Robot-Enhanced ABA Therapy: Exploring Emerging Artificial Intelligence Embedded Systems in Socially Assistive Robots for the Treatment of Autism

by

Eduardo Calle Ortiz

A Thesis
Submitted to the Faculty
of the
WORCESTER POLYTECHNIC INSTITUTE
In partial fulfillment of the requirements for the
Degree of Master of Science
in
Computer Science
by

August 2019

APPROVED:

Professor Jacob Whitehill, Master Thesis Advisor

Professor Gregory S. Fischer, Master Thesis Co-Advisor

Professor Berk Calli, Thesis Reader

Professor Craig E. Wills, Head of Department

Acknowledgements

I would like to acknowledge those individuals who have supported me as a graduate student at Worcester Polytechnic Institute, as I have investigated the use of socially assistive robots in ABA therapies, and working to enhance therapeutic treatments for those affected by ASDs. I wish to recognize the U.S. Fulbright Program, LASPAU, and the Fulbright Commission of Ecuador for providing the opportunity to study in the United States through the Fulbright program. My heartfelt gratitude to Terri Anne Camesano, Dean of Graduate Studies, for your continuous support during my graduate studies at WPI.

A special thank you to my primary thesis advisor, Professor Gregory Fischer, for the opportunity to apply my interests in artificial intelligence, robotics, and computer science with aims to serve others. Your sincere investment in my work has made my academic life richer. I wish to express my sincere gratitude to Dr. Laurie Dickstein-Fischer for her clinical guidance on this project. Also, a special thank you to Professor Jacob Whitehill, my thesis and academic advisor, for his support throughout my academic studies. Thank you, Professor Berk Calli, for dedicating the time to read drafts of my work as my thesis reader. I would also like to thank my colleagues in the robotics lab as we share a sincere commitment to learning and creative problem-solving, while advancing innovative technologies that will serve society at large.

Finally, I wish to acknowledge every individual who believes in the power of an education to transform, enrich, and inspire. For it is to you that, rather than a fixation on the end result of this project, which is so frequently the case of considerable personal or professional undertakings, I have sincerely enjoyed the process of investigation, strengthening my belief in transformation through education.

Eduardo Calle Ortiz

Dedication

A Sarah,
Gracias por ser mi compañera de viaje,
sin tu apoyo y tu amor esto no hubiera sido posible.

A mi familia,
Por recordarme siempre qué es lo verdaderamente importante.

A tí,
Por guiar siempre mi camino

Abstract

In the last decade, socially assistive robots have been used in therapeutic treatments for individuals diagnosed with Autism Spectrum Disorders (ASDs). Preliminary studies have demonstrated positive results using the Penguin for Autism Behavioral Intervention (PABI) developed by the AIM Lab at WPI to assist individuals diagnosed with ASDs in Applied Behavioral Analysis (ABA) therapy treatments. In recent years, power-efficient embedded AI computing devices have emerged as a powerful technology by reducing the complexity of the hardware platforms while providing support for parallel models of computation. This new hardware architecture seems to be an important step in the improvement of socially assistive robots in ABA therapy. In this thesis, we explore the use of a power-efficient embedded AI computing device and pre-trained deep learning models to improve PABI's performance. Five main contributions are made in this work. First, a robot-enhanced ABA therapy framework is designed. Second, a multilayer pattern software architecture for a robot-enhanced ABA therapy framework is explored. Third, a multifactorial experiment is completed in order to benchmark the performance of three popular deep learning frameworks over the AI computing device. Experimental results demonstrate that some deep learning frameworks utilize the resources of GPU power while others utilize the multicore ARM-CPU system of the device for its parallel model of computation. Fourth, the robustness of state-of-the-art pre-trained deep learning models for feature extraction is analyzed and contrasted with the previous approach used by PABI. Experimental results indicate that pre-trained deep learning models overcome the traditional approaches in some fields; however, combining different pre-trained models in a process reduces its accuracy. Fifth, a patient-tracking algorithm based on an identity verification approach is developed to improve the autonomy, usability, and interactions of patients with the robot. Experimental results show that the developed algorithm has potential to perform as well as the previous algorithm used by PABI based on a deep learning classifier approach.

Contents

List of Figures

List of Tables

List of Acronyms

ASD Autism Spectrum Disorder

ABA Applied Behavior Analysis

DTT Discrete Trial Training

RAT Robot Assisted Therapy

SAR Socially Assistive Robot

DNN Deep Neural Network

AI Artificial Intelligence

SoC System On Chip

Chapter 1

Introduction

Rapid technological innovations, particularly in the field of robotics, offers innovative treatment options for individuals diagnosed with Autism Spectrum Disorders (ASDs). As such, incorporating robots in autism therapy has acquired a substantial amount of attention over the past twenty years (Diehl et al. 2012; H.-L. Cao, Pablo G. Esteban, et al. 2019). Potential benefits of robot-enhanced autism therapies are diverse. Clinical results indicate that robot-enhanced therapy (RET) has potential to perform similar to that of human standard therapies for individuals with ASDs(H.-L. Cao, Pablo G. Esteban, et al. 2019). As innovative technologies in robotics and artificial intelligence are included in autism treatment therapies, there is great potential for diminished treatment costs and improved quality of life for individuals diagnosed with ASDs.

Preliminary studies demonstrate positive results using the Penguin for Autism Behavioral Intervention (PABI ©Dickstein-Fischer) developed by

the AIM Lab (Laurie A Dickstein-Fischer et al. 2018) at WPI to assist individuals diagnosed with ASDs in ABA therapy treatments. Power-efficient embedded AI computing devices have emerged in recent years as powerful technologies, reducing complexity of the hardware platforms and providing support for parallel models of computation. This new hardware architecture indicates an important step in the improvement of socially assistive robots in ABA therapy. In this thesis, the use of a power-efficient embedded AI computing device and pre-trained deep learning models are explored to improve PABI's performance.

Emerging technologies suggest a rethinking the design of software architectures. Chapter two focuses on this task. In the preliminary part of this chapter, a robot-enhanced ABA therapy framework is designed. This framework combines the requirements of ABA therapy with best practices in the design of social robots. Seven subsystems integrate the framework in order to provide reactive and deliberative therapeutic behaviors in the robot: low-level feature extraction system, attention system, high-level and intelligent perception system, ABA intelligent tutoring system, physical control system, behavior system, and therapy configuration system. Chapter two also explores the implementation of a multilayer software architecture, serving as the pattern to implement the framework. The use of deep learning pre-trained models over this pattern merges both the framework and the AI computing device of PABI. The main characteristics looked for

with this pattern are real-time, multitasking, modularity and expandability.

In the proposed software architecture, deep learning frameworks play a key role on creating systems that are both real-time and multitasking. If the inference engine of a deep leaning framework is inefficient on the AI device, real-time tasks are affected. On the other hand, in the inference engine uses all of the resources of the embedded platform, multitasking performance decreases substantially. Chapter three focuses on the evaluation of three popular deep learning frameworks in order to evaluate efficiency an resources usage.

Pre-trained deep learning models to make the robot expandable and modular are explored in chapter four. Using pre-trained models on PABI has several advantages. First, the hardware platform of PABI is designed to deploy deep learning architectures efficiently. Second, the concept of black-box inherits to deep learning models, enabling future improvements in the robot without modification of the coding. Third, deep-learning models exhibit better robustness than some traditional computer vision approaches. For example, in face detection, deep-learning models have demonstrated robustness to changes in pose, illumination, age, among others.

deep-learning models have surpassed traditional computer vision approaches, particularly under uncontrolled conditions as extreme pose face. Chapter three focuses on testing the robustness of state-of-the-art pre-

trained deep learning models for feature extraction under extreme poses. The results are contrasted with traditional approaches used in the previous version of PABI. Three low-level features are evaluated: face detection, face landmarks detection, and face recognition because of they are essential components for building high-level human-robot interaction applications.

Finally, the software architecture previously designed is used to improve the usability of the robot in an ABA therapy context. Using a patient-tracking algorithm based on an open-set approach surpasses the constraints inherent in close-set approaches. In close-set approaches a set of patient images are taken and a deep learning model trained. The open-set algorithm presented in chapter 5, combines patient data, a face encoding deep learning model, and unsupervised methods to improve the usability and autonomy of the PABI robot during ABA therapies.

This work re-imagines the design of socially assistive robots in ABA therapy for autism treatments using emergent technologies in both software and computational hardware of the robots. The core motivation for this project is to improve the quality of life for autistic individuals, forging advancements in robot-enhanced therapeutic treatments and providing new opportunities within the standard-of-care treatment of individuals with ASDs.

Chapter 2

Background and Literature Review

This chapter provides an overview of the research literature related to Autism Spectrum Disorders (ASDs). It addresses many of the challenges in treating ASDs, defined by a broad spectrum of disorders with a wide range and variation of symptoms. Previous research based on the literature in the field of autism therapy is further explained with specific focus on Applied Behavior Analysis (ABA), a skill-based therapy that looks at the science of behavior and the principles of motivation and learning to modify behaviors. ABA therapies are underscored because they have been scientifically proven to effectively teach or eliminating undesired behaviors exhibited by individuals on the autism spectrum. Moreover, ABA is one of the leading therapies in the treatment of ASDs (Fisher et al. 2019), whose results have been supported with measurable outcomes in scientific research (Reichow et al. 2012; Heitzman-Powell et al. 2014). In addition to an overview of the fundamental principles of ABA, it's potential benefits in relation to socially

assistive robots is explored. Rapid technological innovations, particularly in the field of robotics, offers innovative treatment options for individuals with ASDs. Robot-enhanced therapy (RET), for example, has the potential to achieve an equivalent performance compared to that of standard human therapies. More specifically, socially assistive robots (SAR) have been proven to benefit individuals treated for ASDs, helping in the acquisition of social skills and language development in the treatment of autism. This chapter provides a brief overview of the Penguin for Autism Behavioral Interventions (PABI) as a socially assistive robot that was created by the AIM Lab at WPI with its innovative software architecture specifically designed for use in ABA therapy. Finally, thesis contributions are detailed, providing a brief overview of the contents of each chapter.

2.1 Autism Spectrum Disorders (ASDs)

According to the Diagnostic and Statistical Manual of Mental Disorders-Fifth Edition (DSM-5), ASDs is a generic term under which falls the pervasive developmental disorders characterized by the exhibition of challenges associated with language development, social communication, social interaction, excessive or stereotyped repetitive behaviors or resistance to change in one's daily routine. Rather than ASDs being a singular, easily identifiable condition based on a common standard of symptoms across the board, it is rather a broad spectrum of disorders defined by an array of severity

levels, resulting in widespread variation in patient types.

Autism Spectrum Disorders include Autistic Disorder, Childhood Disintegrated Disorder, Asperger's Disorder, Rett's Disorder, and Pervasive Developmental Disorder Not Otherwise Specified (*DSM-IV Diagnostic Classifications* 2019)(Zachor and Merrick 2012).

ASDs are identified by abnormalities in social interactions and communications, repetitive behavior and confined interests (*DSM-IV Diagnostic Classifications* 2019; H.-L. Cao, Pablo G. Esteban, et al. 2019). A person diagnosed on the spectrum with low functioning abilities may have intense behaviors, impeding their ability to lead an independent life, little to no communication skills, and an inability to interact in environments without additional support. A higher functioning person would have the stereotypical behavior, yet would be capable of communication and interact with others while still experiencing challenging in these areas. Symptoms of ASDs are vast and varied, presenting a unique set of challenges in care and treatment of individuals diagnosed with ASDs. ASDs may be recognizable in an individual who is nonverbal and uses an iPad to verbalize concerns or express their ideas. Yet a different individual diagnosed with ASDs may be presented with a different set of challenges. For example, they may be extremely talented in a specific subject area such as mathematics, yet struggle to make eye contact and when nervous or upset may be prone to emotionally shutdown. Early indicators of ASDs include social exclu-

sion, poor balance skills, persistent impairments in social communications skills, impairments in verbal and nonverbal communications skills, avoidance of social situations, as well as sensitivity to oral textures. Identified individuals with ASDs may exhibit one or more of these behaviors.(Bellini 2004).

A common concern regarding ASDs is its widespread prevalence in children. According to the Centers for Disease Control and Prevention, in the United States, 1 in 59 children have been identified with ASDs (Baio 2018). Furthermore, ASDs is not correlated with racial, ethnic and socioeconomic groups, reaching global concern. In the late 1980s and early 1990s legislation was passed that provided individuals with educational materials to help those diagnosed with ASDs. As a result, with increased awareness of the symptoms of autism, more individuals have been able to receive therapeutic treatments.

Despite therapies exceptional success in significantly improving the lives of many autistic individuals, therapies for ASDs are accompanied by a number of challenges. One such concern is the high cost of therapy, acting as a barrier to families who need access to therapy. It is estimated that the cost of supporting a person with ASDs during his or her lifespan is 1.4 million USD in the United States. Furthermore, if the person has an intellectual disability, the cost practically doubles to 2.4 million USD (Baio 2018). The primary expenditures for children with ASDs include

special education programs and productivity loss of parents with children diagnosed with ASDs. Furthermore, behavioral intervention programs, for example early intervention for children, requires a significant amount of human workload to carry out therapeutic sessions, as well as to manage a child's performance data. Increased risk of poverty is a concern for families as therapies contribute to a reduction in family household wealth. Medical costs and residential care for individuals with ASDs is another concern as prices continue to mount.

2.2 Applied Behavior Analysis (ABA) for ASDs

ASDs have not been be associated with any particular biological marker and a cure to this disorder has not yet been discovered. As a result of the varying symptoms and severity of ASDs, no singular therapy is capable of being effective for every autistic individual. However, there are several available treatments that are well-studied in literature that have been proven to assist individuals diagnosed with ASDs. According to (Green et al. 2006)(Howard et al. 2005) the most frequently used therapies include Speech Therapy, Visual Schedules, Sensory Integration, Applied Behavior Analysis (ABA), and Social Stories. They have been demonstrated to serve as viable therapeutic treatments.

Applied Behavior Analysis (ABA) is a skill-based therapy that looks at the science of behavior the principles of motivation and learning to

modify behaviors. ABA has been scientifically proven to effectively help individuals on the autism spectrum by teaching or eliminating challenging behaviors exhibited. ABA is one of the leading therapies which results have been supported by scientific research and measurable results (Granpeesheh and Tarbox 2009). ABA is a technique in which a therapist conducts observations, observes the environment, and restructures it in order to assist individuals with ASDs in the development of social skills and forge bonds that will help them overcome challenges associated with ASDs.

ABA is not synonymous for autism therapy yet because of its proven success in helping modify behavior in individuals diagnosed with ASDs, it is commonly associated with autism therapy. ABA is incorporated in teaching individuals with autism and assumes that all activities that people do can be considered a form of behavior and, as such, ABA seeks to either strengthen or weaken behaviors by rewarding or avoiding their consequences. It hinges on the premise that by controlling or alternating behavioral consequences, it can be modified.

Four principles on which ABA stands include reinforcement, extinction, stimulus control, and generalization. Reinforcement generates desirable or averse consequences of a behavior depending if we want to strengthen or weaken it. Using reinforcement systematically, a determinate behavior may be increased or reduced. For example, cheering a child taking their first steps conveys to the child a positive reinforcement, helping them learn

to walk and develop the necessary skills to lead a full life. ABA therapy simply expands this training in intensive ways until each child struggling with autism experiences new milestones and develops the necessary habits to lead a full life.

ABA changes behavior by looking at what reinforces a certain behavior. It further examines the principles of learning in order to alter a specific behavior or to reinforce a given behavior in order to encourage its continuity in the future. Examples of positive reinforcement include praise, something the individual enjoys, or an edible item that demonstrates to the individual that their behavior is what the therapist is seeking. Undesired behaviors can be mitigated through positive reinforcement, negative reinforcement or automatic reinforcement. Unlike positive reinforcement, extinction seeks to deter or reduce negative behavior. Stimulus control, on the other hand, implies that behavior is present only when a specific stimulus is present and cannot exist in its absence. Daily behaviors are controlled by perceived stimuli. Generalization proliferates a behavior in the presence of one stimulus to another and becomes a challenging process for autistic individuals.

2.3 Socially Assistive Robots in Autism Therapy

Rapid technological innovations, particularly in the field of robotics and machine learning, offers innovative and personalized treatment options for

individuals with ASDs. Clinical results indicate that robot-enhanced therapy can obtain an equivalent performance compared to that of human standard therapy for children with ASDs (H.-L. Cao, Pablo G. Esteban, et al. 2019). As innovative technologies in robotics and artificial intelligence are included in the therapy, therapy costs diminish and overall quality of life for individuals with ASDs is improved. Incorporating robotics machines in autism therapy has consequently acquired a substantial amount of attention over the past twenty years (Diehl et al. 2012; H.-L. Cao, Pablo G. Esteban, et al. 2019) due to the potential benefits of incorporating robots in autism therapy.

Hyper-systematizing theory proposes that individuals with ASDs have a hyper-systematized brain which makes them adept at interacting within highly predictable systems such as when working with computers or machines (Baron-Cohen 2002; Baron-Cohen 2006). However, these same individuals often fail to interact in low, unpredictable systems such as in the development of interpersonal relationships or when placed in social settings. Individuals diagnosed with ASDs are influenced by entirely lawful systems, indeed highly lawful systems and in some cases moderately lawful systems, depending on the severity level of the disorder. Interestingly, individuals diagnosed with ASDs exhibit particular interest in robots because they are highly predictable machines (Robins, Dautenhahn, and Dubowski 2006).

Robotic interactions are highly motivating environments for individu-

als with ASD due the simple, predictable nature of robots compared to humans (Srinivasan et al. 2015). Predicting a behavior or intention is a fundamental aspect of all human beings as they aid in survival in changing environments. In order to predict changes in the environment, individuals utilize a process of systematization. Systematizing, that explores the law that governs the relationship between inputs and outputs, is the most powerful way to predict changes (Baron-Cohen 2006). Systems, like computers, are highly predictable with low variability while other systems, such as human behaviors, are defined by low predictability with high variation. Scientific evidence demonstrates that robots are particularly useful to improve the social skills of individuals diagnosed with ASDs (Brian Scassellati et al. 2018).

Both artificial intelligence and Machine Learning have opened the door to the creation of personalized treatments for individuals with ASDs. (Rudovic et al. 2018). Socially assistive robots (SAR) are robots which assist people in social interactions (Breazeal 2004),(Feil-Seifer and Mataric 2005). Scientific literature demonstrates a variety of studies when both robots and individuals with ASDs interact (Begum, Serna, and Yanco 2016), (Pennisi et al. 2016). The main three clinical approaches in which SAR has been used in autism are to eliciting certain behaviors, model behaviors, teaching and practicing a skill-specific task, each while providing feedback and encouragement to the individual.

Elicit behaviors imply the use of a robot as a stimulus to generate the desired behavior. Using robots to elicit behavior has not only been used in people with ASDs. For instance, (Hiolle et al. 2012) robots can help to elicit caregiving behavior in human-robot interaction. In ASDs, SAR has been used to elicit behaviors for diagnostic purposes (Hashim et al. 2013), (Tapus, Mataric, and B. Scassellati 2007) and elicit pro-social behavior (E. S. Kim et al. 2013). Modeling, teaching, and practicing skills, is one of the most commonly used approaches in autism therapies and other fields in which SAR is used (*Robots Help Teach STEM Concepts to Students With Autism* 2018),(*Towards Autonomous Robotic Systems* 2011), (Waltz 2018), (Zheng et al. 2016). Finally, some studies use SAR to provide feedback or encouragement as a part of patient care treatment (Costa et al. 2009), (Miskam et al. 2014).

2.4 Penguin for Autism Behavioral Intervention

Penguin for autism behavioral intervention (PABI ©Dickstein-Fischer) is a research platform developed by the Automation and Interventions Medicine Laboratory of Worcester Polytechnic Institute (WPI) in partnership with Salem State University(Laurie A Dickstein-Fischer et al. 2018), to guide ABA Discrete Trial Training (DTT) therapies (Pereira 2017). Figure 2.1 illustrates the research platform.

The platform includes a robot platform and an interface provided by a

Figure 2.1: Penguin for Autism Behavioral Intervention research platform

tablet. Capable of conveying facial expressions, mobilizing its body, and providing verbal feedback, the robot is both intuitive and responsive. It's tracking system collects patient information in order to gather data on the emotional state of patients. The tablet provides the necessary interface for applying ABA therapy and it is linked to the robot using a WIFI connection. A modified ABA therapy application is installed in the tablet to generate the outputs and inputs necessaries during the therapy. Figure 2.2 illustrate the flow chart of the ABA process.

Prior work on PABI included various iterations of both mechanical and software systems was realized. The last version of the robot has nine degrees of freedom, each of which is actuated by a servomotor. The servomotors are controlled by a servo-motor controller which interchange informa-

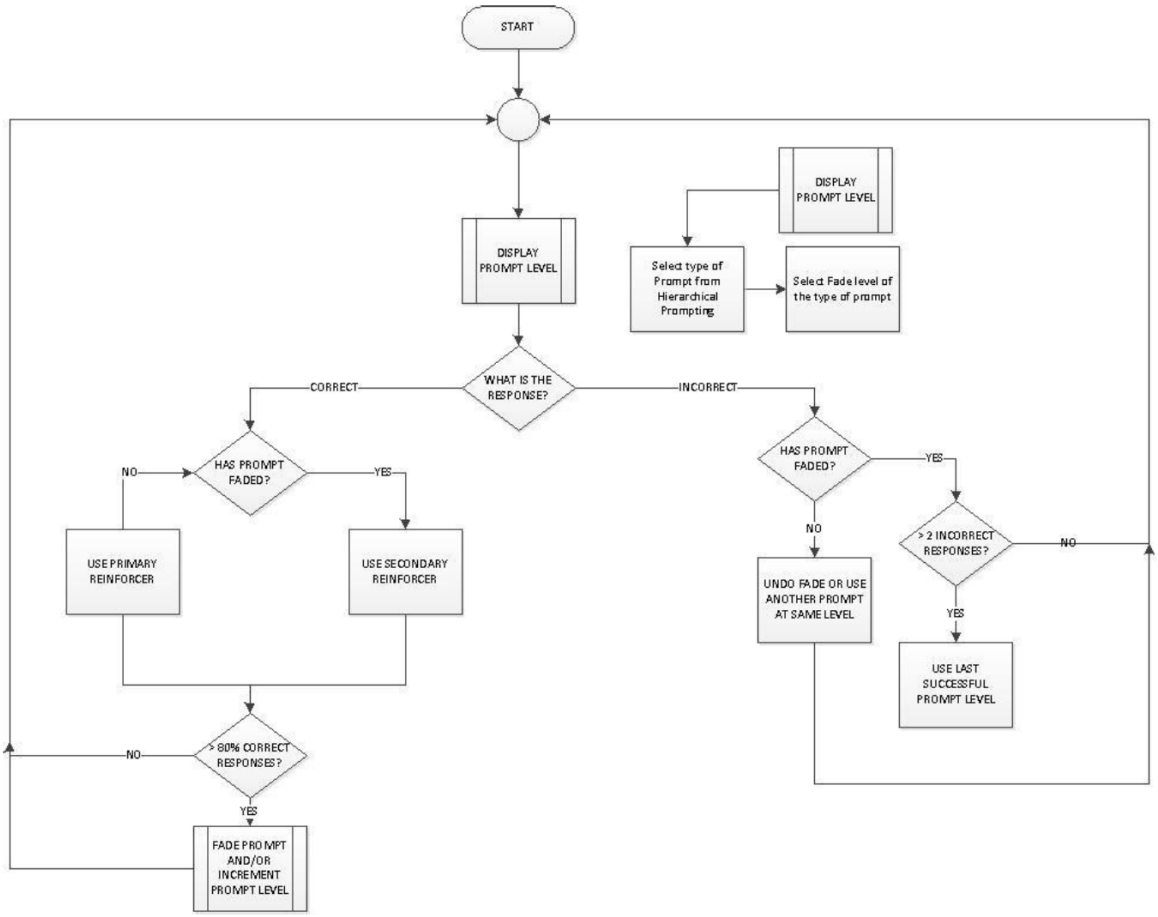

Figure 2.2: Flow chart of the modified ABA therapy. Reprinted from *Adaptive Applied Behavior Analysis (ABA) Therapy for Autistic Children using a Tablet Application alongside a Humanoid Robot*, by A. Fathima, 2017, USA, Worcester Polytechnic Institute - AIM Lab

tion with the main computer. The main computer was a PC-based x86-64 system.

The software architecture of the system used a micro-services pattern. The insight of this pattern is modularity. Thus, this pattern uses fine-grained services and lightweight protocols. This pattern is commonly used to the development in web application. PABI uses a database as the primary service, as illustrated in Figure 2.3. The software included algorithms for face detection, face landmarks detection, and tracking.

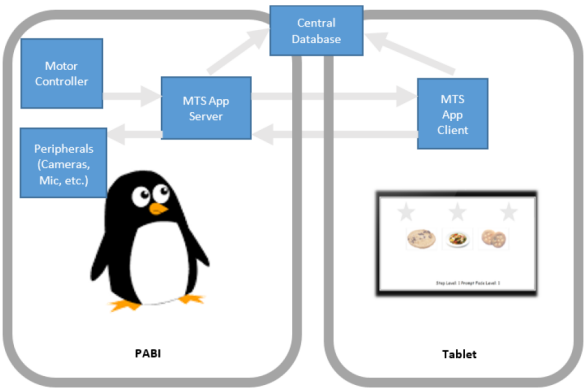

Figure 2.3: Overall Software Architecture of PABI. Reprinted from *Interactive Behavior for Humanoid Robot Mediated Applied Behavioral Analysis Autism Therapy*, by R. Pereira, 2017, USA, Worcester Polytechnic Institute

2.5 Summary

Austism Spectrum Disorders (ASDs) cannot be associated with any particular biological marker and a cure to this disorder has not yet been discovered. Thus, ASDs are of widespread concern. While symptoms vary across the spectrum, early indicators of ASDs include social exclusion, avoidance

to social situations, and sensitivity to oral textures. Individuals identified with ASDs may exhibit one or more of these behaviors. Interestingly, individuals with autism exhibit a high level of interest in machines because they are predictable systems.

