Sympy.”

[122] Evan You, “Vue.” [Online]. Available: https://vuejs.org/. [Accessed: 10-Nov-2019].

[123] Facebook, “React.”

[124] Goolge, “Angular.” [Online]. Available: https://angular.io/. [Accessed: 14-Aug-2019].

[125] Sophia, “Angular vs React vs Vue: Which is the Best Choice for 2019?,” 2019. [Online]. Available: https://hackernoon.com/angular-vs-react-vs-vue-which-is-the-best-choice-for-2019-16ce0deb3847. [Accessed: 10-Nov-2019].

[126] S. Daityari, “Angular vs React vs Vue: Which Framework to Choose in 2019,” 2019. [Online]. Available: https://www.codeinwp.com/blog/angular-vs-vue-vs-react/. [Accessed: 10-Nov-2019].

[127] Altexsoft, “The Good and the Bad of Angular Development.” [Online]. Available: https://www.altexsoft.com/blog/engineering/the-good-and-the-bad-of-angular-development/. [Accessed: 10-Oct-2019].

[128] A. Barros, “Angular / React / Vue pros and cons.” [Online]. Available: https://medium.com/@afonsobarros/angular-react-vue-pros-and-cons-75e161311e86. [Accessed: 10-Oct-2019].

[129] C.-H. Hjortsjo, *Man’s Face and Mimic Language*. Lund: Studentlitteratur, 1970.

[130] A. Freitas-Magalhães, *Facial Action Coding System 3.0: Manual of Scientific Codification of the Human Face*, 1st ed. Porto: Leya, 2018.

[131] S. Velusamy, H. Kannan, B. Anand, A. Sharma, and B. Navathe, “A method to infer emotions from facial action units,” *ICASSP, IEEE Int. Conf. Acoust. Speech Signal Process. - Proc.*, pp. 2028–2031, 2011.

[132] G. Patil and P. Suja, “Emotion recognition from 3D videos using optical flow method,” *2017 Int. Conf. Smart Technol. Smart Nation*, pp. 825–829, 2017.

[133] J. M. Sun, X. S. Pei, and S. S. Zhou, “Facial emotion recognition in modern distant education system using SVM,” *Proc. 7th Int. Conf. Mach. Learn. Cybern. ICMLC*, vol. 6, no. July, pp. 3545–3548, 2008.

[134] A. Nicolai and A. Choi, “Facial Emotion Recognition Using Fuzzy Systems,” *Proc. - 2015 IEEE Int. Conf. Syst. Man, Cybern. SMC 2015*, pp. 2216–2221, 2016.

[135] Suchitra, Suja P., and S. Tripathi, “Real-time emotion recognition from facial images using Raspberry Pi II,” *2016 3rd Int. Conf. Signal Process. Integr. Networks*, pp. 666–670, 2016.

[136] H. Sadeghi, A. A. Raie, and M. R. Mohammadi, “Facial expression recognition using geometric normalization and appearance representation,” *Iran. Conf. Mach. Vis. Image Process. MVIP*, pp. 159–163, 2013.

[137] I. Tautkute, T. Trzcinski, and A. Bielski, “I Know How You Feel: Emotion Recognition with Facial Landmarks,” pp. 1991–1993, 2018.

[138] F. De La Torre, W. S. Chu, X. Xiong, F. Vicente, X. Ding, and J. Cohn, “IntraFace,” *2015 11th IEEE Int. Conf. Work. Autom. Face Gesture Recognition, FG 2015*, 2015.

[139] P. Viola and M. Jones, “Robust Real-time Object Detection Paul,” *Int. J. Comput.*

Vis., no. 57, 2001.

[140] F. Abdat, C. Maaoui, and A. Pruski, “Human-computer interaction using emotion recognition from facial expression,” *Proc. - UKSim 5th Eur. Model. Symp. Comput. Model. Simulation, EMS 2011*, pp. 196–201, 2011.

[141] M. F. Valstar and M. Pantic, “Fully Automatic Recognition of the Temporal Phases of Facial Actions,” *IEEE Trans. Syst. Man, Cybern. Part B*, vol. 42, no. 1, pp. 28–43, Feb. 2012.

[142] R. Verma and M. Y. Dabbagh, “Fast facial expression recognition based on local binary patterns,” *Can. Conf. Electr. Comput. Eng.*, pp. 1–4, 2013.

[143] Z. Y. Peng, Y. Zhou, and P. Wang, “Multi-pose face detection based on adaptive skin color and structure model,” *CIS 2009 - 2009 Int. Conf. Comput. Intell. Secur.*, vol. 1, pp. 325–329, 2009.

