Artificial Intelligence as a Service Architecture: an innovative approach for Computer Vision applications

Larissa Ferreira Rodrigues Moreira

Universidade Federal de Uberlândia
Faculdade de Computação
Programa de Pós-Graduação em Ciência da Computação

Uberlândia
2024

Larissa Ferreira Rodrigues Moreira

Artificial Intelligence as a Service Architecture: an innovative approach for Computer Vision applications

Tese de doutorado apresentada ao Programa de Pós-graduação da Faculdade de Computação da Universidade Federal de Uberlândia como parte dos requisitos para a obtenção do título de Doutora em Ciência da Computação.

Área de concentração: Ciência da Computação

Orientador: Prof. Dr. André Ricardo Backes
Coorientador: Prof. Dr. Bruno Augusto Nassif Travençolo

Uberlândia
2024

Ficha Catalográfica Online do Sistema de Bibliotecas da UFU
com dados informados pelo(a) próprio(a) autor(a).

M838 2024

Moreira, Larissa Ferreira Rodrigues, 1994-
Artificial Intelligence as a Service Architecture: an innovative approach for Computer Vision applications [recurso eletrônico] / Larissa Ferreira Rodrigues Moreira. - 2024.

Orientador: André Ricardo Backes.
Coorientador: Bruno Augusto Nassif Travençolo.
Tese (Doutorado) - Universidade Federal de Uberlândia, Pós-graduação em Ciência da Computação.
Modo de acesso: Internet.
Disponível em: http://doi.org/10.14393/ufu.te.2024.675
Inclui bibliografia.
Inclui ilustrações.

1. Computação. I. Backes, André Ricardo,1981-, (Orient.). II. Travençolo, Bruno Augusto Nassif ,1981-, (Coorient.). III. Universidade Federal de Uberlândia. Pós-graduação em Ciência da Computação. IV. Título.

CDU: 681.3

Bibliotecários responsáveis pela estrutura de acordo com o AACR2:

Gizele Cristine Nunes do Couto - CRB6/2091
Nelson Marcos Ferreira - CRB6/3074

UNIVERSIDADE FEDERAL DE UBERLÂNDIA
Coordenação do Programa de Pós-Graduação em Ciência da Computação
Av. João Naves de Ávila, 2121, Bloco 1A, Sala 243 - Bairro Santa Mônica, Uberlândia-MG, CEP 38400-902
Telefone: (34) 3239-4470 - www.ppgco.facom.ufu.br - cpgfacom@ufu.br

ATA DE DEFESA - PÓS-GRADUAÇÃO

Programa de Pós-Graduação em:	Ciência da Computação				
Defesa de:	Tese, 41/2024, PPGCO				
Data:	23 de setembro de 2024	Hora de início:	13:40	Hora de encerramento:	17:10
Matrícula do Discente:	12023CCP004				
Nome do Discente:	Larissa Ferreira Rodrigues Moreira				
Título do Trabalho:	Artificial Intelligence as a Service Architecture: an innovative approach for Computer Vision applications				
Área de concentração:	Ciência da Computação				
Linha de pesquisa:	Ciência de Dados				
Projeto de Pesquisa de vinculação:	----------				

Reuniu-se por videoconferência, a Banca Examinadora, designada pelo Colegiado do Programa de Pós-graduação em Ciência da Computação, assim composta: Professores Doutores: Bruno Augusto Nassif Travençolo - FACOM/UFU (Coorientador), Alessandra Aparecida Paulino - FACOM/UFU, João Henrique de Souza Pereira - FACOM/UFU, Dalcimar Casanova - UTFPR, João Batista Florindo - IMECC/ Unicamp e André Ricardo Backes- DC/UFSCar, orientador da candidata.

Os examinadores participaram desde as seguintes localidades: André Ricardo Backes - São Carlos/SP, Dalcimar Casanova - Pato Branco/PR e João Batista Florindo - Campinas/SP. Os outros membros da banca e o aluno participaram da cidade de Uberlândia.

Iniciando os trabalhos o presidente da mesa, Prof. Dr. André Ricardo Backes, apresentou a Comissão Examinadora e o candidato, agradeceu a presença do público, e concedeu á Discente a palavra para a exposição do seu trabalho. A duração da apresentação da Discente e o tempo de arguição e resposta foram conforme as normas do Programa.

A seguir o senhor presidente concedeu a palavra, pela ordem sucessivamente, aos examinadores, que passaram a arguir á candidato. Ultimada a arguição, que se desenvolveu dentro dos termos regimentais, a Banca, em sessão secreta, atribuiu o resultado final, considerando o candidato:

Aprovado

Esta defesa faz parte dos requisitos necessários à obtenção do título de Doutor.

O competente diploma será expedido após cumprimento dos demais requisitos, conforme as normas do Programa, a legislação pertinente e a regulamentação interna da UFU.

Nada mais havendo a tratar foram encerrados os trabalhos. Foi lavrada a presente ata que após lida e achada conforme foi assinada pela Banca Examinadora.

Documento assinado eletronicamente por **André Ricardo Backes**, **Usuário Externo**, em 25/09/2024, às 15:09, conforme horário oficial de Brasília, com fundamento no art. 6º, § 1º, do Decreto nº 8.539, de 8 de outubro de 2015.

Documento assinado eletronicamente por **Dalcimar Casanova**, **Usuário Externo**, em 25/09/2024, às 17:12, conforme horário oficial de Brasília, com fundamento no art. 6º, § 1º, do Decreto nº 8.539, de 8 de outubro de 2015.

Documento assinado eletronicamente por **João Batista Florindo**, **Usuário Externo**, em 25/09/2024, às 22:37, conforme horário oficial de Brasília, com fundamento no art. 6º, § 1º, do Decreto nº 8.539, de 8 de outubro de 2015.

Documento assinado eletronicamente por **João Henrique de Souza Pereira**, **Professor(a) do Magistério Superior**, em 26/09/2024, às 10:23, conforme horário oficial de Brasília, com fundamento no art. 6º, § 1º, do Decreto nº 8.539, de 8 de outubro de 2015.

Documento assinado eletronicamente por **Alessandra Aparecida Paulino**, **Professor(a) do Magistério Superior**, em 26/09/2024, às 14:09, conforme horário oficial de Brasília, com fundamento no art. 6º, § 1º, do Decreto nº 8.539, de 8 de outubro de 2015.

Documento assinado eletronicamente por **Bruno Augusto Nassif Travençolo**, **Professor(a) do Magistério Superior**, em 26/09/2024, às 15:37, conforme horário oficial de Brasília, com fundamento no art. 6º, § 1º, do Decreto nº 8.539, de 8 de outubro de 2015.

A autenticidade deste documento pode ser conferida no site https://www.sei.ufu.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0, informando o código verificador **5687011** e o código CRC **FD857360**.

Referência: Processo nº 23117.059853/2024-21 SEI nº 5687011

To my family.

Acknowledgements

Earning a PhD is far more than just achieving academic milestones. It represents the culmination years filled with challenges, sacrifices, and most importantly, learning. It is not merely about mastering theories and techniques, but rather an unyielding pursuit of understanding and crafting solutions that, in some way, can contribute to the greater good.

Throughout this process, I realized that technical knowledge, as valuable as it is, is merely a tool for larger purposes. True wisdom lies in knowing how to apply that knowledge to serve others and, finally, to glorify God in all that we do. Thus, I thank God, whose boundless grace sustained me at every step of the way. I acknowledge that everything I have achieved is not by my own strength but by His grace, which works in us both to will and to act according to His purpose.

I am grateful to my parents, Joaquim and Abadia, for their love and encouragement. You were my first life teacher, imparting me the value of knowledge and the importance of perseverance in the face of challenges from an early age. Your unconditional support and prayers are the foundations upon which I built this journey. I thank my brother Leonardo for your love, prayer, partnership, and support at all times.

To my loved husband Rodrigo, my partner, not only in life but also in our shared academic and professional journey. Your love, understanding, and joy have made your support more meaningful. I am grateful for your presence on my side through every step, for sharing knowledge, exchanging ideas, and encouraging me to persevere, especially during difficult times. This work reflects on what we are building together.

To my in-laws Donizete and Claudia and my brothers-in-law Thiago and Júnior, I am grateful for your prayers, support, and encouragement throughout this journey.

I extend my gratitude to my entire family for their support, love, and encouragement throughout at every stage of my life.

I am grateful to my advisors, Professors Dr. André Backes and Dr. Bruno Travençolo, for their guidance and support throughout this study. Your expertise, patience, and encouragement were essential for shaping this work. Beyond the academic mentorship, I

am grateful for the friendship that developed through our collaboration.

Gratitude to the professors of the Graduate Program in Computer Science at the Federal University of Uberlândia (UFU), especially those who honored me with their teachings: Dra. Gina Oliveira, Dr. Henrique Fernandes, Dra. Maria Camila Barioni, Dr. João Henrique Pereira, and Dr. Paulo Ribeiro Gabriel.

I thank Professor Dr. Flávio de Oliveira Silva for friendship with our family and for research collaboration along this way.

I would like to thank the Postgraduate Secretariat for the promptness and diligent management of academic procedures.

I express my gratitude to the board examiners Dra. Alessandra Paulino (UFU), Dr. Dalcimar Casanova (Federal University of Technology - Paraná), Dr. João Florindo (University of Campinas), and Dr. João Henrique Pereira (UFU). Thank you for your time, review, and feedback throughout the evaluation process.

I thank my undergraduate students at the UFU and Federal University of Viçosa (UFV): Emanuel T. Martins, João Vítor S. de Lima, Lucas Nardelli B. F. Saar, Thiago V. Machado, Victor F. Jansen, and Yasmin S. Lima, for collaboration with my research.

I am grateful to the UFV for its functional support, resources, and opportunities for my professional development. I would also like to extend my thanks to my colleagues in the Information Systems undergraduate course for their partnership in the teaching mission.

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Finance Code 001. I am grateful for the financial support provided by CAPES throughout my PhD research.

To all who contributed, both directly and indirectly, I extend my heartfelt thanks.

"Yours, LORD, is the greatness and the power
and the glory and the majesty and the splendor,
for everything in heaven and earth is yours.
Yours, LORD, is the kingdom;
you are exalted as head over all.
Wealth and honor come from you;
you are the ruler of all things.
In your hands are strength and power
to exalt and give strength to all.
Now, our God, we give you thanks,
and praise your glorious name."
(I Chronicles 29.11-13)

Resumo

Nos últimos anos, o uso da Inteligência Artificial (IA) experimentou um crescimento exponencial em diversos domínios da vida cotidiana, incluindo transporte, saúde e segurança. No entanto, a atual implementação de concepção e implementação de serviços inteligentes torna desafiador aumentar o uso personalizado, organizado e em larga escala da IA, especialmente para lidar com tarefas complexas para criação de recursos inteligentes. Neste contexto, esta tese propõe e avalia uma Arquitetura de IA como Serviço (IAaaS) que visa abordar a falta de métodos atuais para implantar e fornecer serviços inteligentes para dispositivos heterogêneos e múltiplos usuários. O principal objetivo deste projeto é pesquisar, desenvolver e validar uma Arquitetura de Inteligência Artificial como um Serviço (IAaaS) que ofereça soluções e recursos baseados em IA sob demanda para usuários e aplicativos. Para isso, é levantada uma hipótese sobre a viabilidade e adequação de fornecer recursos de IA de forma simples no campo da visão computacional, permitindo o gerenciamento da entrega e incorporação de serviços cognitivos para dispositivos e usuários. Nesta tese, a hipótese foi deduzida e diferentes aspectos da arquitetura IAaaS foram validados. Foram exploradas as capacidades da arquitetura IAaaS para computação de borda, incluindo dispositivos convencionais e de baixo custo. Os resultados demonstraram que modelos de IA leves, apesar de sua simplicidade, são adequados para implantação em dispositivos de baixo custo, enquanto modelos mais profundos podem ser usados para tarefas de predição nesses dispositivos. Os resultados indicam que a estrutura de controle de inteligência de borda proposta facilita efetivamente a comunicação e gerencia o ciclo de vida de modelos de IA em um ambiente distribuído. Além disso, foi explorado o gerenciamento da plataforma, o treinamento de modelo e as funcionalidades de gerenciamento de conjuntos de dados por meio de experimentos que consideraram o aprendizado federado. A abordagem proposta fornece um ambiente flexível e escalável que suporta vários paradigmas de IA e permite a implantação e o gerenciamento eficientes de modelos em dispositivos heterogêneos, ao mesmo tempo em que equilibra a eficiência computacional com o desempenho do modelo. Por fim, foi demonstrada a funcionalidade do otimizador de modelo da arquitetura IAaaS usando otimização de hiperparâmetros com

estratégias baseadas em Algoritmo Genético, Busca Aleatória e Otimização Bayesiana. As descobertas estão alinhadas com o objetivo da Arquitetura IAaaS, que é fornecer aos usuários e aplicativos soluções e recursos inteligentes facilmente acessíveis. Ao incluir a otimização de hiperparâmetros, os usuários podem aproveitar as vantagens das instalações eficientes da IAaaS e criar classificadores de alto desempenho sem exigir configurações manuais extensas ou conhecimento especializado. Além disso, a abordagem facilita soluções de IA personalizadas e escaláveis, promovendo a inovação e acelerando a implantação de aplicações inteligentes em diversos contextos, tornando-a adequada para cenários do mundo real.

Palavras-chave: Inteligência Artificial como Serviço. IAaaS. Visão Computacional. Aprendizado Profundo. Computação de Borda. Otimização.

Abstract

In recent years, Artificial Intelligence (AI) has experienced exponential growth across various domains of daily life, including transportation, healthcare, and security. However, the current implementation of conceiving and implementing intelligent services makes it challenging to increase the personalized, organized, and large-scale use of AI, particularly to deal with complex tasks to create intelligent resources. In this context, this thesis proposes and evaluates an AI as a Service (AIaaS) architecture that aims to address the lack of current methods for deploying and delivering intelligent services for heterogeneous devices and multiple users. The main goal of this project is to research, develop, and validate an Artificial Intelligence as a Service (AIaaS) architecture that offers AI-based solutions and resources on demand for users and applications. To this end, we proposed a hypothesis regarding the feasibility and suitability of efficiently delivering AI resources in the field of computer vision, allowing the handling of cognitive service delivery and embodiment for devices and users. In this thesis, we deduced our hypothesis and validated different aspects of AIaaS Architecture. We explored the capabilities of our AIaaS for edge computing, including low-cost and conventional devices. Our results demonstrate that despite their simplicity, lightweight AI models are well suited for deployment on low-cost devices, whereas deeper models can be used for prediction tasks on these devices. The results indicate that the proposed edge-intelligence control framework effectively facilitates communication and manages the lifecycle of the AI models in a distributed environment. In addition, we exploited platform management, model training, and dataset management functionalities through experiments that considered federated learning. Our proposed approach provides a flexible and scalable environment that supports various AI paradigms and enables efficient deployment and management of models across heterogeneous devices while balancing computational efficiency with model performance. Finally, we demonstrated the model optimizer functionality of the AIaaS architecture using hyperparameter optimization with strategies based on a Genetic Algorithm, Random Search, and Bayesian Optimization. Our findings are in line with the aim of AIaaS Architecture, which is to provide users and applications with easily accessible intelligent solutions and

resources. By incorporating hyperparameter optimization, users can take advantage of efficient AIaaS facilities and create high-performance classifiers without requiring extensive manual configurations or specialized knowledge. Furthermore, our approach facilitates personalized and scalable AI solutions, fostering innovation and expediting the deployment of intelligent applications across diverse contexts, making it suitable for real-world scenarios.

Keywords: Artificial Intelligence as a Service. AIaaS. Computer Vision. Deep Learning. Edge Computing. Optimizing.

List of Figures

List of Tables

List of Algorithms

Acronyms list

3GPP 3rd Generation Partnership Project

AI Artificial Intelligence

AIaaS Artificial Intelligence as a Service

ALL Acute Lymphoblastic Leukemia

API Application Programming Interface

AutoML Automated Machine Learning

CNN Convolutional Neural Network

COVID-19 Corona Virus Disease-2019

CPU Central Processing Unit

CT Computed Tomography

DRL Deep Reinforcement Learning

EC Edge Computing

ETSI European Telecommunications Standards Institute

FL Federated Learning

FedAvg Federated Averaging

FedOpt Federated Optimization

FedProx Federated Proximity

GA Genetic Algorithm

GPU Graphics Processing Unit

GUI Graphical User Interface

IaaS Infrastructure as a Service

IAI Interpretable Artificial Intelligence

ILSVRC ImageNet Large Scale Visual Recognition Challenge

IID Independent and Identically Distributed

KNN K-Nearest Neighbors

LSTM Long Short-Term Memory

ML Machine Learning

MLP Multi Layer Perceptron

MLaaS Machine Learning as a Service

MEC Mobile Access Edge Computing

MLOps Machine learning operations

NWDAF Network Data Analytics Function

non-IID non-Identically Distributed

PaaS Platform as a Service

PAIaaS Pervasive Artificial Intelligence as a Service

PoC Proof of Concept

RT-PCR Reverse transcription-polymerase chain reaction

SGD Stochastic Gradient Descent

SaaS Service as a Service

SVM Support Vector Machine

TPU Tensor Processing Units

WS World Size

XAI Explainable AI

Contents

I hereby certify that I have obtained all legal permissions from the owner(s) of each third-party copyrighted matter included in my thesis, and that their permissions allow availability such as being deposited in public digital libraries.

Larissa Ferreira Rodrigues Moreira

Chapter 1

Introduction

AI has been considered the most prevailing technology over the past few decades. According to a report by the International Data Corporation (IDC), global spending on AI is projected to reach a cumulative global economic impact of $19.9 trillion by 2030 (FIORETTI et al., 2024). Gartner's reports have highlighted AI-driven development as a top trend in strategic technology and identified innovations that offer significant importance, including cloud AI services, computer vision, edge AI, and model operationalization (COSTELLO; RIMOL, 2019; JAFFRI; SICULAR, 2023).

The rapid expansion of computational services that support increasingly pervasive and sophisticated connectivity for smart application delivery, particularly using AI, is transforming daily tasks across various domains, including healthcare, elder care, entertainment, education, precision agriculture, and security (ZHANG et al., 2021; ISMAIL; BUYYA, 2022; MA et al., 2023). Currently, applications are contextually driven and smart and are built for individual users, providing a unique and objective experience in task execution. However, such applications generate vast amounts of data and the underlying infrastructure requires methods to harmonize hardware from different functions and vendors (SONG et al., 2022; NAHAVANDI et al., 2022). Technologies and methods that enable smart application deployment are continuously evolving and guiding various research avenues in the AI field.

Embedded systems have emerged as an important component in the development of intelligent and ubiquitous applications, particularly in the field of deep learning-based image classification. These systems, which are characterized by their cost-effectiveness and low energy consumption, offer an ideal platform for implementing advanced algorithms in practical settings (SINGH; GILL, 2023). The embedded AI market is expected to grow significantly, with forecasts predicting it will reach $22.4 billion by 2030, driven by an annual growth rate of 13.8% (TRIVEDI et al., 2024). Despite their promise, embedded systems face notable challenges including limited computational resources, energy constraints, and the need for real-time processing capabilities (SINGH; GILL, 2023; SHARMA; TOMAR; HAZRA, 2024).

In this sense, AI service deployment must be tailored and customized for each context (NAHAVANDI et al., 2022; AHMED; JEON; PICCIALLI, 2022). For example, in the case of deploying Machine Learning (ML) algorithms for disease identification based on images, training, validation, testing, and deployment are essential steps that are reimplemented for each developed application. These tasks require considerable hardware resources and occur repeatedly, even in different applications. Thus, there is an opportunity for techniques and methods that systematically organize and provide AI artifact life cycle management and service delivery.

An artifact or resource of AI is a set of intelligent components that can be conceptualized as trained machine-learning models. In addition, there is a service for refining and optimizing machine-learning models, datasets for model training, and source code snippets for deploying models in the development of intelligent applications (AHMED; JEON; PICCIALLI, 2022). Similar to virtualization, there were well-known techniques such as Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Service as a Service (SaaS), which advanced the state-of-the-art by enabling large quantities of computational resources to be shared among different tenants, thereby optimizing the resource utilization (MANVI; KRISHNA SHYAM, 2014).

This thesis innovates by applying and extending the concept of AIaaS, which entails a service delivery model for AI, where users can access ML models and deploy them in different ecosystems. Here, we propose an innovative layer for delivering AIaaS that fills the gaps in cognitive service delivery, intelligence embedment on low-cost devices, and artifact life cycle management.

1.1 Motivation

The use of AI methods combined with mobile and embedded devices has driven advancements in various technologies, enabling the development of intelligent devices such as smartphones, smartwatches, and sensors integrated into different appliances. In this context, it is essential to explore the use of deep neural networks on mobile and embedded devices to support inference needs in comprising sensors, which will pave the way for the next generation of intelligent devices required for future Internet of Everything-based applications (HUSSAIN, 2017). Outsourcing and personalizing AI resources empower application providers to contract AI services and enhance the robustness and efficiency of their applications, particularly during software updates, by selectively applying AI services.

Furthermore, delivering AI resources as a service provides a method of organizing these artifacts, particularly by facilitating the improvement and adjustment of AI artifacts within a single platform. For example, ML models require updates over time, and with an organized platform, it becomes possible to update the model without requir-

ing modifications to the application that provides intelligent service. It also consolidates large-scale ecosystems in various contexts and applications.

To support this thesis, we provide an overview of the current state of deep-learning-based image classification on embedded devices and analyze research published in the literature between 2013 and 2023. The search was conducted across four databases (ACM Digital Library, IEEE Xplore, PubMed, and Scopus), yielding 1,038 candidate papers, of which 111 were selected as relevant. All the studies analyzed are listed in Appendix A.

Although deep learning has been applied to image classification since 2012, we find that research aimed at making it suitable for embedded devices began only in 2015, as shown in Figure 1. However, this research topic was affected by the Corona Virus Disease-2019 (COVID-19) pandemic, which shifted the global focus towards combating the virus and developing diagnostic and treatment solutions instead of embedded device applications (MORADIAN et al., 2020; BHATTACHARYA et al., 2021). After the pandemic, new research on image classification based on deep learning for embedded devices has stabilized.