Emergent technologies such as socially assistive robots (SAR) have been successfully used in recent years to support autism treatments. The three main clinical approaches in which SAR have been used in autism are: eliciting behaviors; modeling, teaching and practicing a skill; and providing feedback and encouragement. Experimental studies have shown that using socially assistive robots have a similar performance measures as human interventions.

Penguin for Autism Behavioral Interventions (PABI) is a socially assistive robot created by the AIM Lab at WPI with the aim of supporting autism treatments. PABI's architecture is especially designed for applying ABA therapy.

2.6 Thesis Contributions

Previous work with PABI has demonstrated positive results for its use in ABA therapy session. This thesis work is built upon previous research and investigation of both thesis papers and project works. This thesis has achieved the following contributions:

2.6.1 Primary Contributions

Contribution 1: Designing a robot-enhanced ABA therapy framework goes beyond the hardware architecture of the robot and provide a baseline for future innovations in both the hardware and the software of PABI.

Contribution 2: Exploring a multilayer pattern software architecture for a robot-enhanced ABA therapy framework provides a point of reference for contrasting and comparing future architectures.

Contribution 3: Benchmarking deep learning frameworks in the Jetson TX2 enhance the selection of software-hardware pair and provides a baseline for future developments.

Contribution 4: Contrasting deep learning models with another approaches for a socially assistive robot in ABA therapy, contributes to build a knowledge frame that help to build more efficient technology in autism treatment.

Contribution 5: Improving the tracking system in a socially assistive robot in ABA therapy is a keystone in the usability of the robot. The new system not only enhances the human-robot interaction but also strengthen the autonomy of the robot.

2.6.2 Secondary Contributions

Contribution 1: Writing a C++ Library with the protocol to communicate the servocontroller driver with the embedded system help to accelerate

future developments.

Contribution 2: Implementing a C++ Library with the software architecture proposed help to accelerate future developments.

Contribution 3: Analyzing the kinematics of the robotic platform and implementing a C++ Library to control the movement of the robot help to accelerate future developments.

Chapter 3

Designing a New Software Architecture and Framework for Robot-Assisted ABA Therapy

Socially assistive robots have been used in therapeutic treatments for individuals diagnosed with Autism Spectrum Disorders (ASDs) in recent years. Preliminary studies have demonstrated positive results using the Penguin for Autism Behavioral Intervention (PABI) developed by the AIM Lab at WPI to assist individuals diagnosed with ASDs in ABA therapy treatments. On the other hand, in recent years, power-efficient embedded AI computing devices have emerged as a powerful technology by reducing the complexity of the hardware platforms while providing support for parallel models of computation. This new hardware architecture seems to be an important step in the development of socially assistive robots in ABA therapy.

This chapter discusses the new hardware and software architecture of architecture for SARs and its implementation on PABI. The chapter begins with an analysis of the new power-efficient embedded AI computing

device incorporated to the hardware architecture of PABI. Additionally, the chapter focuses on the software architecture in which three main ideas are discussed: an analysis of the interactions in a robot-enhanced ABA therapy context, the design of a robot-enhanced ABA framework, and the use of a multilayer pattern for a software architecture.

3.1 Motivations and requirements

Previous experimental studies with PABI demonstrated positive results (Laurie A Dickstein-Fischer et al. 2018) while using the robot during an ABA therapy session (Pereira 2017). In recent years, power-efficient embedded AI computing devices have emerged as a powerful technology by further reducing the complexity of the hardware platforms while providing support for parallel models of computation. The small size of those devices as well as their low power consumption allow innovation in the mechanical designs of the autonomous machines thus making this new hardware architecture an important step in the improvement of socially assistive robots in ABA therapy. Motivated by the previous successful results of using PABI in ABA therapy and the emergent AI computing devices, a new version of the computing framework of PABI is proposed in this chapter. The AI embedded system architecture used in PABI includes a multicore ARM-CPU system with 256 GPU cores. While there is great interest in cloud computing and high speed networking such as 5G to offload processing and potentially further reducing costs of onboard components, at the present, there is no guarantee of adequate network coverage in the location of therapy and thus embedded intelligence is critical for widespread deployment. The small size and the low power consumption of the embedded system are also advantages that allow improvements to the mechanical design of the robot physical body. Because of the ARM processors, the previous software

architecture developed for PABI cannot be deployed on the new hardware. Therefore, a new software architecture exploiting the computational power of the new hardware reduces previous limitations.

3.2 Primary Contributions

The primary contributions of this chapter are two-fold. A robot-enhanced ABA therapy framework is proposed and later a multilayer software architecture for the new hardware architecture of PABI is designed. The robot-enhanced ABA therapy framework is the result of analyzing a robot-enhanced ABA therapy scenario combined with best practices in the design of social machines. The main components of a robot-enhanced ABA therapy are extracted and conceptualized in several systems to build the framework. The systems are then combined to provide reactive and deliberative social behaviors in the robot. A software architecture using a multi-layer pattern in proposed. The proposed architecture articulates the robot-enhanced ABA therapy framework with the embedded platform used by PABI.

3.3 Background and Related Work

Diverse architectural patterns to build multiple software architectures have been developed.(Raj, Raman, and Subramanian 2017). However, architec-

tures for social robots contribute an additional layer of complexity in the design process because of the inherent constraint within those systems. Thus, social robots usually require low-level characteristics like real-time, multitasking, modularity, etc., combined with high-level traits such as social behaviors and social interactions, among others. Social robots are also implemented over embedded systems that add additional constraints to the design. To overcome these challenges, different architectures have been proposed in previous literature. (Ahmad and Babar 2016; H.-L. Cao, Pablo Gómez Esteban, et al. 2017). For instance, Breazeal proposed a combination of both low and high level procedures to a human-inspired framework for a synthetic nervous system (Breazeal 2004).

In another research project, Kim (J. Kim et al. 2008) explored a different approach using an intelligent software architecture that consisted of deliberate, sequenced, and reactive behaviors. (Pakkar et al. 2018), on the other hand, simplified the design for a low cost social robot platform. Research Ferland (Ferland et al. 2013) presents an approach especially focused on interaction. Finally, Saerbeck (Saerbeck 2009) summarized best practices in the design of social robots, presenting a framework for the design of software architectures of social robots.

3.4 PABI Hardware Architecture

As Figure 3.1 illustrates, the new version of PABI is integrated using five systems. The heart of the new architecture is a multicore ARM-based CPU with 256 GPU cores embedded system. This high-performance system provides the computational power as well as additional capacities such as wifi connection, SPI, I2C, UART protocols among other characteristics. The sensory system perceives both verbal and non-verbal clues during the entirety of an ABA therapy session. Two systems are used for the communication with the patient and therapist. First, the interface system provides the communication between the patient and intelligent ABA tutoring system. Second, the physical body system is responsible for the expressiveness of the robot. The power system is the last system and it is responsible for providing a source of energy for the hardware architecture.

3.4.1 The Embedded System: NVIDIA Jetson TX2 Platform

Jetson TX2 is a low-power embedded platform with specific focus on accelerated AI-based applications. This embedded platform is ideal for autonomous machines which require high computational power with low-power consumption. The Jetson TX2 can compute 1.3 Teraflops with 7.5W-15W power consumption. Additionally, the small size of this system (50mmx87mm) enables ease of incorporation in robotics systems. A difference of systems like a PC-based system, this embedded platform can

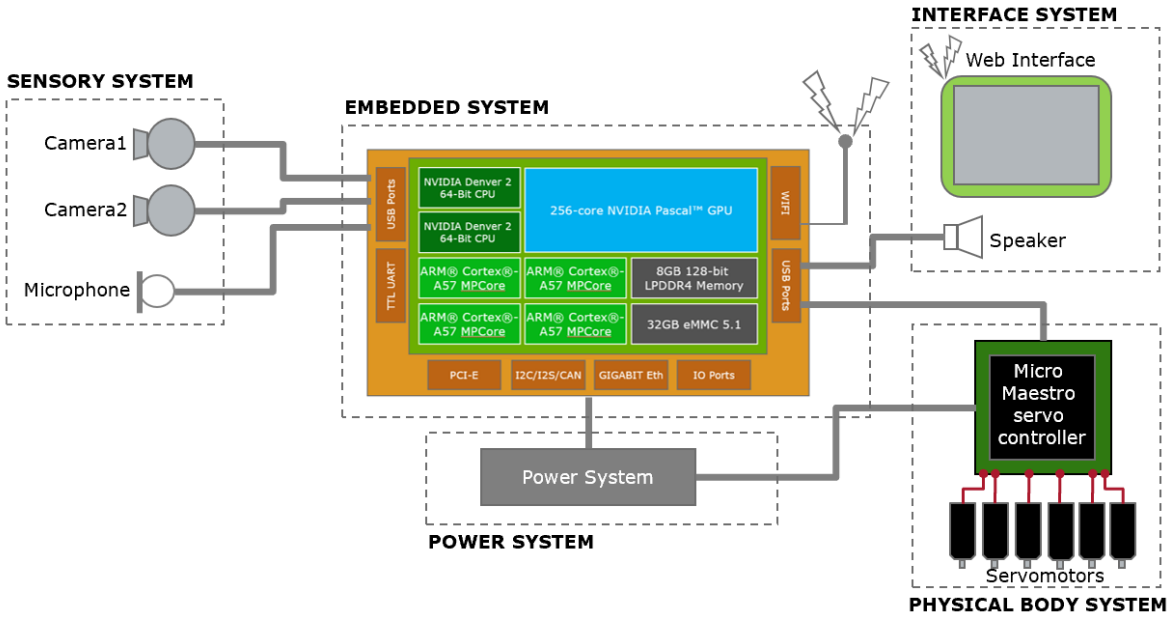

Figure 3.1: Proposed new hardware architecture for Penguin for Autism Behavioral Intervention

communicate with additional devices using different standards such as TTL UART, I2C, SPI, CAN, among others. Figure 3.2 illustrates the Jetson TX2 architecture.

The heart of the JetsonTX2 is an NVIDIA Tegra Jetson TX2 System-On-Module Block Diagramystem-on-Chip (SoC) module. This module incorporates both CPU and GPU processors in one chip. Tegra X2 includes four 64 bits ARM® Cortex®- A57 MPCore, two 64 bits NVIDIA Denver 2, a 256-core NVIDIA Pascal™ GPU and 8GB 128-bit LPDDR4 Memory. Figure 3.3 illustrates the components of the Tegra X2.

Different modes of operation control the performance of the Jetson TX2. Thus, the embedded system can switch from a low-power consumption mode to a high-power computation mode. This characteristic allows build-

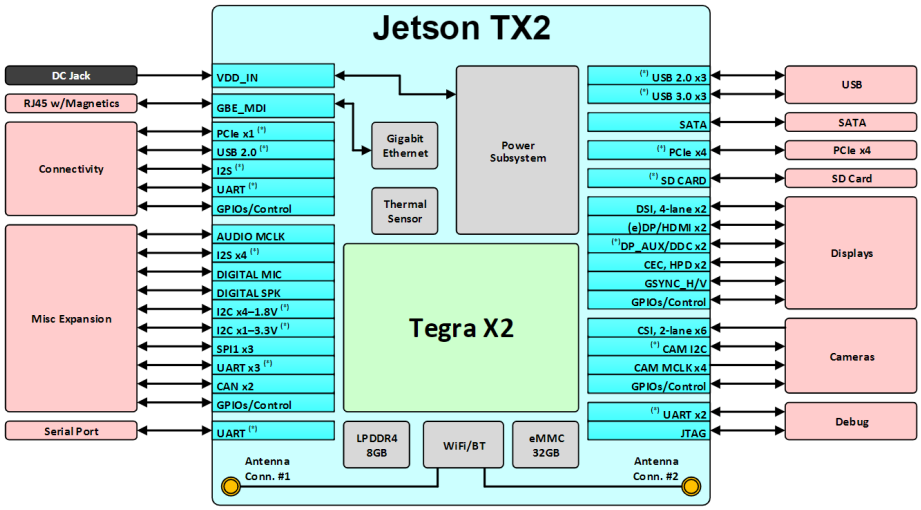

Figure 3.2: Jetson TX2 Block Diagram. Reprinted from *Nvidia Tegra X2. Architecture and Design* by J.Barker, B.Wu., 2017, Rochester Institute of Technology

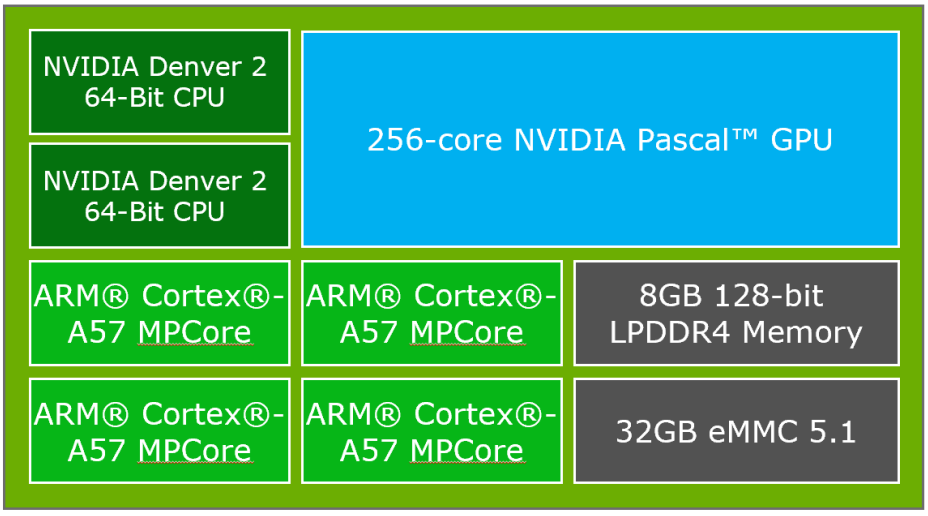

Figure 3.3: Tegra X2 SoC Architecture

Table 3.1: Jetson TX2 modes of operation

Mode	Mode Name	Denver 2	Frequency	ARM 57	Frequency	GPU
0	MAX-N	2	2.0GHz	4	2.0GHz	1.3GHz
1	MAX-Q	0		4	1.2GHz	0.85GHz
2	MAX-P Core All	2	1.4GHz	4	1.4GHz	1.12GHz
3	MAX-P ARM	0		4	2.0GHz	1.12GHz
4	MAX-P Denver	1	2.0GHz	1	2.0GHz	1.12GHz

ing more power-efficient autonomous machines. Table 3.1 illustrates the Jetson TX2 operation modes.

Three modes of operation are present in the Jetson TX2. They include: Best power-efficiency mode (MAX-Q), best power and performance balance mode (MAX-P), and best computation performance mode (MAX-N). In MAX-Q mode, this embedded system consumes only 7.5W and uses four processors plus the GPU power. MAX-P mode offers additional variations: MAX-P Core All, MAX-P ARM, and MAX-P Denver. Distinct CPUs configurations are established in each case, as Table 3.1 illustrates. Power consumption in this mode varies between 7.5W and 15W, depending on the selected configuration. In MAX-N mode, all CPUs and the GPU at maximum frequency are used. The power consumption in this mode is 15W, and the maximum computational power of the Jetson TX2 is accessible.

3.4.2 The Sensory System

The sensory system illustrate in Figure 3.1 is responsible for perceiving stimuli during the ABA therapy session. The sensory system is consisted

of two USB cameras model USBFHD01M and a USB microphone. The USB camera produces 1280x760 frames at 60 frames per second. Fish-eye lens are mounted on the USB cameras capable of observing the physical environment, broadening its field of vision.

3.4.3 The Interface System

The interface system is the mechanism used for communication between the patient and the intelligent ABA tutoring system (Figure 3.1. This system also can be used by the therapist to set parameters during the session. PABI uses a tablet as the primary interface system. The tablet is linked with the robot's web server through a wifi connection. Any device that supports wifi connections and web browser navigation can be used as part of the output system. The interface system also includes a speaker to generate verbal positive reinforcers aimed to increase the likelihood of a particular behavior in the patient with ASDs during the therapy.

3.4.4 The Physical Body System

A robotic platform constitutes the physical body system. The platform includes ten degrees of freedom, each of which is actuated by a servomotor (Figure 3.1. The heart of the physical body is the servo controller "Micro Maestro servo controller". The servomotor controller has a USB connection within the embedded system.

3.4.5 The Power System

It provides the electricity to the system. The Jetson TX2 runs off of 12V DC power and that while it is typically run off of a wired power adapter it is designed to be portable and run off of a battery. The embedded system as a low power consumption.

3.5 Robot-Enhanced ABA framework and software architecture

3.5.1 Robot-Enhanced Therapy Analysis

Figure 3.4 illustrates a robot-enhanced ABA therapy session using PABI. The session is integrated by the therapist who can be a behavior analyst, teacher, aide, parent, or other caregiver, the patient , and the robot. Different stages are executed during the therapy session. First, prior to the session the therapist configures the goals of the therapy session and inputs the patient's information. Second, during the session, PABI generates social behaviors to help the patient to achieve these therapeutic goals. Social behavior generated by PABI is affected by three factors: the information provided by the therapist before the session, the information received through the interface system, and PABI's perception of the session. The data initially provided by the therapist assists PABI to establish proper robot behavior during the session.

Information such as the patient's severity of ASDs, session goals, among others, can help set the parameters for PABI's social interaction during the therapeutic session. The information received through the interface system determines the kind of reinforcement that PABI provides to the patient. For example, if during the ABA therapy, the child's response to a predefined stimulus is correct, PABI will intuitively provide a positive reinforcement to the patient.

Likewise, if the response is wrong, PABI provide the correct response to encourage the child to improve during the next attempt. The sensory system of the robot will provide the information in real-time to serve in future behaviors. For example, tracking is an action that the robot makes based on its perception of the scene. The third stage during a therapy session is data processing. After the session, data collected is analyzed to improve the therapy and the social behavior of the robot.

Figure 3.4 illustrates the interaction in an enhanced-robot ABA therapy session. Before the session the robot interacts more in the therapist loop; however, during the therapy the robot is focused on the patient loop.

3.5.2 Robot-Enhanced ABA Therapy Framework

The basic scenario shows that different processes and interactions during an ABA therapy session exits. As Figure 3.6 illustrates, seven subsystems have been identified. First, a real-time low-level feature extraction system

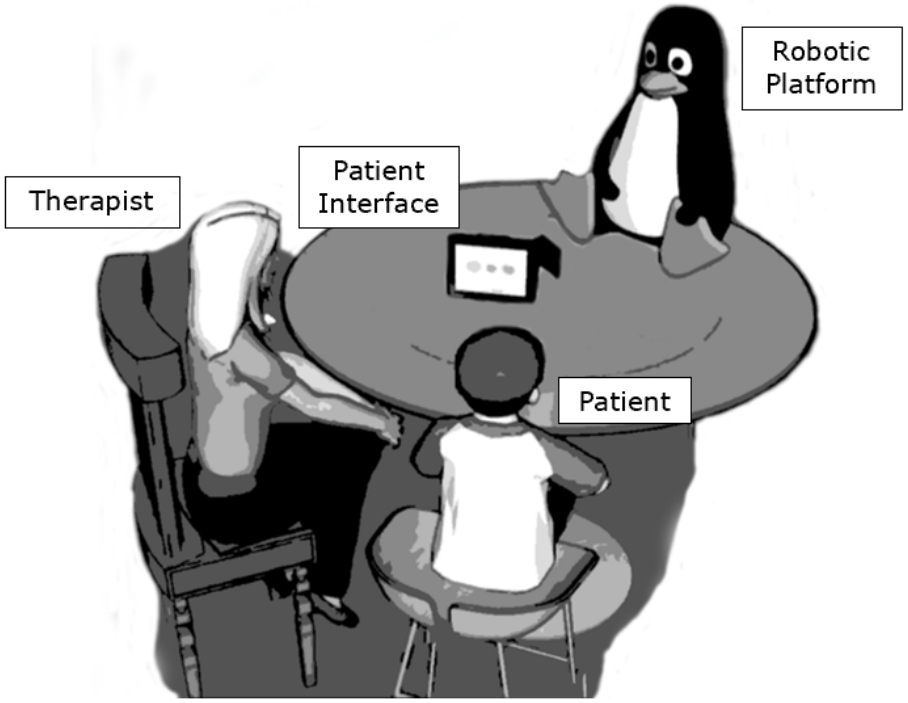

Figure 3.4: Example Configuration of a Robot-Enhanced Therapy Session. Adapted from *Interactive Behavior for Humanoid Robot Mediated Applied Behavioral Analysis Autism Therapy*, by R. Pereira, 2017, USA, Worcester Polytechnic Institute

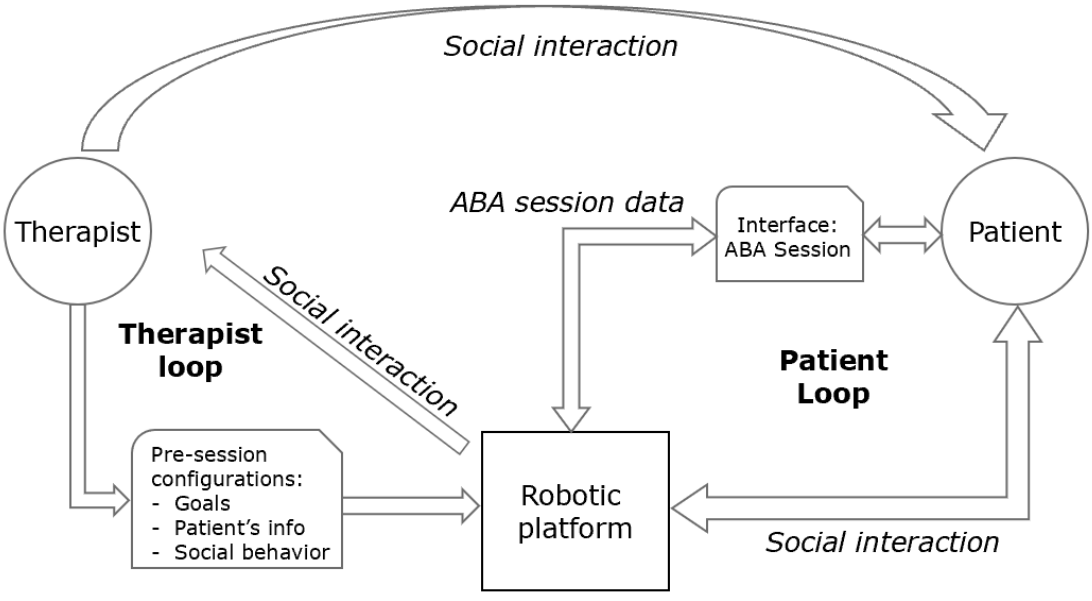

Figure 3.5: Robot-enhanced ABA Therapy Interaction

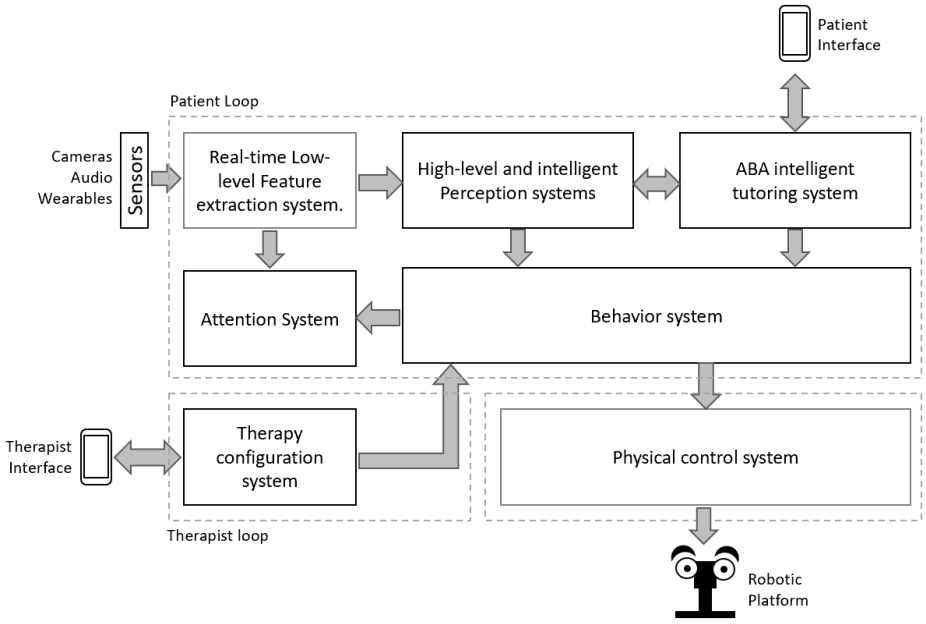

Figure 3.6: Robot-enhanced ABA Therapy Framework

analyzes the information of the scene and extract low-level features. The extracted features are consisted of an additional two systems: the attention system and the high-level and intelligent perception system. The first system is responsible for maintaining the robot focus on the patient or the therapist. The second system combines the low-level features to produce stimuli that are consumed by the behavior system. The ABA intelligent tutoring system is focused on applying ABA therapy to the patient while the therapy configuration system is focused on the therapist. Finally, a physical control system is responsible for producing the robot's physical movements.

The seven subsystems proposed allow to the robot exhibit both reactive and deliberative behaviors. The reactive behavior maintain the social interaction while the deliberative behavior allows the therapeutical use of

the robot. Subsystems are described below and summaries of the primary tasks are included.

Real-time Low-Level Feature Extraction System

Scene perception and extraction of features than help the robot process the data is one of the first stages of social interaction. The features are extracted from different sources like images, sounds, or other kind of data. The real-time low-level feature extraction is the system responsible for reading to raw data and transforming it into useful features. Real-Time processing is one of the keystones in this module. The faster the system can produce the features, the faster the robot can react to the changes in the scene. Some examples of low-level feature face detection, landmark detection, pose detection, among others.

Attention System

A scene can have different stimuli. The attention system selects one of them and excludes extraneous information. This system is important because it helps the robot to allocate its limited resources in the most essential tasks. In the ABA therapy, one of the most important tasks is monitoring the state of the patient in order to maintain interactions.

High-Level and Intelligent Perception System

This system is responsible for organizing, combining, and interpreting low-level features in order to have a better comprehension of the scene. This system also combines the low-level features with the goals set by the therapist or the learning progress exhibited by the patient. As a result of this combination, a stimulus is generated for the behavior system.