[144] Z. Y. Peng, Y. H. Zhu, and Y. Zhou, “Real-time facial expression recognition based on adaptive Canny operator edge detection,” *2010 Int. Conf. Multimed. Inf. Technol. MMIT 2010*, vol. 2, pp. 154–157, 2010.

[145] K. Mistry, L. Zhang, S. C. Neoh, M. Jiang, A. Hossain, and B. Lafon, “Intelligent Appearance and shape based facial emotion recognition for a humanoid robot,” *Ski. 2014 - 8th Int. Conf. Software, Knowledge, Inf. Manag. Appl.*, 2014.

[146] M. Suk and B. Prabhakaran, “Real-time mobile facial expression recognition system-a case study,” *IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. Work.*, pp. 132–137, 2014.

[147] V. B. Bekezhanova and O. N. Goncharova, “Study of the convective fluid flows with evaporation on the basis of the exact solution in a three-dimensional infinite channel,” *J. Phys. Conf. Ser.*, vol. 899, no. 3, 2017.

[148] P. Lucey, J. F. Cohn, T. Kanade, J. Saragih, Z. Ambadar, and I. Matthews, “The Extended Cohn-Kanade Dataset (CK+): A complete dataset for action unit and emotion-specified expression,” in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Workshops*, 2010, vol. 45, no. 7, pp. 94–101.

[149] M. Lyons, S. Akamatsu, M. Kamachi, and J. Gyoba, “Coding facial expressions with Gabor wavelets,” *Proc. - 3rd IEEE Int. Conf. Autom. Face Gesture Recognition, FG 1998*, pp. 200–205, 1998.

[150] X. Zhang *et al.*, “A high-resolution spontaneous 3D dynamic facial expression database,” in *2013 10th IEEE International Conference and Workshops on Automatic Face and Gesture Recognition (FG)*, 2013, vol. 38, no. July, pp. 1–6.

[151] A. T. Lopes, E. De Aguiar, and T. Oliveira-Santos, “A Facial Expression Recognition System Using Convolutional Networks,” *Brazilian Symp. Comput. Graph. Image Process.*, vol. 2015-Octob, pp. 273–280, 2015.

[152] A. Mollahosseini, D. Chan, and M. H. Mahoor, “Going deeper in facial expression recognition using deep neural networks,” in *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2016, pp. 1–10.

[153] B.Kennedy and A. Balint, “EmotionNet.” [Online]. Available: https://github.com/co60ca/EmotionNet. [Accessed: 02-May-2019].

[154] B. Doerrfeld, “20+ Emotion Recognition APIs That Will Leave You Impressed, and Concerned,” 2018. [Online]. Available: https://nordicapis.com/20-emotion-recognition-apis-that-will-leave-you-impressed-and-concerned/. [Accessed: 30-Dec-2018].

[155] Affectiva, “Affectiva.” [Online]. Available: https://www.affectiva.com/. [Accessed: 30-Dec-2018].

[156] eyeris, “Multi-Ethnic Emotional Vocabulary.” [Online]. Available: http://www.eyeris.ai/technology/. [Accessed: 30-Dec-2018].

[157] Nvisio, “ADVANCING HUMAN POTENTIAL,” 2019. [Online]. Available: https://www.nviso-insights.com/en.

[158] Microsoft, “Cognitive Services.” .

[159] “FaceReader.” [Online]. Available: https://www.noldus.com/human-behavior-research/products/facereader.

[160] P. Kruchten, “Architectural Blueprints — The ‘ 4 + 1 ’ View Model of Software Architecture,” no. May, 2014.

[161] sheetjs, “xlsx.” [Online]. Available: http://sheetjs.com/opensource. [Accessed: 14-Aug-2019].

[162] “charjs.” [Online]. Available: https://www.chartjs.org/. [Accessed: 14-Aug-2019].

[163] “jszip.” [Online]. Available: https://stuk.github.io/jszip/. [Accessed: 14-Aug-2019].

[164] “file saver.” [Online]. Available: https://www.npmjs.com/package/file-saver. [Accessed: 14-Aug-2019].

[165] P. Viola and M. Jones, “Rapid object detection using a boosted cascade of simple features,” in *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, 1997, vol. 1, no. 1–3, pp. I-511-I–518.

[166] “Tensorflow.js.” [Online]. Available: https://js.tensorflow.org/. [Accessed: 02-Jul-2019].

[167] G. Inc., “GitHub.” .

[168] Atlassian, “BitBucket.” [Online]. Available: https://bitbucket.org. [Accessed: 14-Aug-2019].