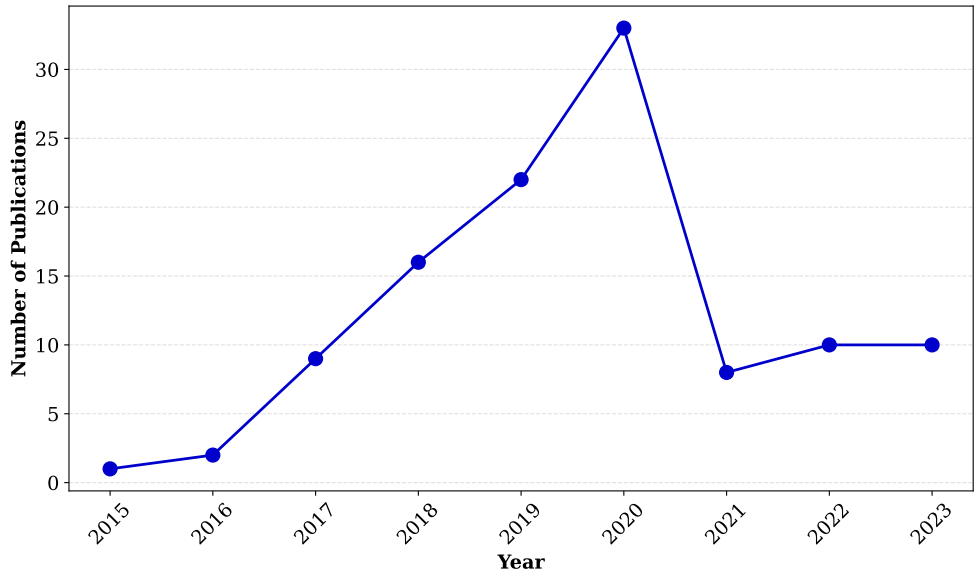


Figure 1 – Number of publications per year for deep learning image classification using embedded devices.

In addition, we analyzed the low-cost devices used in deep learning applications for embedded systems, as depicted in Figure 2. We observed that the Raspberry Pi and ARM Cortex processors remain dominant in low-cost deep learning applications for embedded systems. This popularity can be attributed to the affordability, ease of use, and robust community support of the Raspberry Pi and ARM Cortex versatility in handling computational tasks relevant to embedded applications. In contrast, smartphones and tablets, which are less frequently used, signify a growing trend towards leveraging ubiquitous consumer devices for edge-computing tasks. Their inclusion highlights the potential for

leveraging the existing hardware infrastructure to deploy deep learning solutions without requiring specialized embedded systems.

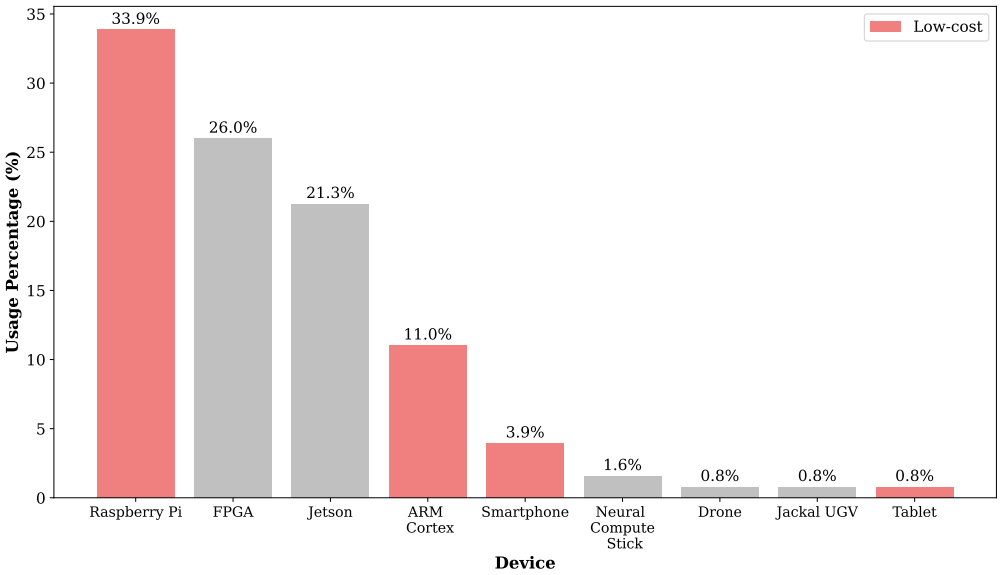


Figure 2 – Devices usage for deep learning image classification. Low-cost devices are indicated in red.

In this context, there is a clear gap between the current way of conceiving and deploying intelligent services and the significant potential of delivering AI resources to applications in a personalized, organized, and large-scale manner (PARASKEVOULAKOU; KYRIAZIS, 2023). Essentially, bridging this gap constitutes the main contribution of this thesis.

Our AI approach can be applied in various fields, such as creating AI models for detecting bone fractures in medical imaging analysis, recognizing activities to support smart surveillance, and managing pests and diseases in agriculture. The main advantage of our AIaaS lies in its ability to support different machine learning paradigms in architecture. This allows for both distributed and centralized learning, enabling clients or model subscribers to pull AI models into their application domains.

1.2 Research Hypothesis

This thesis is based on the following hypothesis:

Hypothesis: *AI as a Service Architecture improves life cycle management in Computer Vision field, such as model training, validation, testing, and model fine-tuning management by providing to users and applications baseline infrastructure and methods to handle cognitive service delivery.*

1.3 Goals

This thesis aims to provide an **Architecture for delivering AI as a Service**, dedicated to offering on-demand solutions for users and applications. An AI service can take various forms, including: 1) managing ML models; 2) providing fine-tuning services for trained models; and 3) managing datasets. Thus, the general objective of this thesis is to research, develop, and validate such an architecture to deliver a reliable AI service. The specific goals are as follows:

- ❏ Evaluate the quality of the proposed approach through an extensive set of experiments on different image datasets, and comparing the results.
- ❏ Propose a new Architecture to deliver AI as a service.
- ❏ Map the challenges of applying AIaaS architecture in the context of edge prediction.
- ❏ Investigating a new way of adjusting hyperparameters in CNN in the context of cloud-computing.
- ❏ Explore different data augmentation techniques in order to generate synthetic images to overcome the limitation of small and unbalanced data sets that can lead to overfitting.

1.4 Contributions

The main contributions of this thesis are:

- ❏ A new AIaaS Architecture to deliver AI as a Service and is dedicated to providing on-demand solutions for users and applications.
- ❏ Evaluation of the computational cost of embedding deep learning models in different devices.
- ❏ An analysis to assess the relationship between the computational cost of a CNN and the number of its parameters.
- ❏ AI service design different forms, being: ML model management; trained model tuning service; and dataset management.
- ❏ A proof of concept to showcase the effectiveness of our proposed architecture in delivering AI models in a novel way.

1.5 Thesis Organization

This thesis is structured into seven chapters to explain the theoretical concepts, techniques, and results. This chapter provides an overview of the motivation, goals, main contributions, and structural organization of this thesis. Chapter 2 describes the theory related to this research. Chapter 3 delves into related works, providing an essential context, highlighting trends, and situating the present study within a wider research landscape. Chapter 4 presents our proposed solution, Artificial Intelligence as a Service Architecture, detailing its design and components. This provided an empirical investigation of the research questions in various case studies.

Chapter 5 explores the performance and capabilities of AIaaS architecture in edge. This chapter investigates the deployment of AI models on heterogeneous devices and evaluates key metrics such as classification performance, resource consumption, and model size. Additionally, we validated the potential of FL in enhancing the AIaaS architecture's ability to manage cognitive service delivery in decentralized environments while also ensuring efficient model execution and lifecycle management. Chapter 6 evaluated how AIaaS to dynamically adapt and optimize models for specific applications and user needs. The evaluation emphasizes the flexibility and robustness of the architecture in supporting scalable AI services while also balancing computational efficiency with high-performance model deployment across different computer vision datasets.

Finally, Chapter 7 presents the conclusions and main contributions to the state-of-the-art research. In addition, we present directions for the next steps, future developments, and scientific contributions of this thesis. Figure 3 illustrates how the deduction of the hypothesis is organized throughout the structure of this thesis.

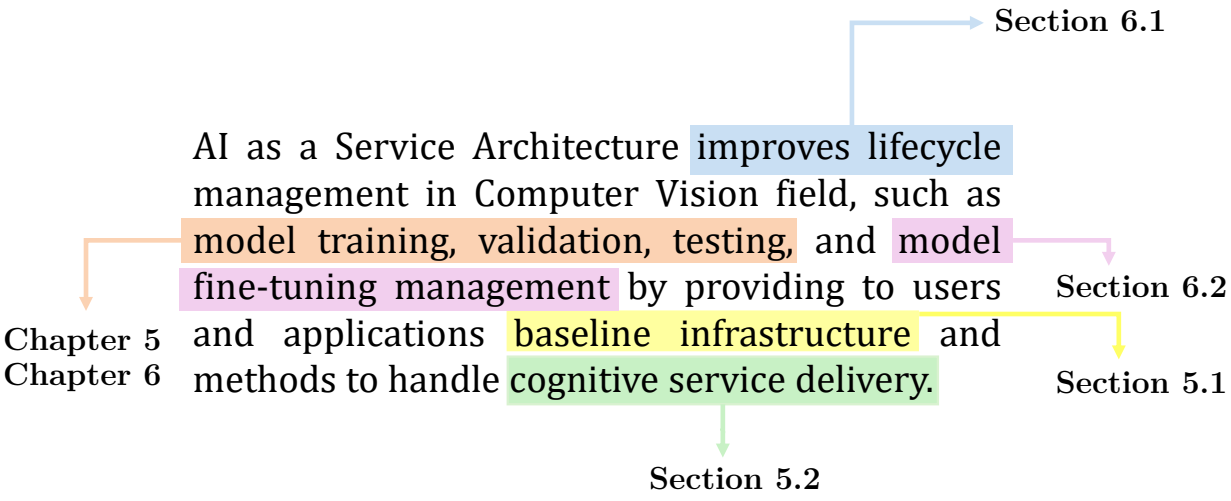


Figure 3 – Hypothesis deduction through the structure of this thesis.

Chapter 2

Background Concepts

This chapter aims to provide a background by introducing key concepts that are essential to grasp the main challenges of our problem, such as Artificial Intelligence, Computer Vision, Convolutional Neural Networks, Federated Learning, Edge Computing, and Computer Infrastructure.

2.1 Artificial Intelligence

Artificial Intelligence (AI) is a field that aims to create systems capable of performing tasks that normally require human intelligence, such as speech recognition, decision-making, language translation, among others. These systems are trained using large amounts of data to learn patterns and perform tasks autonomously (RUSSELL; NORVIG, 2009).

AI encompasses a wide range of algorithms that demonstrate intelligent behavior, but not all of them are capable of evolving on their own and solving their own problems. In this context, we highlight Machine Learning (ML) as a subset of AI that learns from experience or datasets rather than just instructions. ML methods used mathematical and statistical procedures to enable computational systems to learn from data. They operate on the principle of generalization, where a model is trained with input data to make predictions based on new data. These algorithms are designed to identify patterns, correlations, and meaningful information in the training data, and gradually adapt to improve their predictive accuracy (FACELI et al., 2011).

Among the best-known ML algorithms are neural networks, which are effective in modeling complex nonlinear relationships between variables. Neural Networks consist of a massively distributed parallel processor made up of simple processing units, also known as neurons or nodes, which have the capacity to store and use knowledge. They acquired knowledge from examples through a learning process. This learning process occurs by adjusting the network parameters (synaptic weights) in a process known as training. Thus, the type of learning is determined according to the changes made to

parameters (RUSSELL; NORVIG, 2009). The mathematical model of the artificial neuron is given by Equation 1.

$$y_k = f\left[\left(\sum_{i=1}^{n} x_i \times w_{ki}\right) + b_k\right] \tag{1}$$

Where y_k represents the output of the neuron k. The input vector given by x_i and synaptic weights w_i interact multiplicatively, generating the output of the linear combiner, which is added to the *bias* term represented by b. The term bias allows one to increase or decrease the input of the activation function f, which is responsible for restricting the amplitude of the output signal by conditioning the activation of the signal to exceed a certain threshold by the value of the weighted sum of the inputs (RUSSELL; NORVIG, 2009; FACELI et al., 2011). Figure 4 illustrates the mathematical model of an artificial neuron known as the perceptron model.

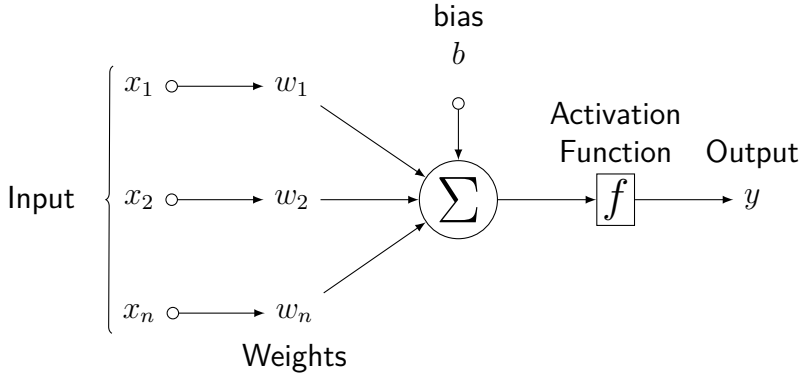


Figure 4 – Perceptron Model.

Training a neural network is iterative, and its quality is related to the choice of network architecture and number of epochs required for training to overcome the two main challenges in machine learning: underfitting and overfitting. Underfitting occurs when the model cannot obtain a sufficiently low error value in the training set. Overfitting occurs when the interval between the training error and the test error is too large, resulting in limited generalization (RUSSELL; NORVIG, 2009) .

Neural networks have the ability to identify patterns, improve through examples, and generalize learned information; they can also be advanced to more intricate models using Deep Learning, a subset of AI and ML that consists of an input layer, an output layer, and many intermediate hidden layers. The name "deep" is explained by the large number of internal hidden layers (GOODFELLOW; BENGIO; COURVILLE, 2016; LECUN; BENGIO; HINTON, 2015; PONTI et al., 2017).

Figure 5 illustrates the position of ML and deep learning within the AI field. As shown in the illustration, deep learning is a subset of ML, which is also a subset of AI.

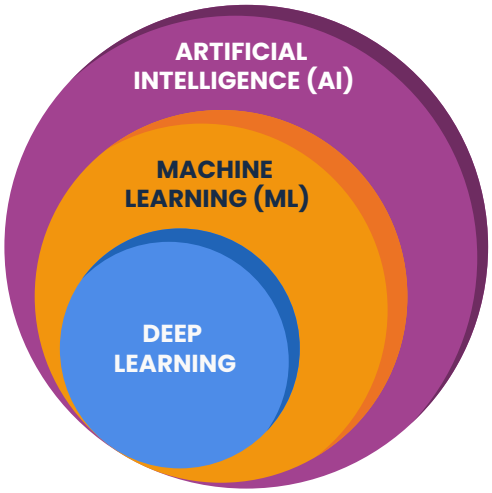


Figure 5 – ML and Deep Learning within the domain of AI inspired by Sarker (2022).

Furthermore, ML and Deep Learning are considered essential AI technologies, being frontiers for the development of intelligent systems and automation of processes enabling AI to reach a new level for use in the AIaaS architecture.

2.2 Computer Vision

Computer vision is a key research area in AI, which enables computers to interpret and understand visual information from images and videos (RUSSELL; NORVIG, 2009). This involves the ability to recognize, identify, and classify images as well as extract useful information from them (RUSSELL; NORVIG, 2009; BACKES; SÁ JUNIOR, 2016; SZELISKI, 2022). Figure 6 illustrates the areas that influence computer vision.

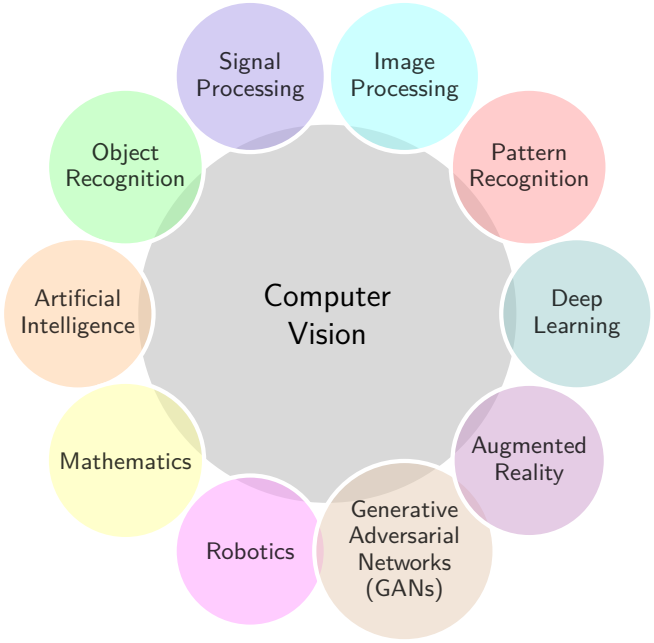


Figure 6 – Various fields of computer vision.

The field of computer vision has experienced remarkable progress in recent decades due to technological advancements such as the development of GPU and Tensor Processing Units (TPU) and the creation of deep learning algorithms (PONTI et al., 2017; SZELISKI, 2022). These innovations have led to the creation of highly precise and efficient computer vision systems. Additionally, computer vision is essential in emerging technologies such as medical computer aided diagnosis based on image (SOUSA et al., 2020; RODRIGUES; NALDI; MARI, 2020; SANTOS et al., 2021; BACKES, 2022; ROCHA; LANDINI; FLORINDO, 2023), precision agriculture (MOREIRA et al., 2022; DA SILVA et al., 2023), management network (MOREIRA et al., 2022; MOREIRA; RODRIGUES MOREIRA; OLIVEIRA SILVA, 2023), and industry (SOUSA REIS et al., 2023; MUMBELLI et al., 2023; BERGMANN et al., 2024).

2.3 Convolutional Neural Networks

Convolutional Neural Networks (CNNs) are a type of artificial neural network designed to handle large amounts of data. Unlike other artificial neural networks, CNN uses a multistage image classification technique that incorporates spatial context and weight sharing among pixels to extract high-level hierarchical representations of data (GOODFELLOW; BENGIO; COURVILLE, 2016; PONTI et al., 2017).

The initial concepts that laid the groundwork for CNN development emerged in the late 1970s (FUKUSHIMA, 1980) and were applied to medical image analysis in 1995 to aid the detection of lung nodules in X-ray images (LO et al., 1995). Subsequently, in 1998, the first successful real-world application of a CNN was achieved using the LeNet architecture, which was employed to recognize handwritten digits (LECUN et al., 1998).

Despite the early promise of CNNs, their utilization was limited until advances in computational resources, primarily driven by massive parallel and distributed computing, became available. CNNs gained popularity from 2012 onward, when the AlexNet architecture triumphed in the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) competition, significantly surpassed previous feature extraction methods (KRIZHEVSKY; SUTSKEVER; HINTON, 2012). In subsequent years, advancements involving related and deeper architectures have taken place (RUSSAKOVSKY et al., 2015), solidifying CNNs as state-of-the-art in the field of computer vision. In addition, CNNs have a greater ability to generalize and lead to better performance than both traditional algorithms and human capabilities (GOODFELLOW; BENGIO; COURVILLE, 2016; ZHOU et al., 2021).

Convolution is the primary operation performed by CNNs applied to input images to derive the feature maps for classification (GOODFELLOW; BENGIO; COURVILLE, 2016). Convolutions involve a mathematical operation conducted pixel-by-pixel, where a filter slides across the entire image, computing the sum of element-wise products at each position. This process yields values for a new image, denoted by $g(x, y)$, obtained from an

input image $f(x, y)$ and a filter $w(x, y)$ with dimensions $m \times n$, as defined by Equation 2.

$$g(x, y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s, t) f(x - s, y - t), \tag{2}$$

where $a = \frac{(m-1)}{2}$ and $b = \frac{(n-1)}{2}$.

Figure 7 illustrates a convolution operation. The size and values of the filters depend on the objectives of the feature extraction.

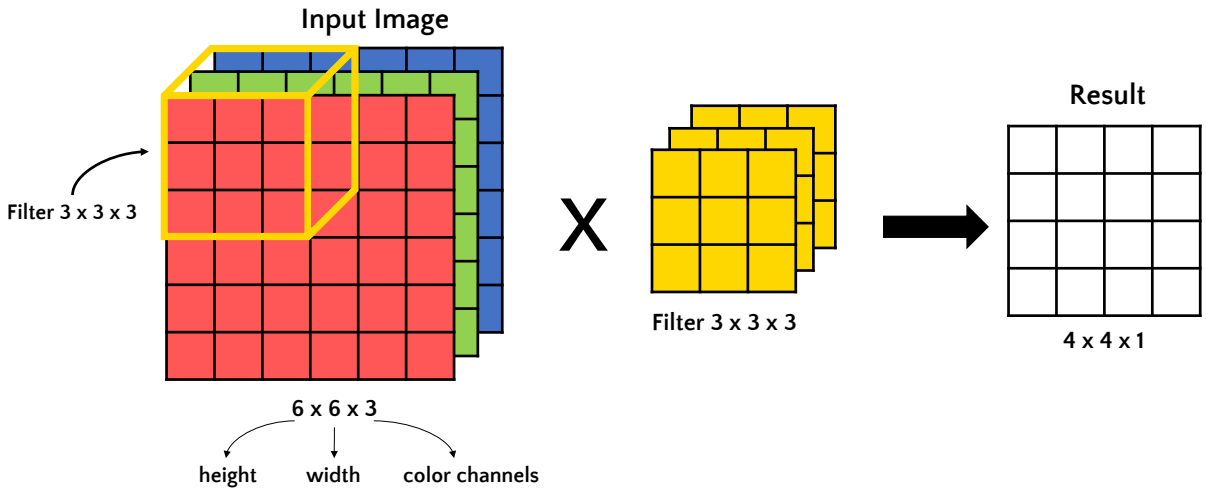


Figure 7 – Representation of a convolution operation.

A typical CNN consists of three fundamental layers: a convolutional layer, pooling layer, and a fully connected layer (PONTI et al., 2017):

- **Convolutional Layer:** performs convolution operations and generates feature maps. This layer applies filters that slide across the input data, computing element-wise multiplications and summations to produce feature maps. By employing weight sharing and spatial hierarchy, these layers effectively extract relevant features, enabling the network to recognize patterns of increasing complexity, which are essential for tasks such as image classification and object detection. The ability of the convolutional layer to extract meaningful information directly from raw data significantly contributes to the success of CNNs in computer vision tasks.