Behavior System

Different social behaviors can be observed by the robot during ABA therapy. Those behaviors are designed to have a therapeutic effect on the patient. From the robot's perspective, each behavior is a well defined state. Thus, the social behavior system can be considered a finite state machine. Every low-level feature generates by the low-level feature extraction system can be observed as an element in a features-alphabet. The high-level intelligent perception system uses the alphabet to generate strings that will be applied to the behavior system, the finite state machine of the robot. Figure 3.7 illustrates the process.

ABA Intelligent Tutoring System

One of the essential objectives of PABI is to provide support during the ABA therapy session. The ABA intelligent tutoring system is responsible for applying and evaluating the ABA therapy. Applying Intelligent tutoring

systems in ABA therapy can be used to create personalized treatments.

Therapy Configuration System

This system provides the tools and interfaces to the therapist to set the robot-enhanced social intervention. Through the interfaces, the therapist can introduce not only descriptive information of the therapy but also can configure the robot with new social behaviors, or new intervention, for instance.

Physical Control System

This system is responsible for controlling the physical robotic platform and show the expressiveness of the robot. It is directly linked with the behavior system. The physical control system includes the servomotor's control, kinematic analysis, protocols of communications, among other tasks.

The previous robot-enhanced ABA framework exhibit constraints that the software architecture needs consider. Below these restrictions are described.

Real-Time Performance

Interaction with the patient needs a real-time approach to create a positive perception of the robot. Actions like face detection, pose detection, tracking, among others, need to respond immediately maintain the robot involved in the session.

Multitasking

The robot-enhanced ABA therapy framework previously analyzed is constituted by various systems and several tasks need to be executed simultaneously. Some examples of tasks that need to be run concurrently are social interaction, communication with the external interface, communication with the servomotors, among others.

Modular

Including modularity in a software architecture provides some advantages. From a develop point of view, modularity allows build and test systems independently. From a robot perspective, modularity helps to build fault tolerant system.

Expandable

Understand the scene of the therapy requires a low-level feature that can change in the future. Some new features can be added and some eliminated. Therefore, being expansible is a useful capacity of the system.

3.5.3 Multilayer Software Architecture

A multilayer software architecture is proposed to combine the robot-enhance ABA therapy framework with the AI embedded platform. Because of the deep learning approach of the Jetson TX2, deep learning frameworks and

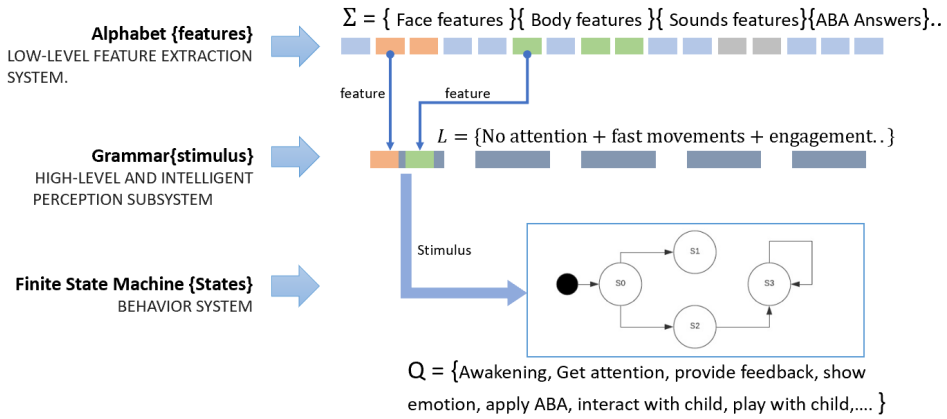

Figure 3.7: Behavior System Generating Process of a Stimulus

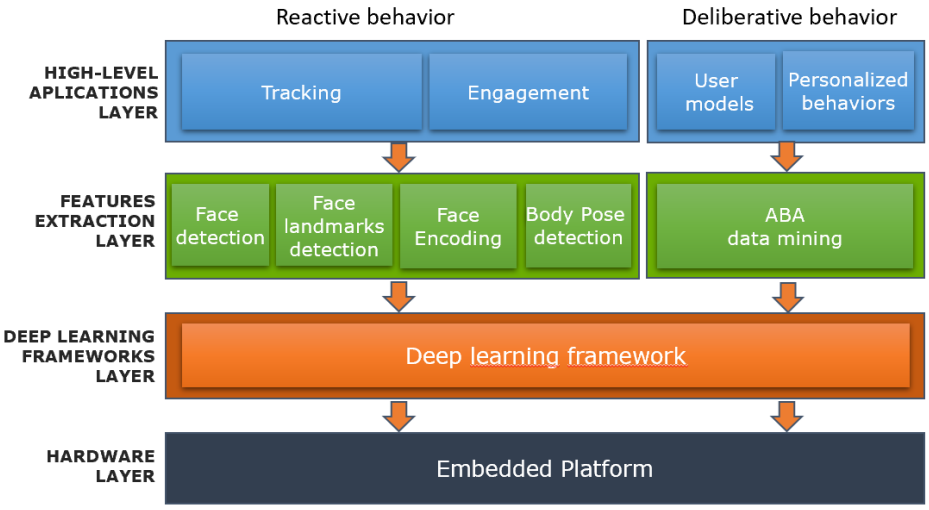

Figure 3.8: Layer Architecture for Implementation on a Robot-Enhanced Framework on Jetson TX2

models play a key role in the proposed software architecture. The multi-layer architecture also allows building complex applications while maintain modularity and expandability. The use of deep learning models also helps to execute complex tasks in real-time.

Each layer proposes a different level of abstraction. Figure 3.8 illustrate the multilayer architecture proposed.

Below the layers are described, and the main function of each of them

is then summarized.

Hardware Layer

This layer is an abstraction of the hardware architecture that supports the upper layers. PABI uses the Jetson TX2 embedded system in its hardware layer.

Deep Learning Frameworks Layer

This is the second layer of abstraction. The layer implements a deep learning framework that takes advantage of the multicore ARM CPU and GPU power of the platform.

Features Extraction Layer

This is the third layer of abstraction. In this layer, deep learning pretrained models are deployed. The objective of this layer is extracting the features necessaries for supporting the next layer.

High-Level Application Layer

This is the fourth layer of abstraction. In this layer, applications that support the social behavior of the robot, as well as intelligent applications, are implemented. The applications communicate between them to create complex systems.

3.6 Summary

In this chapter, the hardware architecture of PABI is analyzed, a robot-enhanced ABA therapy framework is created, and a multilayer software architecture are designed.

The heart of the new hardware architecture of PABI is a power-efficient embedded AI computing device (NVIDIA Jetson TX2). This embedded system is characterized by providing CPU and GPU support. The central System-on-Chip (SoC) of the system is a NVIDIA Tegra X2 which provides a multicore ARM-CPU system that includes a 256 GPU core system. Both the CPU and the GPU systems can be reconfigured to achieve different results.

The robot-enhanced ABA therapy framework is designed as the result of the analysis of a robot-enhanced ABA therapy scenario combined with the best practices in the design of social machines. The design began by analyzing the interactions between the patient, the therapist, and the robot in a robot-enhanced ABA therapy session. Then, the main components of the interaction are extracted and conceptualized into several systems that constituted the proposed framework. Seven systems integrate the framework and provide reactive and deliberative social behaviors in the robot.

Finally, a 4-layer software architecture is designed, articulating the robot-enhanced ABA therapy framework with the AI embedded platform used

in PABI. The designed architecture creates a multitasking, real-time, and modular platform, taking advantage of the parallel model of computing of the embedded system.

Chapter 4

Exploring the Impact of Deep Learning Frameworks: Implementing a Real-Time and Multitasking Robot-Enhanced ABA Software Architecture

Multitasking and real-time are two characteristics required by a robot-enhanced ABA therapy framework and the software architecture detailed in Chapter 2. The parallel model of computation of the AI computing device used by PABI can support both of them. However, this embedded system must be combined with software that properly exploits its architecture.

In this chapter, a benchmark of three deep learning frameworks is presented. TensorRT, Caffe, and the deep learning module of OpenCV are evaluated on the NVIDIA Jetson TX2. A within-subject multifactorial experimental design was implemented for the analysis. Two variables are studied: the velocity of the inference engines and resources usage of each framework. The first variable is essential for building real-time systems, while the second is vital for building multitasking systems. Two factors

were used for testing conditions: workload and hardware configuration. The experimental results are discussed at the final of the chapter.

4.1 Motivations and requirements

A socially assistive robot in the context of ABA therapy involves tasks like image processing, servomotors control, social behavior, among others. Some tasks, for example, image processing, require high computational power. Others, such as social behavior, require that different actions be executed simultaneously. The multilayer architecture proposed in Chapter 2 included both multitasking and real-time processing characteristics.

Appropriate pairing of both software and hardware is essential when using an embedded system as the core computational system of an autonomous system. An appropriate hardware-software match improves critical tasks on the robots. Therefore, a device like the Jetson TX2 must be combined with a software that exploits its architecture properly. Because of the parallel computing approach of the Jetson, deep learning frameworks are studied in this chapter. Several studies analyzing deep learning framework on PC-based platforms exist in the literature. However, there are no formal studies that exist about how deep learning frameworks perform on the Jetson TX2.

4.2 Primary contributions

The aims of this chapter are two-fold. First, a benchmark of three popular deep learning frameworks is underscored using the Jetson TX2 as the hard-

ware platform. Three different modes of operation of the Jetson TX2 are utilized for the benchmark: the most power-efficient mode (MAX-Q), the best power efficient and computational performance balance mode (MAX-P), and the best computational performance mode (MAX-N). For each mode of operation, an analysis of the inference time and usage of hardware resources by every deep learning framework is analyzed. The second contribution presented is the implementation of an abstraction layer for the bench-marked deep learning frameworks. The abstraction objective is to build an expandable architecture for the robot, making it adaptive to change and social in essence when interacting interacts with patients in ASDs therapy sessions.

4.3 Background and Related work

Deep learning has allowed a dramatic improvement in areas such as computer vision, speech recognition, natural language processing, etc. (LeCun, Bengio, and Hinton 2015). In fact, artificial neural networks have outpaced human capacities in some tasks. For instance, ImageNet illustrates that a state-of-the-art neural network reaches an error rate of 3.57% in image classification while the human error rate is 5.1%.(Russakovsky et al. 2015). This demonstrates the ability of artificial neural networks outperform humans in image classification and they have a smaller margin of error as compared to humans.

Modern GPU-based hardware architectures have played a vital role in the success of deep learning. Parallel models of computation of the GPUs has reduced the training time of artificial neural networks. Embedded GPU-based systems play a vital role in deploying pre-trained neural networks. These integrated systems have assisted in the implementation of real-time applications, popularizing the deep learning approach.

Development of open-source frameworks have had an essential role in the successful implementation of deep learning by facilitating the process of training and deployment of artificial neural networks. Deep neural networks (DNN) frameworks like Caffe from UC Berkeley (Jia et al. 2014), CNTK from Microsoft (Deng 2012), TensorFlow from Google (Abadi et al. 2016), Torch (Collobert, Kavukcuoglu, and Farabet 2011), Nvidia TensorRT, for example, have been developed in the past decade.

Frameworks with a C++ API support are underscored in this chapter: Caffe, TensorRT, and the deep learning module of OpenCV (DNN OpenCV). Caffe is one of the earlier frameworks and it has been used to train different networks architectures. Consequently, a plethora of pre-trained models are available for this framework. NVIDIA TensorRT is a framework included with the JetPack 3.2 set of libraries provided by NVIDIA and installed in the Jetson TX2. Finally, the deep neural module of OpenCV (DNN OpenCV) was selected because OpenCV has been utilized as the development framework of the PABI project.

Below the frameworks are described, and the main characteristics of each is summarized.

4.3.1 CAFFE Framework

. Caffe was developed by the Berkeley Vision and Learning Center (BVLC) at UC Berkeley and released in 2014 under a BSD 2-Clause license. (*caffe.berkeleyvision.org* 2019)

This framework allows for training, testing, fine tuning and deployment of deep neural network architectures. Caffe combines GPU and CPU implementations and its libraries are C++ based. However, it has Python and Matlab bindings for rapid prototyping.

CUDA is used to implement the parallel programming model in this framework. The CPU-GPU implementation uses a synchronized memory model to share data. In this memory model, data is transferred by the framework on demand between the CPU memory and the GPU memory. Figure 4.1 shows the synchronization model. Once a pre-trained model is loaded into the memory, the model is processed by the Caffe engine using a CPU inference or a GPU inference that is dependent upon the hardware capacities.

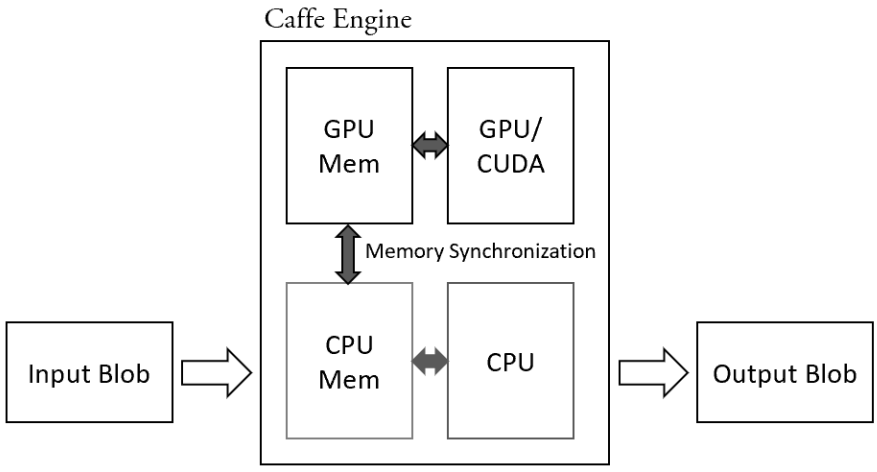

Figure 4.1: Synchronization Model of Caffe Engine

4.3.2 NVIDIA TensorRT Framework

NVIDIA developed TensorRT focused on leveraging the parallel programming model of its GPUs. A core distinction between Caffe and TensorRT is regarding the high performance achievement in the inference process of a deep learning model. Once a network is loaded in memory, TensorRT builds and optimizes an inference Engine. The optimization procedure maximizes throughput by quantitating models FP16 or INT8 without loss of accuracy, optimizing GPU memory.

Figure 4.2 shows the NVIDIA TensorRT optimization model. Similar to Caffe, TensorRT libraries are C++ based.

4.3.3 Deep Learning module from OpenCV framework

.

The deep learning module for OpenCV (DNN-OpenCV) was developed

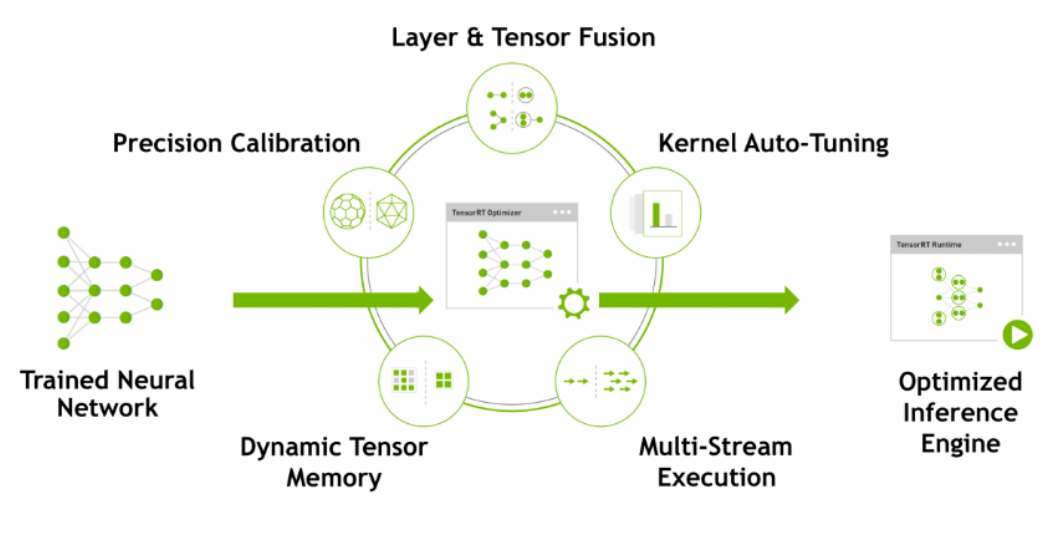

Figure 4.2: NVIDIA TensorRT Optimization Model. Reprinted from *Nvidia Developer*, by NVIDIA, 2019, Retrieved from https://developer.nvidia.com/tensorrt, Copyright 2019 by NVIDIA.

to expand the capacity of this popular computer vision framework (*Deep Learning in OpenCV* 2019). DNN-OpenCV allows training, testing, and deployment of deep neural network architectures. Remarkable aspects of the DNN-OpenCV include the following features: open-source, simplicity of use, high compatibility with pre-trained models from other frameworks, and the optimization of the networks through the use of OpenCL. A key difference of Caffe and TensorRT is that DNN OpenCV does not support a parallel computation model through CUDA.

4.4 Experimental methods

An within subject multi-factorial experimental design was used to benchmark the deep learning frameworks. Two factors were used for the experi-

ment design: mode of operation of the Jetson TX2 and workload. For the first factor, three different modes of operation of the Jetson TX2 were used: MAX-Q, MAX-P, and MAX-N. For the second factor, two levels were chosen: a high number of parameters and a low number of parameters. Caffe, TensorRT, and the DNN-OpenCV, was selected as experimental units.

The inference time and the GPU/CPU usage were measured for each deep learning framework. The inference time was measured by tracking the number of clock-cycles used in a framework to propagate input data through the artificial neural network. The input data was propagated several times through each deep learning framework and the number of clock-cycles used each time was registered. The average time is reported. Batch processing performance was not evaluated during the experiments.

In order to measure the Jetson TX2 resource usage, a shell utility included with JetPack called tegrastats were used. Tegrastats were configured in order to obtain the Jetson TX2 status every 100 milliseconds and save the samples in a log file. The following command line: ./sudo tegrastats –interval 100 –logfile data.txt was used to set the configuration of tegrastats. Below an example an example of a tregrastats is presented:

RAM 7405/7851MB (lfb 2x512kB) CPU[0%@1463, 0%@345, 0%@345, 0%@1575, 0%@1575, 0%@1575] EMC_FREQ 1%@1866 GR3D_FREQ 40%@114 APE 150 MTS fg 0% bg 0% BCPU@34.5C MCPU@34.5C GPU@40.5C PLL@34.5C AO@34C Tboard@31C Tdiode31.5C PMIC@100C thermal34.5C VDD_IN 2074/2074 VDD _CPU 307/307 VDD_GPU 153/153 VDD_SOC 460/460 VDD_WIFI 19/19 VDD _DDR 363/363

Table 4.1 illustrates the meaning of each field. The logfile was later

Table 4.1: *Tegrastats* fields description

FIELD	MEANING
RAM 7405/7851MB (lfb 2x512kB)	Used RAM/Total RAM (number of blocks and size of the larger block)
CPU [0%@1463,0%@345,0%@345, 0%@1575,0%@1575,0%@1575]	CPU's statistics. CPU usage%frequency
EMC_FREQ 1%@1866	Percent of the EMC memory bandwidth being used/ EMC frequency in MHz
GR3D_FREQ 40%@114 midrule APE 150	GPU statistics. GPU usage%frequency Audio Processing Engine frequency
MTS fg 0% bg 0%	Time spent in foreground tasks, Time spent in background tasks
BCPU@34.5C MCPU@34.5C GPU@40.5C PLL@34.5C AO@34C Tboard@31C Tdiode@31.5C PMIC@100C thermal@34.5C	Temperatures in Celcius in different reference points on the Jetson TX2. According to NVIDIA, the PMIC doesn't allow software to read an actual temperature. Software can only read whether various thermal thresholds have been crossed. The first threshold is higher than 100C. So, the PMIC driver *assumes* temperature is 100C.
VDD_IN 2074/2074 VDD_CPU 307/307 VDD_GPU 153/153 VDD_SOC 460/460 VDD_WIFI 19/19 VDD_DDR 363/363	Voltages in different reference points on the Jetson Tx2.

processed using a script in Matlab.

Hardware Platform

The JETSON TX2 platform and JetPack 3.2 were installed to conduct the experiments. The JetPack 3.2 includes Ubuntu 16.04 LTS and CUDA 9.0. OpenCV 3.4.1 for ARM processors and CUDA support were built from the source code, installed, and later used for the experiments. Three operation modes for the JETSON TX2: best power efficiency (MAX-Q), power and performance balance (MAX-P), and best performance (MAX-N) were selected. The details of the hardware configuration for each mode are shown below in Table 4.2.

Table 4.2: Hardware configuration for the experiments

Mode	Mode Name	Cores	GPU
0	MAX-N	4 ARM57 CPUs at 2.0 Ghz 2 Denver CPUs at 2.0 Ghz	256 cores at 1.30 Ghz
1	MAX-Q	4 ARM57 CPUs at 1.2 Ghz	256 cores at 0.85 Ghz
2	MAX-P Core-All	4 ARM57 CPUs at 1.4 Ghz 2 Denver CPUs at 1.4 Ghz	256 cores at 1.12 Ghz

Table 4.3: Deep Learning Frameworks used for the experiments

Framework	Version	Additional Information
DNN-OpenCV	3.4.1	
Caffe	1.0.0 rc3	Caffe version that supports SSD architecture
TensorRT	3.0	Include in JetPack 3.2

Deep Learning Frameworks

Three popular deep learning frameworks with C++ API interface available was selected for the experiments. The software versions and related libraries in the experiments are shown in Table 4.3. For TensorRT a FP32 quantization model was used in all experiments.

Neural networks and input data

. Two models that can be loaded and deployed in all three of the deep learning frameworks tested were selected. The first model was GoogLeNet(Szegedy et al. 2014). GoogLeNet is focused on image classifying. It was the winner of the ImageNet Large Scale Visual Recognition Competition in 2014. szegedyGoingDeeperConvolutions2014. Both TensorRT and OpenCV uses

Table 4.4: Experimental setup of neural networks

Deep Learning Model	Input	Output	Layers	Number of parameters
GoogLeNet	224x224	1x20	22	6 Millon
OpenPose	320x240	57x30x40	∼90	5 Billon

a Caffe version of GoogLeNet to evaluate its frameworks. A Caffe version of a GoogLeNet model is included with JetPack 3.2 which has been used for the experiments. GoogLeNet is considered to have a low number of parameters model in the experiment.

The second model is OpenPose. This is a relatively new model focused on human pose estimation. It was developed by the Perceptual Computing Lab at Carnegie Mellon University (Z. Cao et al. 2017).This model Won the Commons Objects in Context keypoints challenge in 2016 and it has been selected for two reasons. First, the model has a high number of parameters requiring a high level of computation. Second, this model can be useful in the ABA therapy to understand a patients' behavior. Table 4.4 shows the details of the neural networks configuration.

Experiment Procedure

The experiments were implemented using OpenCV 3.4.1 with CUDA support and C++11. Four stages were considered in the realization of each experiment. First, a baseline was built. The usage of Jetson TX2 resources by the operating system was measured within the span of ten seconds.

The input data was then loaded to memory and the inference engine was

built. Only one framework was tested every time. Next, prior to the start of the inferences, a pause of four seconds was realized for the posterior analysis of the data. This ensured separation of the build-time from the inference-time. Finally, the inference of the input data two executed hundred times. Figure 4.3 shows the four stages of the experiment and an example of the GPU performance after the experimentation stages.

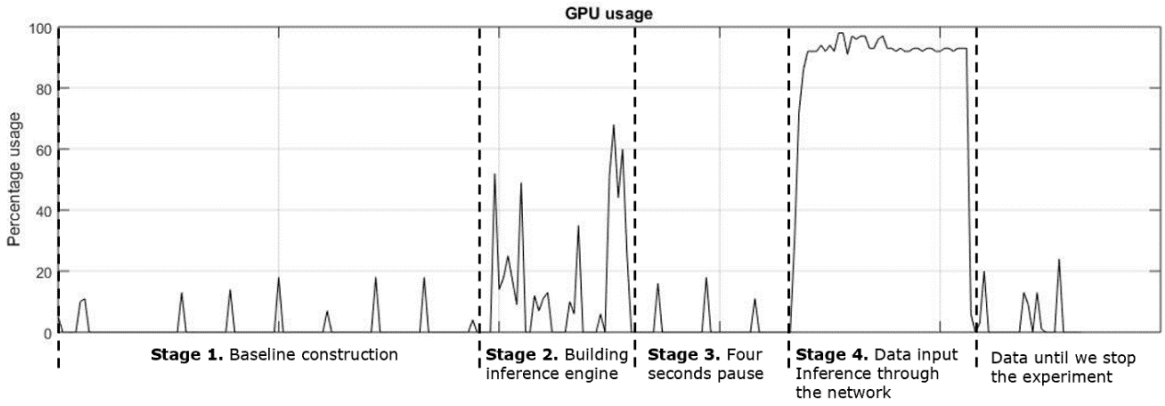

Figure 4.3: Stages during the experiments

Table 4.5 illustrates experiments conducted.

4.5 Experimental Results

The experimental results are organized into two sections. The results of using a low-parameters model and the three modes of the Jetson TX2 are presented in the first section. The results of using a high-parameters model and the three modes of the Jetson TX2 is presented in the second.

Table 4.5: Summary of experiments

Frameworks	**Jetson TX2 modes**	**Deep learning models**	
		GoogLeNet	OpenPose
DNN-OpenCV	MAX-Q	x	x
	MAX-P Core All	x	x
	MAX-N	x	x
Caffe	MAX-Q	x	x
	MAX-P Core All	x	x
	MAX-N	x	x
TensorRT	MAX-Q	x	x
	MAX-P Core All	x	x
	MAX-N	x	x

4.5.1 Experimental Results Using GoogLeNet model

Experimental Results in MAX-Q Mode

Caffe was the fastest framework in this test. It was 6% faster than TensorRT, and 88% faster than the DNN module of OpenCV. The inference time was 31ms for Caffe, 33ms for TensorRT and 278ms for DNN-OpenCV as is illustrated in Figure . 4.4

Caffe used 93% of GPU's cores of the Jetson TX2 for the deep learning inference process. This indicates a 40% difference as compared to the percentage of GPU's cores used by TensorRT. Caffe was also the framework with the lower use of the CPUs, leaving them available for other processes. A singular high usage of CPU number four can be noted in Figure 4.5. Overall, Caffe used only 22% of the total CPU power of the Jetson TX2.