[169] Google, “Firebase.” [Online]. Available: https://firebase.google.com/. [Accessed: 14-Aug-2019].

[170] Microsoft, “Typescript.” .

[171] H. Catlin, N. Weizenbaum, C. Eppstein, and J. Anne, “SASS.” [Online]. Available: https://sass-lang.com.

[172] NPM, “NPM.” [Online]. Available: https://www.npmjs.com/. [Accessed: 14-Aug-2019].

[173] Python Software Foundation, “Python.” [Online]. Available: https://www.python.org/. [Accessed: 14-Aug-2019].

[174] Python Software Foundation, “Pip.” [Online]. Available: https://pypi.org/. [Accessed: 14-Aug-2019].

[175] Ian Bicking and The Open Planning Project, “virtualenv.” [Online]. Available: https://virtualenv.pypa.io/en/latest/. [Accessed: 14-Aug-2019].

[176] D. Lundqvist, A. Flykt, and A. Öhman, “The Karolinska Directed Emotional Faces - KDEF.” CD ROM from Department of Clinical Neuroscience, Psychology section, Karolinska Institutet, ISBN 91-630-7164-9, 1998.

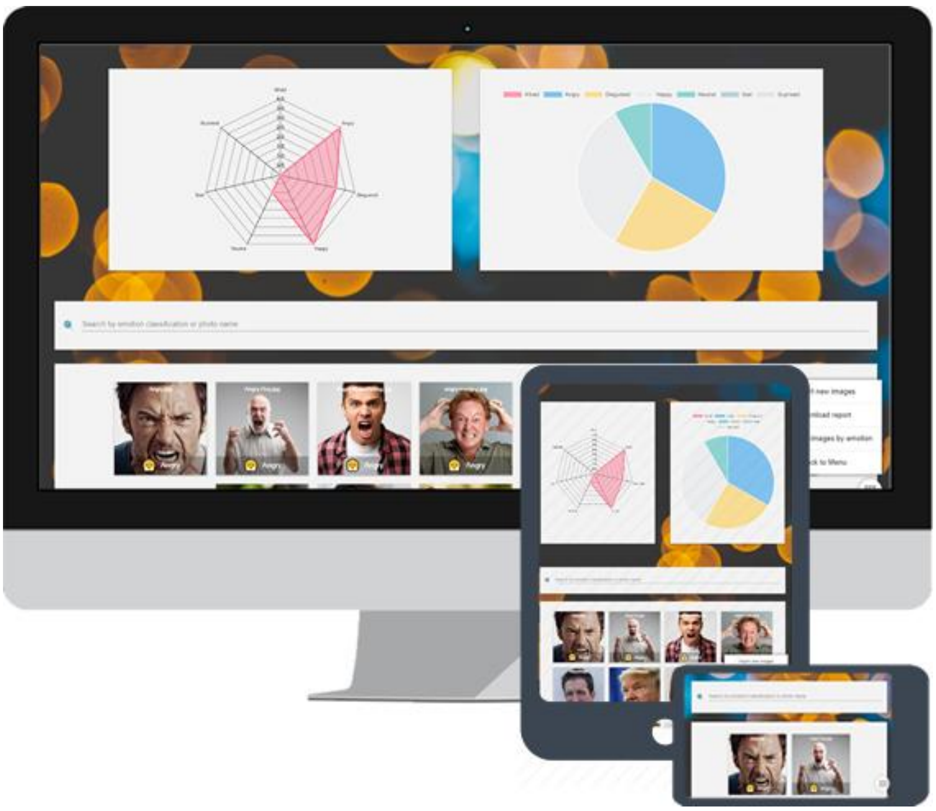
[177] L.-F. Chen and Y.-S. Yen, “Taiwanese Facial Expression Image Database.,” *Brain Mapp. Lab. Inst. Brain Sci. Natl. Yang-Ming Univ.*, 2007.