- **Pooling Layer:** is responsible for reducing the size of the feature maps. Pooling operations, such as max pooling or average pooling, reduce the spatial dimensions of feature maps while preserving the most important information. These operations summarize local regions and capture the most salient features, improving the network's ability to recognize patterns and reducing the computational load. By eliminating redundant information, pooling layers help manage overfitting and

maintain the hierarchical structure of the extracted features, leading to improved generalization and pattern recognition capabilities.

- ❑ **Fully Connected Layer:** serves as the final stage in a CNN, connecting all neurons from the previous layer. Unlike the convolutional and pooling layers, which focus on spatial patterns, the Fully Connected layer captures complex relationships and high-level features by interconnecting with all neurons from the previous layer. The layer combines the extracted features and makes decisions based on these aggregated representations, playing a crucial role in tasks such as classification. It ultimately produces the final predictions or classifications, completing the network's computational process.

Depending on the specific architecture of a CNN, some of these layers may be omitted or alternative modules may be introduced to replace their functionalities. Some architectures may use skip connections to maintain spatial information, or utilize different forms of downsampling instead of traditional pooling layers (KHAN et al., 2020; ZHAO et al., 2024). Additionally, certain models may incorporate attention mechanisms and normalization techniques, or employ more complex skip connections between non-adjacent layers. The flexibility of CNN architectures lies in their ability to adapt to various tasks, data types, and computational resources, which allows customization of the network to meet the specific requirements of a project.

CNN architectures demonstrated diverse structural features such as residual connections and dense blocks. In our literature review, we observed a correlation between the depth of CNN and its generalization performance, suggesting that very deep CNNs with skip connections are effective for generalization. However, shallow CNNs are more suitable for embedded platforms. In this thesis, we aim to explore the feasibility of embedding different CNNs models (shallow and deep).

2.3.1 Shallow CNNs

Shallow CNNs, also known as lightweight architectures, prioritize the efficiency while maintaining a moderate level of accuracy. These networks have fewer layers than their deep counterparts, and are computationally less demanding. They are particularly well suited for resource-constrained devices, such as mobile phones and embedded systems, where memory and processing power are limited (ZHOU et al., 2021; CHEN et al., 2024).

Despite their reduced depth, they can still capture essential features from input data and perform adequately in tasks, including image recognition and classification. These architectures often employ techniques such as channel pruning, parameter reduction, and efficient building blocks to balance computational efficiency and predictive performance (CHEN et al., 2024). Shallow CNNs address the challenges of deploying machine learning

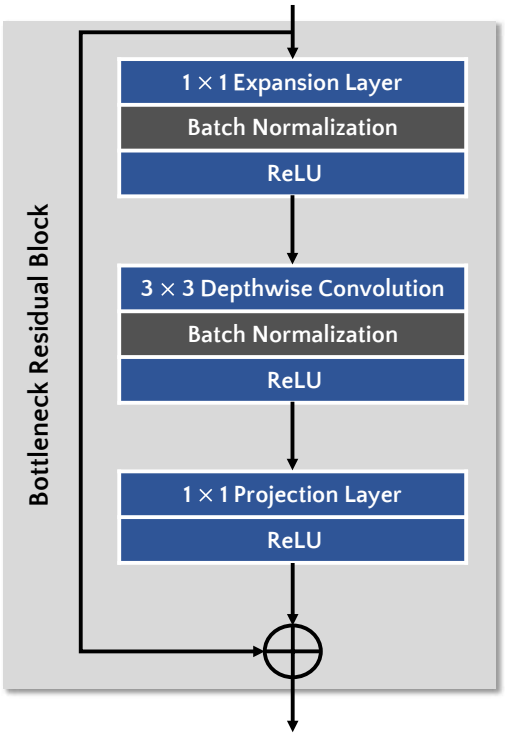
models on devices with constrained resources, making them a valuable choice for real-world applications with practical limitations. The shallow CNNs evaluated in this study are described as follows:

- **MobileNet:** was developed for deploying deep learning applications in mobile or portable devices (HOWARD et al., 2017). The first version of MobileNet, known as MobileNet-V1, is based on depthwise and pointwise convolutions. Depthwise is a depth-level operation that uses different convolution kernels for each input channel. The pointwise convolution is a traditional convolution and considers a kernel of 1×1. Based on V1, MobileNet-V2 (SANDLER et al., 2018) introduces inverted residual and linear bottleneck blocks to overcome the gradient vanishing problem. In addition, compared with MobileNet-V1, MobileNet-V2 has a smaller model with fewer parameters. In this thesis, we considered MobileNet-V2. Figure 8(a) illustrates its basic structure.

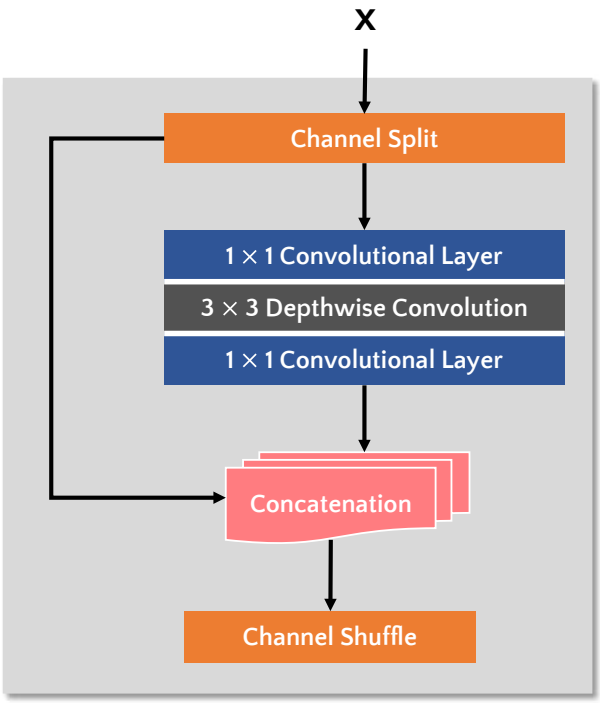
- **ShuffleNet:** was developed for mobile devices with low computing power. The first version of Shufflenet introduces the pointwise group convolution and channel shuffling to reduce computational cost. However, the disadvantage of pointwise group convolution is that it is derived only from a small part of the input channel. To overcome this limitation, ShuffleNet-V2 applies a channel split operator and merges three basic operations into a single operation: concatenation, channel shuffle, and channel split, as illustrated in Figure 8(b) (MA et al., 2018).

- **SqueezeNet:** is a lightweight architecture that uses compression technique (HAN et al., 2016). It introduces a "fire module", which is comprised of a "squeeze" convolutional layer that feeds into an "expand" layer that is a mix of 1×1 and 3×3 convolutions. The squeeze layer helps to limit the number of input channels and helps to limit the number of input channels. The fire module is illustrated in Figure 8(c), and the s_1 indicates the number of 1×1 filters in the expand layer, and e_3 indicates the number of 3×3 filters in the expand layer (IANDOLA et al., 2016).

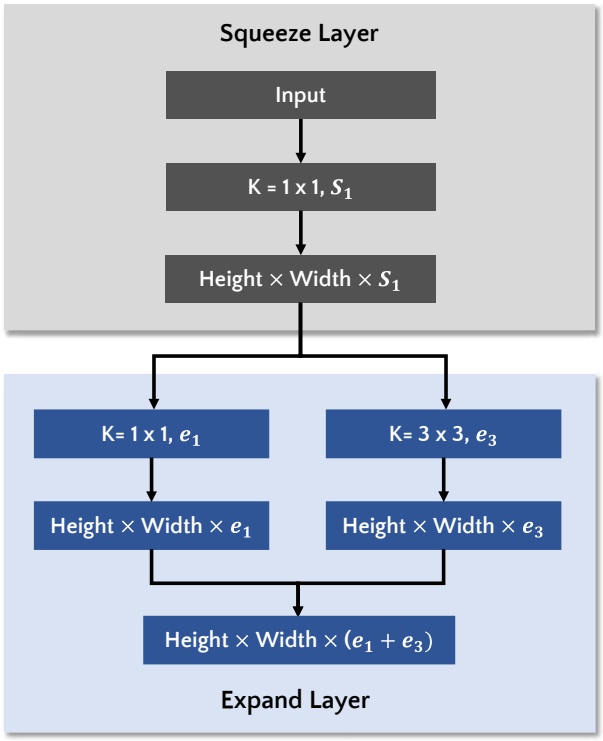
- **EfficientNet:** is an architecture that employs a compound scaling method to uniformly scale all dimensions of depth, width, and resolution using a set of fixed coefficients. The variations in EfficientNet are designated as EfficientNet-B0 through EfficientNet-B7. Each version progressively increases in complexity and capacity with more layers and parameters. The base model EfficientNet-B0 integrates inverted bottleneck residual blocks and squeeze-and-excitation blocks, resulting in fewer parameters (TAN; LE, 2019). In this thesis, we considered an EfficientNet-B4.



(a) MobileNet-V2.



(b) Basic unit in ShuffleNet.



(c) Fire module in SqueezeNet.

Figure 8 – Structure of each shallow CNN evaluated.

2.3.2 Deep CNNs

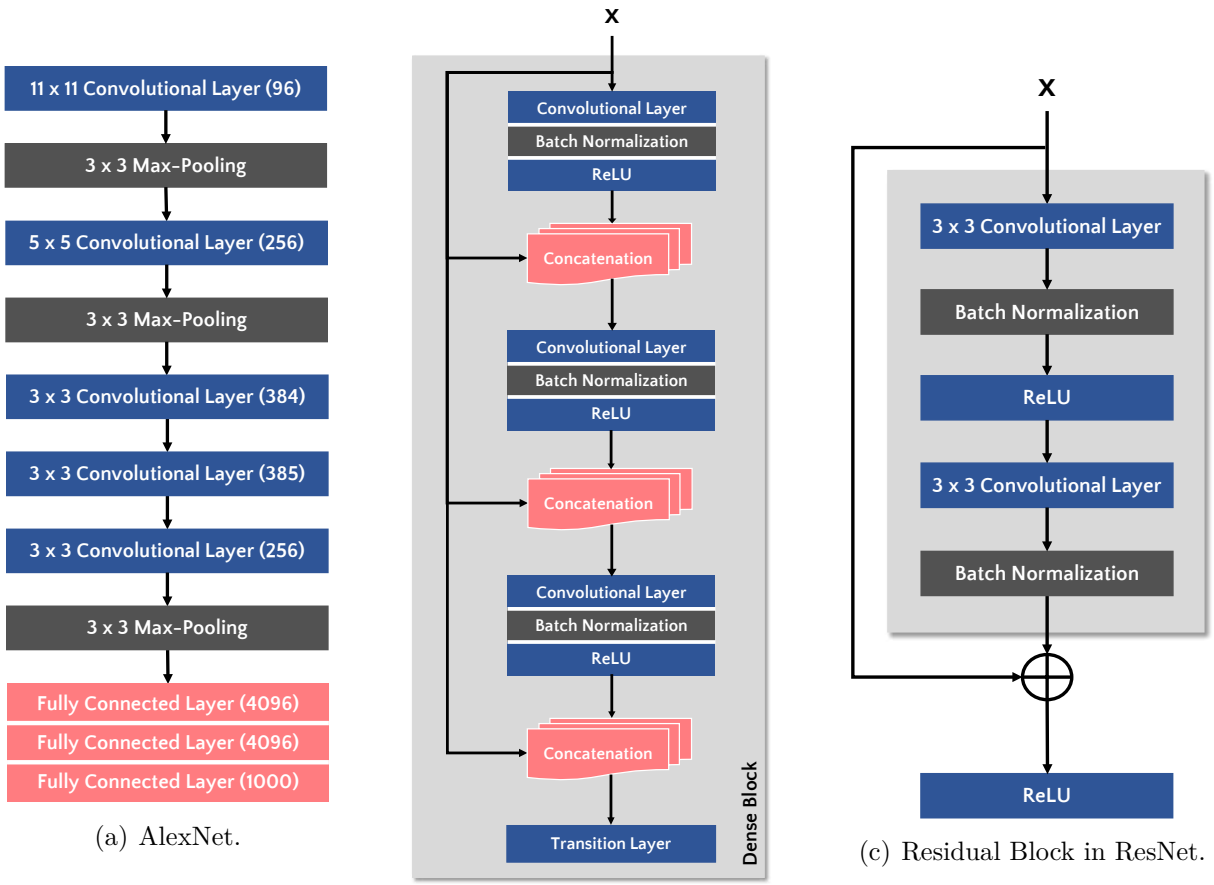
Deep CNN architectures are known for their many layers that gradually learn increasingly complex features from input data (LECUN; BENGIO; HINTON, 2015; GOODFELLOW; BENGIO; COURVILLE, 2016). These networks can capture intricate patterns in images, audio, and other types of data. Deep architectures incorporate multiple convolutional and pooling layers, enabling them to model complex relationships within the data (PONTI et al., 2017).

However, deeper networks can encounter challenges such as vanishing gradients and overfitting, leading to the development of techniques such as skip connections, batch normalization, and residual networks to address these issues (KHAN et al., 2020). These architectures can extract abstract and nuanced information from data, making them a cornerstone of modern machine learning applications (LECUN; BENGIO; HINTON, 2015). The deep CNNs evaluated in this study are described as follows:

- **AlexNet:** represents the first advance in CNNs architectures, winning the ImageNet Large Scale Visual Recognition Challenge 2012. It contains five convolutional layers, three fully connected layers, max-pooling after each convolution, and approximately 60 million parameters. Additionally, it introduces a dropout strategy to reduce overfitting (KRIZHEVSKY; SUTSKEVER; HINTON, 2012). The structure of AlexNet is illustrated in Figure 9(a).

- **DenseNet:** uses dense block structures to connect convolutional layers. Traditional CNNs such as AlexNet contain L layers and L connections that connect each layer and its subsequent layer. On the other hand, DenseNet has $L\ (L+1)/2$ connections, and the features maps are used as input in all subsequent layers. Thus, DenseNet reduces the number of required parameters, strengthens feature propagation, encourages feature reuse, and overcomes the vanishing gradient problem (HUANG; LIU; WEINBERGER, 2016). The dense block structure is illustrated in Figure 9(b). In this study, we considered a DenseNet with 121 layers.

- **ResNet:** won the ILSVRC 2015 challenge and it is a Residual Network proposed to avoid vanishing gradients (HE et al., 2016b). Previous CNNs such as AlexNet are composed of several stacked layers. In ResNet, the residual is extracted instead of the extracted features. Thus, the residual is defined as the deduction of the highlights gained from the contribution of that layer. Figure 9(c) shows the structure of a residual block. We used ResNet-18, which contains 18 convolutional layers with a 3×3 filter.



(a) AlexNet.

(b) DenseNet.

(c) Residual Block in ResNet.

Figure 9 – Structure of each deep CNN evaluated.

2.4 Federated Learning

Federated Learning (FL) is a distributed ML approach that enables model training across multiple decentralized devices or servers that hold local data samples without requiring direct access to their local data (KONEČNÝ et al., 2016; MCMAHAN et al., 2017). In FL each client (for example a smartphone or embedded device) trains a model locally using its private data and then shares only the learned model model weights with the central server. The server aggregates these updates from multiple clients to form a global model, which is then redistributed to the clients for further local training (KONEČNÝ et al., 2016; MCMAHAN et al., 2017; LIU et al., 2024).

This technique is suitable for applications where data privacy and security are critical, as it mitigates the need to share sensitive information, thereby adhering to regulations and reducing the risk of data breach (KONEČNÝ et al., 2016; MCMAHAN et al., 2017). For example, in the medical or agricultural fields, FL allows institutions to collaboratively train robust models using sensitive image data without sharing data. This approach preserves data privacy and ensures compliance with data-protection regulations (ANTICO; RODRIGUES MOREIRA; MOREIRA, 2022; GUAN et al., 2024).

In FL, each client computes a local loss function to evaluate the model's performance on its dataset, as defined in Equation 3. The global model w_t is sent to the edge devices, which then trains the model locally using their data. The central server aggregates the updates to refine the global model. Specifically, each device k optimizes the local model based on its local data D_k by minimizing the local loss function F_k (Equation 3).

$$F_k(w) = \frac{1}{|D_k|} \sum_{i \in D_k} \ell(w; x_i, y_i) \tag{3}$$

Where F_k represents the loss function for client k and w is the model parameter being optimized. Function F_k calculates the average loss over the client's local dataset D_k and each $\ell(w, x_i, y_i)$ measures the discrepancy between the model's prediction for input x_i and the actual output y_i. Finally, factor $\frac{1}{|D_k|}$ normalizes the sum by the number of data points in the dataset of the client.

To illustrate the concept of FL, Figure 10 summarizes the basic structure, highlighting how multiple servers collaboratively train a global model while maintaining decentralized data.

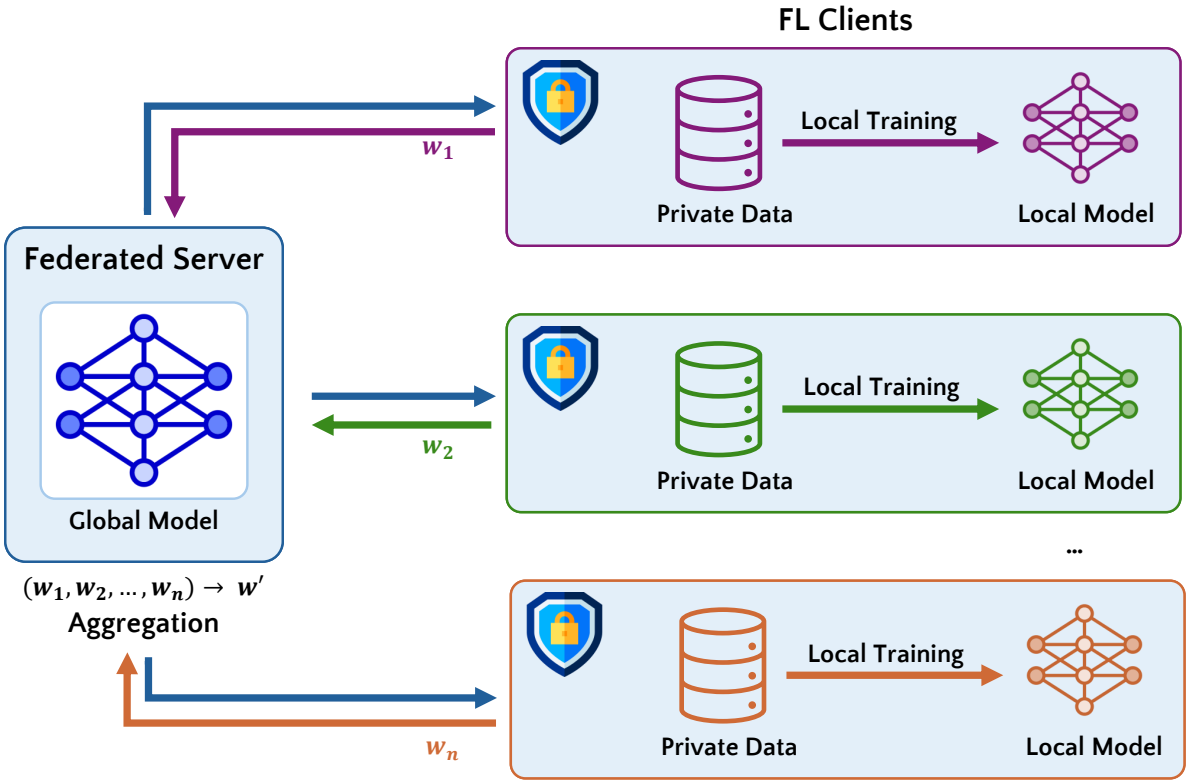


Figure 10 – FL basic structure.

The distribution of data across clients in FL is a critical aspect that affects the training process and model performance. This data can be classified as Independent and Identically Distributed (IID) or non-Identically Distributed (non-IID) (KONEČNÝ et al., 2016; LIU et al., 2024).

- ❑ **IID:** each client's dataset is drawn from the same distribution and is statistically similar, which simplifies the aggregation of updates and generally leads to better convergence of the global model.

- ❑ **non-IID:** is more common in real-world scenarios, means that data varies significantly among clients. This may be due to different data collection methods, user behavior, or data sparsity, leading to challenges such as biased model updates and slower convergence.

The main operations of FL include local model updates, global model aggregation, and client selection (KONEČNÝ et al., 2016; MCMAHAN et al., 2017; LIU et al., 2024). Each of these operations is explained in detail as follows:

❑ Local Model Update

Each client i trains its local model using its private dataset D_i over multiple epochs. The local update to the model's weights $\mathbf{w}_t^i$ after E local epochs can be represented as defined by Equation 4.

$$\Delta \mathbf{w}_t^i = \mathbf{w}_t^i - \mathbf{w}_t \tag{4}$$

Where $\mathbf{w}_t^i$ is the locally updated model weights after training on client i and $\mathbf{w}_t$ is the global model weights at the beginning of the round.

❑ Global Model Aggregation

The central server aggregates the updates from all clients to update the global model. The aggregation can be done using different methods such as Federated Averaging (FedAvg) (MCMAHAN et al., 2017), Federated Optimization (FedOpt) (REDDI et al., 2021), and Federated Proximity (FedProx) (LI et al., 2020).

In this thesis, we applied the FedAvg algorithm as the aggregation method due to simplicity and efficiency in distributed scenarios (LIU et al., 2024). Also, this iterative process enables the global model to improve over time while the data remains securely on the devices, making it a scalable and efficient solution for training ML models in real-world scenarios (YANG et al., 2019; GUAN et al., 2024). FedAvg computes the new global model w_{t+1} as a simple average of the updated defined by Equation 5.

$$\mathbf{w}_{t+1} = \frac{1}{C} \sum_{i \in \mathcal{S}} \mathbf{w}_t^i \tag{5}$$

Where $\mathbf{w}_{t+1}$ is the updated global model, C is the number of clients that participated in the training round, $\mathcal{S}$ is the set of clients selected in the round, and $\mathbf{w}_t^i$ is the model of client i after local training.

❑ Client Selection

Client selection entails identifying the clients that participate in each training round, as defined by Equation 6. Common selection strategies include random selection, in which a subset of clients is chosen randomly, and systematic selection, which may prioritize clients based on data quality or other criteria (LIU et al., 2024). In this thesis, we selected all clients to participate in each training round, ensuring that the global model benefits from updates across the entire dataset.

$$\mathcal{C}_i = \begin{cases} 1 & \text{if client } i \text{ is selected} \\ 0 & \text{if client } i \text{ is not selected} \end{cases} \tag{6}$$

Where $\mathcal{C}_i$ represents the client i. Finally, the global model is update as defined by Equation 7.

$$\mathbf{w}_{t+1} = \mathbf{w}_t + \eta \cdot \frac{1}{\mathcal{C} \cdot \mathcal{N}} \sum_{i=1}^{\mathcal{N}} \mathcal{C}_i \cdot \frac{n_i}{n} \cdot \Delta \mathbf{w}_t^i \tag{7}$$

Table 1 summarizes the main parameters in FL that influence the efficiency of the training process, determining the convergence, accuracy, and resource utilization.