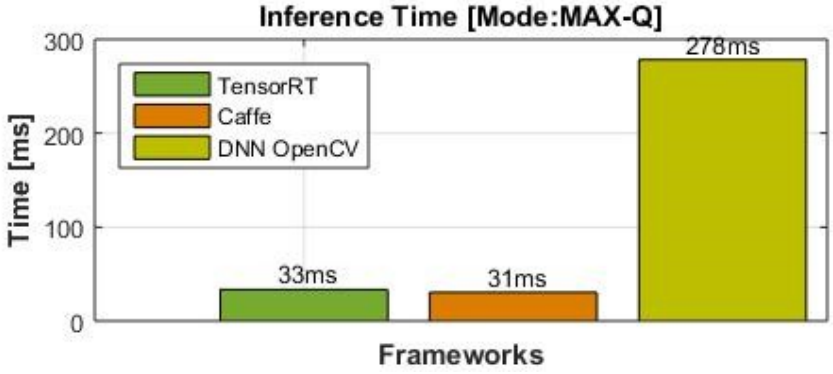

Figure 4.4: Inference Times in MAX-Q Mode Using GoogLeNet model

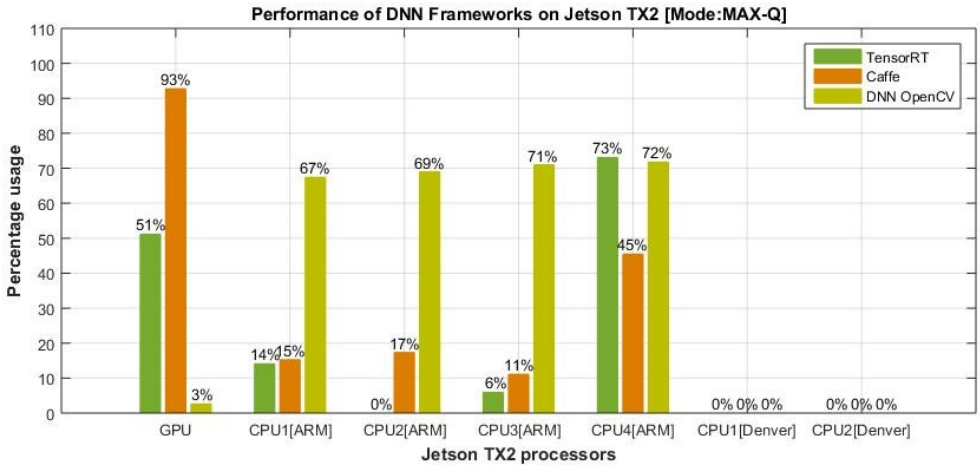

Figure 4.5: Resource usage during the inference process in MAX-Q Mode Using GoogLeNet model

TensorRT, the second-fastest framework, used 51% of GPU's cores of the Jetson TX2 for the deep learning inference process. However, the performance of this framework was just 6% lower than Caffe. TensorRT used 36% of the total CPU power available on the Jetson TX2. However, similar to Caffe, a singular high usage of CPU number four can be noticed in Figure 4.5. TensorRT was evaluated only in FP32 quantization mode. TensorRT could have a better performance with this particular deep learning models with FP16 and INT8 quantization models.

The DNN-OpenCV was the framework with the higher inference time. In fact, the inference time was almost eight times higher than the other frameworks. DNN-OpenCV used only 3% of the GPU's cores of the Jetson TX2 as illustrated in Figure 4.5 . One possible explanation could be the fact that OpenCV does not support CUDA language for the parallel model of computation. However, it is interesting that DNN OpenCV shows an excellent parallel computation using the CPUs. As figure 4.5 illustrates, DNN-OpenCV used 70% of the CPU power of the Jetson TX2.

Experimental Results in MAX-P Core-All Mode

Figure 4.6 illustrates an improvement of the performance in all three deep learning frameworks. The highest improvement was reached by the DNN-OpenCV, which reduced its inference time by 54% with respect to MAX-Q mode. Two reasons explain this improvement. First, the presence of two

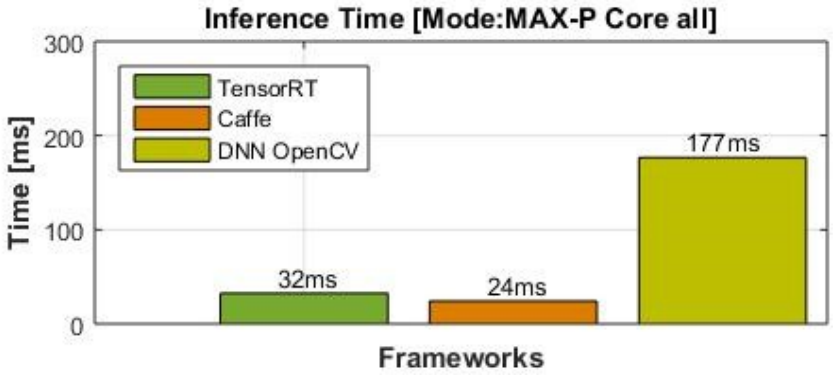

Figure 4.6: Inference Times in MAX-P Mode Using GoogLeNet model

new Denver Processors assists the DNN-OpenCV to parallelize the calculations. Furthermore, the increase of the clock-frequency of the CPUs 16% respect for the MAX-Q mode. The second-best improvement was reached by Caffe which ran 22% faster than in MAX-Q mode. This improvement is a consequence of the increment of the GPU's clock-frequency in a 31% respect of the MAX-Q mode. Containing faster CPUs also improves the performance of this framework, which uses 24% of the total CPU capacity of the JETSON TX2. This usage is consequent to the amount of power used in MAX-Q mode. Figure 4.7 show the performances in MAX-P mode.

TensorRT had an improvement of just 3% in the inference time. This was the framework with the least improvement. Curiously, this framework increased the GPU's core usage from 51% to 59% on average. TensorRT decreased the use of the CPU from 36% to 28%. One reason to explain the lower improvement of the TensorRT is that the model could not be optimized on TensorRT in FP32 quantization mode. 4.8 illustrates the GPU's usage by TensorRT as compared to the CPU's usage by Caffe.

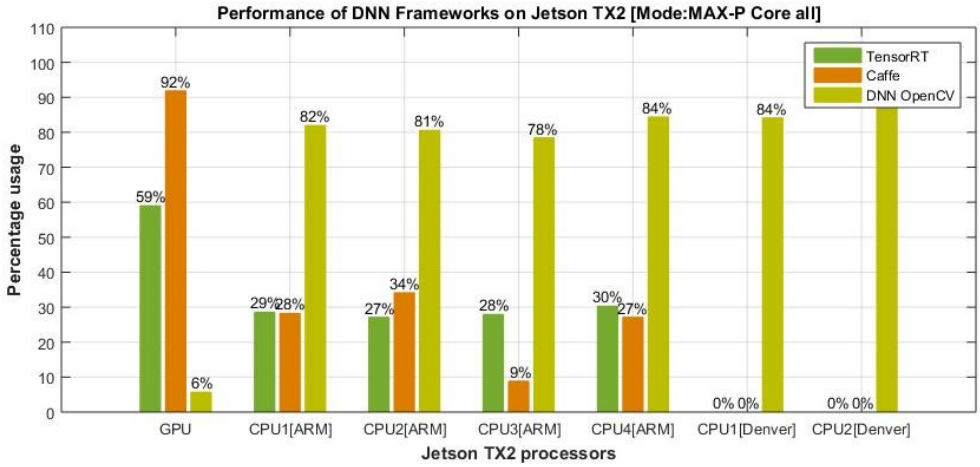

Figure 4.7: Resource usage during the inference process in MAX-P Mode Using GoogLeNet model

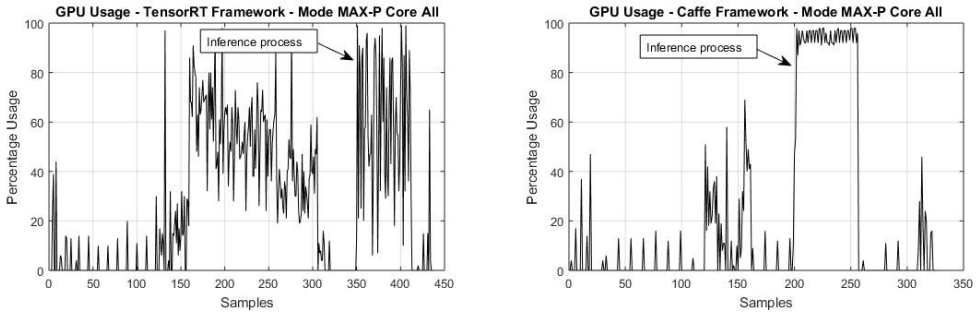

Figure 4.8: GPU Usage by TensorRT and Caffe with GoogLeNet model

DNN-OpenCV was the framework with the best improvement. Figure 4.7 confirms the CPUs approach for the computation parallel model of this framework. DNN OpenCV incorporated the available new CPUs and consumed 83% of the total CPU power on the Jetson. This framework used a mere 6% of the GPU's cores available in the JETSON TX2.

Experimental Results in MAX-P Core-All Mode

Caffe is still the fastest framework in the fastest mode of the Jetson TX2. This framework had an improvement of 12.5% respect to the previous MAX-P mode. The increment of the GPU's clock-frequency a 16% is the main factor for that improvement. However, the new GPU's clock-frequency does not help to TensorRT which has an improvement of only a 3% being the framework with the lest improvement. DNN-OpenCV was again the framework that had the better improvement respect the MAX-P mode being a 28% faster. The increase of 30% in the clock-frequency of the CPUs can explain the improvement of this framework. Figure 4.9 illustrate the inference time of the frameworks.

Figure 4.10 shown that the deep learning frameworks maintained a resources usage's profile similar that in MAX-P Core All Mode. Thus, Caffe took 92% of the GPU's cores and a 16% of the total CPU power. TensorRT took a 56% of the GPU's cores and a 19% of the total CPU power. Finally, OpenCV took 8% of the GPU's cores and a 82% of the total CPU power.

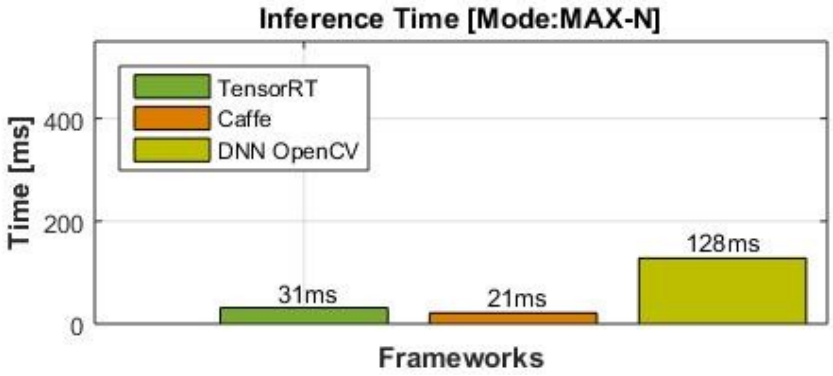

Figure 4.9: Inference Times in MAX-N Mode Using GoogLeNet model

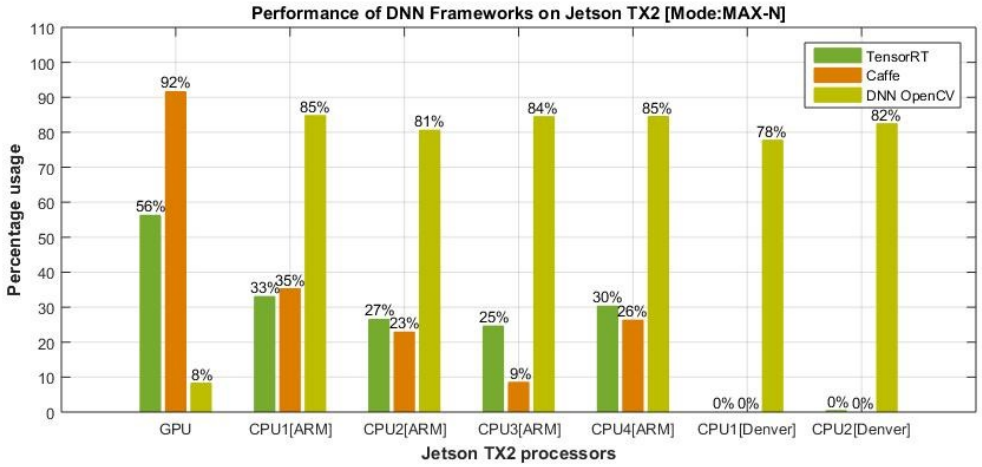

Figure 4.10: Resource usage during the inference process in MAX-N Mode Using GoogLeNet model

Table 4.6 summarize the resource usage's profiles using the deep learning model GoogLeNet.

4.5.2 Experimental Results Using OpenPose Model

Experimental Results in MAX-Q Mode

For this model, TensorRT was the fastest framework. It was 50% faster than Caffe and 30 times faster than DNN OpenCV as illustrated in Figure 4.11. The considerable difference between DNN OpenCV and the other

Table 4.6: Summary of resources usage in the experiment with GoogLeNet model

Framework	MAX-Q Mode	MAX-P Core-All	MAX-N
Caffe	93% GPU-22%CPU	92%GPU-24%GPU	92%GPU-16%CPU
Tensor RT	51%GPU-36%CPU	59%GPU-28%CPU	56%GPU-19%CPU
DNN-OpenCV	3%GPU-70%CPU	6%GPU-83%CPU	8%GPU-82%CPU

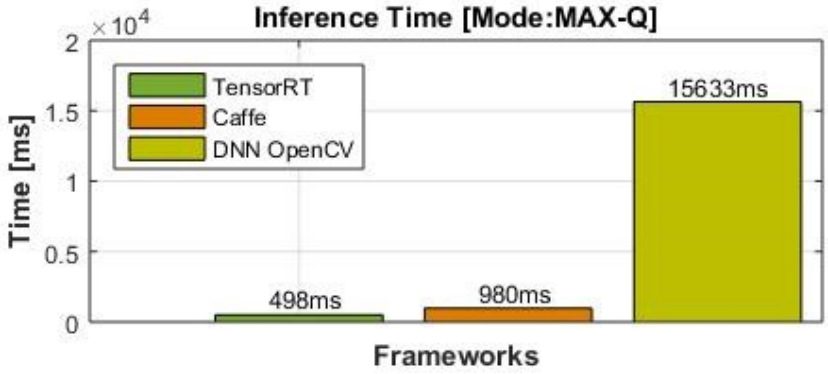

Figure 4.11: Inference Times in MAX-Q Mode using OpenPose model

frameworks is due to the parallelization model of this framework. As the previous experiments confirmed, DNN-OpenCV uses only the multicore CPU system. In contrast, TensorRT and Caffe have the potential to parallelize the deep learning model using 256 GPU cores plus the multicore CPU system.

Figure 4.12 illustrate than TensorRT used 92% of the GPU's cores and 2% of the CPU power of the JETSON TX2. Similarly, Caffe used a 97% of the GPU's cores and 0% of the CPU power. However, TensorRT was 50% faster compared to Caffe. This suggests that TensorRT has a better implementation for the NVIDIA platform Jetson TX2. DNN OpenCV exhibited a similar resource usage's profile as in the previous deep learning model using a 0% of the GPU and a 80% of the total CPU capacity.

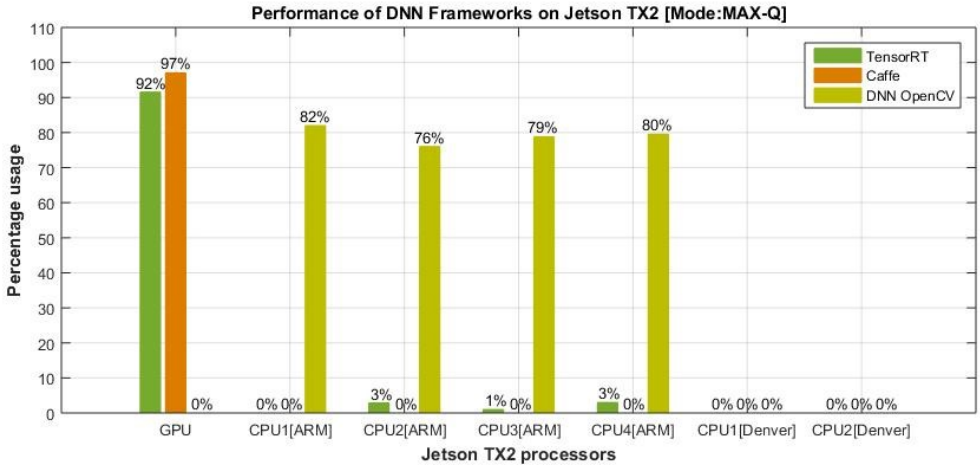

Figure 4.12: Resource usage during the inference process in MAX-Q Mode using OpenPose model

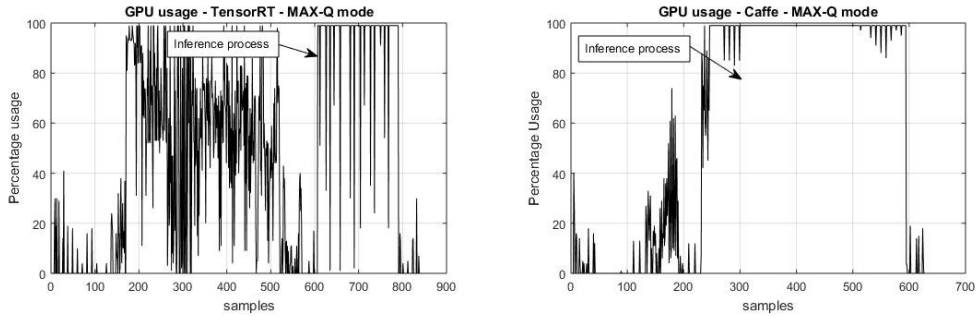

Figure 4.13: GPU Usage by TensorRT and Caffe with OpenPose model

Figure 4.13 illustrates that Caffe is still the most stable framework in terms of GPU usage during the inference process. Although TensorRT is the fastest framework, it still has more variability using the GPU. It is important to note that performance results reported in these experiments are based on the understanding of usage of these tools by the author of this work and do not necessarily reflect the best possible results possible of being achieved.

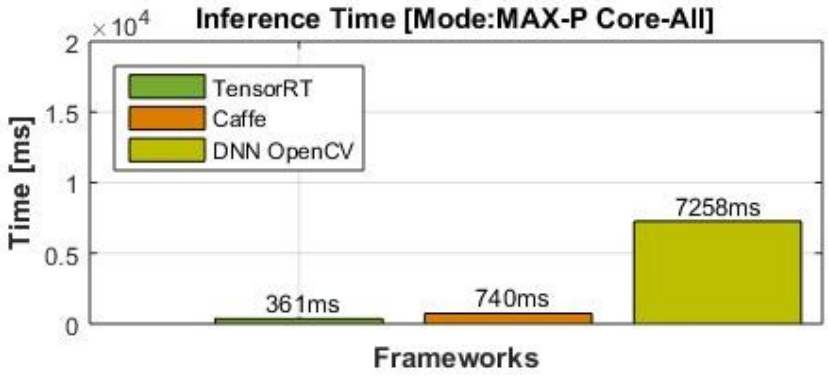

Figure 4.14: Inference Times in MAX-P Mode using OpenPose model

Experimental Results in MAX-P Core-All Mode.

DNN-OpenCV was the framework with the best improvement compared to the MAX-Q mode. This framework reduced its inference time by 54%. The improvement of DNN-OpenCV results from the two new Denver Processor and the clock-frequency increment of the CPUs. The improvement for TensorRT and Caffe was very similar, being 27% and 25% respectively in regards to the MAX-Q mode. This improvement was a consequence of the new clock-frequency of the GPU, which increased by 31%. Figure 4.14 illustrate the inference times this mode.

TensorRT was again the faster framework and used 90% of the GPU's cores and 3% of the total CPU power on the Jetson TX2. Its inference time is 52% lower than Caffe even do it is using a 9% less GPU capacity. Caffe maintained its previous resource usage's profile using 97% of the GPU's cores and 1% of the total CPU power. Caffe otherwise is still the best framework in terms of using the GPU capacity. DNN-OpenCV in

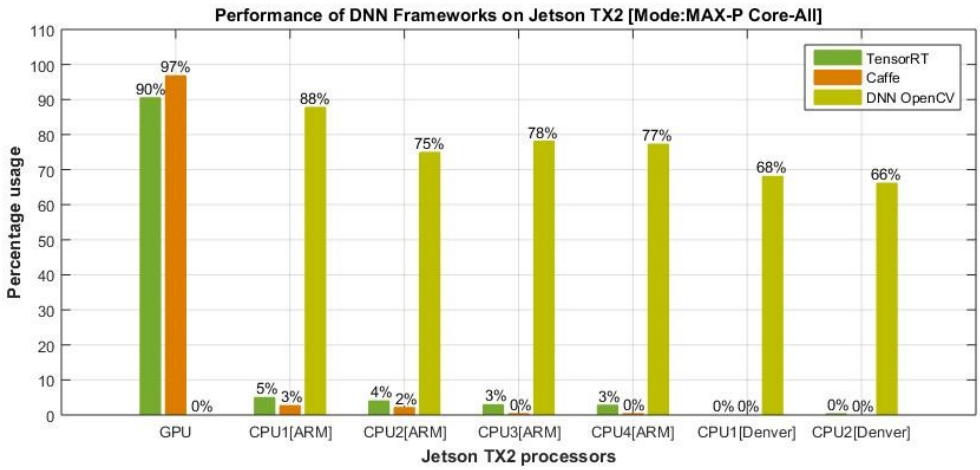

Figure 4.15: Resource usage during the inference process in MAX-P Mode using OpenPose model

this mode takes 0% of the GPU's cores and 75% of the CPU total power. Figure 4.15 illustrate the resource usage.

Experimental Results in MAX-N Mode.

TensorRT remains 51% faster than Caffe in the fastest mode of the Jetson TX2 as figure 4.16 illustrates. DNN OpenCV, similar to the aforementioned case study, was the framework with the best improvement regarding the MAX-P mode, 30% faster in this mode than in MAX-P mode. Both TensorRT and Caffe had an overall improvement of 10%. This improvement results from the new clock-frequency of the GPU, which increased by 16%.

Figure 4.17 illustrate that Caffe is the best framework in terms of usage of GPU's power. However, TensorRT can reach a better performance than Caffe but using 7% less GPU's cores. This confirms that the implementation of TensorRT on the NVIDIA platform is better.

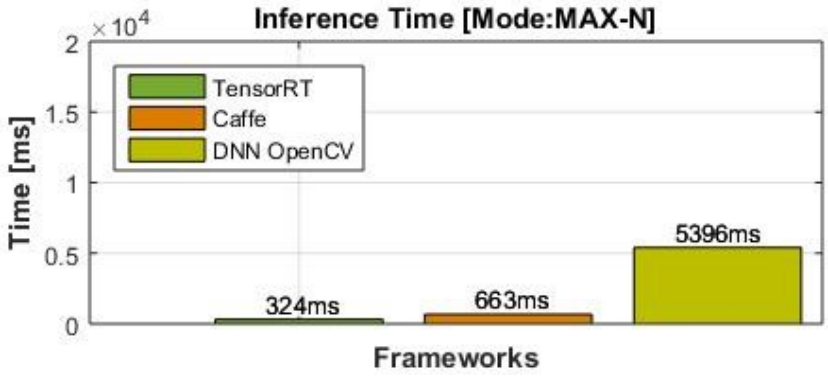

Figure 4.16: Inference Times in MAX-N Mode using OpenPose model

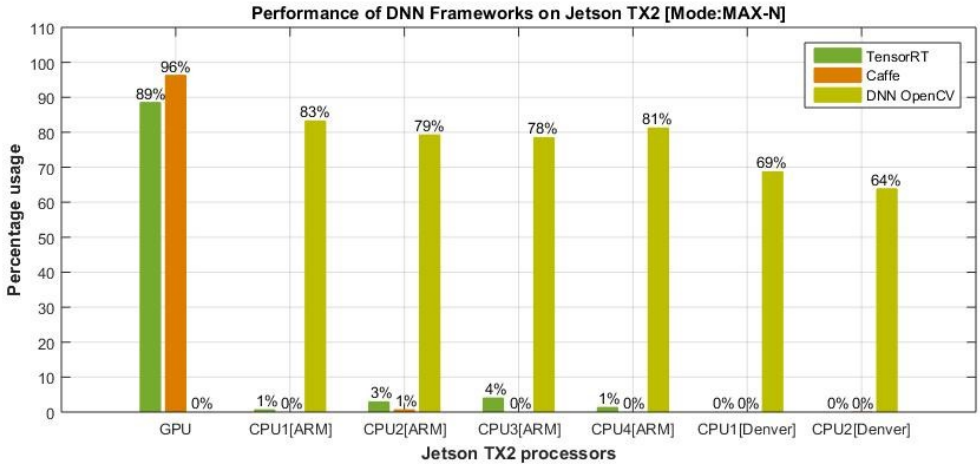

Figure 4.17: Resource Usage during the inference process in MAX-N Mode Using OpenPose model

TensorRT and Caffe are focused on using the GPU power of the Jetson TX2 while DNN OpenCV is focused on using the multiple CPUs architecture. In this mode, OpenCV does not use GPU power, rather it uses 75% of the total CPUs capacity. Table 4.7 despicts a comparison between the frameworks' usage profile in the different modes of the Jetson TX2.

Table 4.7: Summary of Resource Usage in the OpenPose Model Experiment

Framework	**MAX-Q Mode**	**MAX-P Core-All**	**MAX-N**
Caffe	97%GPU-0%CPU	97%GPU-1%CPU	96%GPU-0%CPU
Tensor RT	92% GPU-2%CPU	90%GPU-3%GPUU	89%GPU-1%CPU
DNN-OpenCV	0%GPU-80%CPU	0%GPU-75%CPU	0%GPU-75%CPU

4.6 Discussion

The GoogLeNet's experiment illustrated that Caffe was the fastest framework. On the other hand, OpenPose's experiment resulted that the TensorRT was the fastest framework. In general terms, the inference time of these two frameworks during the tests were quite similar. However, TensorRT has proven to be the best framework considering the inference time in comparison to the amount of resources consumed. Utilizing TensorRT with INT8 or FP16 quantization modes in the GoogLeNet experiment, demonstrated its potential to outperform the inference time of Caffe. From an implementation standpoint, TensorRT is more difficult to implement in C ++ compared with the other frameworks. A disadvantage of this framework is that in order to optimize the engine, TensorRT constructs static engines, preventing a change in shape of the deep learning engine with a different input size. In contrast, Caffe and DNN OpenCV have the capacity to reshape the network, allowing various input sizes. TensorRT is also a relatively new framework, so the version tested does not support some neural network architectures. As a result, pre-trained models cannot be

deployed using this framework.