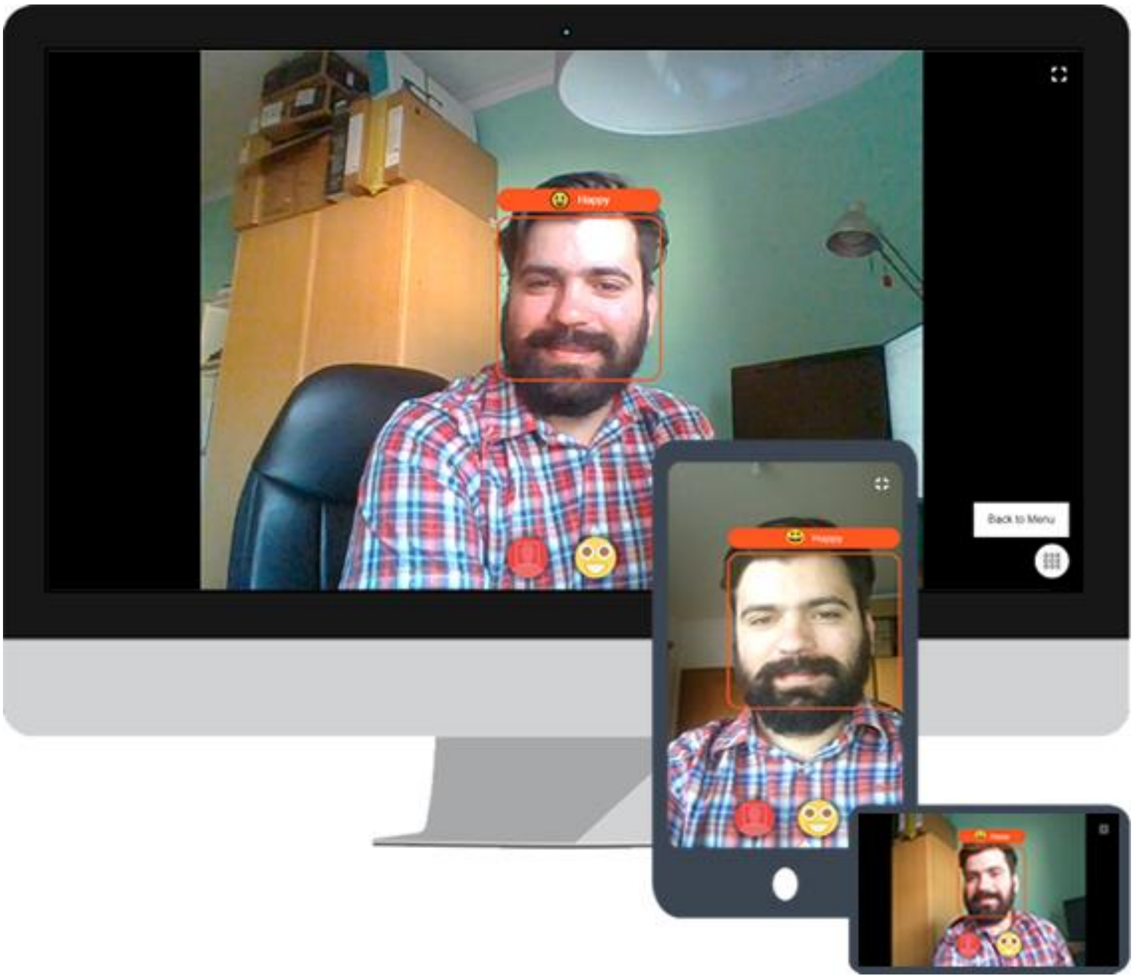
[178] G. Righi, J. J. Peissig, and M. J. Tarr, “Recognizing disguised faces,” *Vis. cogn.*, vol. 20, no. 2, pp. 143–169, Feb. 2012.

[179] Lomus Lab, “The Human Cognitive Project.” [Online]. Available: https://www.lumosity.com/hcp. [Accessed: 17-Aug-2019].