Table 1 – Main parameters in FL.

Parameter	**Description**	**Impact on Training**
World Size ($\mathcal{WS}$)	The total number of clients (devices) participating in the training.	Influence of aggregation of updates and model convergence.
Local Epochs ($\mathcal{E}$)	The number of epochs each client trains locally before sending updates to the server.	Affects the balance between the local learning and model divergence.
Learning Rate (η)	The learning rate used in the model training process.	Determine the speed of the model updates locally and globally.
Client Fraction ($\mathcal{C}$)	The proportion of clients selected to participate in each round of training.	Influence of representativeness of updates and training efficiency.
Training Round ($\mathcal{R}$)	A complete cycle of communication and model updates between the central server and participating clients.	Gradually enhances global model accuracy by integrating client updates over multiple iterations.

FL allows the creation of models that can learn from diverse and dispersed data while ensuring privacy. In addition, this strategy facilitates the development of models that utilize distributed data.

2.5 Edge Computing

Edge Computing (EC) is a distributed computing model aimed at bringing data processing closer to the data-source point. In edge computing, network and computing devices, such as routers, servers, and sensors, are deployed in close proximity to end-users, where data are generated, processed, and stored. This enables data to be processed at higher speeds, thereby reducing latency and enhancing network efficiency (SHI et al., 2016).

EC is also crucial for the Internet of Things (IoT) because it enables connected devices to perform tasks locally, reducing the need for constant communication with a central server. This not only lightens the load on the network infrastructure, but also enables devices to function even when there is intermittent or unreliable connectivity. Essentially, EC empowers devices to become smarter and more independent, contributing to the seamless functioning of our interconnected world. Furthermore, EC can assist in decreasing the volume of data that must be transmitted over a network, conserving bandwidth, and lowering storage costs. This is particularly significant in applications that generate substantial real-time data, such as the Internet of Things (IoT), autonomous vehicles, and video processing (LI; OTA; DONG, 2018; RAY; DASH; DE, 2019; O'GRADY; LANGTON; O'HARE, 2019).

Figure 11 shows an example of EC. The illustration depicts a smart city environment in which various connected devices, such as drones, cameras, and sensors, operate within an urban setting integrated with the AIaaS Architecture proposed in this study. Instead of transmitting all the data generated by these devices to a remote cloud server for processing, EC is applied. Each device was equipped with its own computing capabilities, allowing it to perform local data processing and analysis.

Figure 11 – Edge Computing in a Smart City, enabling real-time urban management and optimization through AIaaS integration.

This approach reduces data transmission delays and enables swift decision-making, such as triggering an immediate alert within the home upon motion detection. This decentralized processing at the edge of the network represents the essence of EC, providing real-time responsiveness and optimizing the resource utilization (SHARMA; TOMAR; HAZRA, 2024).

2.6 Computational Infrastructure

Computational infrastructure is a cloud-based resource that enables the delivery of computing services over the Internet. Instead of hosting their own servers, users can lease computational resources such as servers, storage, networks, and software from cloud service providers who deliver these resources via the Internet. Cloud service providers offer a diverse array of cloud computing services, which can be grouped into three main categories: (i) Infrastructure as a Service (IaaS), (ii) Platform as a Service (PaaS), and (iii) Software as a Service (SaaS) (KAVIS, 2014). The hierarchical pyramid in Figure 12 illustrates how IaaS, PaaS, and SaaS are organized into distinct levels of service provisioning, providing a visual representation of how infrastructure, platform, and software services are organized.

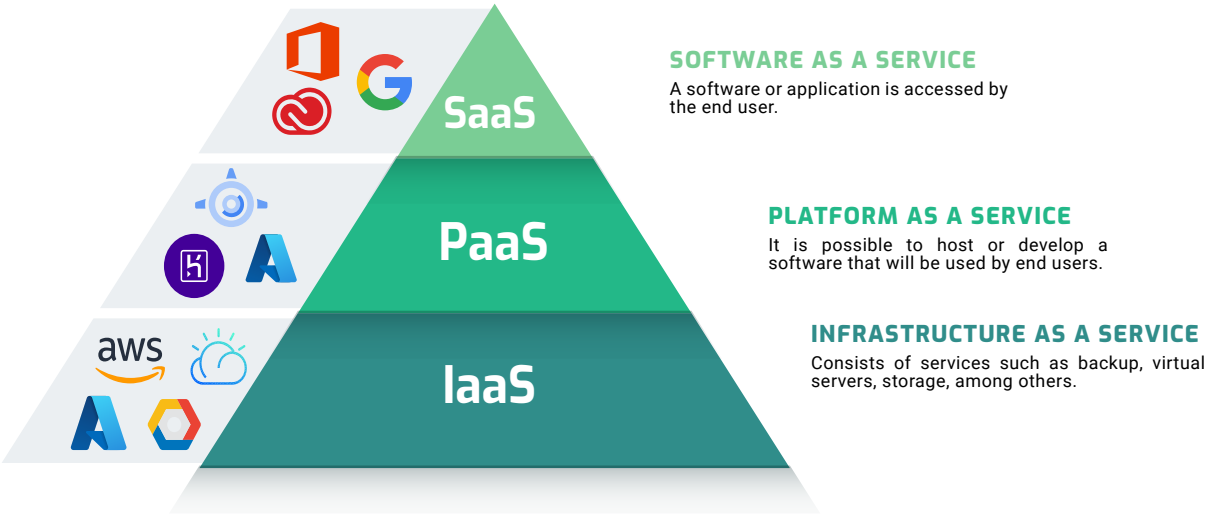


Figure 12 – Hierarchy of IaaS, PaaS, and SaaS layers.

2.6.1 Infrastructure as a Service (IaaS)

Infrastructure as a Service (IaaS) is a cloud computing model that provides access to infrastructure resources, such as servers, storage, and networking, as a service over the Internet. With IaaS, users can access virtualized infrastructure components such as virtual machines, storage, and networking resources on a pay-as-you-go basis. IaaS providers handle the underlying hardware and offer a range of configurable options, enabling users to create, configure, and manage virtualized environments to meet their specific needs. This

approach reduces the burden of capital-intensive investments and enables businesses to swiftly scale up or down in response to changing demands (MANVI; KRISHNA SHYAM, 2014).

For example, a company aiming to host its website on a server can rent a virtual server (virtual machine) for a specific duration rather than purchasing a physical server and managing it internally. Thus, the IaaS grants users enhanced flexibility and scalability concerning technological resources, all without the need to invest in expensive hardware. Leading providers offering IaaS include Amazon Web Services (AWS), Microsoft Azure, Google Cloud Platform, IBM Cloud, and Oracle Cloud.

2.6.2 Platform as a Service (PaaS)

Platform as a Service (PaaS) is a cloud computing model that provides a comprehensive platform for developers to build, deploy, and manage applications. PaaS providers offer a preconfigured environment with essential tools, such as operating systems, development frameworks, and databases. This enables developers to focus solely on creating and enhancing their applications, without the complexity of managing the underlying infrastructure. PaaS not only accelerates application development but also facilitates collaboration among development teams, ensuring seamless integration and streamlined workflows (ARMBRUST et al., 2010).

PaaS enables developers to focus on coding and innovation by minimizing the time spent on setting up and managing technical complexities. Platforms such as Heroku, Google App Engine, and Microsoft Azure provide PaaS solutions that simplify the entire development life-cycle, from coding and testing to deployment and scaling. This model provides a collaborative environment, enabling multiple developers to work together concurrently, resulting in increased productivity and faster time-to-market for applications.

2.6.3 Software as a Service (SaaS)

Software as a Service (SaaS) is a software delivery model in which the software is hosted in the cloud and users can access it through the Internet, typically via a web browser. Instead of purchasing and installing software on their own computers, users pay for a subscription to access the software hosted in the cloud. In this way, the SaaS provider is responsible for managing the necessary computational resources, network, and storage to run the software as well as keeping it up-to-date and secure (KAVIS, 2014).

The use of SaaS offers significant benefits to both individual users and businesses. For individual users, SaaS eliminates the need for software installation and provides easy access to tools such as email, document editing, and collaboration from any device. Businesses, on the other hand, benefit from the scalability and cost-effectiveness of SaaS, as they can pay for software on a subscription basis without the upfront costs associated

with purchasing and maintaining licenses. Additionally, SaaS applications often include automatic updates and security features, ensuring that users have access to the latest functionalities and protection against potential threats. This model enhances productivity, fosters collaboration, and aligns software expenditure with actual usage, making SaaS a transformative approach to software delivery in the digital age (ALI et al., 2019). Examples of software offered as a service include Microsoft Office 365, Google Apps, Adobe Creative Cloud, and Slack.

2.6.4 Artificial Intelligence as a Service (AIaaS)

Artificial Intelligence as a Service (AIaaS) is an innovative paradigm that harnesses the power of cloud computing to make AI capabilities accessible to a wider audience (LINS et al., 2021). This model provides on-demand access to AI algorithms, machine learning models, and data processing capabilities via the cloud. AIaaS providers enables developers, data scientists, and businesses to integrate AI functionalities into their applications without requiring extensive expertise in AI model training and deployment (ELSHAWI et al., 2018). This democratization of AI technology enables the creation of intelligent applications that can analyze data, recognize patterns, and make informed decisions.

As illustrated in Figure 13, AIaaS can be divided into three layers, with each layer organized as a stack according to the level of abstraction provided by its capability, which is inspired by traditional models of cloud services (LINS et al., 2021).

- ❏ **AI software service:** provides ready-to-use AI applications and building blocks similar to conventional SaaS cloud service.
- ❏ **AI developer service:** helps developers incorporate code to take advantage of AI capabilities and is akin to a conventional PaaS cloud layer.
- ❏ **AI infrastructure service:** offers raw computational power for building and training AI algorithms, and network and storage capacities to store and share data. This is related to conventional IaaS cloud services.

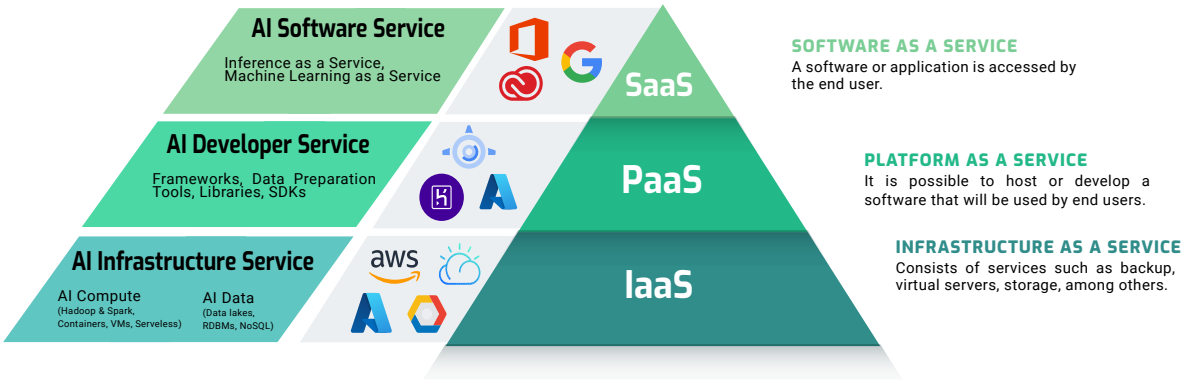


Figure 13 – AIaaS stack and its relationship with traditional cloud services.

In recent years, several researches have emerged in the domain of AIaaS, encompassing topics such as the development and evaluation of AI service models (BOAG et al., 2018; ELSHAWI et al., 2018; SINGH et al., 2023) , the adoption and effectiveness implementations (PANDL et al., 2021; ALHOSANI; ALHASHMI, 2024), the detection of potential misuse cases by users (JAVADI et al., 2020; CHAN et al., 2024), and the understanding of inherent challenges and vulnerabilities (TRUEX et al., 2021; PAN et al., 2024). These research strands collectively contribute to a comprehensive understanding of the multifaceted landscape surrounding AIaaS, encompassing aspects ranging from service design and operational aspects to the security and ethical considerations involved in this emerging paradigm.

2.7 Final Considerations

These models will be made available through an Application Programming Interface (API), enabling the integration of AI into specific applications tailored to user needs. Additionally, the integration of FL into our AIaaS architecture provides a secure and scalable solution for training ML models on edge devices, thereby reducing the need for a costly centralized infrastructure. By leveraging the computational power of distributed devices, FL enhances the adaptability, flexibility, security, and cost-effectiveness of AIaaS architecture. In the following chapters, the related work and the proposed approach will be presented, along with the results of the performed experimental evaluations for different scenarios.

Chapter 3

Related Work

In this chapter we provide a comparative survey of state-of-the-art solutions that relate to the subject of our approach.

Chan et al. (2013) proposed the PredictionIO architecture to integrate multiple machine learning processes into a horizontally scalable distributed system with a Hadoop-based distributed computing component. Baldominos et al. (2014) have also developed a platform built on Hadoop and tested it on an ad recommendation system on the web capable of serving up to 30 requests at the same time while maintaining a response time of less than one second. However, neither architecture is flexible enough to add multiple functionalities of AI for different applications because they are limited to their domain. This limitation arises from their confinement to specific domains, as adapting the Hadoop framework to a new approach is not a trivial task, despite it being an open-source platform.

Ribeiro, Grolinger e Capretz (2015) proposed a Machine Learning as a Service (MLaaS) architecture in order to create a flexible and scalable machine learning as a service, focusing on prediction tasks. MLaaS was tested in an application scenario to estimate the electricity demand of a company in the Canadian electricity sector, using classical supervised learning algorithms including Multi Layer Perceptron (MLP), K-Nearest Neighbors (KNN) and Support Vector Machine (SVM). However, MLaaS has not been adapted to handle other types of applications such as pattern recognition, clustering, or anomaly detection. Moreover, ML models cannot be offered as a feature for low-cost devices.

Li et al. (2017) constructed a distributed and scalable MLaaS architecture for Uber, a technology and transportation company that operates globally. The developed architecture supports different types of machine learning algorithms, where those based on deep learning are implemented using TensorFlow and Caffe frameworks, and traditional machine learning algorithms are implemented with Spark MLlib and XGBoost. The developed models were used for real-time predictions, such as estimating the delivery time of an order in the UberEats app. Despite being a comprehensive architecture, it currently lacks the resources to serve customers in a customized manner and is unable to train

deep learning models in a scalable manner with GPU resource allocation and real-time optimization. In addition, they are not oriented towards low-cost devices and it is a proprietary architecture.

Yao et al. (2022) developed the VenusAI architecture, based on AI, for large-scale computing scenarios in scientific research, leveraging the structure of resource scaling for heterogeneous resources. The VenusAI platform utilizes deep learning algorithms to conduct complex simulations, and has been tested in various use cases, including energy prediction, material analysis, and flight planning for unmanned aerial vehicles. However, VenusAI has limitations in providing support for large-scale scientific computing.

To comprehend the significance of AIaaS in today's society, it is important to explore the emerging applications of this groundbreaking technology. AIaaS has numerous potential applications in various fields, including autonomous driving. Caro et al. (2022) developed an AIaaS toolkit for an autonomous driving personalization system that uses an automatic driver stress recognition algorithm in a cyber-physical system. The toolkit includes a subsystem that gathers data from wearables and cameras to determine the driver's stress level in a cyber-physical system.

Shah et al. (2022) proposed the application of AIaaS in combination with SaaS and IaaS to prevent the spread of immoral content in social networks. They also proposed using AIaaS to detect and remove such content, which was classified into the immoral, cyberbullying, and dislike components. This approach leverages AI-driven optimization techniques to handle polarized immoral content. However, the authors focused mainly on traditional supervised learning methods, such as Decision Tree, SVM, and Naive Bayes, and did not evaluate the performance of deep learning techniques despite arguing that their strategy is suitable for handling large datasets.

Lewicki et al. (2023) performed a comprehensive review of the AIaaS market by presenting a taxonomy of AI services based on the levels of autonomy granted to users. They analyzed various classifications of AIaaS and described how these services can result in biases or cause harm in end-user applications. Fortuna et al. (2023) conducted a study to examine the significance of deploying on-premise AIaaS for small and medium-sized companies. The authors evaluated the functionality and technology stack of AIaaS and analyzed the feasibility of implementing it using open-source, user-friendly technologies that are suitable for on-premise environments, providing full control over data, processes, and costs while avoiding dependence on third-party vendors or the risk of vendor lock-in.

Guntupalli e Rudramalla (2023) presented a refined algorithm for ensuring data compression, data integrity, and data confidentiality in AIaaS deployments. AIaaS platforms typically feature an online repository containing numerous machine learning services and tools that users can access to fulfill their needs. However, selecting the optimal machine learning model or AI service is not always a straightforward task. To address this challenge, Cerar e Hribar (2023) proposed a dynamic approach that utilizes Deep

Reinforcement Learning (DRL) to select the most appropriate machine learning model. The authors demonstrated the effectiveness of their method in a specific case of energy consumption forecasting.

A low-code approach called the AI-Toolkit was proposed by Lomonaco et al. (2023) for decentralized machine intelligence, which seeks to bring Research and Development closer together through an open-source, microservice-based architecture. The AI-Toolkit gathers and utilizes AI microservices for the Teaching project (BACCIU et al., 2021) that outlines a computational graph as a docker-composed setup, consisting of multiple microservices that perform as data producers, consumers, or both. The authors assert that the AI-Toolkit offers substantial assistance with the creation of AI functionalities within the application, and can be customized for specific use cases and hardware platforms. In addition, the AI-Toolkit provides native support for recurrent neural networks for time-series forecasting and classification, generally applicable predictor for stress recognition based on physiological data, native support to predictors personalization based on reinforcement learning, and anomaly detection applied in Cybersecurity using Long Short-Term Memory (LSTM) Autoenconder architecture.

Zhang et al. (2023) proposed a model deployment architecture that can be configured for AIaaS used in EC. This architecture allows for the joint configuration of data quality ratios and model complexity ratios for AI tasks. The advantage of using AIaaS, supported by edge computing, is that it can deliver AI capabilities with low latency. Experiments and simulations were conducted on both ImageNet classification and COCO object detection tasks using state-of-the-art deep-learning models. The results demonstrate that the proposed approach can effectively improve the energy-delay performance of edge AI tasks. However, the authors did not evaluate the proposed approach in a real application scenario using mobile devices or low-cost devices. Our approach goes further by testing in a real scenario and proposing a hybrid Edge Predictor Service that consists in a centralized edge that is not limited to the resources available only at the edge.

Hajipour, Hekmat e Amini (2023) proposed a business plan that combines AI products and services through AIaaS. This plan outlines a strategic approach, step-by-step plan, and heuristic pricing model that can serve as a guide for AIaaS companies. The approach also includes a case study to demonstrate how the business plan can be applied in the real world. The goal is to provide a comprehensive process for commercializing AIaaS and utilizing the benefits of AI in the current business landscape. However, the proposed approach is limited to a business plan and has not been implemented in practice, while we present a detailed reference architecture.

Merluzzi et al. (2023) bases on Mobile Access Edge Computing (MEC) to delivery trained models for delivering trained models in the context of the Hexa-X Project. However, this approach is limited by the MEC, while in this thesis, we use multiple AI facilities.

Napisa et al. (2023) conducted a study on the utilization of AIaaS in the context of green smart cities, with a particular emphasis on the potential of Explainable AI (XAI) and Interpretable Artificial Intelligence (IAI) models to improve urban living standards and foster sustainable growth. Despite the advantages offered by these solutions, issues such as the intricacy of XAI models and concerns regarding data privacy and security continue to represent significant challenges to their implementation.

To stay ahead in the rapidly advancing world of smart digital technology, B5G have been introduced to improve a wide range of intelligent services and cutting-edge applications by rapidly deploying advanced AI across global networks. Moreover, the integration of AI-powered services is a critical missing element in previous generations, and will be the primary driving force behind the 6G leap in capabilities (PIVOTO et al., 2023).

In this sense, Baccour et al. (2023) proposed a novel platform architecture for deploying zero-touch Pervasive Artificial Intelligence as a Service (PAIaaS) in 6G networks, supported by a blockchain-based smart system. To demonstrate the capabilities of the proposed platform, they presented a use case for FL, where the service consumer deployed distributed training on the 6G infrastructure using widespread devices.

Oliveira et al. (2024) proposed a framework for anomaly detection in 5G using ML algorithms in Network Data Analytics Function (NWDAF), which is a component of the 5G and B5G network architectures responsible for collecting and analyzing data from various network elements and performing data analytics to monitor network performance, detect anomalies, and optimize operations (3GPP, 2023). Nadar e Härri (2024) also proposed an architecture for integrating NWDAF with AIaaS platforms to enhance 5G network analysis. However, the NWDAF standardization model is managed only by network operators, which hinders third-party innovation in cognitive service delivery (BRILHANTE et al., 2023; GKONIS et al., 2024; ISHTEYAQ et al., 2024).

To address this challenges, the approach presented in this thesis combines and evolves NWDAF with AIaaS to specify how third-party services can be integrated and specified within the network framework. Additionally, we proposed an architectural framework for lifecycle management and delivery of AI services that aims to seamlessly deploy and embed intelligent capabilities to applications or edge devices.

In the context of the industry, major companies, such as Amazon, Google, and Microsoft, have launched proprietary platforms. However, external developers do not have access to code to add specific functionalities to different application scenarios. In response to this, the AIaaS proposed in this thesis innovates by facilitating the inclusion of new machine learning models, their enhancement, and adaptation to other applications. Furthermore, the related architectures can be integrated into the proposed architecture to support various AI applications.

Table 2 summarizes the related works described above and highlights the differences that demonstrate how our proposal represents a significant advancement in the field. The

"Native API for smartphone" column characterizes the solutions to offer native API in mobile applications. The solutions proposed by Chan et al. (2013) and Li et al. (2017) are examples of this type of support. The "Native API for low-cost devices" column characterizes the solution in terms of its ability to support low-cost devices. Predominantly, state-of-the-art solutions require expensive devices and high computational resources. The same occurs when analyzing the "Dataset Management" column, which refers to the possibility of managing different sets of data, where it can be seen that this is a feature that is absent from all state-of-the-art solutions.