Caffe has proven to be a very reliable framework with a fast inference engine. The GPU's cores usage of this framework is both efficient and stable. Caffe performance during the experiments is similar to TensorRT. A primary disadvantage of Caffe is there only exists a small amount of information about the steps to use its C ++ API library that does the work hard. Additionally, Caffe does not accept models from other frameworks, reducing the possibility of using Caffe as a unique framework. Finally, Caffe is being improved based on a community model, so there are different versions of this framework. Each framework is personalized, and as a result, not all of the frameworks support the same networks.

Finally, DNN-OpenCV has proven to be the framework with the lowest performance for the Jetson TX2 due to its inability to take full advantage of GPU power. However, this is an excellent framework for systems with multiple CPUs without GPU support. Additionally, DNN-OpenCV has proven to have some clear advantages over the other frameworks. For instance, DNN-OpenCV was the most straightforward framework to use and it is compatible with almost all of the current neural network architectures. DNN- OpenCV is also a very well documented framework in both C ++ and Python languages. For the previous reasons this framework is an excellent option for fast developments, however, it is not deal for optimized applications.

4.7 Summary

As underscored in this chapter, optimal inference is indeed a critical factor for the implementation of real-time systems. Optimal resources usage is essential to build a multitasking system. Inference time and resource usage were both tested to evaluate their performance capacities as both real-time and multitasking are characteristic of the software architecture of PABI.

In this chapter, the performance of three deep learning frameworks was evaluated on the Jetson TX2. TensorRT, Caffe, and the deep learning module of OpenCV was tested under controlled conditions. The main factors analyzed were inference time and resource usage.

A multi-factorial experiment was designed in order to evaluate the frameworks. Two independent variables were chosen for the experiment: workload and hardware configuration. Workload included two deep learning models with a medium and later with a large number of parameters, respectively. The hardware configurations corresponded to three different modes of operation of the Jetson TX2.

The results indicate that TensorRT is the best framework in terms of velocity and resource usage. However, this framework was arduous to implement as compared with other frameworks due to its constraints in flexibility and compatibility issues. Caffe showed decent performance in both velocity and resources usage. The primary disadvantage of Caffe is the lack of information for its C++ API. Caffe also had compatibility restrictions.

The deep learning module, OpenCV, proved to be a framework designed for multicore CPU architectures, yet not for GPU architectures. However, this framework had excellent compatibility with other frameworks and is well-documented.

Chapter 5

Evaluating Performance of Pre-Trained Deep Learning Models for Low-Level Feature Extraction in a Robot-Enhanced ABA Therapy Framework

Modularity and expandability are two characteristics suggested by the software architecture designed in Chapter 2. These two characteristics are particularly important in an ABA therapy context because of the nature of the interactions. People with ADS have behaviors that increase the complexity of human-robot interaction. Modularity and expandability allow building flexible and adaptable systems while maintaining good performance.

In this chapter, the use of deep learning pre-trained models to build modular and expandable systems are explored. Two main factors incentive the use of deep learning models in PABI. First, the hardware architecture the PABI specially developed for efficiently deploying deep learning models. Second, the scientific literature that state that deep learning models are

more robust than classical approaches under uncontrolled conditions. Face detection, face landmarks detection, and face codification are analyzed. These feature are important for developing human-robot interactions. At the end of the chapter, the experimental results are discussed.

5.1 Motivation and Requirements

Face detection or face recognition using computers is difficult, particularly under uncontrolled conditions. However, new approaches, such as deep neural networks and convolution neural networks, have reached performances that overcome human perception (Russakovsky et al. 2015). The main applications of these new algorithms have been for devices like personal computers, tablets, or smartphones. In these platforms, the user interacts directly with the device most of the time. Therefore, the algorithms or models are trained for peak performance when they work with frontal faces.

Introducing a robot as part of ABA therapy presents significant challenges. First, the patient has behaviors that are not present in people without autism, increasing the complexity of the interaction. Second, the robot is not the main interface. The robot is only an observer or partial participant of the therapy in some cases. The primary attention of the patient varies between people or other devices during the treatment. Therefore, frontal faces are not available all time, doing the image process and the interaction harder.

5.2 Primary contributions of this chapter

The main contribution of this chapter is the analysis of deep learning pre-trained models for low-level feature extraction in an ABA therapy context. Face detection, face landmark detection, face recognition are basic features for human-robot interaction. Therefore, the analysis is focused on these features. The study includes the performance of the algorithms/models on the Jetson TX2 as well as their robustness working with non-frontal face images; a condition commonly found during the ABA therapy. Mainly deep neural network models are analyzed considering the parallel architecture of the Jetson TX2.

5.3 Background and related work

As described in chapter one, the three key uses of social robots in autism are eliciting behaviors; model, teach and practice a skill; and provide feedback and encouragement. In all these cases, the robot needs to have real-time social interaction skills. For ABA therapy, an essential interaction skill is patient and therapist tracking (Laurie A. Dickstein-Fischer et al. 2017). Once people are tracked, other high-level capabilities like engagement level detection, attention detection, among others, can be observed during the therapy. However, in order to build high-level skills, a set of low-level skills needs to be developed. Some of the low-level skills proposed by this

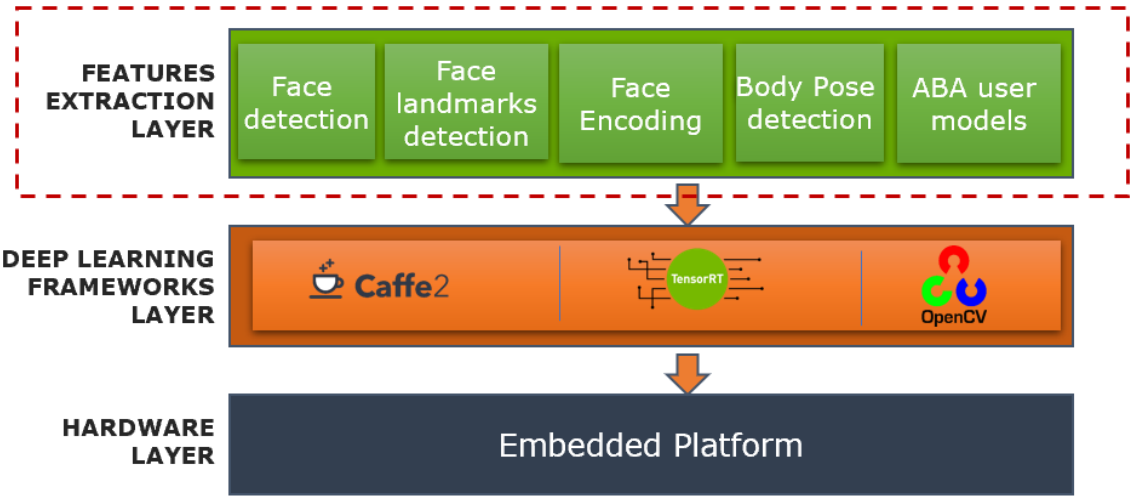

Figure 5.1: Low-level feature extraction layer in the multilayer software architecture

work are face detection, face landmark detection, face recognition, and pose detection. All these low-skill are built in the feature detection layer according to the proposed model in chapter 2 and shown in Figure 5.1.

5.3.1 Face Detection

Face detection is one of the basic steps in social interaction. There is some scientific evidence that child born with some information about the structure of the human face. (Morton and Johnson 1991). Human face detection from images has been studied for several years, and algorithms with different performances have been developed. Traditional approaches like pattern recognition were initially used; however, those approaches were very sensitive to the illumination, pose, among others factors (Rowley, Baluja, and Kanade 1996; Yow and Cipolla 1997). Moreover, the processing time required by those methods made them useless for real-time face detection

applications. Viola et al proposed in 2001 the first method that has been extensively used in real-time face detection. (Viola and Jones n.d.) The develop of the Deep Learning and the Convolutional Neural Networks (CNN) in the last years have contributed to build more robust and fast face detection systems for real-time applications. The deep learning architectures more used in object detection are Faster R-CNNs (Girshick et al. 2013), You Only Look Once (YOLO) (Redmon et al. 2015), Single Shot Detectors (SSDs) (Wei Liu et al. 2016). Faster R-CNN, proposed in 2015, has a high accuracy. However, Faster R-CNN has also a high inference time. In contrast, YOLO, proposed in 2016, has a lower inference time than Faster R-CNN but lower accuracy. Finally, Single Shot Detectors which was introduced also in 2016, makes a tradeoff between the inference time and the accuracy being one of the most used architectures nowadays. The three previous architectures are built over another two networks: VGG(Simonyan and Zisserman 2015) and RESNET(He et al. 2015) which acts like features extractors. A third architecture called MobileNets proposed in 2017 is being used to build object detectors in mobile devices. Any of the previous detectors can be trained with human faces allowing the development of real-time systems for face detection. Figure 5.2 illustrate an example of face detection using a classical HAAR Cascade detector and a deep learning model approach.

Figure 5.2: Example of face detection. In red the bounding box generated by HAAR Cascade algorithm and in green the bounding box generated by a deep learning model

5.3.2 Face Landmark Points Detection

Once a face is detected, locate the fiducial facial landmark points around the face is important for different face analysis tasks. The number of the landmark points detected depend of the subsequent use of that information. Two number of landmarks are mainly used: sixty-eight points and five points. The five points model is in general faster than the sixty-eight model and it is useful for tasks like face alignment or face pose estimation for instance (Murphy-Chutorian and Trivedi 2009). On the other hand, the sixty-eight points model provides more information about the face and it is useful for tasks like non-verbal communication, emotion recognition (Pantic and Rothkrantz 2000), among others. Figure 5.3 illustrate these two different approaches.

According (Wu and Ji 2019) the landmark points detection algorithms can be classify in three categories: holistic methods, Constrained Local Model (CLM) methods, and the regression-based methods. Wu et al states

Figure 5.3: Examples of landmark points detection. Left) 68 points model, Center) 5 points model, Right) Eyes and lips model

that Regression-based methods achieve better performance than holistic methods and CLM methods. Regression methods are also faster which is particularly useful in real-time application like PABI. Convolutional Network methods for landmark points detection follow the regression-based frameworks. Some of the popular frameworks based on regression methods are presented in (Kazemi and Sullivan 2014; Ren et al. 2014; Wu, Hassner, et al. 2015)

5.3.3 Face Recognition

On step forward in social robot interaction is the capacity of the robot to recognize specific people. Recognizing people means to identify or verify the identity of a person based on images. This is a very challenging problem which has been studied for more than forty years. In fact, one of the earliest researches was done in 1970 (Kelly 1971). One of the main problems that face recognition revise is the variability of the face images in the real world. (Trigueros, Meng, and Hartnett 2018) classify the approaches of

face recognition in five categories: geometry-based methods, holistic methods, feature-based methods, hybrid methods, and deep learning methods. Particularly, the develop of the last approach has allowed to supersede the human capacity of face recognition in the last years. (Weiyang Liu et al. 2017; Schroff, Kalenichenko, and Philbin 2015; Simonyan and Zisserman 2015) The deep learning approach has also allowed the real-time implementation of face recognition.

5.3.4 Pose Detection

Human pose detection has been extensively studied in recent years. From a social robot point of view, pose detection provides a new way to understand social interaction. (Rudovic et al. 2018) uses pose detection to understand the behavior of patients in autism therapies. Human pose detection consists on locate the joints of the human body in images. Then, a specific pose can be detected. For pose estimation, there are various approaches. First,(O'Rourke and Badler 1980; Y. Yang and Ramanan 2013; Hogg 1983) present a classical approach. In these approaches human pose is represented as a collection of parts organized in a deformable configuration. Every rigid part is connected by a spring which allows different configurations. The second approach is based on deep learning models. This approach has been growing faster in last years (Toshev and Szegedy 2014). Deep learning models for pose detection outperformed classical models. Most popular

deep learning approaches using deep learning to generate is a discrete representation of the probability than a joint occurred at each pixel called a heatmap.

5.4 Experimental methods

Different experiments were run to evaluate the robustness of the deep learning models. For face detection, landmarks face detection, and face recognition the metric was the robustness of the model to different faces poses and the inference time on the Jetson TX2. For the pose detection, the analysis was focus on the inference time of the models on the JetsonTX2.

5.4.1 Experimental setup: Images Dataset and hardware

Images Dataset

Robustness of the face detection, landmarks face detection, and face recognition algorithms were evaluated using the Pointing'04 face data set(Gourier, Hall, and Crowley n.d.). This database consists of two sets of 15 people in different face poses. Each pose is defined by a vertical angle and a horizontal angle. The vertical angles used by this dataset are -90, -60, -30, -15, 0, +15, +30, +60, +90 degrees while the horizontal angle are -90, -75, -60, -45, -30, -15, 0, +15, +30, +45, +60, +75, +90 degrees. The database provides 2790 images in total. Figure 5.4 illustrates an example of the several images of one user in the dataset. The resolution of the images in the

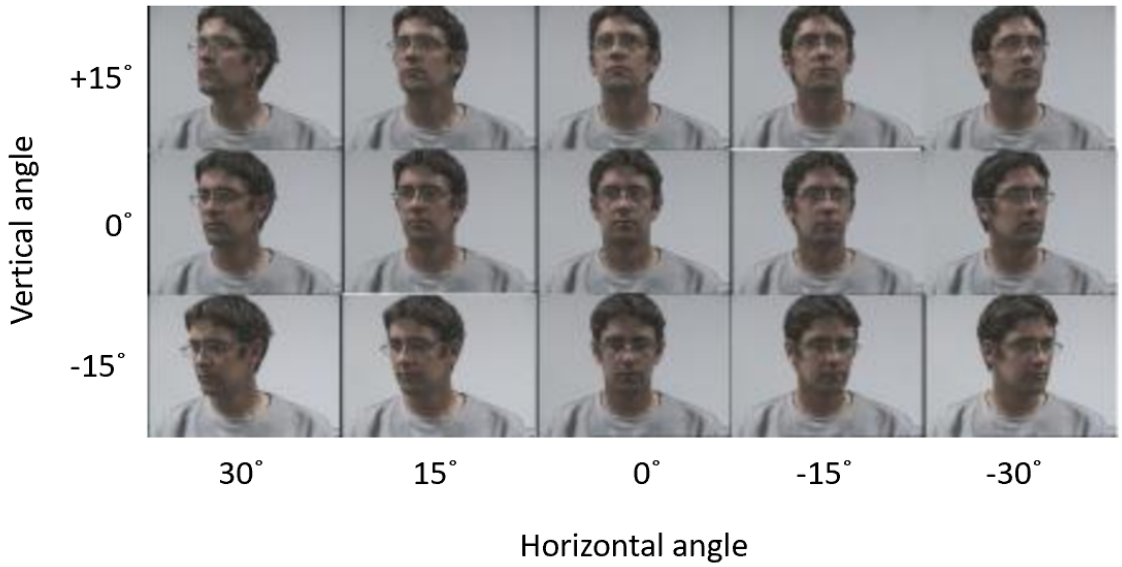

Figure 5.4: Multipose Pointing'04 face data set

dataset is 384x288 pixels.

Hardware platform

For the experiments a Jetson TX2 platform with JetPack 3.2 installed was used. The JetPack 3.2 included Ubuntu 16.04 LTS and CUDA 9.0. OpenCV 3.4.1 for ARM processors and CUDA support was built from the source code and installed and used for the experiments. The Jetson TX2 was configured in MAX-N mode.

5.4.2 Face Detection Experimental Setup

Face detection models were evaluated by using two metrics: processing time and face detection performance. For the first metric, the number of clock-cycles was used to measure the inference time. For the second metric, the

result of the face detector on faces in different poses was registered. Then, the true positive detections and the false positive detections were used to build a probability map.

Neural networks

Two approaches were analyzed for face detection. The first approach was the classical HAAR Cascade detection algorithm. This method was used like a baseline for contrasting the performance of a deep learning model. The previous version of PABI used this algorithm for the face detection stage (Pereira 2017). The second approach is based on a deep learning model.

The deep learning model used in the experiment consisted of a Single Shot Detector pipeline with a RESNET-10 architecture as a backbone. This model was trained for human face detection by the OpenCV group using Caffe framework. This model is one of the most popular pre-trained models for face detection. This model is included with OpenCV 3.4.1

Experiment procedure

The experiments were executed using OpenCV 3.4.1 with CUDA support and C++11. Two experiments were built for each model. The first experiment was focused on measure the face detection performance while the second experiment was focused on measure the processing time.

For the first experiment, every face in the dataset was evaluated through

Table 5.1: Tested angles. Each pose from each user was tested using a HAAR and DNN algorithm. The sizes of the input images during the experiment were 266x200, 400x300 and 666x500 pixels.

		Horizontal Angles												
Vertical Angles		-90	-75	-60	-45	-30	-15	0	+15	+30	+45	+60	+75	+90
	-90							x						
	-60	x	x	x	x	x	x	x	x	x	x	x	x	x
	-30	x	x	x	x	x	x	x	x	x	x	x	x	x
	-15	x	x	x	x	x	x	x	x	x	x	x	x	x
	0	x	x	x	x	x	x	x	x	x	x	x	x	x
	+15	x	x	x	x	x	x	x	x	x	x	x	x	x
	+30	x	x	x	x	x	x	x	x	x	x	x	x	x
	+60	x	x	x	x	x	x	x	x	x	x	x	x	x
	+90							x						

the face detectors. Then, the result was registered and classified as a true positive, or false positive. Finally, the probability that a face detector detects a face in a specific pose was calculated as:

$$P(TruePositive[H_Angle, V_Angle]) = \frac{TruePositive[H_Angle, V_Angle]}{NroFaces[H_Angle, V_angle]} \tag{5.1}$$

For each pose, a total of 30 true positives cases are available (2 sets x 15 people). The experiment was repeated using different sizes for the input images. The sizes used was 266x200, 400x300 and 666x500 pixels. Table 5.1 shows the experiments executed.

For the second experiment, 200 random images from the dataset were

taken and the processing time was calculated. Nine different size of the input image were used to calculate the inference time.

5.4.3 Face Landmark Points Detection Experimental Setup

For landmarks face detection two metrics was used: processing time and the error in the landmarks position. For the first metric, the number of clock-cycles was used to measure the processing time. For the error, the distance between every detected landmark and its labeled position were calculated using equation 5.2.

Neural networks

For face landmarks detection two approaches were evaluated. The first approach is the classical Local Binary Feature (LBF) landmarks detection algorithm (Ren et al. 2014). This algorithm, based on a regression approach, is one of the fastest non-deep learning approaches used for landmarks detection. This method as a baseline for landmarks detection comparison. The second method is a deep learning approach presented in (Wu, Hassner, et al. 2015). This model was chosen because of authors state that the model has a good performance for extreme poses.

The Pointing'04 face dataset described in the previous section was used as input. However, landmark points were manually labeled by the author of this thesis, in order to calculate the error of the tested models using this

Figure 5.5: Example of a manually labeled image

dataset. Figure 5.5 shows an example of labeling.

Experiment procedure

The experiments were executed using OpenCV 3.4.1 with CUDA support and C++11. Two experiments were built for each algorithm/model. The first experiment was focused on measuring the face landmarks detection performance while the second experiment was focused on measure the processing time. For the first experiment, every face in the set one was evaluated through the face landmark detectors. Then, the Euclidian distance between every calculated landmark and the corresponding manually la-

beled landmark was calculated. A Euclidean distance equal to zero means no error. The bigger the Euclidian distance the bigger the error. The average error for a pose with an H horizontal angle and V vertical angle is calculated as:

$$Error[H,V] = \frac{\sum_1^{NroFaces[H,V]} \left(\sum_1^{NroLandmarks} \sqrt{(Pcx_i - Plx_i)^2 + (Pcy_i - Ply_i)^2} \right)}{NroFaces[H,V]} \tag{5.2}$$

where Pc_i is the calculate landmark point and Pl_i is the manually labeled point.

In order to measure the inference time, 1000 random images of the dataset and the processing time were calculated. We repeat the experiment for 9 different sizes of the input image.

5.4.4 Face Encoding Experimental Setup

Face encoding analysis was focused on deep learning approaches due to the high accuracy reported in the literature. Evaluating a deep learning-based face encoder is difficult because of a preprocessing stage is necessary. Any modification in the preprocessing stage can affect the performance a face encoder. Figure 5.6 illustrates the preprocessing stage before applying a face encoder. Three procedures are necessary before applying a face

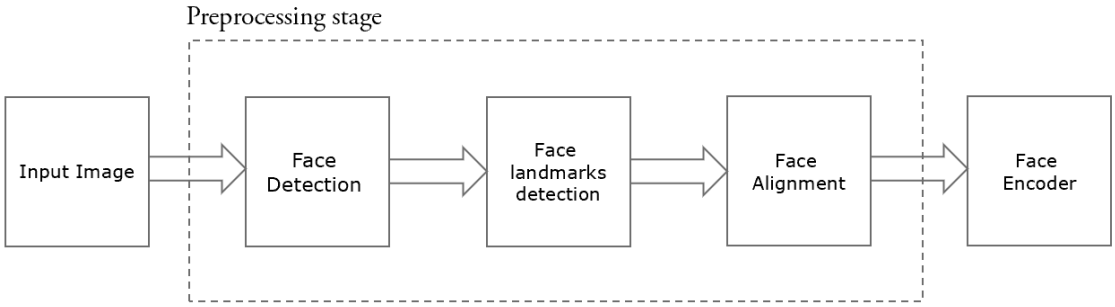

Figure 5.6: Preprocessin stage for face encoding

encoder: face detection, face landmark detection and face alignment. The first and second procedures were described in the previous sections. For the face alignment procedure, the landmark points detected are used to obtain a normalized rotation, translation, and scale representation of the face.

Face encoding is sensitive to face alignment. A good alignment generates a good performance of the face encoder. Similarly, the quality of the face alignment is affected by the performance of the face landmarks detector. High error in the landmarks produce a bad face alignment. Therefore, there are an accumulated error in the processing stage that is transferred to the performance of the face encoder. For the experiments the face detector was set to the SSD-RESNET model and the landmarks detector to a vanilla model.

Two metrics were used to evaluate the face encoder models. First, the inference time of each model on the Jetson TX2 was measure. Second, the capacity of the models for verifying if two faces belong to the same person

was evaluated.

Neural networks and input data

Three different models were evaluated: OpenFace (Schroff, Kalenichenko, and Philbin 2015), SphereNet (Weiyang Liu et al. 2017), and VGGFace (Simonyan and Zisserman 2015). The first model proposes a loss function called triplet-loss to train the network. The triplet consists in two matching faces and one non-matching face. The loss function objective is to separate the correct pair from the incorrect pair using a distance margin. This model mapped a face in a 128-dimensional Euclidean space. The distances in this space directly correspond to faces similarity. Faces of the same person have small distances while faces of different people have large distances. Thus, identical faces have a Euclidian distance of 0 while two totally different identities have a Euclidian distance of 4.

The second model propose a more complex loss function call Angular softmax to learn angular discriminate features. The main idea of SphereFace respect to OpenFace, is to separate the feature by imposing an angular margin instead of a distance. Exist different pre-trained models of SphereFace. The reported model mapped faces in a 512-dimensional space. Figure 5.7 illustrate the intuition behind OpenFace and SphereFace

Finally VGGFace architecture in more complex than FaceNet but it also uses a triplet-loss to train the network. However, this model mapped a face

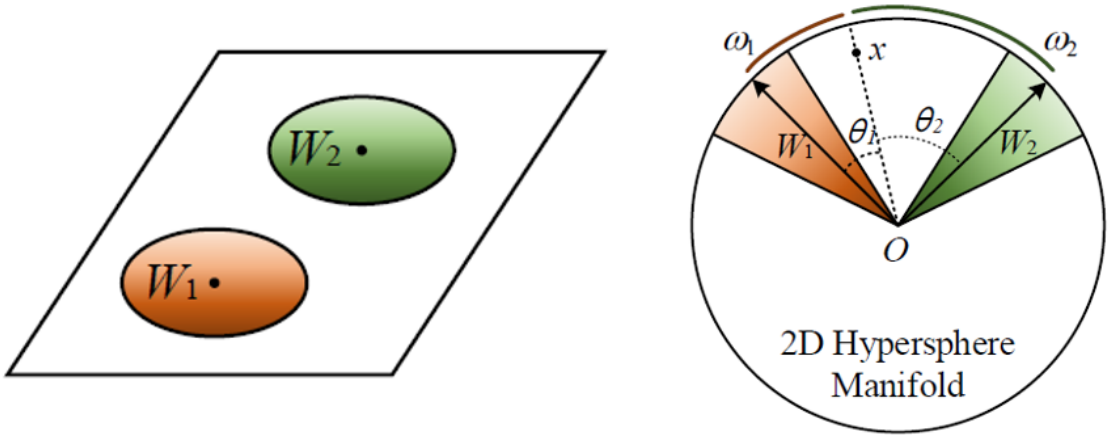

Figure 5.7: Difference between euclidian margin loss (used by OpenFace) and A-softmax loss (used by SphereNet). Adapted from *SphereFace: Deep Hypersphere Embedding for Face Recognition* by W.Lui et al, 2017, 2017 IEEE Conference on Computer Vision and Pattern Recognition, 6738-6746

in a 4096-dimensional Euclidean space.

Experiment procedure

. The experiments were completed using OpenCV 3.4.1 with CUDA support and C++11. Two experiments were developed. The first experiment consisted on measuring the capacity of the model of matching the same person. For this, five thousand random pairs of images that match the same person were chosen using the dataset aforementioned. Each face was encoded using a deep learning model and a metric was calculated in order to measure the similarity of the faces. Similarly, five thousand random pair of images that unmatched the same person was chosen and the similarity calculated. After that, a threshold that maximizes number of correct matches pairs and correct unmatched pairs was calculate. The true positives, true

Table 5.2: Experimental setup for face encoding

Model	Framework used	Number of features used to the face codification	Face detector	Landmarks detector
OpenFace	Caffe	128	SSD-ResNet	Vanilla 5 points
SphereFace	Caffe	512	SSD-ResNet	Vanilla 5 points
VGGFace	Caffe	4096	SSD-ResNet	Vanilla 5 points

negatives, false positives and false negatives was registered.