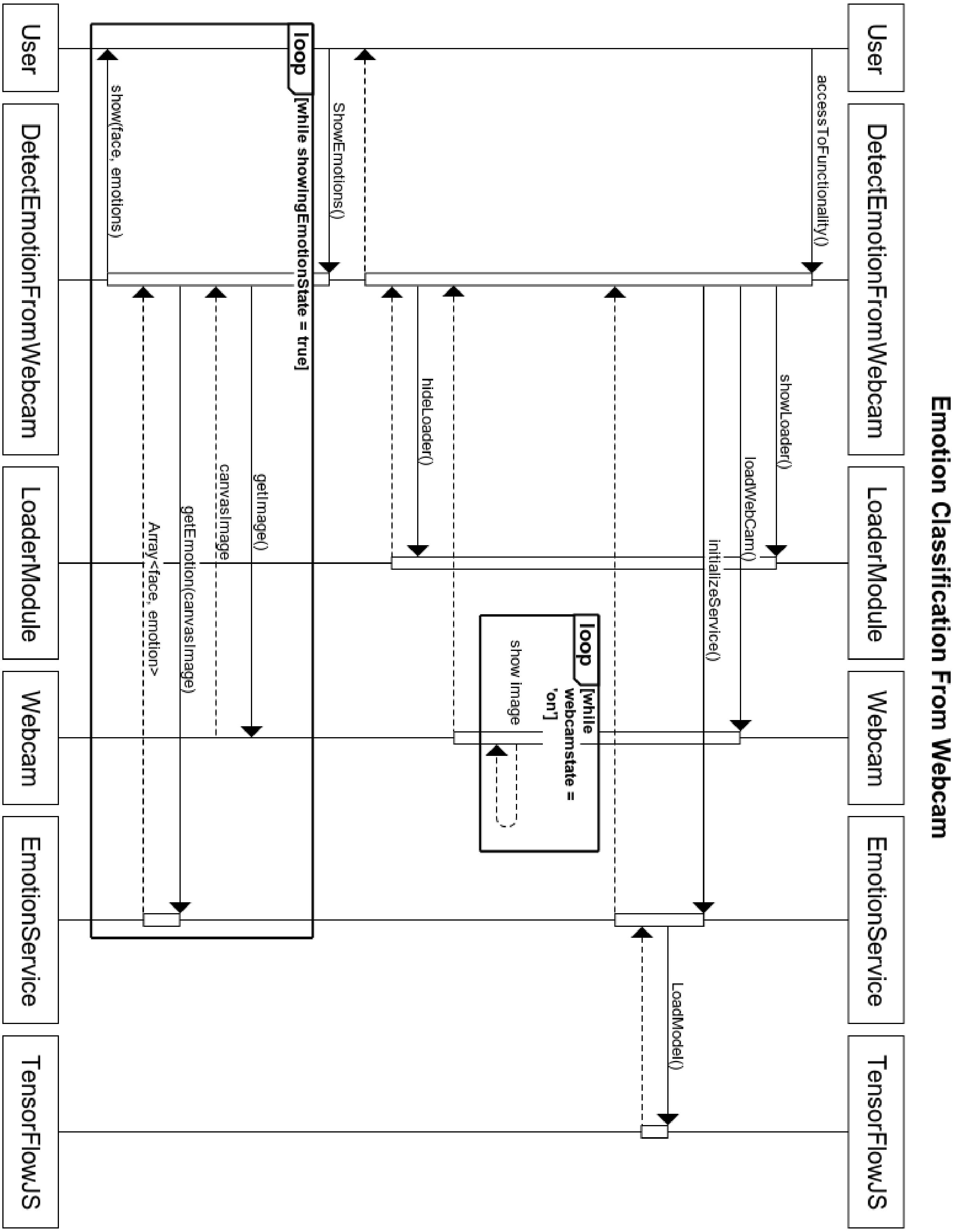
[180] O. M. Parkhi, A. Vedaldi, and A. Zisserman, “Deep Face Recognition,” no. Section 3, pp. 41.1-41.12, 2015.