The "Multiple AI Facilities" column characterizes the solutions to offer multiple AI facilities such as different ML algorithms, optimization methods, and feature extraction. The approaches proposed by Ribeiro, Grolinger e Capretz (2015), Lomonaco et al. (2023) and Cerar e Hribar (2023) are examples of this type of support. Finally, the "Computer Vision" column refers to the use of computer vision methods for different studies. Some solutions such as proposed by Li et al. (2017), Caro et al. (2022) and Zhang et al. (2023) make use of computer vision.

Table 2 – Short state-of-the-art survey.

Approach	Native API for smartphone	Native API for low-cost devices	Dataset Management	Multiple AI Facilities	Computer Vision
Chan et al. (2013)	●	○	○	○	○
Baldominos et al. (2014)	○	○	○	○	○
Ribeiro, Grolinger e Capretz (2015)	○	○	○	●	○
Li et al. (2017)	●	○	○	●	●
Yao et al. (2022)	○	○	○	●	○
Caro et al. (2022)	○	○	○	●	●
Shah et al. (2022)	○	○	○	●	○
Lewicki et al. (2023)	○	○	○	○	○
Fortuna et al. (2023)	○	○	○	●	○
Guntupalli e Rudramalla (2023)	○	○	○	●	○
Cerar e Hribar (2023)	○	○	○	●	○
Lomonaco et al. (2023)	○	○	○	●	○
Zhang et al. (2023)	○	○	○	●	●
Hajipour, Hekmat e Amini (2023)	○	○	○	●	○
Merluzzi et al. (2023)	○	○	○	○	○
Napisa et al. (2023)	○	○	○	●	○
Baccour et al. (2023)	○	○	○	○	○
Oliveira et al. (2024)	○	○	○	●	○
Nadar e Härri (2024)	○	○	○	●	○
Our approach	●	●	●	●	●

3.1 Final Considerations

This chapter presents a panorama of related works that offers a comprehensive perspective on the existing landscape. These previous studies have showcased the multifaceted applications of AI in diverse domains, underscoring the transformative potential of AI technologies. Although these studies have provided important insights, the proposed architecture is an innovative step forward by incorporating edge computing, service management, and model training. Through its holistic approach, the architecture strives to address the limitations and challenges of existing solutions, paving the way for efficient, adaptable, and real-time AI deployment in different fields.

Chapter 4

Proposal

This chapter details the contributions of this thesis. We present the rationale for the proposed AIaaS architecture considering its design and practical implementation. Also, we offer a comprehensive understanding of its functionalities, use-cases, and potential applications in real-world scenarios.

The proposed architecture represents an innovative approach for providing AI resources as a service in edge computing. To achieve this, several technologies, contributions, and developments need to be systematically organized into research fronts. Consequently, the hypothesis deduction is subdivided into three technological domains, from which activities are developed within the scope of the project.

- **Infrastructure:** The infrastructure of the proposed architecture is rooted in cloud computing, and aims to deliver services in the form of IaaS or PaaS. IaaS is a flexible and scalable cloud service that provides complete computing infrastructure (e.g., Amazon Web Services - AWS). On the other hand, PaaS is a comprehensive platform for application development and hosting, which helps expedite the development process and reduce infrastructure costs (e.g., Google Colaboratory). In this thesis, we utilize the IaaS services offered by conventional providers available in the market. Using these high-performance servers, the API of the AIaaS architecture API was constructed.

- **Edge Predictor:** This domain involves the instantiation of all AI resources on-demand in mobile devices or low-cost devices for prediction tasks. For instance, it categorizes diseases within a specific domain by using a mobile application. Edge devices request the most suitable AI service through the infrastructure domain using the API.

- **Service Management:** This entails the management of the essential functionalities of the proposed architecture. In this technological domain, mechanisms for the control and management of API, which operates within a high-performance computing infrastructure, will be developed.

An overview of the proposed architecture is shown in Figure 14. Each component is detailed in the following subsections, which highlight the opportunities and contributions that will be achieved.

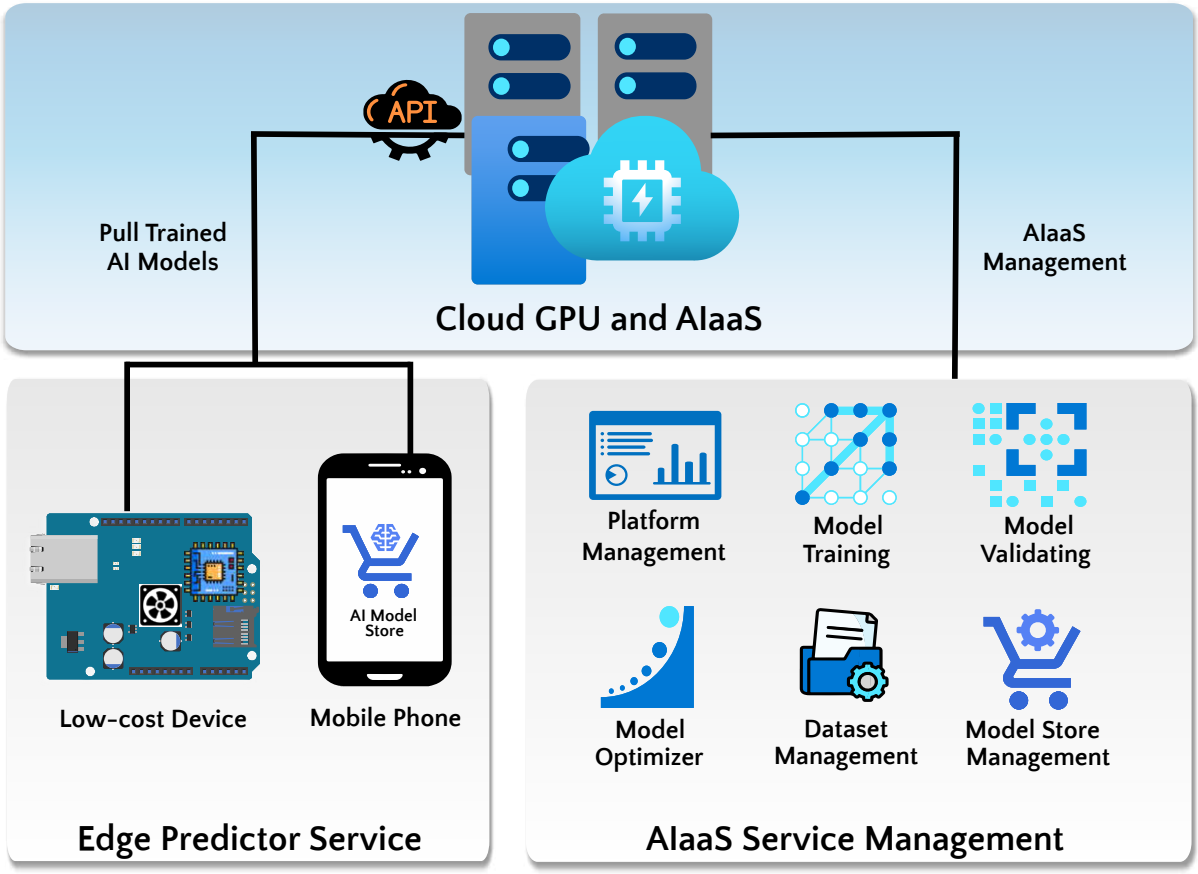


Figure 14 – General Contribution: AIaaS Architecture.

4.1 Infrastructure

The **Infrastructure** uses high-performance cloud-computing services for instantiation of GPU and hardware resources. Two functions are performed within this infrastructure: hosting the functional modules of the proposed architecture and providing the necessary hardware resources for executing computationally expensive tasks such as training or optimizing ML models. Well-established frameworks in the community and industry, such as PyTorch, TensorFlow, and Keras, will be employed to handle specific tasks such as model creation, training, validation, and optimization in the realm of ML (e.g., CNN and Supervised Algorithms). Furthermore, communication with Edge Prediction via API and Service Management is facilitated by the architecture manager.

Edge Predictor requests access to appropriately trained models that are downloaded onto mobile or low-cost devices. By training, optimizing, and storing models with various datasets for different contexts, the proposed architecture aims to robustly serve various needs, such as identifying different medical diseases or classifying different agricultural

crops. **Service Management** will be employed to train ML models in cloud computing environments and to handle the management, storage, optimization, and validation of these models.

To create the Infrastructure API for the proposed Architecture, the following services are required:

- ❑ **GPU-based ML Trainer Allocation:** This service is used by the architecture administrator to train a specific ML model.
- ❑ **Resource Manager:** This service is used by the architecture administrator to check the available computing resource quotas.
- ❑ **ML Model Retrieval:** This service is employed when a user (Edge Predictor) requests a pre-trained and existing model for a specific prediction task.
- ❑ **Dataset Storage:** The architecture administrator intends to upload datasets for ML tasks.
- ❑ **ML Model Storage:** This service is used by the architecture administrator to train and validate a ML model available through the API.
- ❑ **ML Model Optimizer:** This service is used when the architecture's administrator aims to enhance a trained ML model.

4.2 Edge Predictor

The **Edge Predictor** encompasses the user side of the interaction with the AIaaS architecture in order to acquire or manage the trained models available within it. This is termed **Edge Prediction** because it employs the novel concept of distributed computing to process and store data on devices situated closer to the application scenario, rather than sending them to a central server. In this way, edge devices (such as smartphones, notebooks, Arduino, and Raspberry Pi) can process and analyze samples locally and in real time, alleviating the burden of network traffic on communication networks, which can be non-existent or of low quality in scenarios such as remote or rural areas.

In this phase, we developed a mobile application (for widely accepted mobile operating systems) that interacts with the Infrastructure API and retrieves pretrained ML models. This application enables users to make predictions on given samples using the chosen pre-trained ML model, as illustrated in Figure 15. Furthermore, the application will facilitate the real-time deployment of models onto low-cost devices, allowing users with varying levels of expertise to harness the potential of robust ML models.

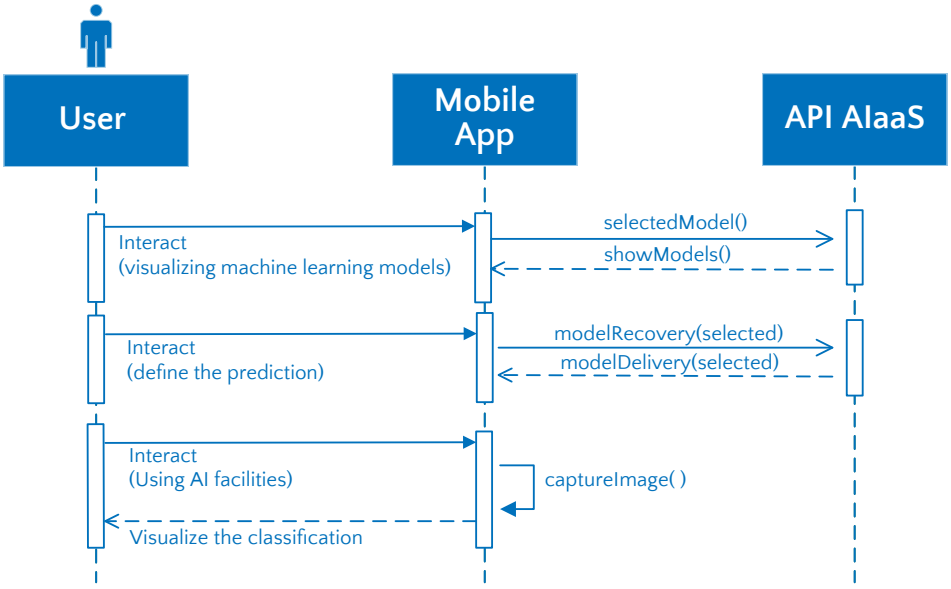


Figure 15 – Behaviors that need to be investigated and refined in the Edge Predictor.

4.3 Service Management

The **AIaas Management Service** facilitates the administration and operational aspects of the architecture. Our proposed method includes **Platform Management** with a Graphical User Interface to support the manager handling the architecture, as depicted in Figure 16, where the features are available on the left menu.

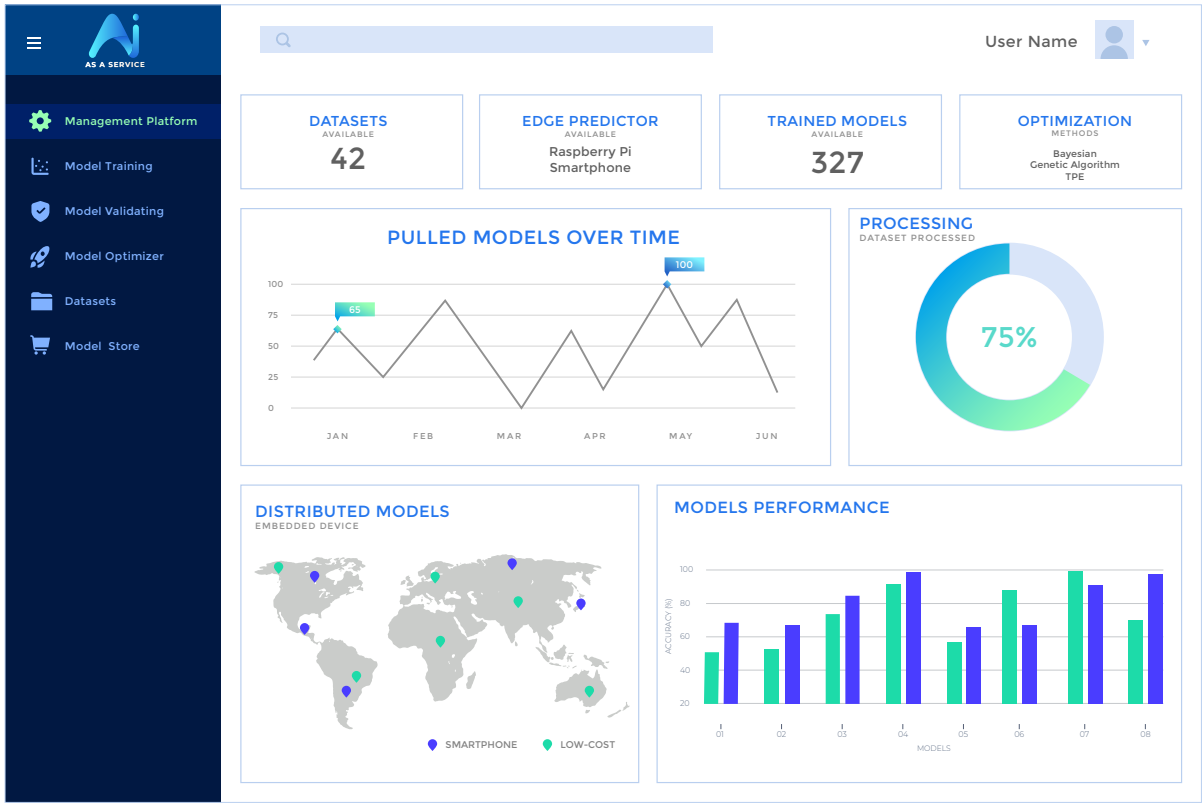


Figure 16 – Proposed AIaaS Management Platform GUI.

The **AIaaS Management Service** provides a dashboard with a general view of AIaaS, showing the number of datasets available, edge predictors used by devices, number of AI models adequately trained and evaluated in the cloud GPU, and model optimizer available to improve the AI models during the training enhancement stage. Our dashboard provides the status of the dataset processing, geographical location of the downloaded models, and number of pulled models over time. Finally, Graphical User Interface (GUI) shows the general performance of the trained models.

Service Management is accountable for the operation and administration of the architecture through the five submodules tasked with managing the functioning of the entire architecture. Figure 17 illustrates the administrator's interaction with the architecture, as well as the behaviors that need to be investigated, expanded, and implemented within the scope of project development. Subsequently, the functionalities of the Service Management carried out by the architecture administrator are delineated.

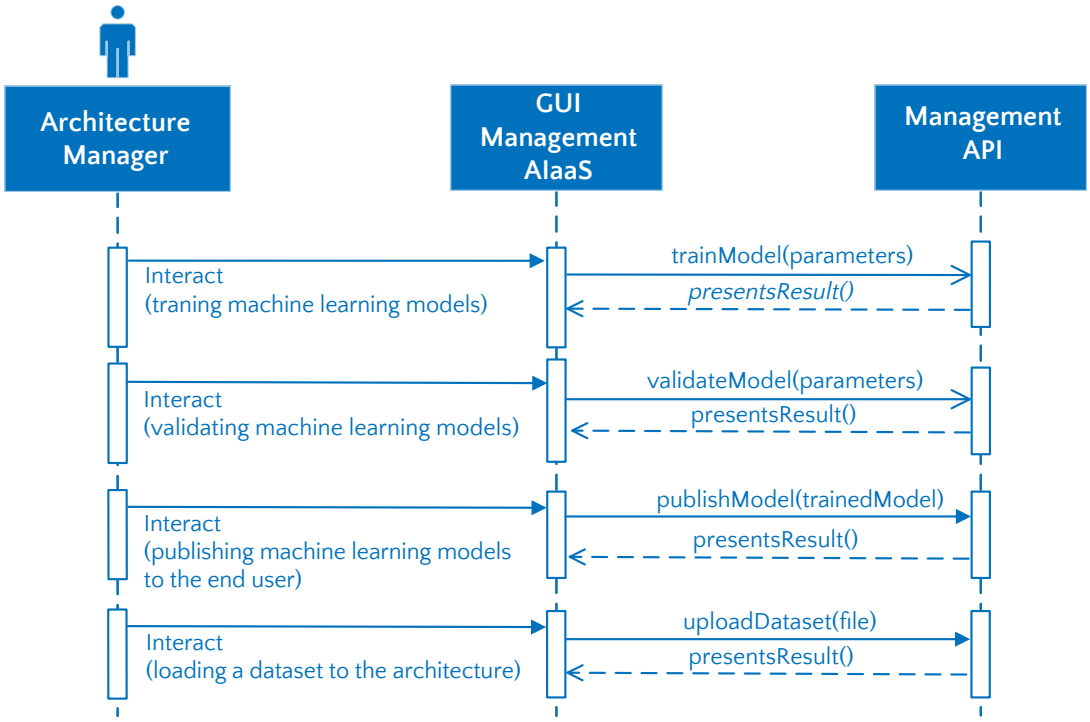


Figure 17 – Behaviors that need to be investigated and refined in the Service Management.

4.3.1 Platform Management

Platform Management serves as a management platform with a user-friendly GUI, aimed at facilitating the administration of architectural functionalities. This empowers administrators to oversee a spectrum of tasks, including the management of available datasets, utilization of edge prediction, access to available model optimizer, and other essential features. A comprehensive dashboard provides an overarching perspective of the entire proposed architecture, aiding decision-making and administrative actions.

4.3.2 Model Training

The model training phase enables the development of diverse AI models, offering the flexibility to select suitable ML paradigms for each application. Consequently, this stage aims to identify the most fitting models for various computer vision contexts and analyze, evaluate, and refine each model. This process entails the examination of state-of-the-art models representing leading AI techniques along with the potential introduction of simpler yet robust models that harness attributes from the finest models documented in the literature. Our goal is to identify the optimal models for different domains and optimize their performance.

In this context, the model-training process encompasses multiple objectives. First, it explores models that align with specific computer vision needs, focusing on precision, interpretability, and resource efficiency. Second, it involves rigorous model assessment, performance metrics, and potential enhancement strategies. Finally, the process contributes to advancing the field by combining innovative approaches with well-established techniques to develop models that can excel across various applications.

4.3.3 Model Validation

Following the training process, the models were validated using the classical quantitative metrics. Upon successful validation, they were approved and subsequently incorporated into the architecture model catalog.

The validation stage is pivotal for ensuring the efficacy and suitability of the models for their intended tasks. This involves subjecting the trained models to established quantitative metrics that measure their performance across various criteria such as accuracy, precision, recall, and F1-score. The models that met the predefined criteria and demonstrated robust performance were validated.

Once validated, these models were eligible for inclusion in the architecture model catalog. This catalog serves as a repository of pre-trained models covering a diverse range of applications. Users can then seamlessly access and deploy these validated models for specific prediction tasks, thereby enhancing the overall versatility and utility of the architecture.

4.3.4 Model Optimizer

Selecting optimal hyperparameters for training a deep CNN is crucial, but there is no one-size-fits-all method for this task. Common approaches such as grid search, random search, Bayesian optimization, and Genetic Algorithm (GA) are used in ML to tackle this challenge (BERGSTRA et al., 2011; MARCOS; RODOVALHO; BACKES, 2019; MOREIRA et al., 2020; SILVA; ESCARPINATI; BACKES, 2021). However, the computational

cost of grid search is too high, while random and Bayesian approaches are limited to the search space distribution (LIASHCHYNSKYI; LIASHCHYNSKYI, 2019).

Hyperparameters are essential settings that influence the performance of a CNN, determining aspects such as the topology of the CNN, optimization algorithms, and training details (BENGIO, 2012). The selection of hyperparameter values lacks a universally applicable method, and often involves iterative experimentation through trial and error. To address the gap in the state-of-the-art regarding the best hyperparameter setup (ZHOU et al., 2021), we developed a novel approach using GA.

In this thesis, we treated hyperparameters as decision variables and aimed to minimize the loss function to fine-tune their values. The following hyperparameters were used in the analysis:

- **Dropout (d):** dropout is a technique to deal with overfitting. It is based on the random dropping of neurons during the training process, that is, a unit out is temporarily removed from the network along with all its incoming and outgoing connections (SRIVASTAVA et al., 2014).

- **Learning rate (l):** the learning rate is the main tuning parameter, being responsible for improving the Stochastic Gradient Descent (SGD) (LECUN et al., 1998) optimizer runtime. This defines the level of adjustment of the weight connections and network topology applied at each training epoch. A high learning rate may sacrifice accuracy owing to the lack of precision in the adjustments. However, a low learning rate requires more training epochs and longer processing times (BENGIO, 2012).

- **Momentum (μ):** the momentum coefficient (POLYAK, 1964) is responsible for reducing oscillations in the high-curvature regions of the loss function generated by the SGD. By default, its value is set to 1; however, fine-tuning this hyperparameter may lead to improved results (PONTI et al., 2017).

4.3.5 Model Store

The Model Store provides the settings and deployment of machine learning models properly trained, pushing them to the AIaaS API that runs in the cloud computing infrastructure. In addition, it is possible to load the models onto low-cost devices on the fly, enabling high-performance models in different fields of application.