Because of the extreme poses in the database, two different ways of using the code generated by the encoders was tested. The first way was to use directly the code generated by the encoders to calculate the similarity between faces. The second way was to calculate the similarity using an average code. The average code is generated using the original code generated for a face and the code generated using the horizontal mirror of that face. Table 5.2 present the details of the experiment.

5.4.5 Experimental Setup for Body Pose Detection

No formal experiments were built for body pose. Mainly, the pre-trained model used in chapter 3 to evaluate the frameworks was tested. Although there are several neural networks in the literature, only OpenPose model was available. The inference time of OpenPose was measured in the chapter 3.

5.5 Experimental results

5.5.1 Experimental Result for Face Detection

The deep learning model had better performance compared with the HAAR detector algorithm. The model exhibited high robustness to the extreme pose variations. As is illustrated in Figure 5.8 the model covered almost all possible changes in the pose in both vertical and horizontal angles. In contrast, the HAAR detector had a good performance only in poses between -30 and 30 degrees in both the vertical and horizontal angles. The reason is that the HAAR detector uses kernels that were specially designed for frontal faces. Exist HAAR variations for profile detection which was not evaluated in this experiment. Figure 5.8 also illustrate that the deep learning model diminish its performance with small size images.

The inference time of the two approaches is very similar. Figure 5.9 illustrate a comparison of the inference time between HAAR and the deep learning model. For a small input, HAAR is slightly faster than the deep learning model. However, for a higher input size, the deep learning model is marginally faster than HAAR. So, in term of inference time, the models do not present a significative difference. For an input size of 400x300, both methods reached a rate of 18 frames per second in average on the Jetson TX2.

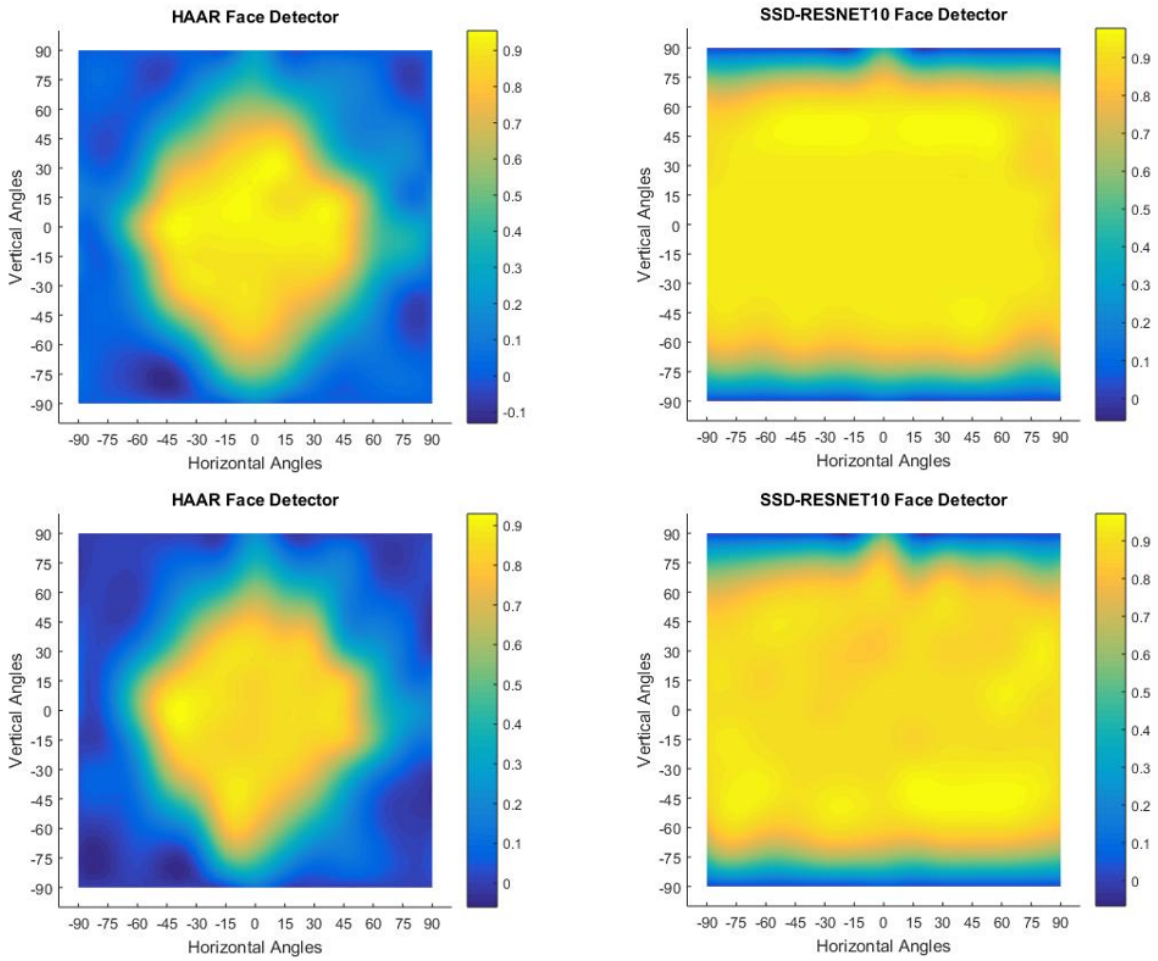

Figure 5.8: Probability of true positives for face detectors. The two upper images illustrate the probability maps of HAAR detector (left) and DNN model detector (right) using an input size of 266x200. The two lower images illustrate results of the detectors using an input size of 400x300

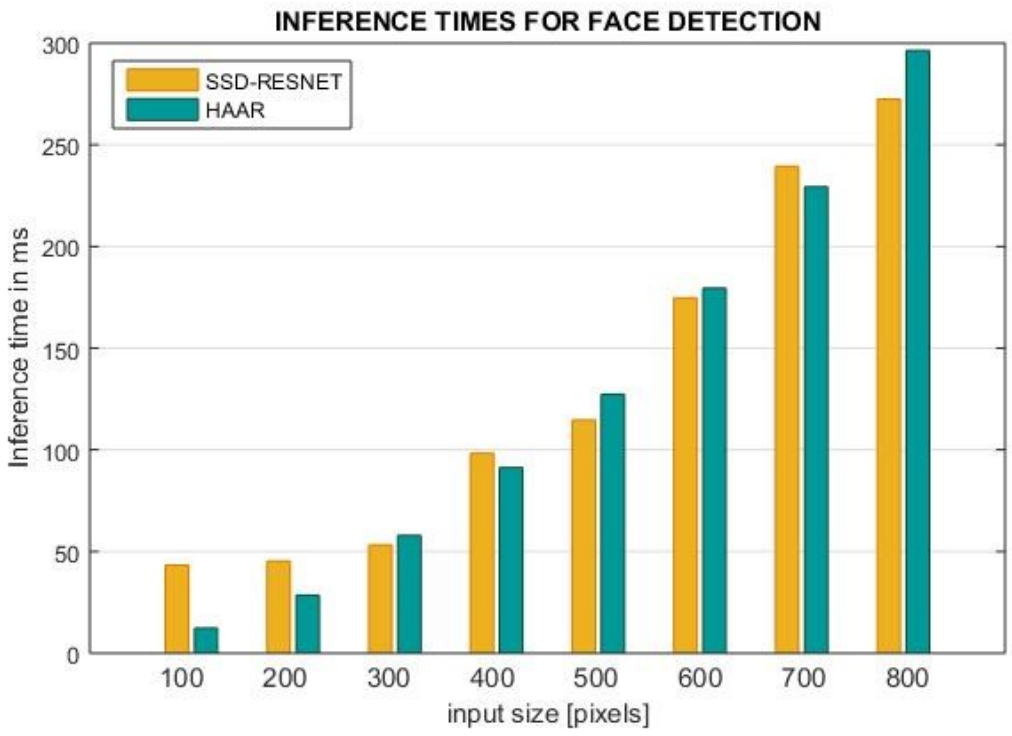

Figure 5.9: Inference time of the two face detector under different sizes

5.5.2 Face landmark points detection

Figure 5.10 illustrate the average error of face landmark points detection algorithms. The deep learning model evidenced better performance that LBF for extreme poses. In the horizontal angle, the model shows a acceptable performance between -60 and 60 degrees while in the vertical angle the acceptable performance is located between -45 and 75 degrees. In contrast, LBF shows an acceptable performance between -30 and 30 degrees in the horizontal angle and -45 and 45 degrees in the vertical angle. Figure 5.11 show the difference between the landmarks generated by the deep learning model (lower row) and the landmarks generated by the LBF algorithm.

Figure 5.12 illustrate the inference time of the two methods. The deep

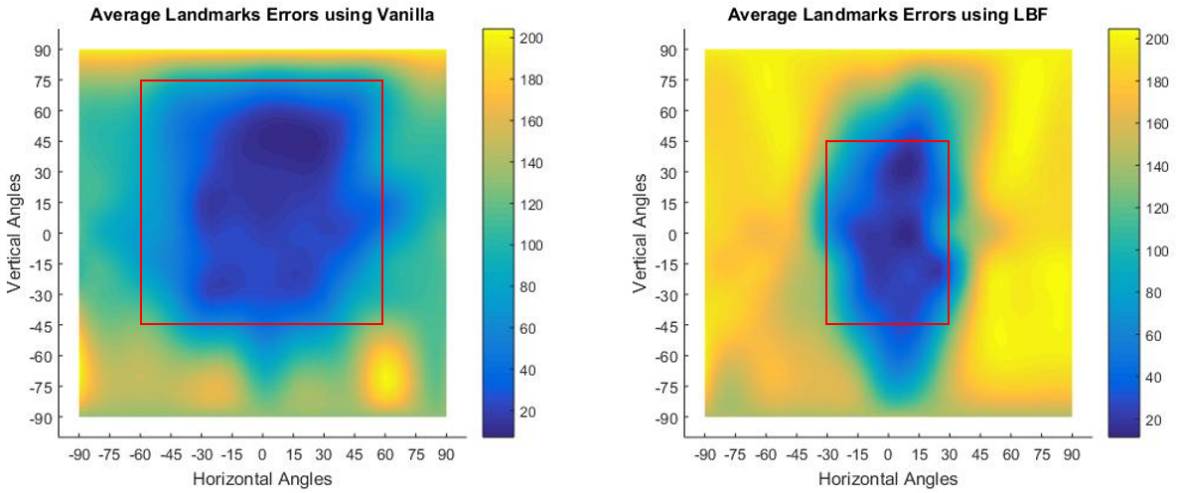

Figure 5.10: Average error for two landmarks detectors. Left) Deep learning model rigth) Local Binary Feature method

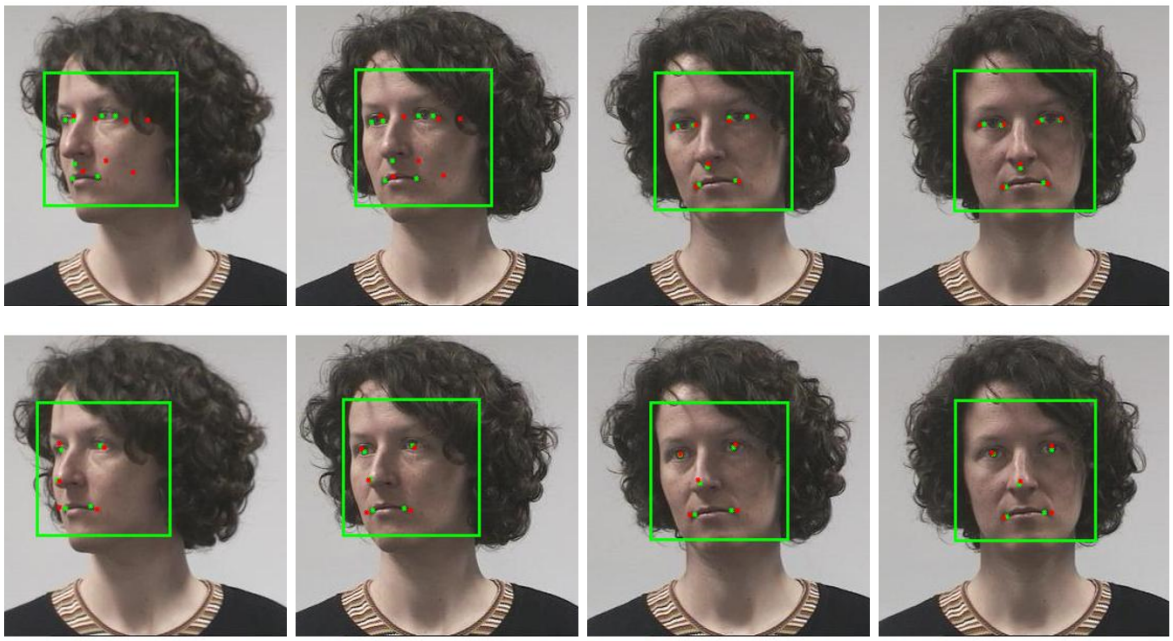

Figure 5.11: Landmark points detected by LBF (upper row) and the deep learning model (lower row)

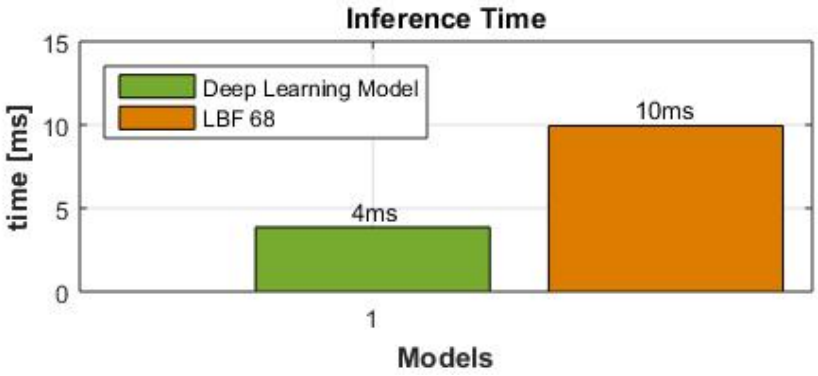

Figure 5.12: Inference time

learning model was faster; however, the LBF methods is calculating more points of reference.

5.5.3 Experimental Results for Face encoding

The results presented in the section are based on the configuration of the experiment defined in section 5.4.4. A better or worst performance of the deep learning face detectors can be reached using a different face encoder and/or a different face landmarks detector.

The first experiment consisted of using the code generated by the encoder without modification, to identify if two people match or not. The upper histograms in Figure 5.13 illustrate the results. According to the experiment, SphereFace was the model with less accuracy. 62.1% of the pairs of faces were correctly classified in matched or non-matched using this model. OpenFace was just 4.7% more accurate than SphereFace.The most accuracy model was VGG face, which was 18.6% more accurate than OpenFace. The metric that give the best performance for OpenFace and

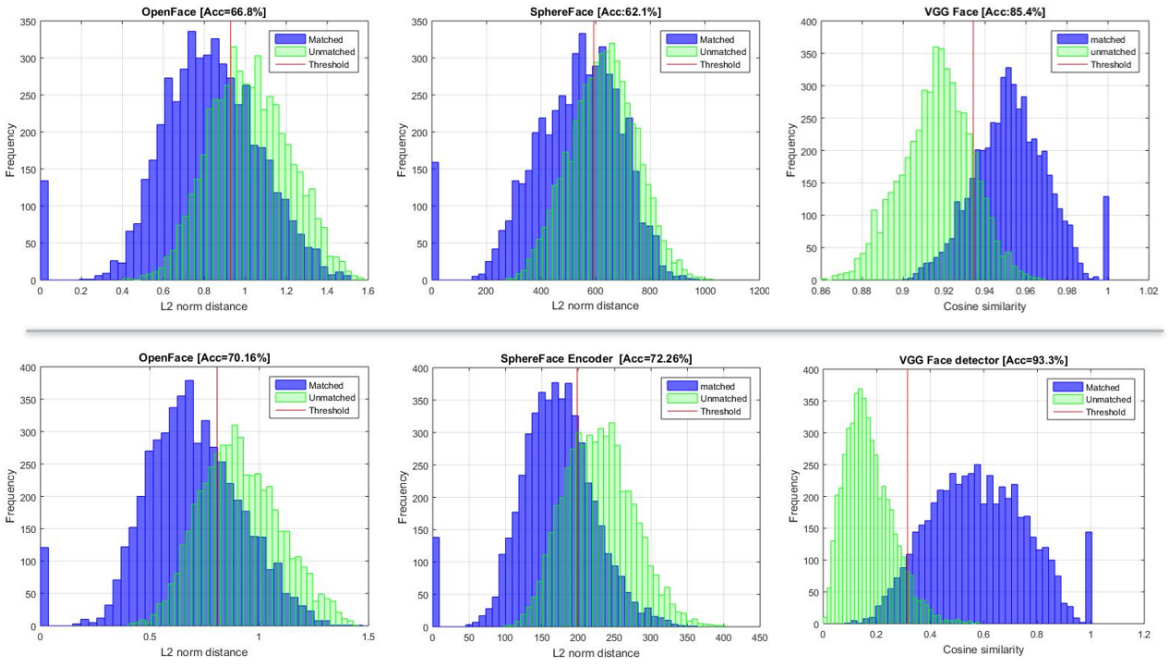

Figure 5.13: Accuracy of the face encoders. The upper row uses a simple code for matching faces while the lower row uses an average code

SphereFace was the Euclidian distance. For VGG was the cosine similarity. The values of the metrics was found using a 10-fold cross-validation. The best thresholds were 0.9279 for OpenFace, 592.57 for SphereFace and 0.31134 for VGGFace.

For the second experiment, two codes were generated and then the average was calculated. The first code corresponds to the original face detected and the second code corresponds to the face horizontally flipped. With this modification a more balance code is expected. Lower histrograms in Figure 5.13 illustrate de results. With the average code, all models were more accurate than using the single code. OpenFace improved 3.36%, SphereFace improved 10.16% and VGGFace improved 7.9%. VGGFace was still the

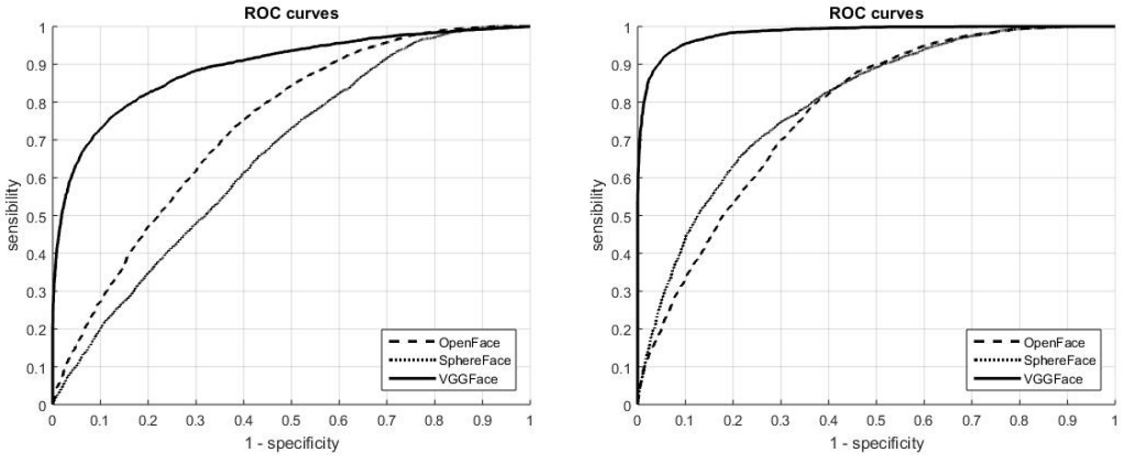

Figure 5.14: ROC curves for the faces encoders. Left) ROC using single code right) ROC using average code

best model, however, the second best model after the modification was SphereFace. The metrics used for each model was the same than in the previous experiment. The new thresholds were 0.806 for OpenFace, 198.98 for SphereFace and 0.3135 for VGGFace. Figure 5.14 illustrates the ROC curves comparing the performance of the three face detectors.

In terms of inference time, SphereFace was the faster model. VGGFace was the slowest model, for instrance, it was five times slower than SphereFace. OpenFace was three times slower than SphereFace. Figure 5.15 illustrates the inference times.

5.6 Discussion

Experiments executed in this chapter demonstrate that deep learning models can outperform the traditional algorithms in image processing. Particularly, for face processing, the evaluated deep learning models has shown

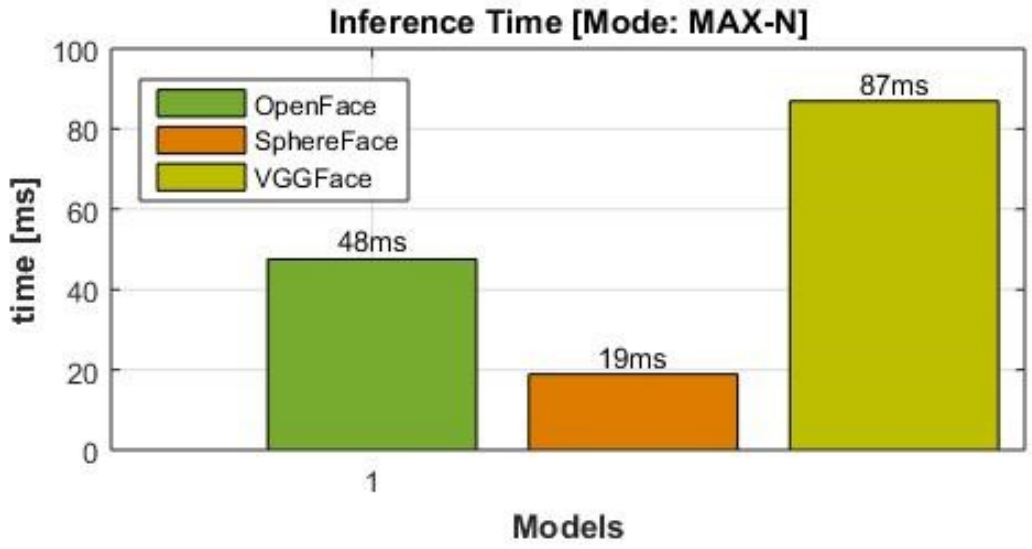

Figure 5.15: Inference time of the face encoders

to have equal or better performance that classical non-deep learning algorithms. Despite the deep learning models have not been trained specifically for their use in ABA therapy, they showed the capability of generalization and response with good performance in this environment. Using pre-trained deep learning models on embedded systems such as the Jetson TX2 produce some advantages. One advantage is that high performance models can be executed in real time. The training process, which is the hardest part of getting a pre-trained model, is not executed on the device. This helps to release resources on the device to improve other tasks. That is excellent for systems like social robots that need executed several tasks simultaneously. Another advantage is that the pre-trained models have countless possibilities for enhancement, improving the overall systems without modifying the code. Using pre-trained models also reduces the amount of code helping to minimize the error produced by bugs in the

code. Some non-deep learning approaches require hard coding and they are more susceptible to run-time mode errors.

Disadvantages of using pre-trained models is that they are inflexible prototypes. Once a model has been trained, parameters cannot be modified to adapt it to different conditions for those which were used in training. This makes it challenging to match models and build larger systems due to pre-trained models that are sensitive to the trained conditions. Landmarks detectors, for instance, are very sensitive to the face detector. If the face detector generates a slight variation in bounding boxes, the performance of the landmark detector is affected, resulting in an inaccurate portrayal of the facial features and a lower overall performance.

5.7 Summary

This chapter explores the use of pre-trained models to support two characteristics of the software architecture proposed: modularity and expandability. The use of deep learning models was incentivated because of the AI hardware architecture of PABI and the performance of deep learning models under uncontrolled conditions. The deep learning models evaluated focused on face detection, face landmark points detection, and face recognition. Those low-level features are the base in many application related with human-robot interaction.

Various experiments were designed to evaluate each deep learning model.

However, in general, two main variables were evaluated: robustness to extreme face poses and time processing. The experimental results indicate that deep learning models overpass traditional approaches, providing a rapid way of adding more characteristics to the robot. The main disadvantage of using deep learning models is the difficulty of combining two or more models in a process. The reason for this is because the deep learning models are very sensitives to the conditions that were used during the training. This problem can be overcome by retraining the models; however, the complexity of some models make that solution daunting. Deep learning models can still be used, but it would result in a lower performance.

Chapter 6

A Patient-Tracking Algorithm to Improve the Usability of PABI in Autism Treatments

Effectively incorporating a robot in a robot-enhanced ABA therapy is a challenging task. The design requires provision of high-level abilities in the robot such as identifying and maintaining attention of a particular individual. On the other side, the design needs to improve the usability of the robot for ease of use by the therapist. In an ABA therapy context in particular, the robot needs to track the patient in order to understand his or her behavior throughout the session. Information collected by the robot will be used later for improving its social interactions with the patient, thus increasing the effectiveness of ABA robot-enhanced therapies.

In this chapter, a patient-tracking algorithm is designed to improve the usability of PABI in autism treatments. The proposed algorithm is focused on improving the usability of the device while providing the robot with a high quality tracking system. One of the key ideas in this algorithm is to use an Open-set approach instead of a classical closed-set approach. Thus,

the problem of identifying an individual, the heart of the tracking system proposed, is constructed using a large-margin features approach instead of a classification approach. This chapter also presents the details of the research methodology implemented to evaluate the algorithm.

6.1 Motivations and Requirements

Attention is a cognitive process that consists on determining the correct stimulus around which a behavior is organized, ignoring other perceivable information.(Breazeal 2004). Maintaining attention in uncontrolled environments is relatively easy for humans, however, it is a challenging tasks for social robots such as PABI. In the framework of ABA therapy, attention begins by first recognizing the patient and the therapist during an ABA therapy session and tracking them, as well as disregarding extraneous information such as other individuals or stimulus in the scene.

Previous versions PABI implemented the aforementioned tracking system using deep learning and close-set approaches. In close set approaches, for each new patient added, a sample of images of the patient are taken and a deep learning model is trained. From a technical perspective, this approach works appropriately; however, from a usability perspective, the training process necessary to add a new patient, increases the complexity of robot use for the therapist. Thus, a tracking process that maintains the technical performance and reduces the complexity of robot use is necessary. Furthermore, the tracking system needs to have a real-time approach and be implemented on the high-level application layer proposed in Chapter 2.