ATTACHMENT I

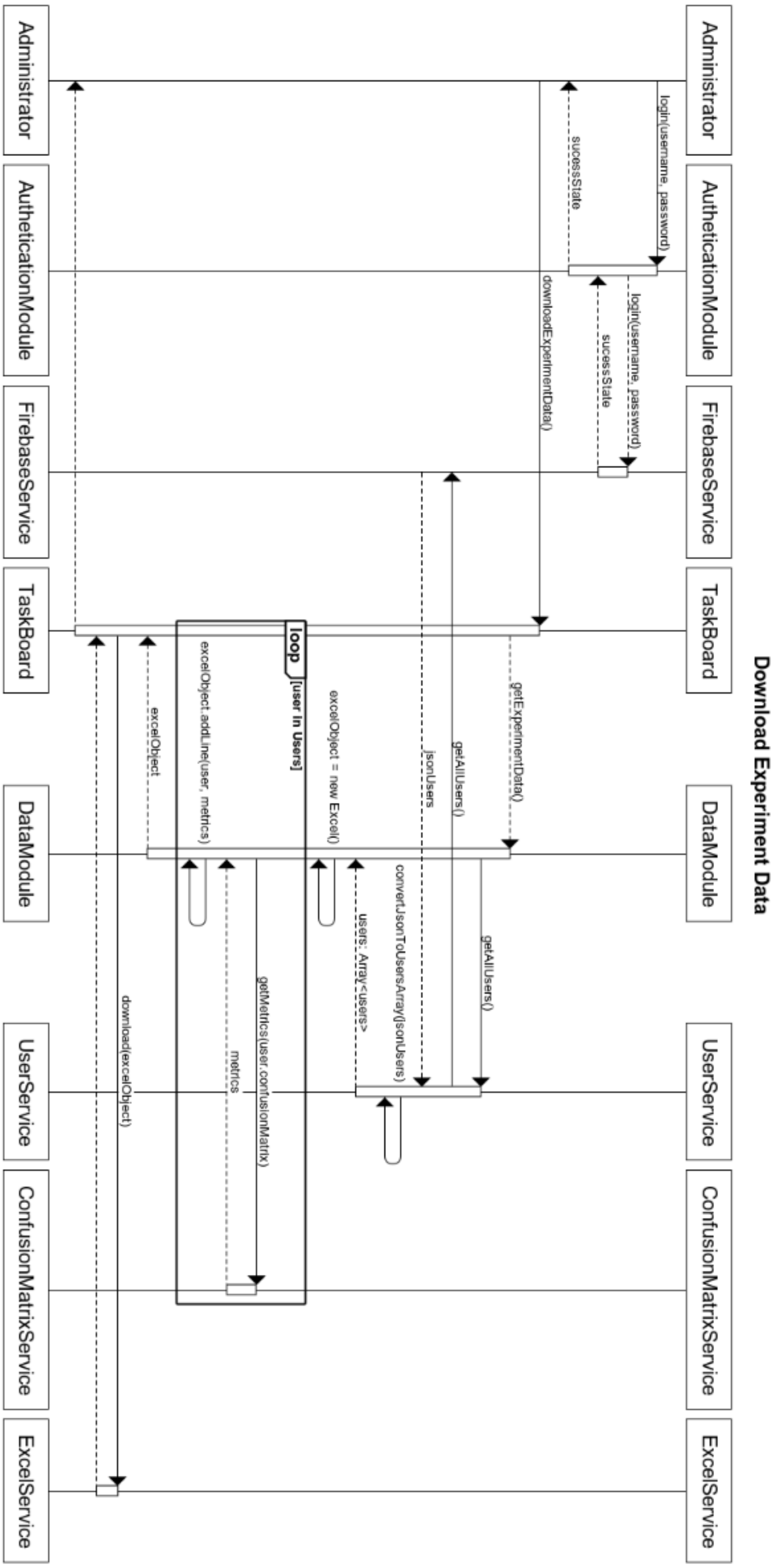



Happy
Back to Menu
Happy