There are distinct views of the Model Store: the architecture administrator can manage the life cycle of models that can be pulled by model subscribers. The model subscribers skimmed the catalog of available models on the platform at a glance and downloaded them on demand. Moreover, the Model Store enables users with different skills to take

advantage of the potential of robust machine learning models, thereby avoiding users requiring expensive computing resources for training and validation workloads.

4.4 Final Considerations

In this chapter, we presented a comprehensive overview of the proposed architecture, emphasizing its innovative approach to providing AI resources at the edge. The foundation of the architecture is rooted in cloud computing, providing IaaS and PaaS capabilities. These facets enable the instantiation of AI models on demand and the development of applications for different scenarios.

Our research also addresses research endeavors in edge prediction, ranging from model adaptation to low-latency inference. These aim to bolster the architecture's robustness, adaptability, and security, ensuring reliable and efficient AI deployment in diverse settings.

In summary, this architecture presents an innovative approach towards efficient, localized, and adaptable AI deployments in Computer Vision tasks. With its emphasis on edge prediction, service management, and model training, this architecture opens new avenues for improved medical image diagnosis, resource management, and enhanced decision support. This chapter sets the stage for subsequent exploration, where the functionalities of the architecture are developed, optimized, and practically implemented.

Chapter 5

Edge Predictor Service Evaluation

In this chapter, we present and analyze the results of the Edge Predictor Service evaluation, focusing on the performance across low-cost devices (Section 5.1) and conventional devices (Section 5.2). For this analysis, we use the term "conventional devices" to refer to typical mobile devices, such as smartphones and tablets, which possess relatively advanced hardware capabilities. The evaluation emphasizes measuring the effectiveness and efficiency of the prediction service on different types of edge devices, with particular attention paid to metrics such as response time, energy consumption, and prediction accuracy. By comparing these metrics between low-cost and conventional devices, we aim to understand the capabilities and limitations of the Edge Predictor Service in real-world scenarios, where device heterogeneity is a critical factor.

5.1 Exploring the Capabilities on Low-Cost Devices

To evaluate the capabilities of low-cost devices in our AIaaS architecture, we focused on the recent COVID-19 pandemic, a highly contagious global public health emergency that affects millions of people worldwide (ROSER et al., 2020). Our evaluation considered the potential applications and limitations of low-cost devices in supporting public health responses during such crises. By analyzing key metrics, such as processing power, energy efficiency, and model classification performance, we assessed the feasibility of deploying these devices in resource-constrained environments for health monitoring and diagnostics.

During the pandemic, it was found that COVID-19 manifested as a respiratory disease that led to lung inflammation and pneumonia (SHI et al., 2020). Furthermore, survivors of the disease are at an increased risk of developing cardiovascular complications such as cerebrovascular disorders, dysrhythmias, and heart failure (XIE et al., 2022). Therefore, early diagnosis is crucial to control the spread of the disease and to provide appropriate treatment.

Reverse transcription-polymerase chain reaction (RT-PCR) has been widely used to support the diagnosis of COVID-19. However, the RT-PCR test has a long time to re-

lease the outcome and presents poor sensitivity owing to high false-negative rates as well as sample collection, transportation, and kit performance limitations (SHARFSTEIN; BECKER; MELLO, 2020). It is crucial to develop alternative diagnostic methods for COVID-19, such as chest X-rays and Computed Tomography (CT) scans, which are effective in detecting and assessing lung damage at various stages of the disease (RUBIN et al., 2020). Although CT is a highly accurate examination, chest X-ray imaging is still beneficial owing to its low cost, speed, and radiation exposure (RUBIN et al., 2020; RODRIGUES et al., 2020). However, approaches based on images are subjective and require visual analysis, and health specialists must look for white patches in the lungs, which can also be confused with pneumonia caused by different pathogens, bronchitis, or tuberculosis (ZHOU et al., 2020; PEREIRA et al., 2020).

Thus, computer-aided diagnostic systems that utilize both AI and Computer Vision are a viable option for extracting features from X-ray images, ultimately leading to an accurate diagnosis of COVID-19 and enabling for critical time to be saved in the disease management and control process.

Previous studies have explored the use of deep learning to diagnose COVID-19 based on X-ray images (RODRIGUES et al., 2020; ISLAM et al., 2021; KHAN et al., 2021; AGGARWAL et al., 2022). However, there are challenges associated with using computational methods based on deep learning to support the COVID-19 pandemic, such as the difficulty of generalizing a detection model and the need for specific preprocessing for data from different locations. Additionally, powerful computing resources are required to train deep learning models and they typically use centralized intelligence on high-performance servers, making it challenging to embed those models on low-cost devices or smartphones for early disease diagnosis (WANG et al., 2020; CHANG et al., 2021; AGGARWAL et al., 2022).

To overcome this gap, we validated our designed AIaaS Architecture, which has a hybrid AI support operation that is both centralized and distributed. Our architecture takes advantage of the AI services offer model, allowing low-cost devices to access and utilize specialized trained models from the cloud for specific contexts. Our approach enables the seamless integration of CNN models on low-cost devices, eliminating the need for resource-intensive training tasks, and enabling the utilization of pre-trained models across multiple domains.

The goal of these experiments was to answer the following Research Questions ($\mathcal{RQ}s$).

- ❏ $\mathcal{RQ}1$: Low-cost devices are able to timely predict X-ray images?
- ❏ $\mathcal{RQ}2$: Among the evaluated CNNs, which ones are more complex in terms of computational cost?
- ❏ $\mathcal{RQ}3$: Which CNN takes the shortest time to predict an X-ray image?

- ❑ $\mathcal{RQ}4$: Do more complex CNNs require more time to predict an X-ray image?
- ❑ $\mathcal{RQ}5$: Considering RAM memory, what is the computational overhead that CNNs impose on the low-cost device hardware?
- ❑ $\mathcal{RQ}6$: What criteria should be considered to embed a CNN on a low-cost device for image classification?

In the following sections, we detail the prior approaches employed to identify COVID-19 on edge devices and the dataset utilized for evaluation.

5.1.1 Previous Approaches to classify COVID-19 on Edge Devices

We analyzed past research that employed image analysis and AI techniques on edge devices to identify COVID-19. We also compared these previous methods with our proposed approach.

The initial method for diagnosing X-ray images using an mobile application was proposed by Sait et al. (2019) and was implemented for pneumonia diagnosis. Nevertheless, the authors did not consider potential hardware limitations. Maghded et al. (2020) proposed a framework to detect COVID-19 using smartphone sensors, which involves reading signal measurements from the sensors and scanning CT images to identify COVID-19 and viral pneumonia. However, they did not implement the proposed framework on a smartphone.

COVID-MobileXpert, a mobile system for detecting COVID-19 from X-ray images, was proposed by Li, Li e Zhu (2020). The system utilized a pre-trained DenseNet-121 model on 100k X-ray images and tested the performance of MobileNetV2 and SqueezeNet on Android devices. Alam e Rahmani (2021) utilized FL for COVID-19 classification and segmentation of X-ray images by employing Raspberry Pi for Internet of Medical Things. Although they evaluated Raspberry Pi, the slow training process required reliance on a central server.

Paluru et al. (2021) introduced Anam-Net, a lightweight CNN, for the purpose of detecting anomalies in CT images related to COVID-19 evaluating the performance on Raspberry Pi, Jetson Xavier, and Android. Bushra, Ahamed e Ahmad (2021) developed an Android application that uses a CNN architecture inspired by GoogLeNet to identify COVID-19 in X-ray images reducing the number of parameters.

Sait et al. (2021) proposed a multimodal framework for COVID-19 detection using X-ray images, breathing sounds, and rapid antigen tests. They utilized an Inception-V3 CNN to extract features and a multilayer perceptron (MLP) as a classifier. They also developed an Android smartphone application and evaluated its performance on three

smartphones with different specifications, but did not provide sufficient details on the mobile application's performance on these devices.

Hosny et al. (2021) proposed a COVID-19 diagnosis system utilizing both chest X-ray images and CT scans, and implemented it on a Raspberry Pi Linux embedded system. They utilized handcrafted feature extraction and conventional classifiers. While the model size was small, the prediction time was relatively long. Rangarajan e Ramachandran (2021) evaluated five pre-trained CNN models and integrated them into a smartphone, but the performance of the CNN is hindered by limitations in terms of the limited memory space and longer average prediction time. Verma et al. (2022) proposed an Android application that detects COVID-19 from CT scans using a CNN. They evaluated several CNN models, and evaluated the mobile application on four smartphones, but they did not consider low-cost devices.

Most of the surveyed papers primarily focused on identifying COVID-19 through X-ray imaging. However, our contribution extends beyond this by proposing and evaluating an AIaaS Architecture. To provide a clear overview of all the approaches, we present Table 3. The "Image" column refers to the type of lung image (CT or X-ray) used by the proposed approach. The "Classes" column indicates the categories of lung images considered. The solutions found predominantly consider COVID-19, Pneumonia, and normal lung. The "Method" column indicates the computational method utilized. The "Device" column indicates the device type considered.

The "Embedded" column describes whether the solution is integrated into the proposed device. Solutions proposed in (LI; LI; ZHU, 2020; ALAM; RAHMANI, 2021; PALURU et al., 2021; SAIT et al., 2021; HOSNY et al., 2021) are examples of embedded solutions. The "Embedded Task" specifies the task that was executed on embedded devices. For instance, the solutions presented in (ALAM; RAHMANI, 2021; PALURU et al., 2021) utilize different devices to perform image segmentation. On the other hand, for the solutions proposed in (SAIT et al., 2019; MAGHDED et al., 2020; RANGARAJAN; RAMACHANDRAN, 2021) this feature is indicated as not applicable (NA) because the approach was not embedded.

Finally, the "Affordable" feature refers to the cost of computing devices. Some solutions, such as (ALAM; RAHMANI, 2021; HOSNY et al., 2021), utilize low-cost devices. However, most solutions found in the literature require the acquisition of devices at a cost of over US$100. In our approach, we go further by using a low-cost device and applying it as a promising alternative with low financial cost to support the automatic diagnosis of COVID-19.

Table 3 – Short State-of-the-Art Survey of approaches that using edge devices to support COVID-19 diagnosis.

Approach	Image	Classes	Method	Device	Embedded	Embedded Task	Affordable
Sait et al. (2019)	X-ray	Normal Lung Pneumonia	CNN	Mobile App Android	○	N/A	○
Maghded et al. (2020)	CT	COVID-19 Pneumonia	CNN	Mobile App Android	○	N/A	○
Li, Li e Zhu (2020)	X-ray	COVID-19 Pneumonia Normal Lung	CNN Supervised Classifiers	Nexus One/ Nexus S Pixel/ Pixel 2 Pixel 2 XL/ Pixel 2 XL	●	Prediction	○
Alam e Rahmani (2021)	X-ray	COVID-19 Pneumonia Normal Lung	Segmentation with U-net and FL	Raspberry Pi 4	●	Segmentation	●
Paluru et al. (2021)	CT	COVID-19 Normal Lung	Segmentation with Anam-Net	Raspberry Pi 4 NVIDIA Jetson Xavier Nokia 5.1 Plus	●	Segmentation	● ○ ○
Bushra, Ahamed e Ahmad (2021)	X-ray	COVID-19 Normal Lung	CNN	Mobile App Android >version 6	○	N/A	○
Sait et al. (2021)	X-ray	COVID-19 Normal Lung Pneumonia	CNN	Mobile App Android >version 6	●	Prediction	○
Hosny et al. (2021)	X-ray CT	COVID-19 Normal Lung	LBP Random Forest Logistic	Raspberry Pi 4	●	Prediction	●
Rangarajan e Ramachandran (2021)	X-ray	COVID-19 Normal Lung Pneumonia	CNN	Mobile App Android >version 6	○	N/A	○
Verma et al. (2022)	CT	COVID-19 Normal Lung	CNN	Nokia 5.1 Plus Xiaomi Note 4 Samsung A30 Samsung Galaxy S2	●	Prediction	○
Our Approach	X-ray	**COVID-19 Pneumonia Normal Lung**	**CNN**	Raspberry Pi 4	●	**Prediction**	●

We proposed the functional viability of constructing an AIaaS Architecture that delivers AI capacities as services to edge computing. Our system concurrently satisfies all the specifications shown in Table 3, introducing a novel approach.

5.1.2 Experiments

The images used in these experiments were obtained from the COVID-19 Radiography Database[1] (CHOWDHURY et al., 2020) (RAHMAN et al., 2021) and consisted of three classes: COVID-19 (3616 images), viral pneumonia (1345 images), and normal lungs (10,192 images). Figure 18 presents the sample images from each class in the dataset.

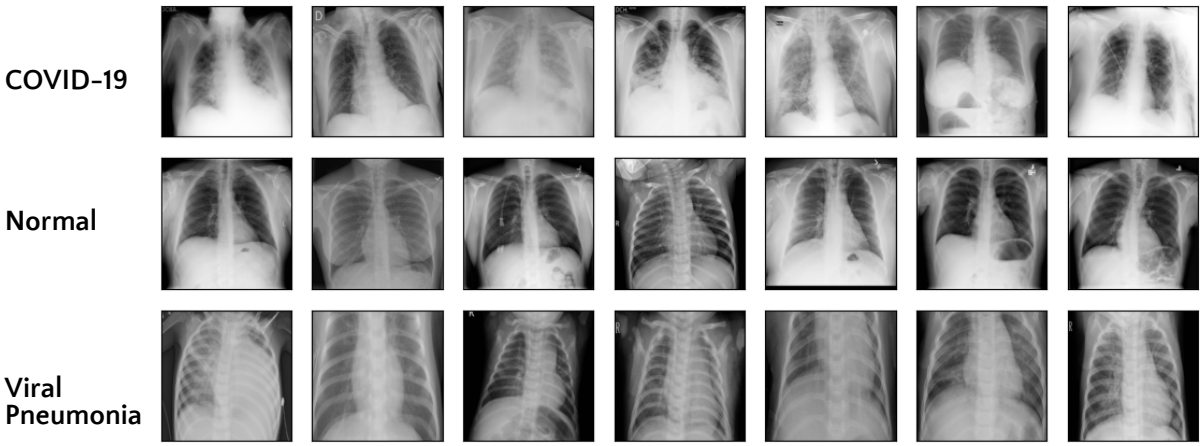


Figure 18 – Sample images from the COVID-19 Radiography Database.

We used Python (version 3.6) and the PyTorch deep learning framework to program the experiments. We partitioned the COVID-19 Radiography Database into three subsets: 80% for training, 10% for validation, and 10% for testing. We also resized all images to 224 × 224 pixels. We conducted training on a Cloud GPU and AIaaS using pre-trained 2012 ImageNet weights (KRIZHEVSKY; SUTSKEVER; HINTON, 2012) and the Adam optimizer (KINGMA; BA, 2017) with a learning rate of 0.0001, batch size of 16, and 30 epochs. We also performed data augmentation with random rotation (between -30° and 30°) and vertical and horizontal flips.

We analyzed six CNN architectures architectures selected due to their success in previous research on COVID-19 classification based on images (KHAN et al., 2021; AGGARWAL et al., 2022). These CNNs present different architectural features, such as residual connections and dense blocks. We observed in the literature the correlation between the depth of CNNs and their performance, leading to the design of very deep CNNs with skip connections (BIANCHINI; SCARSELLI, 2014; WU; SHEN; VAN DEN HENGEL, 2019; OYEDOTUN; ISMAEIL; AOUADA, 2021). On the other hand, shallow CNNs are recommended for embedded platforms (ZHOU et al., 2021). In this thesis, we aimed

[1] Available in: <https://www.kaggle.com/tawsifurrahman/covid19-radiography-database>

to explore the possibility of embedding different models of CNNs, including shallow and deeper models. We considered the following CNNs in this experiment:

- ❏ **Deeper Models:** AlexNet, DenseNet, and ResNet.
- ❏ **Shallow Models:** MobileNet, ShuffleNet, and SqueezeNet.

We provide the results for each CNN in terms of the average accuracy, precision, recall, and F1-Score. Figure 19 shows the loss and accuracy values for both the training and validation sets, demonstrating that the training did not result in overfitting the data and preserving the generalization property of each CNN evaluated.

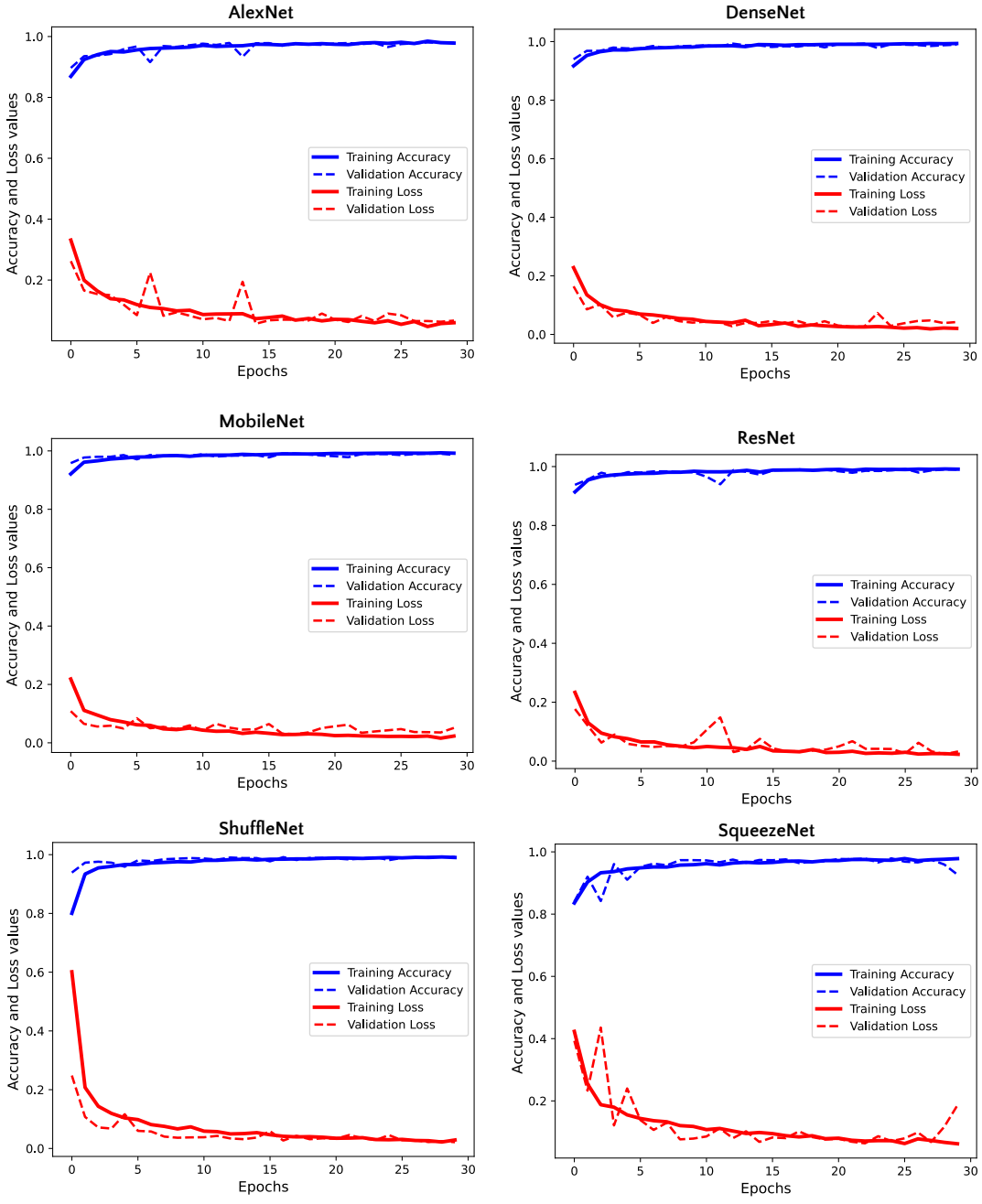


Figure 19 – Accuracy and loss values considering training and validation sets.

Subsequently, we present a overview of each experiment conducted in response to the six Research Questions ($\mathcal{RQ}s$) addressed in this evaluation.

$\mathcal{RQ}1$: Low-cost devices are able to timely predict X-ray images?

To answer $\mathcal{RQ}1$, we embedded the trained CNN models in a low-cost device and verified that they can predict X-ray images in a timely manner. As shown in Figure 20, most of the evaluated CNNs performed the prediction in less than one second. Therefore, our analysis indicates that CNN models can be successfully embedded into low-cost devices.

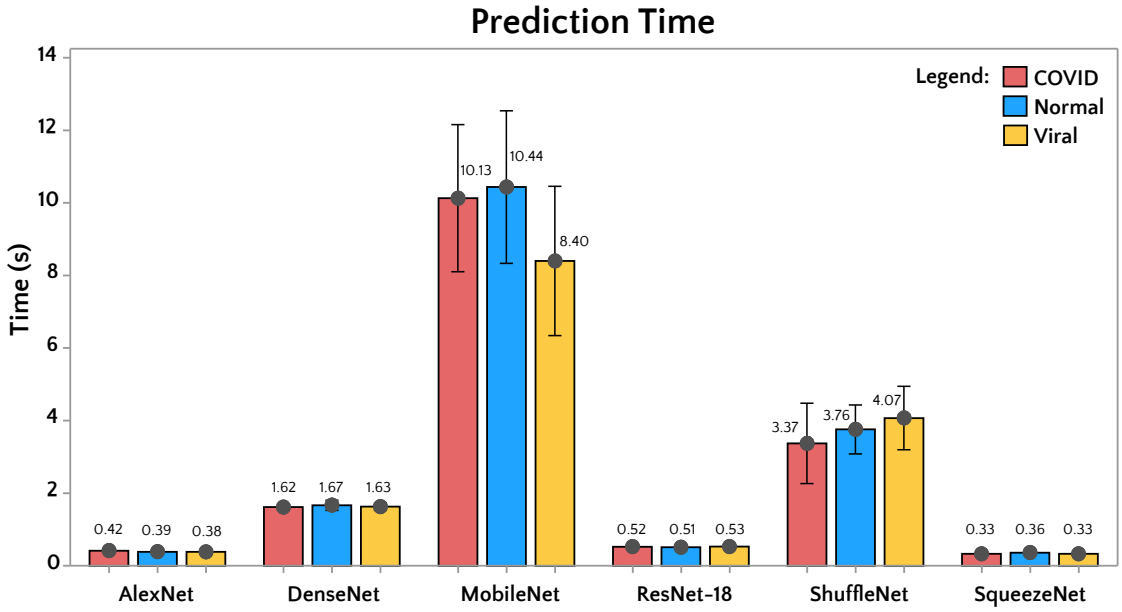


Figure 20 – Prediction time for each CNN considering each class.