6.2 Primary Contributions

The primary contribution of this chapter is the design of a tracking algorithm based on an open-set approach. In the proposed open-set approach, instead of training a new deep learning model each time that a new patient is added, an identity verification approach is used. This process consists in using patient information provided by the therapist before each therapy session, including an image of the patient. Then, a deep learning model for face encoding, combined with unsupervised methods, allows verification of individuals detected in the scene based on the patient image provided by the therapist. The proposed approach increase the usability of the robot by reducing its complexity setting for use during the therapy.

6.3 Background and Related Work

Focusing on the patient is an essential task for a robot during a robot-enhanced ABA therapy. Based on this task, subsequent tasks like face landmark detection or face encoding will be realized. However, although attention is easy for humans, it is a challenging task for the robot. Attention requires one to identify the target and track it. In the context of a robot-enhanced therapy, identifying the target signifies recognizing the patient and tracking him or her.

For the object tracking problem, several algorithms have been developed

in the last decade (Kalal, Mikolajczyk, and Matas 2012; Kalal, Mikolajczyk, and Matas 2010; Henriques et al. 2015; Babenko, M.-H. Yang, and Belongie 2009). Modern approaches of tracking are both fast and accurate. However, tracking alone cannot identify the object that needs to be tracked. This chapter is focused on the patient recognition, which is the base for the tracking system.

Two approaches have developed in the last years for identifying individuals: the closed-set approach and the open-set approach. The first approach addresses the problem of identifying people as a classification problem. The second approach addresses the problem using large-margin features (Weiyang Liu et al. 2017). Below these two approaches are described.

6.3.1 Patient Recognition Using a Closed-Set Approach

Close-set approaches address the problem of patient recognition as a classification problem. Thus, a classifier is trained to identify a patient between a set of patients. Pereira (Pereira 2017), for instance, uses a deep learning approach to train a classifier for a robot-enhanced ABA therapy. 6.1 (left) illustrates the closed-set approach.

A closed-set approach involves several stages. First, a set of patient images is necessary to build a training-set. The training-set includes patient images in different poses, improving the quality of the classifier. In the

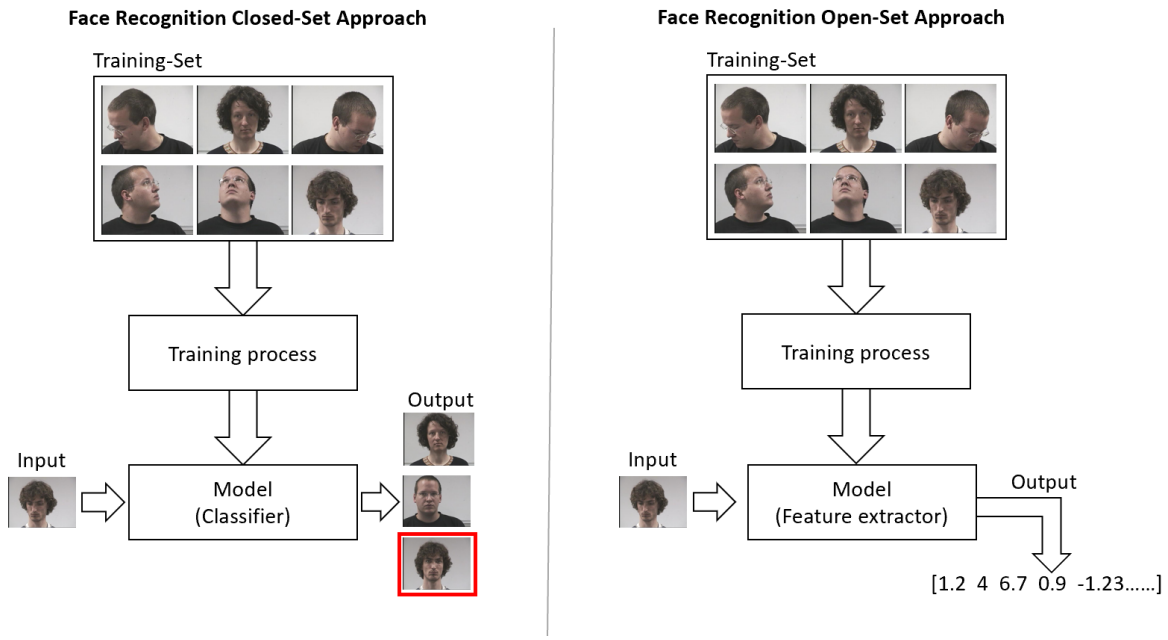

Figure 6.1: Closed-set approach (left) and Open-set approach (rigth)

next stage, the classifier is trained by combining the patient images with a supervised learning algorithm. Then, the trained classifier is used during the therapy session to identify the patient among a set of patients that includes the training set. If a new patient needs to be added, images of the new patient must be added to the training-set and the model must then be re-trained.

6.3.2 Patient Recognition Using a Open-Set Approach

Contrary to closed-set approaches, open-set approaches (Günther et al. 2017) address the problem as a metric learning one rather than one of classification. Figure 6.1 (right) illustrates an open-set approach. In this approach, a training-set is used to train a model that maps faces in a

discriminating n-dimensional features space. For each face, a specific n-dimensional vector is generated. Then, n-dimensional vectors are used to compare two faces in order to verify the identity of two faces. In this instance, it is unnecessary that the patient images are used in the training set. In fact, any set of images can be used to train the model. Additionally, the model needs to be trained once rather than multiple times as, for example, in a closed-set approach. Therefore, adding a new patient does not require retraining the model.

6.4 Tracking-Patient Algorithm Based on Face Verification

The main disadvantage of using a close-set approach for patient-tracking is that the model must be trained every time that a new patient is added to the list of patients. The training process reduces the usability of the robot because it implies additional procedures to use the robot. In some cases, the therapist is not able to execute this procedure without technical support. In contrast, in an open-set approach, the usability of the robot is increased and any procedure performed by the therapist is more streamlined. In the proposed algorithm, the patient profile is used to build an open-set approach for patient-tracking.

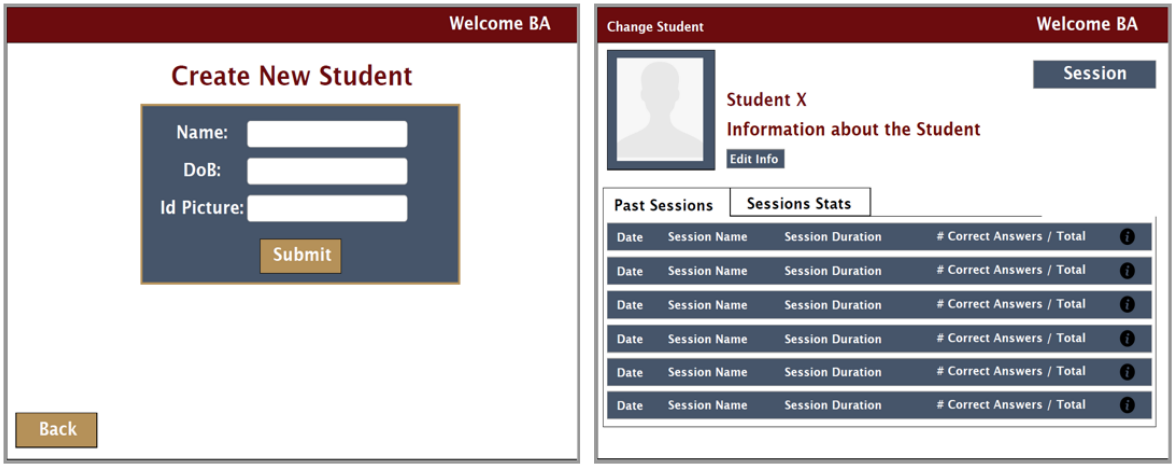

Figure 6.2: Examples of the PABI therapist interface

6.4.1 Robot-Enhanced ABA Therapy Interface.

In a robot-enhanced ABA therapy, prior to the therapy session, some routine procedures need to be completed by the therapist. These procedures involve selecting the patient, for example, or creating a medical profile of the patient if one does not exist or select the therapy to be applied, etc. For this work, the Therapy Configuration System defined in the robot-enhanced ABA therapy framework is used. Figure 6.2 illustrates the session information interface proposed for PABI.

6.4.2 Patient Recognition Algorithm for a Robot-Enhanced ABA Therapy

The proposed patient recognition algorithm is based on a open-set approach. The heart of the algorithm is a face-encoder based in deep learning. In order to recognize the patient during the therapy session, the faces

detected by the robot are compared with the image of the patient register in his or her medical profile. Some constraints are needed in the design of the algorithm due to the nature of the therapy session and reduce computational costs. Constraints of the ABA therapy session include the following:

- Close proximity of both patient and therapist for robot detection.
- Robot positioning with a specific orientation allowing faces to be captured.
- No more than two individuals present during the ABA therapy session.

Figure 6.3 illustrates the algorithm. First, the algorithm begins by detecting all possible faces in the input video frame. For each new frame, a new vector of faces are detected. It is important to note that there is not a particular order in which the faces are detected. Therefore, an element in the vector of faces detected does not represent the same user in each frame.In order to match the faces with the corresponding users, the Hungarian Algorithm is used (Kuhn 1955).

Once each of the faces have been assigned using the Hungarian algorithm, a code for every face is generated using a face encoder. A metric is then calculated between the detected faces and the facial image in the patient's medical profile. To reduce the variability in the metric, a moving average filter is applied. Finally, a threshold is used to identify the patient between the possibles candidates.

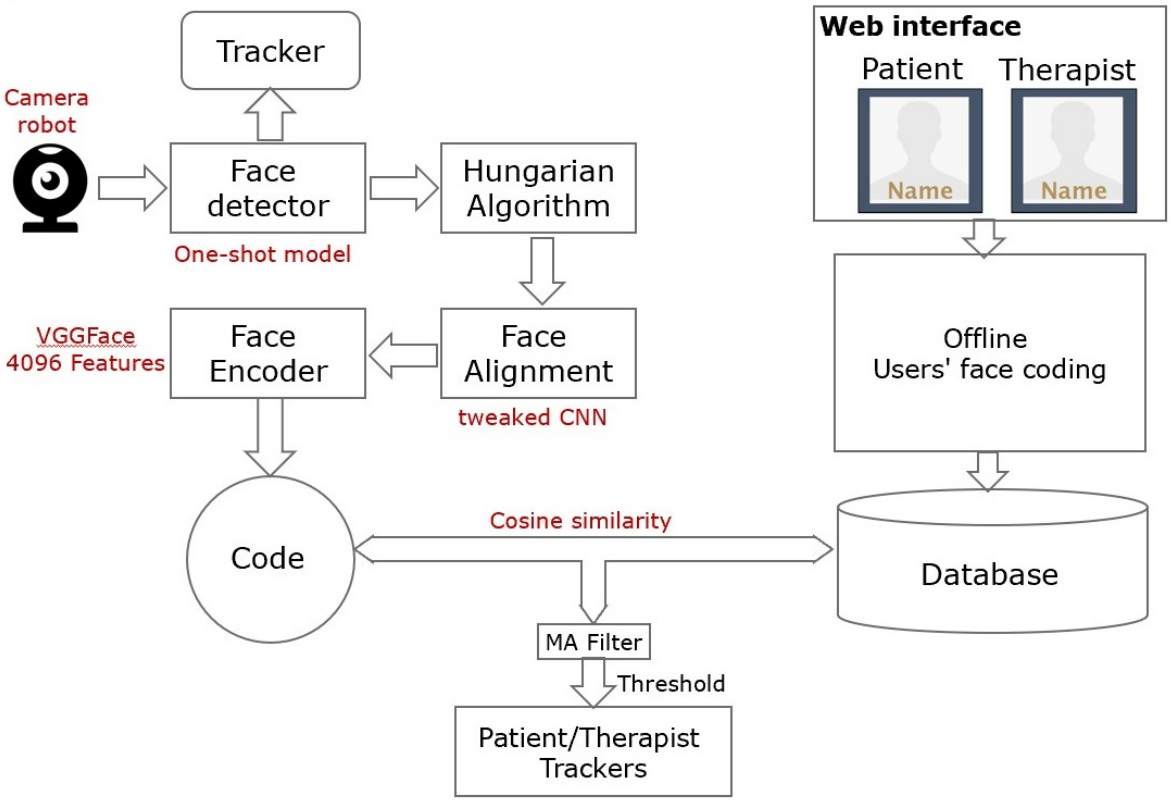

Figure 6.3: Patient Tracking Algorithm Based on an Open-Set Approach

6.5 Experimental Methods

Evaluating the patient recognition algorithm proved to be fraught with difficulty because it is contingent upon several changing factors. For example, the position of the camera, the configuration of the space during the therapy session, the internal parameters of the algorithm, among other things, can change the performance of the algorithm. However, in order to test the algorithm, two different videos where utilized. First, a previously recorded video of an experimental session using PABI was used. Second, a public video available on YouTube (*Lesha Zaslonkin, Invierno 2014, Comienzo de Clases - YouTube* 2019) was used to compare the algorithm's effectiveness in each scenarios, presenting the algorithm with various performance tasks.

In the first video, an ABA therapy is conducted by a therapist while the robot is recording the session. The robot is positioned in front of a young girl, the patient, and the therapist is positioned next to the girl facing the PABI robot. The therapist is not able to be viewed in all of the frames, however, the patient is always front and center, in the focal point of the robot's view. The therapist conducts the ABA therapy session for approximately four minutes and seventeen seconds. There are additional individuals in the background of the video, however, they are undetectable by the robot because they are not close enough in proximity to the PABI robot. The robot video is distorted in its shape due to the robot's fish-eye lens. A head shot of the patient was necessary for the robot to track the

Figure 6.4: Scenario 1: ABA therapy session using PABI and the face of reference used for this scenario

patient and, as such, a screen shot was taken from one video frame during the session. However, any frontal image can be used for this task. So, the image can be captured using a tablet, a phone o uploaded. The aim of this frontal patient photo was to assist the robot in identifying the patient throughout the session. Both therapist and patient face PABI and it is easy to identify the patient throughout the session as a result. Figure 6.4 illustrates a frame example of the ABA therapy and the reference face.

The second scenario is a video recording of about fifteen minutes of an ABA therapy session. There are several noticeable differences in the second scenario as compared to the first. The recording was not taken by PABI, but rather by a normal video recorder. In this case, the therapy

Figure 6.5: Scenario 2: Autism Therapy Session and the Face Used as a Reference in Scenario. Centro de terapia ABA (Producer). (2017, August 12). Lesha Zaslonkin, invierno 2014, comienzo de clases [Video file]. Retrieved from https://www.youtube.com/watch?v=-wro1VYry1g

is conducted with the therapist face-to-face with the patient rather than side-to-side such as in the first scenario. Because the faces of both the patient and therapist are recorded from a lateral view, it prevents a clear image of their faces. Similar to the previous scenario, a head shot of the patient was necessary for the robot to track the patient and a screen shot was taken from one video frame during the session. Figure 6.5 illustrates this scenario.

The applied conditions for the experiment are illustrated in Table 6.1. The moving average filter was used with 10 points and the threshold calculated in section 5.5.3 was utilized.

As a metric, the quality of matching between the patient's faces detected

Table 6.1: Experimental Setup

Parameters	Configuration
Face detector:	SSD-RES10
Face Landmarks detector:	Vanilla 5 Points
Face Encoding:	VGG Face Encoder
Jetson TX2 Mode	MAX-N

and the reference face was used. When a face was correctly matched, the face served as a reference for tracking the patient, however, when the face was lost, a tracking algorithm continued to track the user.

6.6 Experimental Results

6.6.1 Experimental Results Using the ABA Therapy Video

Figure 6.6 illustrates the results of the robot's patient recognition using 2500 frames. In this scenario, a moving average filter was not used. In total, the patient's face was detected in 2236 frames. From these faces detected, 2230 faces were over the threshold, signifying the algorithm matched 99.73% the reference face of the patient to the faces detected during the therapy. All the detected faces of the therapist were under the threshold, so false positives were not generated. Using a moving average filter with size, 100% of the faces were correctly classified. Results using a moving average filter, size 10, are illustrated in the Figure 6.7.

To compare, Figure 6.8 illustrates the results of the algorithm using a SphereFace Encoder. It is important to note that the metrics are different

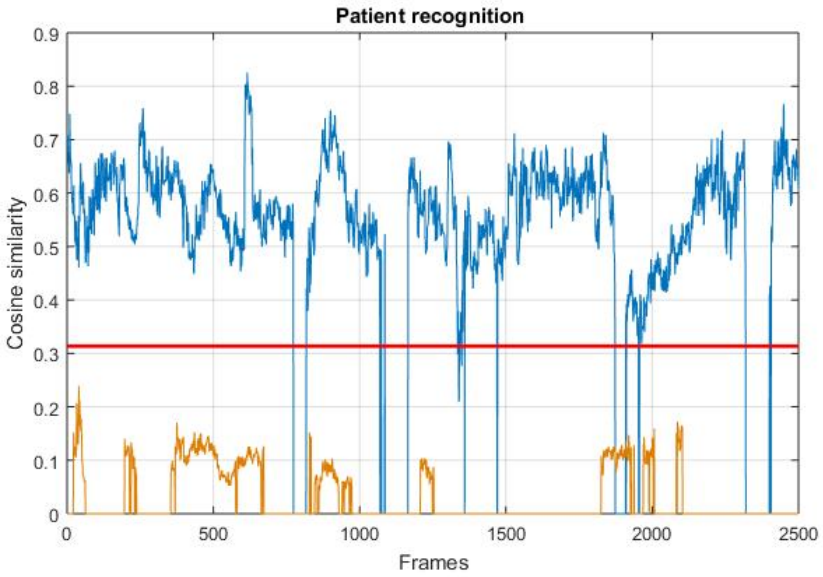

Figure 6.6: Similarity of faces detected with respect to the reference face in Scenario 1. The patient is marked in blue and the therapist in orange. The red line is the threshold used to separate the patient from the therapist.

Figure 6.7: Similarity of the faces detected with respect to the reference face using a 10 element moving average filter in Scenario 1. Blue signifies the patient and orange marks the therapist. The red line is the threshold used to separate the patient from the therapist.

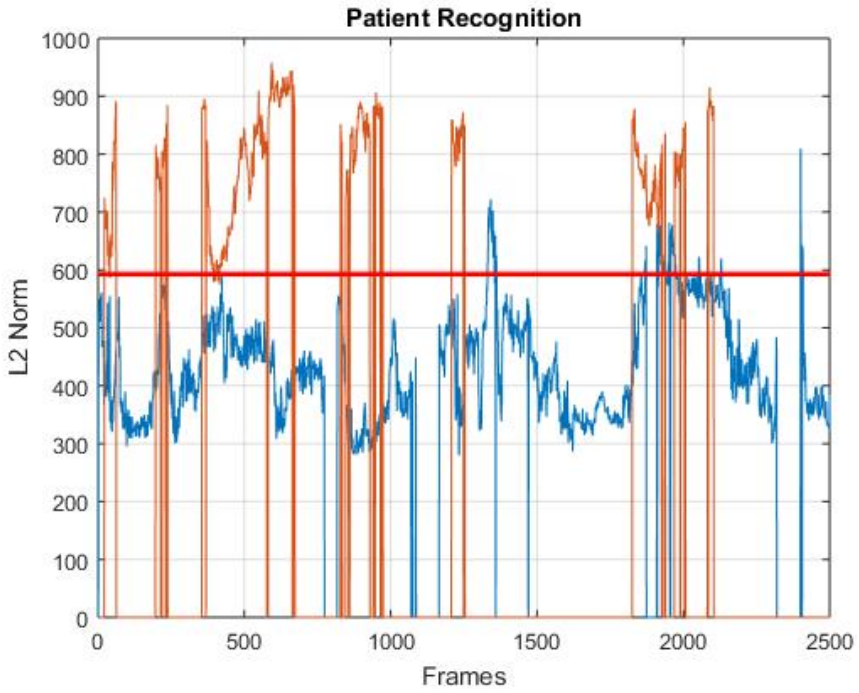

Figure 6.8: Similarity of the faces detected with respect to the reference face SphereFace in Scenario 1. The patient is marked in blue and the therapist in orange. The red line is the threshold used to separate the patient from the therapist.

and in this case, the matching face is under the threshold. The reason for this is that the lower the L2 norm, the higher the probability of two faces matching.

6.6.2 Experimental Results Using the public Therapy Video

Figure 6.9 illustrates the results of the robot's patient recognition using 2500 frames. In this case, a moving average filter was not used. In total, the patient's face was detected in 791 frames. From these faces detected, 476 faces were over the threshold. This means that the algorithm matched 60.17% of the times the reference face of the patient was paired with the faces detected during the therapy. All of the detected faces of the therapist

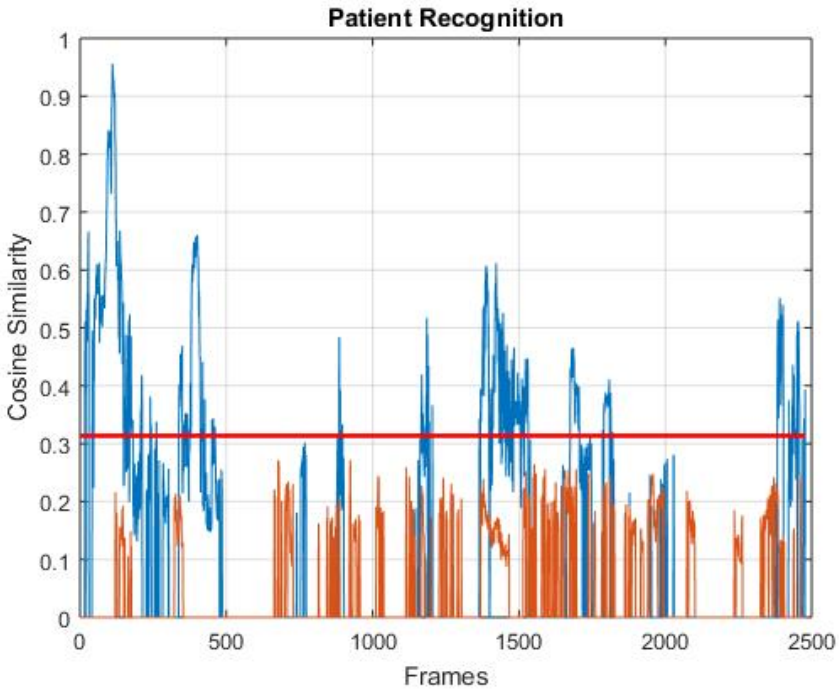

Figure 6.9: Similarity of the faces detected respect to the reference face in Scenario 2. The patient is marked in blue and the therapist in orange. The red line is the threshold used to separate the patient from the therapist.

were under the threshold, so false positives were not generated. Using a moving average filter with size then, 36.60% of the faces were accurately classified. Results using a moving average filter ,size 10, is illustrated in the figure 6.10.

The frame rate of the algorithm using the experimental setting specified in Table 6.1 was 2.4 frames per second. The algorithm is capable of reaching a frame rate of 10 frames per second when using SphereFace instead of VGG Face.

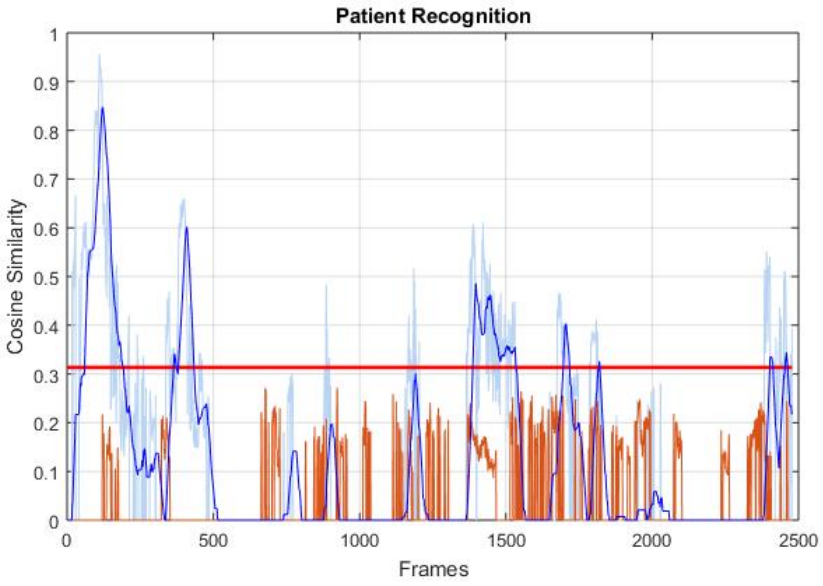

Figure 6.10: Similarity of the faces detected respect to the reference face using a 10 element moving average filter. The patient is marked in blue and the therapist in orange. The red line is the threshold used to separate the patient from the therapist.

6.7 Discussion

Experimental results demonstrate that an open-set approach has the potential to be effective in patient recognition. In Scenario 1 the algorithm had an almost perfect performance despite the varying conditions of the test. In this scenario, the PABI robot was included as a central part of the therapy.

In Scenario 2, performance dropped drastically. In this scenario, the algorithm was tested in an environment in which the robot was not considered. One of the main reasons for the lower performance was the position of the camera. In this scenario, most of the time, the therapist and patient worked face-to-face. It is evident that the location of the robot respect to the patient and the therapist played a key role in the performance of the

algorithm in face detection. Therefore, involving a robot in an ABA therapy requires not only appropriate technology but also a sound methodology to establish a proper work space to incorporate the robot in the therapy effectively.

In the algorithm proposed, complex methods for unsupervised learning were not included. The main reason for this was to improve the velocity of the algorithm. However, incorporating better learning algorithms have the potential to improve the performance in harder scenarios such as the example in Scenario 2.

Finally, the experimental results demonstrate that exploring open-set approaches improve human-robot interaction in a robot-enhanced ABA therapy and should be considered.

6.8 Summary

In this chapter, the use of an open-set approach to identify the patient during an ABA therapy session is explored. Identifying and tracking the patient are two essential tasks for better understanding of the robot-patient relationship. In the proposed approach, a patient image available in his or her medical profile is used by an algorithm in order to identify him or her throughout the duration of the therapy session. This approach varies with the closed-set approach where a set of images of the patient need to be taken, and the system must be re-trained each time a new patient is

added.