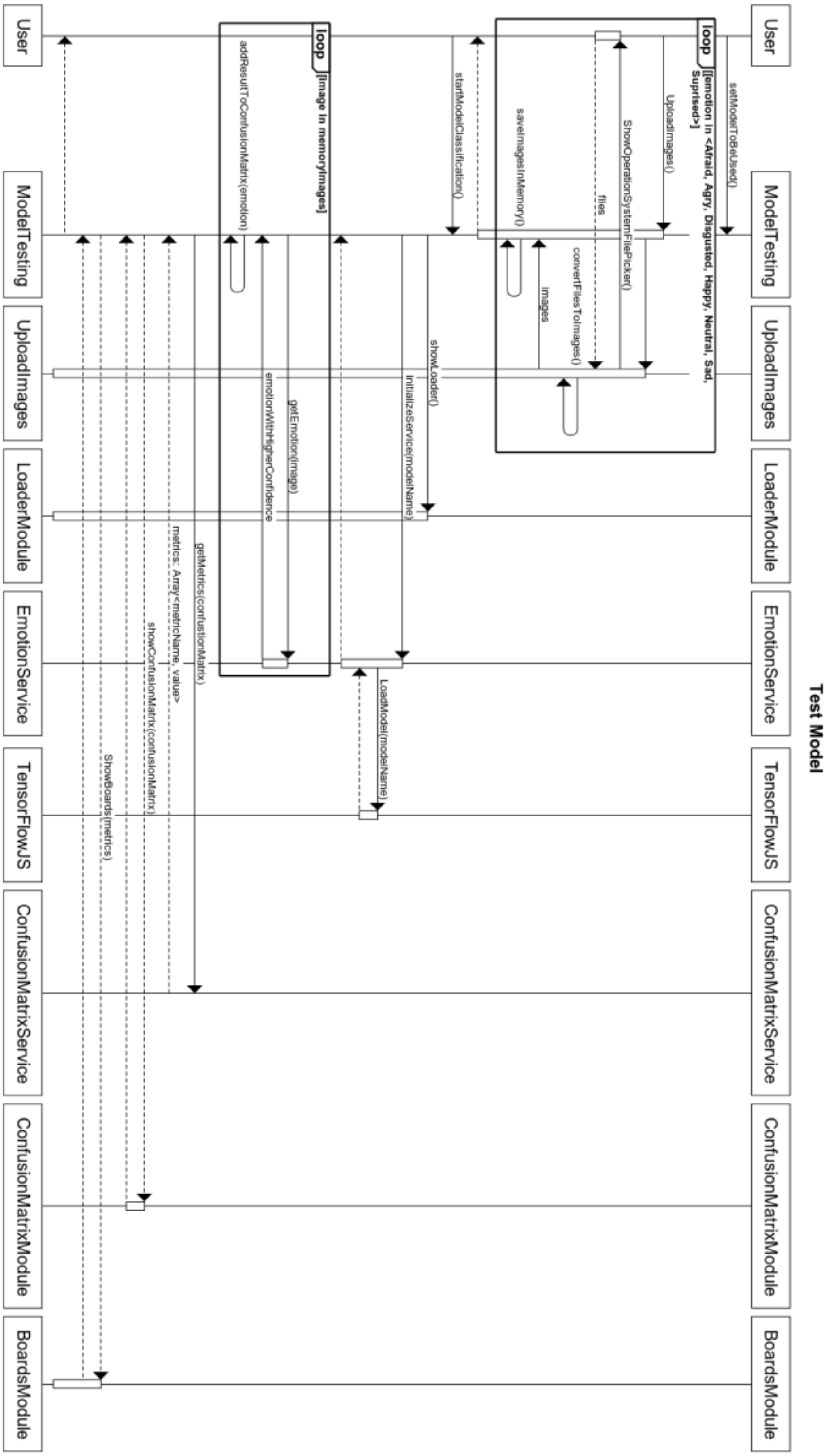
ATTACHMENT II



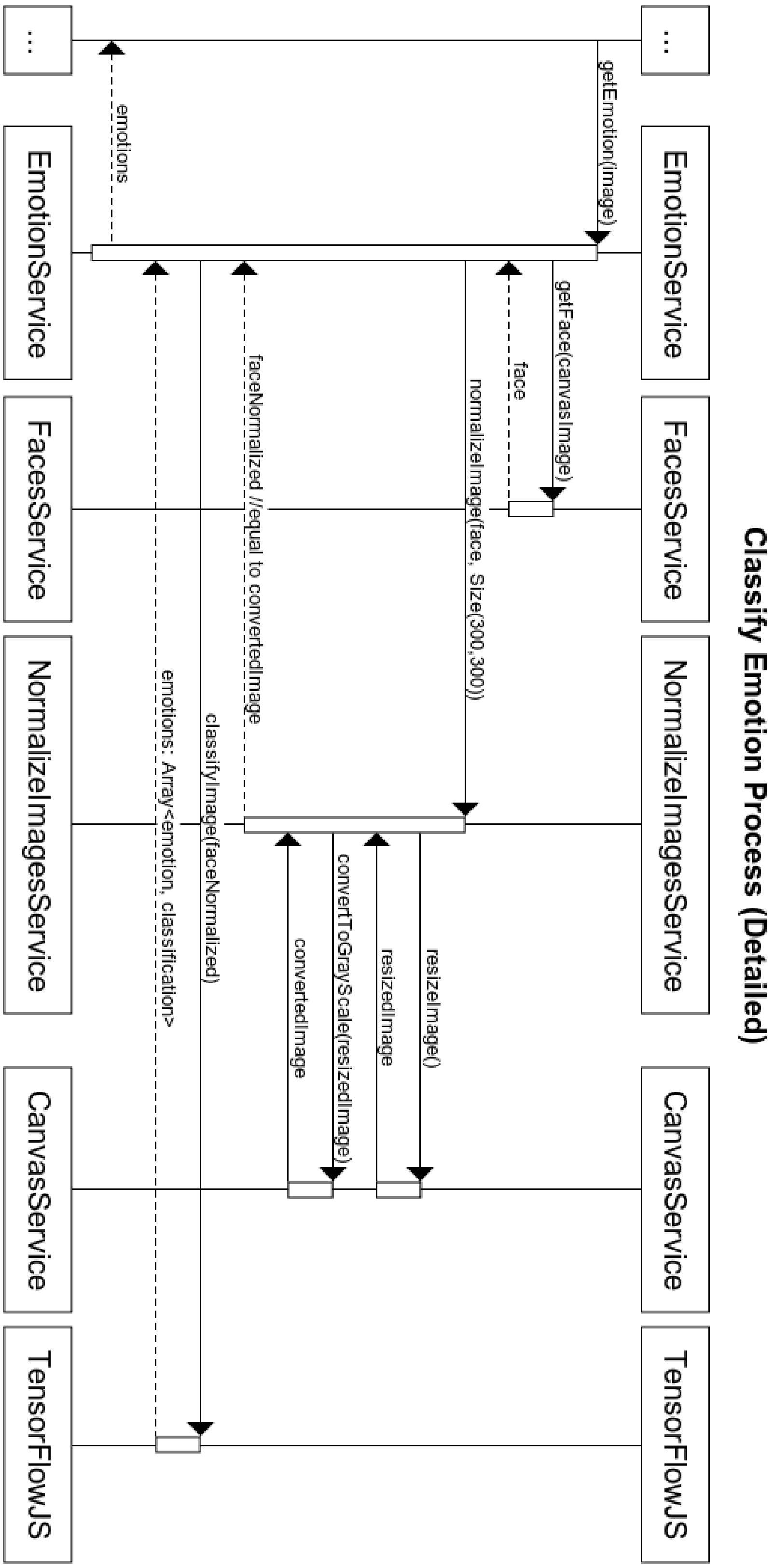
ATTACHMENT III