$\mathcal{RQ}2$: Among the evaluated CNNs, which ones are more complex in terms of computational cost?

We conducted a more in-depth investigation to answer $\mathcal{RQ}2$ and discovered that one of the most significant and influential factors in the image prediction process of embedded CNNs is the last layer, also known as the predictive layer. This layer plays a crucial role since the weights and parameters have already been loaded in the training step on Cloud GPU and AIaaS. As depicted in Figure 21, the number of parameters in the predictive layer is significantly lower than in other layers of the model, except for AlexNet.

We presented the structure of each CNN in Table 4 and evaluated its complexity in terms of the computational cost in FLOPs, considering the Giga multiply–accumulate operations per second (GMac). This measurement is commonly used in state-of-the-art applications to determine the number of floating-point operations a CNN imposes on the hardware, as reported in (ZHANG et al., 2018; MA et al., 2018; MOREIRA et al., 2022).

Thus, the number of operations performed by a CNN on the hardware is what defines its complexity.

We evaluated the size of CNN models and the amount of disk storage needed when analyzing AlexNet, DenseNet, ResNet, and SqueezeNet. The latter three models required more disk storage, with AlexNet consuming approximately 217 MB, DenseNet 27.1 MB, and ResNet 42.7 MB. Additionally, these models had a higher general complexity, requiring more parameters. On the other hand, SqueezeNet demanded less disk storage, with only approximately 2.83 MB needed. Due to its fewer parameters and less complex design, it is intended for use on devices with limited computing resources.

Table 4 – Information about complexity for each CNN.

		Previous Layers		Predictive Layer			
CNN	**Batch Size**	**Complexity GMac**	**Parameters**	**Complexity GMac**	**Parameters**	**Hidden Units**	**Model Size (MB)**
AlexNet	32	0.71	2.47 M	0.055	54.55 M	4096	217.00
DenseNet	32	2.88	6.95 M	0.0	0.003 M	1024	27.10
MobileNet	32	0.32	2.23 M	0.0	0.004 M	1280	8.72
ResNet-18	32	1.82	11.18 M	0.0	0.002 M	512	42.70
ShuffleNet	32	0.15	1.26 M	0.023	0.003 M	1024	4.95
SqueezeNet	32	0.74	0.74 M	0.0	0.002 M	512	2.83

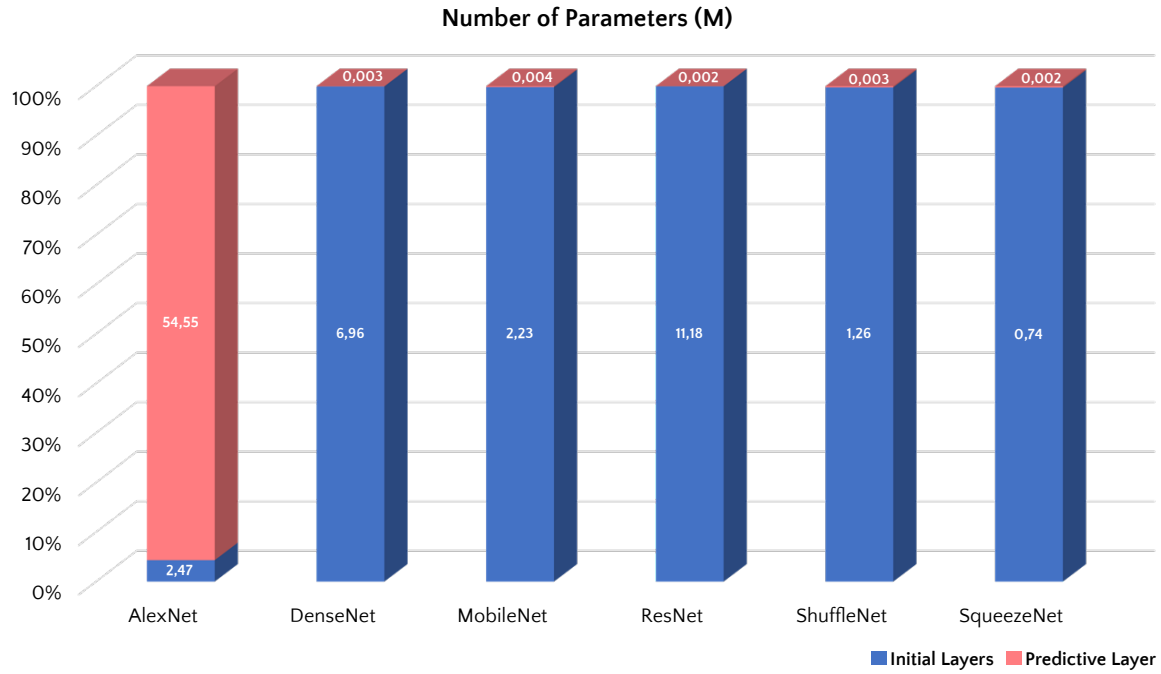


Figure 21 – Parameter Distribution for each CNN.

$\mathcal{RQ}3$: Which CNN takes the shortest time to predict an X-ray image?

To answer $\mathcal{RQ}3$, we analyzed the time it took for each CNN to predict the classes in our study case. As shown in Figure 20, we found that SqueezeNet was the fastest, requiring an average time of 0.34 seconds. This is because SqueezeNet is a lightweight architecture designed for devices with limited computational resources. Although MobileNet and ShuffleNet are also lightweight architectures, they require more time to make predictions: 9.66 seconds and 3.73 seconds, respectively. We observed that among the lightweight CNNs evaluated (MobileNet, ShuffleNet, and SqueezeNet), MobileNet and ShuffleNet have more parameters in the predictive layer. Additionally, MobileNet and ShuffleNet use an inverted residual block in their structure. Therefore, we can infer from this result that more parameters in the predictive layer and architectural structures require more hardware operations and, consequently, a longer time to make predictions. Furthermore, we conducted a hypothesis test with a confidence interval of 95% and found that the average prediction time is statistically equivalent across all classes. Hence, the prediction time is the same regardless of the class being considered.

$\mathcal{RQ}4$: Do more complex CNNs require more time to predict an X-ray image?

To answer $\mathcal{RQ}4$, we observed that AlexNet, DenseNet, and ResNet, which are more computationally expensive and not designed for low-resource devices, require less time to classify an image. Our experiments showed that despite arguments in favor of lightweight CNNs built to be embedded, MobileNet took approximately 24.34 times longer than AlexNet to classify an image. Additionally, when comparing MobileNet with SqueezeNet, we found that it took approximately 28.40 times more time to predict.

Therefore, by answering $\mathcal{RQ}3$ and $\mathcal{RQ}4$, we determined that the CNN requiring the least amount of time to classify an X-ray image is SqueezeNet. However, it should be kept in mind that other CNNs, which are typically associated with high computational demands and memory usage, can also be employed in low-cost devices where prediction time is a significant concern.

$\mathcal{RQ}5$: Considering RAM memory, what is the computational overhead that CNNs impose on the low-cost device hardware?

To answer $\mathcal{RQ}5$, we evaluated the computational overhead that CNNs impose on the hardware on which they are embedded. In the process of measuring memory consumption during prediction, we found that the CNN that consumes the most RAM is DenseNet, at an average of approximately 17.35%. We also conducted a fair comparison of the CNNs in terms of memory consumption, and found that the CNN that consumes the least RAM is ShuffleNet, at about 9.8405%, followed by SqueezeNet, which demands about 9.8431%.

We evaluated all CNNs using the cumulative distribution function (CDF) of memory usage. We determined the 95th percentile of the sample, based on the normal probability distribution for the memory consumption of all evaluated CNNs. As shown in Figure 22, DenseNet's RAM consumption was less than 20.17% in 95% of the samples, followed by 15.91% for MobileNet, 12.12% for ResNet-18, 10.40% for SqueezeNet, and 10.15% for ShuffleNet. Therefore, in response to $\mathcal{RQ}5$, ShuffleNet consumes the least amount of RAM, and there is a direct relationship between the RAM consumption and the size of the generated model. Additionally, it can be noted that the RAM consumption measured in the experiment does not impose a significant overhead on the hardware that runs the embedded CNN. The CDF analysis shows that ShuffleNet and SqueezeNet, being lightweight CNNs, require less RAM when embedded, whereas MobileNet requires more computing resources and consumes 15.91% of RAM in 95% of the samples.

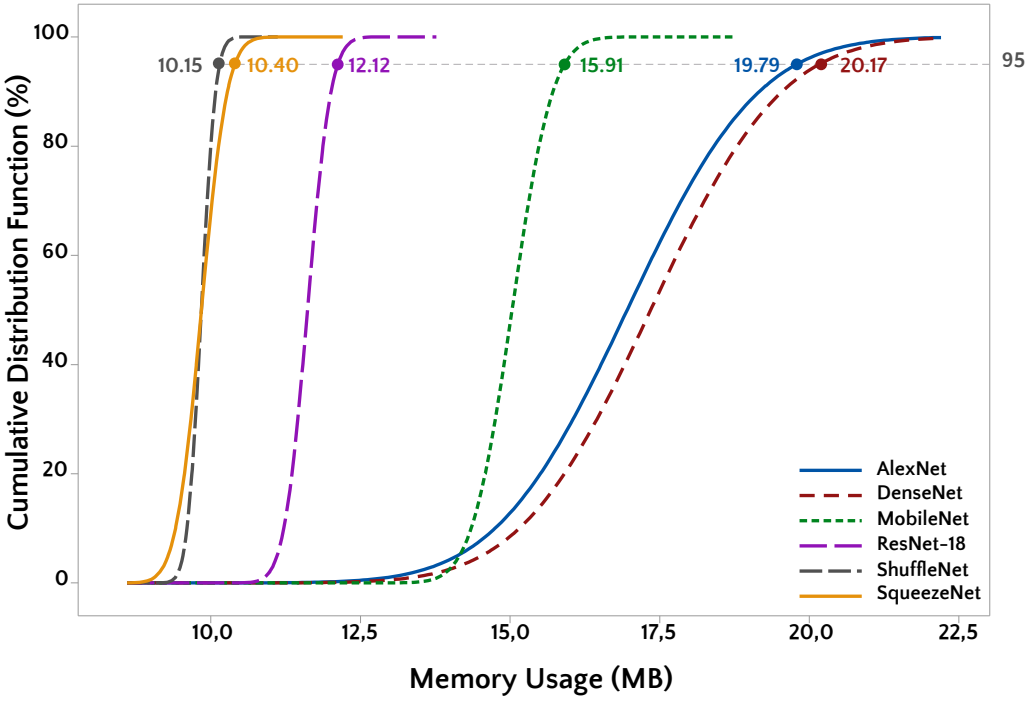


Figure 22 – CDF for Memory Usage considering each CNN evaluated.

$\mathcal{RQ}6$: What criteria should be considered to embed a CNN on a low-cost device for image classification?

We evaluated the classification performance of each CNN on the test set (see Table 5) and found that the DenseNet architecture performed the best, achieving an accuracy of 99.34%, followed by ShuffleNet with 99.21% and MobileNet with 99.08%. However, simply analyzing classification performance is not sufficient to answer $\mathcal{RQ}6$. We need to consider the answers from previous $\mathcal{RQ}s$.

Our case study shows that it is possible to embed CNN models generated by non-lightweight architectures. However, before embedding, we need to check the hardware configurations and available computing resources. Our results demonstrate that while DenseNet achieved the best classification performance, it required more memory to perform the prediction and consumed significantly more memory space compared to lightweight CNNs.

Table 5 – CNN Performance.

CNN	**Accuracy (%)**	**Precision (%)**	**Recall (%)**	**F1-Score (%)**
AlexNet	97.76	96.78	97.42	97.09
DenseNet	99.34	99.22	98.76	98.98
MobileNet	99.08	98.84	98.49	98.66
ResNet	99.01	98.31	98.63	98.47
ShuffleNet	99.21	98.86	98.98	98.91
SqueezeNet	97.83	97.82	95.95	96.86

To address $\mathcal{RQ}6$, we must take into account the available RAM and the expected prediction time on a low-cost device. In this context, our case study indicates that when the prediction time is quick and efficient, using a CNN model with less RAM, such as SqueezeNet or ShuffleNet, is a more suitable option compared to MobileNet lightweight or deeper models.

In Table 6, we compared our best result with other state-of-the-art techniques in the literature in terms of model size and prediction time. Our result is better than the best state-of-the-art technique reported in the literature. Additionally, our AIaaS Architecture enables timely prediction, with a time of 0.34 seconds and a model size suitable for a low-cost device. Our method is approximately 20 times faster than the approaches proposed by Rangarajan e Ramachandran (2021) and Hosny et al. (2021).

Finally, in Table 6 we compared our best result with other state-of-the-art techniques in the literature in terms of model size and prediction time. Our result is better than the best state-of-the-art technique reported in the literature. Additionally, our AIaaS Architecture enables timely prediction, with a time of 0.34 seconds and a model size suitable for a low-cost device. Our method is approximately 20 $\times$ faster than the approaches proposed by Rangarajan e Ramachandran (2021) and Hosny et al. (2021).

Table 6 – Comparison with literature.

Approach	Method	Model Size	Prediction Time	Accuracy (%)
Rangarajan e Ramachandran (2021)	MobileNet-V2	9.84 MB	1.692 s	97.9
	Xception	95.3 MB	2.282 s	98.1
	VGG-16	106 MB	3.627 s	98.6
Hosny et al. (2021)	LBP Random Forest Logistic	3 MB	10 s	99.3
Our Approach	**SqueezeNet**	**2.93 MB**	**0.34 s**	**97.83**
	DenseNet	**27.1 MB**	**1.64 s**	**99.34**

Following these experiments, we confirmed that the AIaaS Architecture is capable of supporting COVID-19 diagnosis using X-ray images. Based on these experiments, we believe that our AIaaS Architecture can be used to provide AI capabilities using low-cost edge computing. In Table 7, we summarize all Research Questions ($\mathcal{RQ}s$) and findings.

Table 7 – Research questions and summary findings for experiments exploring the capabilities of low-cost devices.

Research Question ($\mathcal{RQ}$)	**Finding**
$\mathcal{RQ}1$ Low-cost devices are able to timely predict X-ray images?	• CNN models can be successfully embedded in low-cost devices.
$\mathcal{RQ}2$ Among the evaluated CNNs, which ones are more complex in terms of computational cost?	• AlexNet, DenseNet, and ResNet.
$\mathcal{RQ}3$ Which CNN takes the shortest time to predict an X-ray image?	• SqueezeNet ($\approx$ 0.34 s).
$\mathcal{RQ}4$ Do more complex CNNs require more time to predict an X-ray image?	• Not. • AlexNet, DenseNet, and ResNet are more expensive and require less time.

Table 7: (continued).

$\mathcal{RQ}5$ Considering RAM memory, what is the computational overhead that CNNs impose on the low-cost device hardware?	• RAM consumption does not represent a significant imposition of overhead on the hardware that runs the embedded CNN.
$\mathcal{RQ}6$ What criteria should be considered to embed a CNN on a low-cost device for image classification?	• RAM available and time to perform the prediction.

After identifying the gaps in the state of the art, we evaluated the suitability of AIaaS Architecture for supporting COVID-19 diagnosis. Through our case study, we observed that lightweight CNNs, despite their simplicity, are well-suited for deployment on low-cost devices, while deeper CNNs can be used for prediction tasks on these devices. We also found that heavy models must be trained on powerful computing devices in the cloud and pulled to low-cost devices for use as a predictive service. Our approach achieved the best results, requiring only 2.83 MB of memory and taking less than a second to perform the prediction, outperforming previous works.

5.2 Exploring the Capabilities on Conventional Device and Model Store

To evaluate the capabilities of delivering cognitive services through our AIaaS Architecture and assess the suitability of our solution, we propose a general-purpose image classification task and evaluate its performance, speed, and resource utilization. Our study demonstrates that edge intelligence can be effectively realized by leveraging the concept of the AI Model Store, which serves as an innovative abstraction layer between cognitive service pipelining and the business rules that it supports.

This comprehensive analysis highlights the potential challenges of integrating AI services in a distributed and scalable manner. In this context, AI models can be seamlessly deployed across various heterogeneous edge devices via the AIaaS cognitive service-acquisition process. Once an AI model is embedded in an edge device, it can be applied to various domain applications. The goal of these experiments was to answer the following $\mathcal{RQ}s$.

- ❏ $\mathcal{RQ}1$: How can cognitive services be efficiently retrieved and utilized by heterogeneous edge devices with limited computational resources?
- ❏ $\mathcal{RQ}2$: How effective is general-purpose image classification on edge devices compared to traditional cloud-based solutions in terms of classification performance?
- ❏ $\mathcal{RQ}3$: What is the impact of deploying image classification models on edge versus cloud platforms in terms of the prediction time?
- ❏ $\mathcal{RQ}4$: How does memory usage for image classification models differ when deployed on edge devices compared to cloud-based environments?
- ❏ $\mathcal{RQ}5$: How does the edge-intelligence control framework facilitate communication and manage the lifecycle of AI models in a distributed environment?

5.2.1 Experiments

We developed a cross-platform method that enables edge intelligence and addresses the challenge of retrieving cognitive services from the internet for heterogeneous edge devices with limited computational resources. Our solution includes a general-purpose image classification application that enables users to select an appropriate AI model from the AIaaS Model Store, based on their specific requirements. This selection process involves determining the most suitable model in the context of the application. Our Proof of Concept (PoC) focuses specifically on a software layer that facilitates communication between the edge device application and AIaaS Model Store. The AIaaS architecture manages a variety of AI models and their life-cycles, offering models on a pay-as-you-go basis to meet the demands of the edge devices.

$\mathcal{RQ}1$: How can cognitive services be efficiently retrieved and utilized by heterogeneous edge devices with limited computational resources?

To answer $\mathcal{RQ}1$, we implemented a software layer to facilitate communication between the edge device application and AI Model Store using Python, PyTorch, and the React Native framework. To verify the process of downloading AI models to edge devices, we preloaded the AIaaS architecture with four trained AI models as a part of the experimental setup for this study.

The edge-intelligence control framework is a structured system designed to manage and facilitate the deployment of AI models on edge devices. Its primary goal is to ensure that cognitive services such as image classification can be efficiently accessed and utilized on devices with varying computational capabilities. The framework integrates the following components and processes:

- **User Authentication and Registration:** the framework begins with a user authentication process. Users must register and verify their login credentials before gaining access to cognitive services. This step ensures that only authorized users can utilize edge intelligence resources.
- **Model Retrieval and Management:** once authenticated, users can access a list of AI models tailored to their profiles. The framework checks whether the required model is already available locally on the edge device. Otherwise, it retrieves the model from the AI Model Store within the AIaaS architecture. This process involves querying the centralized store to obtain the necessary model and storing it locally in the device for subsequent use.
- **Application Integration:** after acquiring the model, the edge device can seamlessly integrate and use it for specific tasks such as image classification. The framework supports the ability of the application to handle tasks efficiently by ensuring that appropriate models are available and up-to-date.
- **Operational Workflow:** the framework outlines the workflow for model selection, retrieval, and deployment, guiding the system through a series of steps to ensure smooth operation. This includes checking model availability, retrieving models when necessary, and facilitating their use in edge applications.

Figure 23 shows the workflow of the edge intelligence control framework, illustrating the logical progression of operations, which encompasses user registration, model retrieval, and application integration and offers a coherent visualization of the processes involved in administering cognitive services on edge devices.

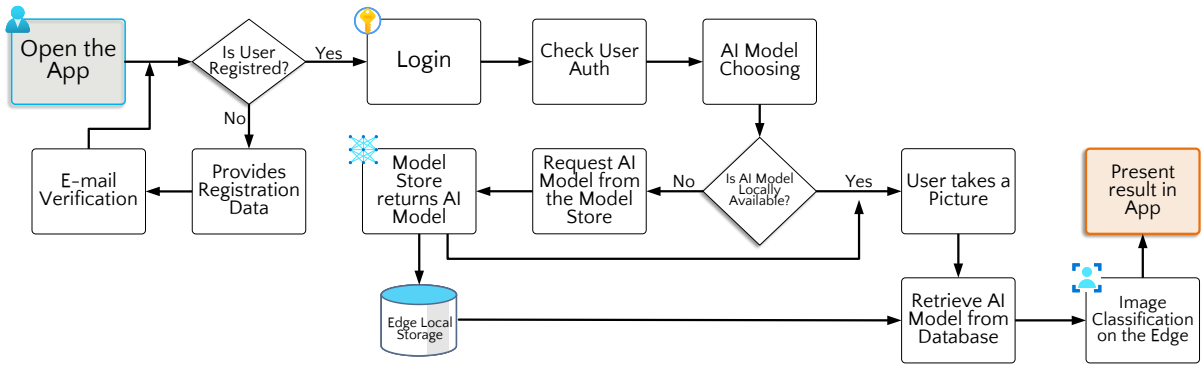


Figure 23 – Control Framework for edge-intelligence enablement.