The algorithm consists of two important stages. After detecting the faces in an input image, the faces are assigned to the correct users using a Hungarian Algorithm. Second, the face of every user is encoded using a deep learning face encoder model. The code generated is compared with the image registered in the database using a specific metric. Finally, a threshold is used to know if the user corresponds to the user in the patient database. Once the patient is identified, a traditional KCF tracking algorithm is used to track the patient.

In this chapter, the algorithm is tested using two scenarios of ABA therapy sessions. In the first scenario, the video of an enhanced-robot ABA therapy is used. In this scenario, the robot is located directly in front of the patient and the therapist. In the second scenario, the algorithm is tested over a regular ABA therapy without the use of a robot-enhanced therapy and the patient has a more severe case of autism.

The experimental results demonstrate that the proposed algorithm has a very high performance in a structured session where the robot is considered and strategically located. However, the performance drops in a regular ABA therapy without the use of a robot primary due to the camera position and position of both therapist and patient.

Finally, the proposed approach is advantageous by improving the usability of the robot. No additional procedures are necessary for adding a new

patient. Further, no additional images of the patient need to be recorded for tracking purposes. This protects the identity of the patient while saving resources in the system.

Chapter 7

Discussion and Future Work

One of the biggest challenges of designing a computer framework for a Socially Assistive Robot in ABA Therapy is the absence of a medical framework to provide insights about the design itself. Scientific literature has made several attempts to incorporate Socially Assistive Robots in autism therapy. However, there is no formal framework to help streamline the process of introducing SARs in therapy. This work presented a first attempt at combining both Social Assistive Robots and Applied Behavioral Analysis within the same framework. The robot-enhanced ABA therapy framework is a small step forward toward building effective robots for the treatment of autism. The framework must be expanded with the contributions of therapists, psychologists, and parents or guardians.

This work also demonstrated that emerging technologies in artificial intelligence and AI embedded devices can be an opportunity to improve the performance of Socially Assistive Robots. Some of the advantages of using the JETSON TX2 were its small size, low-power consumption, and memory protection. The small size of the JETSON TX2 allows innovation

in the mechanical body of PABI, minimizing the constraints due to the size of its computing hardware. Low-power consumption provides autonomy and further reduces the cost of the therapy. The memory protection of the JETSON TX2 provides an extra layer of privacy protection, which is an essential characteristic in medical robots.

In this work, pre-trained models where tested as proof of a specific concept. However, training deep learning models for particular conditions of an ABA therapy session, such as extreme face poses, could improve the overall performance of the robot. Additionally, networks combining various processes such as MTCNN could be further explored.

Improved social interactions with both patients and therapists play a key role in the integration of this new technology as a therapeutic device to support people affected by ASDs. Using the open-set approaches for patient tracking that was proposed in chapter six is a preliminary step toward this aim. The algorithm was designed for a basic ABA therapy session. However, the algorithm could be improved to be used in more complex therapy environments with multiple patients. Using algorithms like Kalman Filter, the tracking system can have the robustness to identify multiple patients while additionally improving the robot's usability.

Below the primary contributions and future work are presented.

Design of the Robot-Enhanced ABA Therapy Framework

The Robot-Enhanced ABA Therapy Framework designed in this work is a novel approach for developing socially assistive robots in the treatment of ASDs. One core advantage of the framework is that it intertwines both reactive and deliberate therapeutic behaviors. The framework makes an abstraction of the physical body and the input/output interfaces are flexible and useful for any robotic platform. In other words, this framework has wide applications because it can be used with any type of robot and any kind of interface.

Future work on the Robot-Enhance ABA Therapy Framework includes fine-tuning of the proposed subsystems based on the feedback of experimental results. A multidisciplinary approach is necessary to do this work. Social interactions that make the robot effective must be studied, combining both technological and therapeutic approaches.

In this work, I proposed a multi-layer architecture software for supporting the proposed framework. However, different architectures can be designed and experimented with using the designed framework.

Impact of Deep Learning Frameworks in the Implementation of Real-Time and Multitasking Software Architecture

Experimental results in this work demonstrate the importance of selecting the proper software for an AI embedded device. Despite the availability and existence of several deep learning frameworks, not all of them present similar characteristics. Experimental results in this research reflect new ways to use the frameworks while in their parallel computing model.

One of the main disadvantages of having several deep learning frameworks is a general lack of standards in order to deploy this technology. Several limitations were discovered due to the incompatibility between frameworks. This limitation increased the complexity of using deep learning to support the designed software architecture. The Open Neural Network Exchange Format (ONNX) is the first attempt of companies, dedicated to deep learning, to tackle this issue, merging technology toward common standards.

Future work in the evaluation and implementation of deep learning frameworks in AI devices includes testing the ONNX standard over the embedded system to increase the efficiency of the deep learning frameworks. Additionally, upgrading the test with a new version of the frameworks is necessary.

Evaluating Performance of Pre-Trained Deep Learning Models for Low-Level Feature Extraction

Experimental results demonstrate pre-trained deep learning models surpassed the traditional approaches. Despite using a model that was not exclusively trained for application in a Robot-Enhanced ABA Therapy context, they exhibited robustness toward extreme poses present in therapy. An additional advantage of using deep learning models is that they aid in the creation of fault tolerant systems.

One of the main disadvantages of using deep-learning models is their sensitivity to training conditions. Therefore, a small deviation in the input with respect to the data that was used for the training, resulted in a drastic decrease in accuracy levels. This issue was observed, for instance, in the face-encoder. The face-encoder was highly sensitive to the output of the face landmarks detector.

Future work in this area includes training specific models to increase the overall performance of the robot for implementation in ABA therapies. Exploring combined approaches such as MTCNN, merging face detection with landmark detection, should be explored further.

Improving the Usability of PABI in Robot-Enhanced ABA Treatments

A simple open-set approach algorithm showed to be powerful enough to successfully identify and track the patient in an ABA therapy session. By

using this algorithm, complexity was decreased when using the robot during therapy, while concurrently increasing the robot's usability.

Future work in this area of investigation suggests improving the algorithm's performance in more challenging environments such as uncontrolled settings. Improvements to the algorithm should include avoidance of unnecessary individuals in the scene, while using the most useful algorithms for tracking both patient and therapist. Integration of the algorithm in the mechanical body should also be completed.

Extrapolating the algorithm in more complex environments is the next challenge. More intelligent algorithms, combining face-detectors with tracking systems, need to be developed in order to achieve greater efficiency.

Bibliography

Abadi, Martín et al. (2016). "TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems". In: arXiv: 1603.04467 [cs].

Ahmad, Aakash and Muhammad Ali Babar (2016). "Software Architectures for Robotic Systems: A Systematic Mapping Study". In: *Journal of Systems and Software* 122, pp. 16–39. ISSN: 01641212. DOI: 10.1016/j.jss.2016.08.039.

Babenko, Boris, Ming-Hsuan Yang, and Serge Belongie (2009). "Visual Tracking with Online Multiple Instance Learning". In: p. 8.

Baio, Jon (2018). "Prevalence of Autism Spectrum Disorder Among Children Aged 8 Years — Autism and Developmental Disabilities Monitoring Network, 11 Sites, United States, 2014". In: *MMWR. Surveillance Summaries* 67. ISSN: 1546-07381545-8636. DOI: 10.15585/mmwr.ss6706a1.

Baron-Cohen, Simon (2002). "The Extreme Male Brain Theory of Autism". In: *Trends in Cognitive Sciences* 6.6, pp. 248–254. ISSN: 1364-6613. DOI: 10.1016/S1364-6613(02)01904-6.

— (2006). "The Hyper-Systemizing, Assortative Mating Theory of Autism". In: *Progress in Neuro-Psychopharmacology and Biological Psychiatry* 30.5, pp. 865–872. ISSN: 02785846. DOI: 10.1016/j.pnpbp.2006.01.010.

Begum, Momotaz, Richard W. Serna, and Holly A. Yanco (2016). "Are Robots Ready to Deliver Autism Interventions? A Comprehensive Review". In: *International Journal*

of Social Robotics 8.2, pp. 157–181. ISSN: 1875-4791, 1875-4805. DOI: 10.1007/s12369-016-0346-y.

Bellini, Scott (2004). "Social Skill Deficits and Anxiety in High-Functioning Adolescents With Autism Spectrum Disorders". In: *Focus on Autism and Other Developmental Disabilities* 19.2, pp. 78–86. ISSN: 1088-3576, 1538-4829. DOI: 10.1177/10883576040190020201.

Breazeal, Cynthia L. (2004). *Designing Sociable Robots*. MIT Press. 294 pp. ISBN: 978-0-262-52431-5.

caffe.berkeleyvision.org (2019). URL: https://caffe.berkeleyvision.org/ (visited on 01/11/2019).

Cao, Hoang-Long, Pablo Gómez Esteban, et al. (2017). "A Survey on Behavior Control Architectures for Social Robots in Healthcare Interventions". In: *International Journal of Humanoid Robotics* 14.04, p. 1750021. ISSN: 0219-8436. DOI: 10.1142/S0219843617500219.

Cao, Hoang-Long, Pablo G. Esteban, et al. (2019). "Robot-Enhanced Therapy: Development and Validation of Supervised Autonomous Robotic System for Autism Spectrum Disorders Therapy". In: *IEEE Robotics & Automation Magazine* 26.2, pp. 49–58. ISSN: 1070-9932, 1558-223X. DOI: 10.1109/MRA.2019.2904121. URL: https://ieeexplore.ieee.org/document/8683968/ (visited on 07/26/2019).

Cao, Zhe et al. (2017). "Realtime Multi-Person 2D Pose Estimation Using Part Affinity Fields". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Honolulu, HI: IEEE, pp. 1302–1310. ISBN: 978-1-5386-0457-1. DOI: 10.1109/CVPR.2017.143.

Collobert, Ronan, Koray Kavukcuoglu, and Clement Farabet (2011). "Torch7: A Matlab-like Environment for Machine Learning". In: p. 6.

Costa, S. et al. (2009). "Applications of Simple Robots to Encourage Social Receptiveness of Adolescents with Autism". In: *2009 Annual International Conference of the IEEE Engineering in Medicine and Biology Society*. 2009 Annual International Conference of the IEEE Engineering in Medicine and Biology Society, pp. 5072–5075. DOI: 10.1109/IEMBS.2009.5334269.

Deep Learning in OpenCV (2019). URL: https://github.com/opencv/opencv/wiki/Deep-Learning-in-OpenCV (visited on 01/11/2019).

Deng, Li (2012). "Three Classes of Deep Learning Architectures and Their Applications: A Tutorial Survey". In: p. 28.

Dickstein-Fischer, Laurie A. et al. (2017). "Interactive Tracking for Robot-Assisted Autism Therapy". In: *Proceedings of the Companion of the 2017 ACM/IEEE International Conference on Human-Robot Interaction - HRI '17*. The Companion of the 2017 ACM/IEEE International Conference. Vienna, Austria: ACM Press, pp. 107–108. ISBN: 978-1-4503-4885-0. DOI: 10.1145/3029798.3038390.

Dickstein-Fischer, Laurie A et al. (2018). "Socially Assistive Robots: Current Status and Future Prospects for Autism Interventions". In: *Innovation and Entrepreneurship in Health* Volume 5, pp. 15–25. ISSN: 2324-5905. DOI: 10.2147/IEH.S138753.

Diehl, Joshua J. et al. (2012). "The Clinical Use of Robots for Individuals with Autism Spectrum Disorders: A Critical Review". In: *Research in Autism Spectrum Disorders* 6.1, pp. 249–262. ISSN: 17509467. DOI: 10.1016/j.rasd.2011.05.006. URL: https://linkinghub.elsevier.com/retrieve/pii/S1750946711000894 (visited on 07/27/2019).

DSM-IV Diagnostic Classifications (2019). URL: http://www.autism-society.org/dsm-iv-diagnostic-classifications/ (visited on 01/11/2019).

Feil-Seifer, D. and M. J. Mataric (2005). "Defining Socially Assistive Robotics". In: *9th International Conference on Rehabilitation Robotics, 2005. ICORR 2005.* 9th International Conference on Rehabilitation Robotics, 2005. ICORR 2005. Pp. 465–468. DOI: 10.1109/ICORR.2005.1501143.

Ferland, François et al. (2013). "Natural Interaction Design of a Humanoid Robot". In: *Journal of Human-Robot Interaction* 1.2. ISSN: 2163-0364. DOI: 10.5898/JHRI.1.2.Ferland.

Fisher, Heidi R. et al. (2019). "Applied Behavior Analysis and Related Treatments". In: *Handbook of Interdisciplinary Treatments for Autism Spectrum Disorder*. Ed. by Robert D. Rieske. Autism and Child Psychopathology Series. Cham: Springer International Publishing, pp. 111–129. ISBN: 978-3-030-13027-5. DOI: 10.1007/978-3-030-13027-5_7.

Girshick, Ross et al. (2013). "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation". In: arXiv: 1311.2524 [cs].

Gourier, Nicolas, Daniela Hall, and James L Crowley (n.d.). "Estimating Face Orientation from Robust Detection of Salient Facial Structures". In: p. 9.

Granpeesheh, Doreen and Jonathan Tarbox (2009). "Applied Behavior Analytic Interventions for Children with Autism: A Description and Review of Treatment Research". In: *ANNALS OF CLINICAL PSYCHIATRY*, p. 13.

Green, Vanessa A. et al. (2006). "Internet Survey of Treatments Used by Parents of Children with Autism". In: *Research in Developmental Disabilities* 27.1, pp. 70–84. ISSN: 08914222. DOI: 10.1016/j.ridd.2004.12.002.

Günther, Manuel et al. (2017). "Toward Open-Set Face Recognition". In: arXiv: 1705.01567 [cs].

Hashim, Hafizan et al. (2013). "Robot-Assisted to Elicit Behaviors for Autism Screening". In: *Applied Mechanics and Materials; Zurich* 393, p. 567. ISSN: 16609336. DOI: http://dx.doi.org/10.4028/www.scientific.net/AMM.393.567.

He, Kaiming et al. (2015). "Deep Residual Learning for Image Recognition". In: arXiv: 1512.03385 [cs].

Heitzman-Powell, Linda S. et al. (2014). "Formative Evaluation of an ABA Outreach Training Program for Parents of Children With Autism in Remote Areas". In: *Focus*

on Autism and Other Developmental Disabilities 29.1, pp. 23–38. ISSN: 1088-3576, 1538-4829. DOI: 10.1177/1088357613504992.

Henriques, João F. et al. (2015). "High-Speed Tracking with Kernelized Correlation Filters". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 37.3, pp. 583–596. ISSN: 0162-8828, 2160-9292. DOI: 10.1109/TPAMI.2014.2345390. arXiv: 1404.7584.

Hiolle, Antoine et al. (2012). "Eliciting Caregiving Behavior in Dyadic Human-Robot Attachment-like Interactions". In: *ACM Transactions on Interactive Intelligent Systems* 2.1, pp. 1–24. ISSN: 21606455. DOI: 10.1145/2133366.2133369.

Hogg, David (1983). "Model-Based Vision: A Program to See a Walking Person". In: *Image and Vision Computing* 1.1, pp. 5–20. ISSN: 0262-8856. DOI: 10.1016/0262-8856(83)90003-3.

Howard, Jane S. et al. (2005). "A Comparison of Intensive Behavior Analytic and Eclectic Treatments for Young Children with Autism". In: *Research in Developmental Disabilities* 26.4, pp. 359–383. ISSN: 08914222. DOI: 10.1016/j.ridd.2004.09.005.

Jia, Yangqing et al. (2014). "Caffe: Convolutional Architecture for Fast Feature Embedding". In: arXiv: 1408.5093 [cs].

Kalal, Zdenek, Krystian Mikolajczyk, and Jiri Matas (2010). "Forward-Backward Error: Automatic Detection of Tracking Failures". In: *2010 20th International Conference on Pattern Recognition*. 2010 20th International Conference on Pattern Recognition (ICPR). Istanbul, Turkey: IEEE, pp. 2756–2759. ISBN: 978-1-4244-7542-1. DOI: 10.1109/ICPR.2010.675.

— (2012). "Tracking-Learning-Detection". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 34.7, pp. 1409–1422. ISSN: 0162-8828, 2160-9292. DOI: 10.1109/TPAMI.2011.239.

Kazemi, Vahid and Josephine Sullivan (2014). "One Millisecond Face Alignment with an Ensemble of Regression Trees". In: *2014 IEEE Conference on Computer Vision*

and Pattern Recognition. 2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Columbus, OH: IEEE, pp. 1867–1874. ISBN: 978-1-4799-5118-5. DOI: 10.1109/CVPR.2014.241.

Kelly, Michael David (1971). "Visual Identification of People by Computer". PhD Thesis. Stanford, CA, USA: Stanford University.

Kim, Elizabeth S. et al. (2013). "Social Robots as Embedded Reinforcers of Social Behavior in Children with Autism". In: *Journal of Autism and Developmental Disorders* 43.5, pp. 1038–1049. ISSN: 0162-3257, 1573-3432. DOI: 10.1007/s10803-012-1645-2.

Kim, Jonghoon et al. (2008). "Intelligent Robot Software Architecture". In: *Recent Progress in Robotics: Viable Robotic Service to Human*. Ed. by Sukhan Lee, Il Hong Suh, and Mun Sang Kim. Vol. 370. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 385–397. ISBN: 978-3-540-76728-2. DOI: 10.1007/978-3-540-76729-9_30.

Kuhn, H. W. (1955). "The Hungarian Method for the Assignment Problem". In: *Naval Research Logistics Quarterly* 2.1-2, pp. 83–97. ISSN: 00281441, 19319193. DOI: 10.1002/nav.3800020109.

LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton (2015). "Deep Learning". In: *Nature* 521.7553, pp. 436–444. ISSN: 0028-0836, 1476-4687. DOI: 10.1038/nature14539.

Lesha Zaslonkin, Invierno 2014, Comienzo de Clases - YouTube (2019). URL: https://www.youtube.com/watch?v=-wro1VYry1g (visited on 05/28/2019).

Liu, Weiyang et al. (2017). "SphereFace: Deep Hypersphere Embedding for Face Recognition". In: arXiv: 1704.08063 [cs].

Liu, Wei et al. (2016). "SSD: Single Shot MultiBox Detector". In: *Computer Vision – ECCV 2016*. Ed. by Bastian Leibe et al. Vol. 9905. Cham: Springer International Publishing, pp. 21–37. ISBN: 978-3-319-46447-3 978-3-319-46448-0. DOI: 10.1007/978-3-319-46448-0_2.

Miskam, Mohd Azfar et al. (2014). "Encouraging Children with Autism to Improve Social and Communication Skills through the Game-based Approach". In: *Procedia Computer Science* 42, pp. 93–98. ISSN: 18770509. DOI: 10.1016/j.procs.2014.11.038.

Morton, John and Mark H Johnson (1991). "CONSPEC and CONLERN: A Two-Process Theory of Infant Face Recognition". In: p. 18.

Murphy-Chutorian, E. and M.M. Trivedi (2009). "Head Pose Estimation in Computer Vision: A Survey". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 31.4, pp. 607–626. ISSN: 0162-8828. DOI: 10.1109/TPAMI.2008.106.

O'Rourke, J. and N. I. Badler (1980). "Model-Based Image Analysis of Human Motion Using Constraint Propagation". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAMI-2.6, pp. 522–536. ISSN: 0162-8828. DOI: 10.1109/TPAMI.1980.6447699.

Pakkar, Roxanna et al. (2018). "Designing a Socially Assistive Robot for Long-Term In-Home Use for Children with Autism Spectrum Disorders". In: p. 7.

Pantic, M. and L. J. M. Rothkrantz (2000). "Automatic Analysis of Facial Expressions: The State of the Art". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 22.12, pp. 1424–1445. ISSN: 0162-8828. DOI: 10.1109/34.895976.

Pennisi, Paola et al. (2016). "Autism and Social Robotics: A Systematic Review". In: *Autism Research* 9.2, pp. 165–183. ISSN: 1939-3806. DOI: 10.1002/aur.1527.

Pereira, Ria (2017). "Interactive Behavior for Humanoid Robot Mediated Applied Behavioral Analysis Autism Therapy". Worcester: Worcester Polytechinic Institute.

Raj, Pethuru, Anupama Raman, and Harihara Subramanian (2017). *Architectural Patterns: Uncover Essential Patterns in the Most Indispensable Realm of Enterprise Architecture*. 1 edition. Packt Publishing. 468 pp.

Redmon, Joseph et al. (2015). "You Only Look Once: Unified, Real-Time Object Detection". In: arXiv: 1506.02640 [cs].

Reichow, Brian et al. (2012). "Early Intensive Behavioral Intervention (EIBI) for Young Children with Autism Spectrum Disorders (ASD)". In: *Cochrane Database of Systematic Reviews* 10. ISSN: 1465-1858. DOI: 10.1002/14651858.CD009260.pub2.

Ren, Shaoqing et al. (2014). "Face Alignment at 3000 FPS via Regressing Local Binary Features". In: *2014 IEEE Conference on Computer Vision and Pattern Recognition*. 2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Columbus, OH, USA: IEEE, pp. 1685–1692. ISBN: 978-1-4799-5118-5. DOI: 10.1109/CVPR.2014.218.

Robins, Ben, Kerstin Dautenhahn, and Janek Dubowski (2006). *Interaction Studies 7:3 (2006), 479–52. Issn 1572–0373 / e-Issn 1572–0381 © John Benjamins Publishing Company.*

Robots Help Teach STEM Concepts to Students With Autism (2018).

Rowley, Henry A, Shumeet Baluja, and Takeo Kanade (1996). "Human Face Detection in Visual Scenes". In: p. 7.

Rudovic, Ognjen et al. (2018). "Personalized Machine Learning for Robot Perception of Affect and Engagement in Autism Therapy". In: *Science Robotics* 3.19, eaao6760. ISSN: 2470-9476. DOI: 10.1126/scirobotics.aao6760.

Russakovsky, Olga et al. (2015). "ImageNet Large Scale Visual Recognition Challenge". In: *International Journal of Computer Vision* 115.3, pp. 211–252. ISSN: 0920-5691, 1573-1405. DOI: 10.1007/s11263-015-0816-y.

Saerbeck, M (2009). *Software Architecture for Social Robots*. OCLC: 8086847112. Technische Universiteit Eindhoven. ISBN: 978-90-5335-234-2.

Scassellati, Brian et al. (2018). "Improving Social Skills in Children with ASD Using a Long-Term, in-Home Social Robot". In: *Science Robotics* 3.21, eaat7544. ISSN: 2470-9476. DOI: 10.1126/scirobotics.aat7544.

Schroff, Florian, Dmitry Kalenichenko, and James Philbin (2015). "FaceNet: A Unified Embedding for Face Recognition and Clustering". In: DOI: 10.1109/CVPR.2015.7298682.

Simonyan, Karen and Andrew Zisserman (2015). "VERY DEEP CONVOLUTIONAL NETWORKS FOR LARGE-SCALE IMAGE RECOGNITION". In: p. 14.

Srinivasan, Sudha M. et al. (2015). "The Effects of Rhythm and Robotic Interventions on the Imitation/Praxis, Interpersonal Synchrony, and Motor Performance of Children with Autism Spectrum Disorder (ASD): A Pilot Randomized Controlled Trial". In: *Autism Research and Treatment* 2015, p. 736516. ISSN: 2090-1925. DOI: 10.1155/2015/736516. pmid: 26793394.

Szegedy, Christian et al. (2014). "Going Deeper with Convolutions". In: arXiv: 1409.4842 [cs].

Tapus, A., M. J. Mataric, and B. Scassellati (2007). "Socially Assistive Robotics [Grand Challenges of Robotics]". In: *IEEE Robotics Automation Magazine* 14.1, pp. 35–42. ISSN: 1070-9932. DOI: 10.1109/MRA.2007.339605.

Toshev, Alexander and Christian Szegedy (2014). "DeepPose: Human Pose Estimation via Deep Neural Networks". In: *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1653–1660. DOI: 10.1109/CVPR.2014.214. arXiv: 1312.4659.

Towards Autonomous Robotic Systems (2011). *Towards Autonomous Robotic Systems: 12th Annual Conference, TAROS 2011, Sheffield, UK, August 31 - September 2, 2011: Proceedings*. Lecture Notes in Computer Science 6856. OCLC: ocn749781525. Berlin: Springer. 435 pp. ISBN: 978-3-642-23231-2.

Trigueros, Daniel Sáez, Li Meng, and Margaret Hartnett (2018). "Face Recognition: From Traditional to Deep Learning Methods". In: arXiv: 1811.00116 [cs].

Viola, Paul and Michael Jones (n.d.). "Robust Real-Time Object Detection". In: p. 25.

Waltz, Emily (2018). *Therapy Robot Teaches Social Skills to Children With Autism*.

Wu, Yue, Tal Hassner, et al. (2015). "Facial Landmark Detection with Tweaked Convolutional Neural Networks". In: arXiv: 1511.04031 [cs].

Wu, Yue and Qiang Ji (2019). "Facial Landmark Detection: A Literature Survey". In: *International Journal of Computer Vision* 127.2, pp. 115–142. ISSN: 0920-5691, 1573-1405. DOI: 10.1007/s11263-018-1097-z. arXiv: 1805.05563.

Yang, Yi and Deva Ramanan (2013). "Articulated Human Detection with Flexible Mixtures of Parts". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35.12, pp. 2878–2890. ISSN: 0162-8828, 2160-9292. DOI: 10.1109/TPAMI.2012.261.

Yow, Kin Choong and Roberto Cipolla (1997). "Feature-Based Human Face Detection". In: p. 23.

Zachor, Ditza A. and Professor Joav Merrick (2012). *Understanding Autism Spectrum Disorder: Current Research Aspects*. New York, UNITED STATES: Nova Science Publishers, Incorporated. ISBN: 978-1-62081-390-4.

Zheng, Zhi et al. (2016). "Robot-Mediated Imitation Skill Training for Children with Autism". In: *IEEE transactions on neural systems and rehabilitation engineering : a publication of the IEEE Engineering in Medicine and Biology Society* 24.6, pp. 682–691. ISSN: 1534-4320. DOI: 10.1109/TNSRE.2015.2475724. pmid: 26353376.