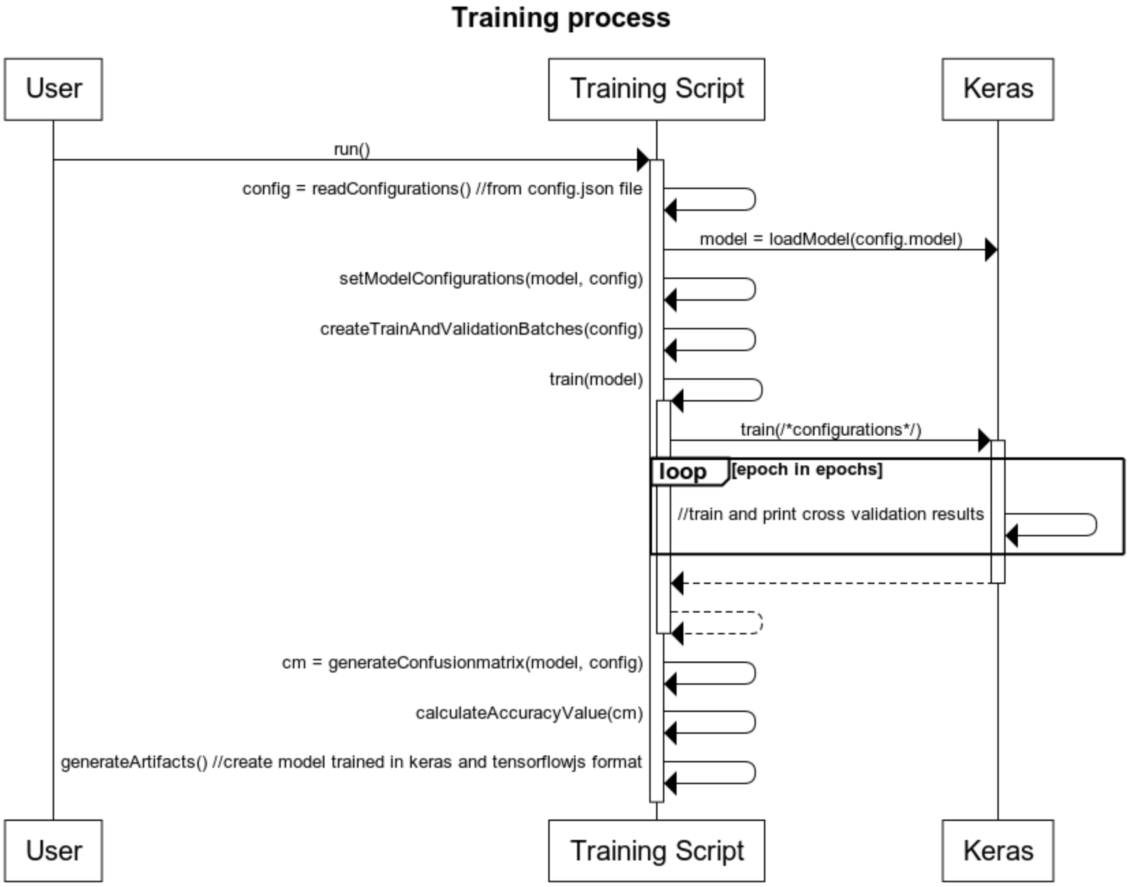
ATTACHMENT IV



ATTACHMENT V



ATTACHMENT VI



ATTACHMENT VII