Our PoC focuses on using image classification to support precision agriculture and offers significant advantages by demonstrating how edge-based image classification can enhance crop management. By utilizing AI models on edge devices, farmers can perform real-time analyses of crop conditions, such as assessing coffee bean quality directly in the field. This approach reduces the need for costly centralized infrastructure, speeds

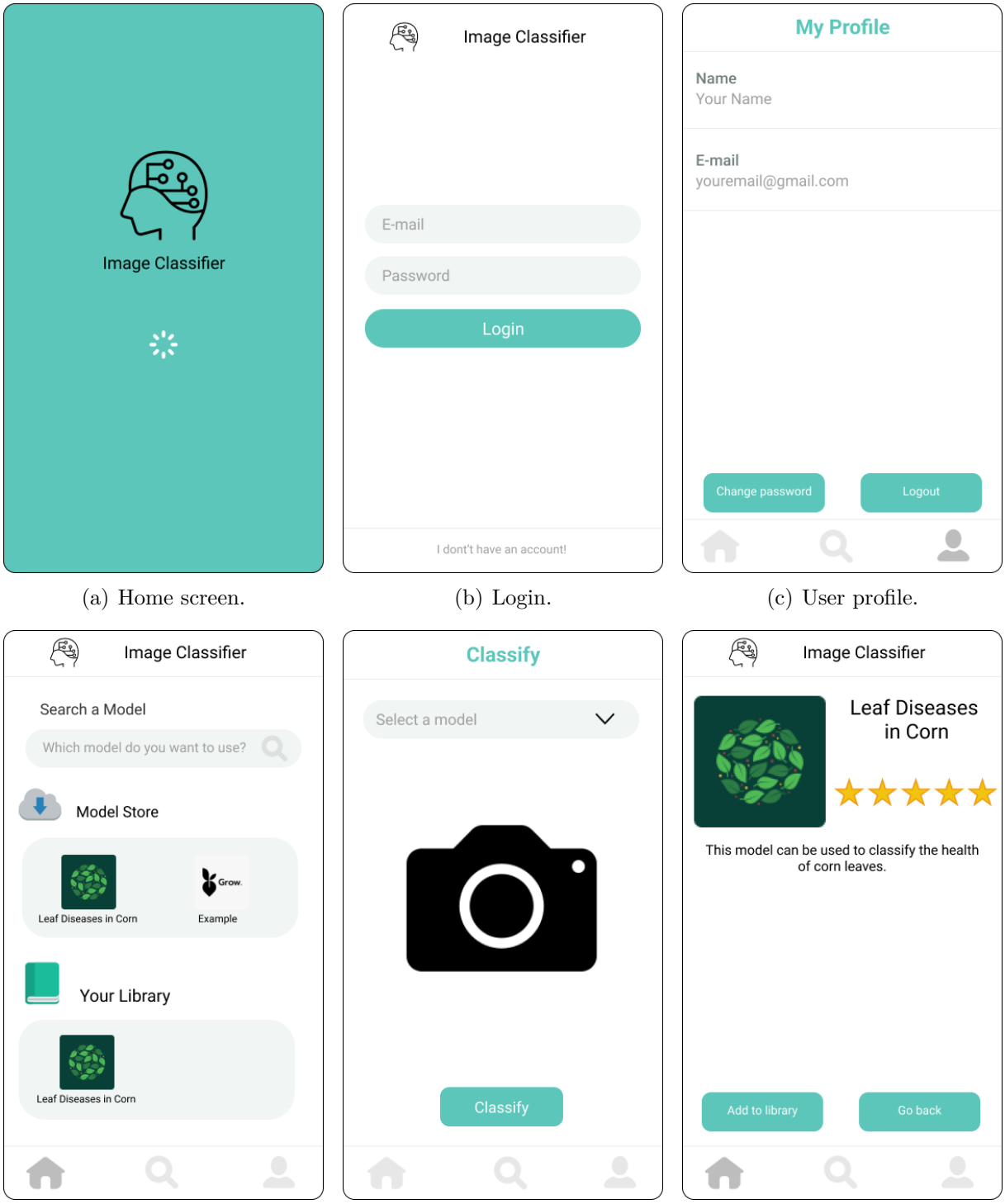
up decision-making processes, and provides actionable insights that can improve crop yields and quality while making advanced technology accessible even in resource-limited settings.

The images utilized in this PoC are sourced from the USK-Coffee dataset (FEBRIANA et al., 2022)[2]. This dataset contains 8,000 images of green coffee beans divided into four distinct categories: peaberry, longberry, premium, and defect. Each category comprised 2,000 images with a resolution of 256 × 256 pixels. Figure 24 presents the selection of images from the dataset and illustrates examples from each class.

Figure 24 – Image samples for each class obtained from the USK-Coffee dataset.

We developed a mobile device application to showcase the functionality and usability of our edge-intelligence framework. The application screens are shown in Figure 25, including the initial screen presented when the user opens the application during loading of its dependencies in Figure 25(a). Authentication, as shown in Figure 25(b), requires the user to log in or create an account to access the application. The profile in Figure 25(c), Home, in Figure 25(d), represents the main interface, showing purchased models and the AI Model Store, respectively. Finally, we classified the image in Figure 25(e) and the Model Store in Figure 25(e).

[2] Available at: <https://comvis.unsyiah.ac.id/usk-coffee/>



(a) Home screen. (b) Login. (c) User profile.

(d) Main interface. (e) Image classification. (f) Model Store.

Figure 25 – Screens of proposed mobile application.

$\mathcal{RQ}2$: How effective is general-purpose image classification on edge devices compared to traditional solutions in terms of classification performance?

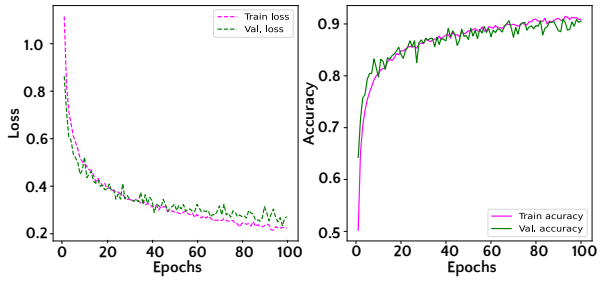
To answer $\mathcal{RQ}2$, we evaluated four CNNs: AlexNet (KRIZHEVSKY; SUTSKEVER; HINTON, 2012), EfficientNet (TAN; LE, 2019), MobileNet (HOWARD et al., 2017), and

ResNet (HE et al., 2016b) for their performance in image classification. We utilized a 5-fold cross-validation scheme and initialized each CNN with pretrained weights from ImageNet. The hyperparameters for the learning rate, batch size, and optimizer were set as described previously by Pereira Neto et al. (2023) and summarized in Table 8. Using Python (version 3.10) and the PyTorch deep learning framework, we resized the images to 224 × 224 pixels and performed random rotations (angles between -15° and 15°) and both vertical and horizontal flips.

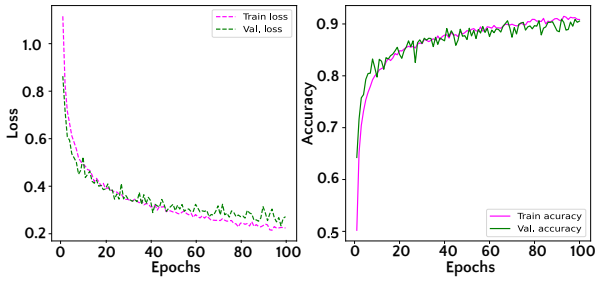
Table 8 – Optimized hyperparameter values defined by Pereira Neto et al. (2023).

Architecture	**Batch Size**	**Learning Rate**	**Accuracy Validation**
AlexNet	16	0.00001	0.9169
EfficientNet	16	0.001	0.9469
ResNet-50	64	0.0001	0.9475
MobileNet	128	0.001	0.9606

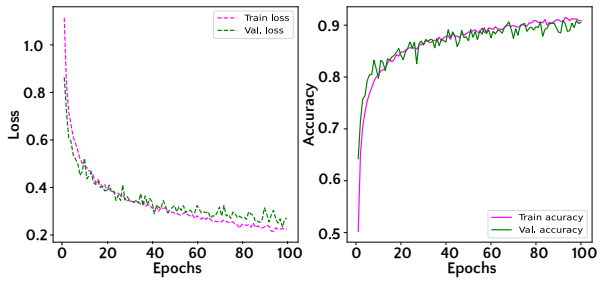
Figure 26 illustrates the learning behaviour during the training phase, focusing on loss and accuracy values from the second iteration of the k-fold cross-validation. The observed trends indicate that the training process effectively avoided overfitting, thereby preserving the generalization capabilities of each evaluated CNN.



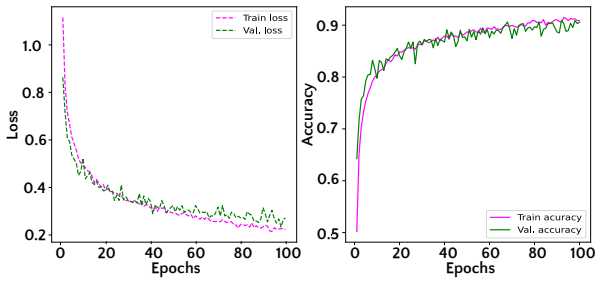
(a) AlexNet.



(b) EfficientNet.



(c) MobileNet.



(d) ResNet.

Figure 26 – Evolution of accuracy and loss values for each CNN.

After training the models, AlexNet, EfficientNet, MobileNet, and ResNet-50, we obtained their accuracies in Table 9. The ResNet-50 model achieved the highest accuracy of 95.11% after the 5-fold training. These models were stored in the AIaaS Architecture

and made available for intelligent edge applications, allowing edge devices to download and perform predictions seamlessly at the edge.

Table 9 – AI Models classification performance.

Model	**Accuracy (%)**	**Precision (%)**	**Recall (%)**	**F1-Score (%)**
AlexNet	90.15	90.22	90.23	90.14
EfficientNet	92.50	92.74	92.51	92.48
MobileNet	91.76	91.92	91.72	91.72
ResNet-50	**95.11**	**95.12**	**95.13**	**95.10**

To demonstrate the effectiveness of the AIaaS architecture for enabling intelligence at the edge, we not only prepared AI models for the AI Model Store but also achieved results that surpassed the current state-of-the-art. This result enabled us to provide models for precision agriculture for a wide range of users across various platforms. Furthermore, we compared our top accuracy results with other studies in the literature. Table 10 presents these comparative outcomes using the same USK-Coffee dataset in therms of accuracy.

Table 10 – Comparison with literature.

Method	**Accuracy (%)**
Febriana et al. (2022)	81.31
Pereira Neto et al. (2023)	88.44
Islamy et al. (2023)	91.50
Our approach	**95.11**

We evaluate the performance of our general-purpose image classification and validate AIaaS architecture service delivery, we designed an experimental setup involving three distinct locations. Specifically, we tested the ability of the AI model to classify coffee beans using the smartphone itself, as well as when the captured image was sent to a local High Performance Computing (HPC) and when it was sent over the Internet, referred to as remote HPC. We used a Motorola Edge 30 Pro device with Android version 14, an octa-core 2.2GHz processor, and 12GB of RAM for the instance and execution of the application. Additionally, both Local and Remote HPC machines have an Intel(R) Core(TM) i5-4430 CPU @ 3.00GHz, 32GB RAM, and an NVIDIA RTX4060Ti 8GB graphics card was used to train the AI models before publishing them to the AI Model Store.

The prediction time and memory usage across the models implemented in the AIaaS architecture were evaluated. This assessment aimed to gauge the efficiency and resource consumption of each model when making inferences about edge devices. Through the measurement of the time required for predictions and the memory footprint, valuable insights were obtained regarding the practicality and scalability of deploying these models in real-world situations, particularly in environments with limited resources.

$\mathcal{RQ}3$: What is the impact of deploying image classification models on edge versus cloud platforms in terms of the prediction time?

To answer $\mathcal{RQ}3$, we conducted experiments to measure the response time required to execute specific actions by using our general-purpose image classification system.We tested three different approaches: prediction directly on a mobile device, local HPC, and remote HPC. Table 11 presents the estimated average times for the image classification task using each AI model in different locations.

Table 11 – Time consumption (ms) in prediction task.

	AlexNet			**EfficientNet**			**MobileNet**			**ResNet-50**		
	Edge	**Local HPC**	**Remote HPC**	**Edge**	**Local HPC**	**Remote HPC**	**Edge**	**Local HPC**	**Remote HPC**	**Edge**	**Local HPC**	**Remote HPC**
Mean (ms)	110.5	424.8	1310.5	185	879.5	1417	101.1	278.6	1311.5	177.9	854.1	1334.7
Std. Deviation	28.441	5.432	35.740	24.922	14.017	108.158	19.473	11.442	164.662	27.586	12.696	126.900

As illustrated in Figure 27, for the same instance of the captured image and AI model, our PoC application was capable of predicting coffee bean quality approximately four times faster than the local HPC and nine times faster than the remote HPC.

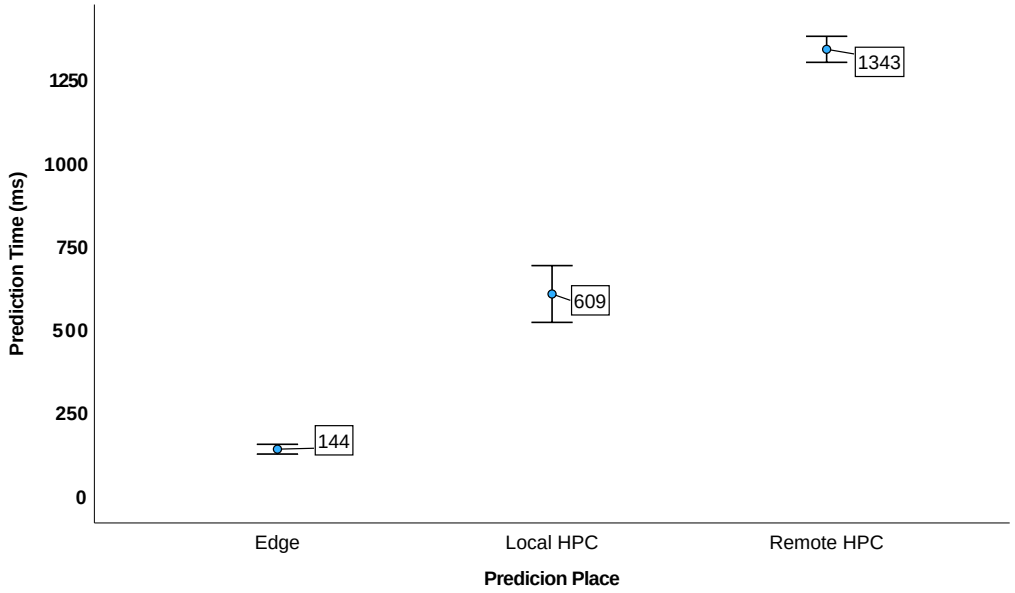


Figure 27 – Prediction time comparison.

When embedding the AI model into a mobile application, the PlayTorch tool optimizes the model natively on mobile devices by reducing its parameters, using the same trained model for the mobile application. This faster performance on edge devices compared with Local HPC can be attributed to reduced latency and the absence of network delays. Remote HPC was included for comparison, as predictions in a client-server model often suffer from lower performance due to Internet transit overhead. These findings support

our hypothesis that edge devices equipped with pre-trained AI models retrieved from the AIaaS architecture can effectively handle requests in a timely manner.

> **$\mathcal{RQ}4$: How does memory usage for image classification models differ when deployed on edge devices compared to cloud-based environments?**

By answering $\mathcal{RQ}4$, our evaluation of memory usage (Figure 28) indicated that the AlexNet model embedded in Local HPC exhibited the lowest memory usage during the image prediction tests compared with other AI models, such as ResNet-50, which required the highest memory usage for the same tests.

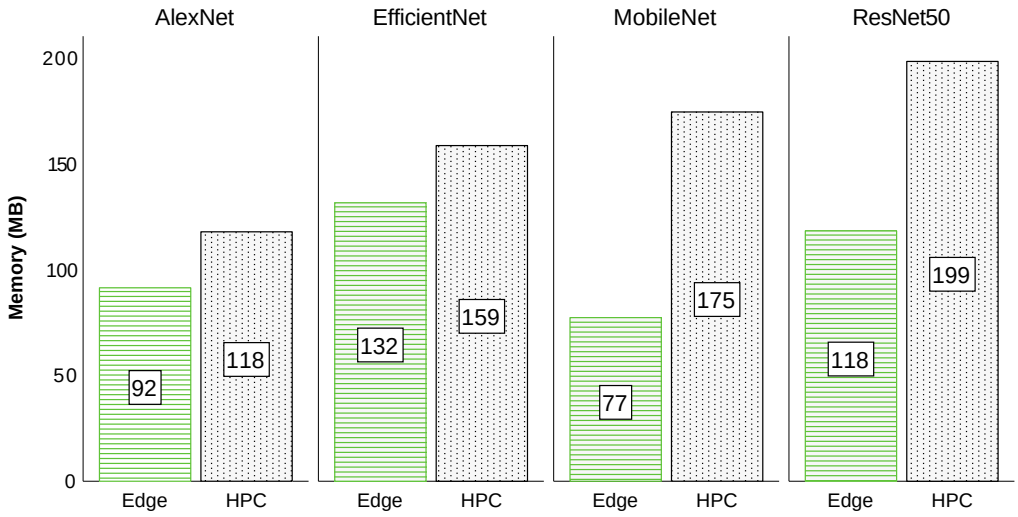


Figure 28 – Memory usage.

For the prediction task on an edge device, we attributed the performance efficiency to the optimization process tailored for mobile devices. We performed this optimization for each model before deployment, significantly enhancing efficiency. In our comparisons, the same AI model required 160MB of storage on HPC; however, after PlayTorch optimization for the mobile device, it only required 100MB. This represents a nearly 40% reduction in memory usage, underscoring the practical and significant impact of optimization on the overall system performance.

> **$\mathcal{RQ}5$: How does the edge-intelligence control framework facilitate communication and manage the lifecycle of AI models in a distributed environment?**

Our experimental results indicate that the edge-intelligence control framework effectively facilitates communication and manages the lifecycle of AI models in a distributed

environment, addressing $\mathcal{RQ}5$. This framework provides a structured approach to model deployment and management, ensuring seamless communication between edge devices and the AI Model Store. It enables for the efficient retrieval and deployment of models tailored to specific applications while overseeing the entire lifecycle of AI models, including storage, versioning, and updates. This ensures that edge devices consistently access the latest and most effective models with minimal latency. By embedding pretrained models optimized for edge devices, the framework reduces computational overhead and enhances prediction speed.

Following these experiments, we confirmed that the AIaaS Architecture effectively supports intelligent image classification on edge devices and manages AI model deployment and retrieval through a Model Store. These findings demonstrate the capability of our AIaaS Architecture to deliver AI functionalities using low-cost edge computing. In Table 12, we summarize the $\mathcal{RQ}s$ and findings.

Table 12 – Research questions and summary findings for experiments exploring the capabilities of conventional devices and the model store.

Research Question ($\mathcal{RQ}$)	**Finding**
$\mathcal{RQ}1$ How can cognitive services be efficiently retrieved and utilized by heterogeneous edge devices with limited computational resources?	• AIaaS architecture provides an efficient framework for retrieving and utilizing cognitive services. • Enabling seamless communication between edge devices and AI Model Store.
$\mathcal{RQ}2$ How effective is general-purpose image classification on edge devices compared to traditional cloud-based solutions in terms of classification performance?	• General-purpose image classification on edge devices performs comparably to cloud-based solutions. • ResNet-50 achieved a high accuracy of 95.11% on edge devices.

Table 12: (continued).

$\mathcal{RQ}3$ What is the impact of deploying image classification models on edge versus cloud platforms in terms of the prediction time?	• Edge-based predictions are significantly faster. • Edge is approximately 4× faster than local HPC • Edge is 9× faster than remote HPC for the same models. • This results demonstrates the efficiency of edge computing in reducing prediction time.
$\mathcal{RQ}4$ How does memory usage for image classification models differ when deployed on edge devices compared to cloud-based environments?	• Memory usage is lower on edge devices compared to cloud-based environments. • Edge devices required 100MB of memory for models, whereas cloud-based HPC required 160MB.
$\mathcal{RQ}5$ How does the edge-intelligence control framework facilitate communication and manage the lifecycle of AI models in a distributed environment?	• The edge-intelligence control framework enables seamless communication and lifecycle management of AI models. • Ensuring that edge devices can access the latest models with minimal latency.

Chapter 6

Service Management Evaluation

In this chapter, we evaluate the elements of our AIaaS architecture by focusing on platform management, model training, and dataset management (Section 6.1). Our evaluation addresses experiments to understand the performance and scalability using FL to assess how different AI applications are delivered to heterogeneous devices and the effect of the AIaaS architecture on reducing operational costs compared with traditional centralized approaches for computer vision applications.

Furthermore, we examined the model optimizer and optimization functionalities within our AIaaS architecture (Section 6.2). This involves experimenting with optimization techniques such as grid search, random search, Bayesian optimization, and evolutionary strategies to fine-tune the AI model.

6.1 Evaluating the Platform, Model Training, and Dataset Management

We present an evaluation of the platform focusing on model training processes and dataset management strategies. The objective is to assess the platform's ability to handle complex machine learning workflows. This evaluation serves as an important step toward understanding how our system supports scalable and efficient AI model development.

We assessed the platform's performance across various metrics to ensure that it meets the demands of both resource-intensive and lightweight applications. The insights gained from this analysis will inform the best practices for managing datasets and training models within our architecture, ultimately contributing to the advancement of AI capabilities in real-world scenarios. The goal of these experiments is to address the following $\mathcal{RQ}s$.

- ❑ $\mathcal{RQ}1$: How can federated training be implemented seamlessly?
- ❑ $\mathcal{RQ}2$: How does the integration of FL into the AIaaS architecture impact the classification performance across different datasets?

- ❏ $\mathcal{RQ}3$: How does the integration of FL into the AIaaS architecture influence efficiency and scalability in resource-constrained environments?
- ❏ $\mathcal{RQ}4$: How can different AI paradigms benefit from AIaaS architecture?

6.1.1 Experiments

We programmed the experiments using Python (version 3.11), PyTorch (PASZKE et al., 2019) and Flower FL framework (BEUTEL et al., 2020). We used the hardware specifications listed in Table 13 to validate our AIaaS in a federated training scenario on top of the Ubuntu 20.04 Long Term Evolution (LTE).

Table 13 – Hardware specifications.

Role	CPU	RAM	GPU
Server and $\mathcal{C}_1$	Intel(R) Core(TM) i5-4430 CPU @ 3.00GHz	32 GB	GeForce RTX 4060 Ti 8GB
$\mathcal{C}_2$	Intel(R) Core(TM) i5-4430 CPU @ 3.00GHz	16 GB	GeForce GTX 1050 Ti 4 GB

We explore the training features of AIaaS, which refers to a cloud-based service model that provides AI capabilities, including model training and deployment, without requiring users to manage the underlying infrastructure. The training functionality within the AIaaS architecture enables users to initiate training of AI models specifically for computer vision tasks. We validated this feature through its application in the medical context, with a focus on aiding the diagnosis of colorectal and breast cancers. The choice of the medical field stems from the need for technological advancements and challenges associated with data privacy. In this scenario, FL is particularly well-suited for examining both the potential of this field and the capabilities of the AIaaS training pipeline.

The first evaluated dataset was biglycan (SILVA NETO et al., 2023). It consists of photomicrographs depicting the immunohistochemical expression of biglycan (BGN) in breast tissue from the pathological archive of the Hospital de Clínicas (HCPA) in Porto Alegre, Brazil. The dataset included a total of 336 images, each 128 × 128 pixels in size, and was divided into two classes: cancer (203 images) and healthy (133 images). Figure 29 shows representative images from the Biglycan dataset for each class.

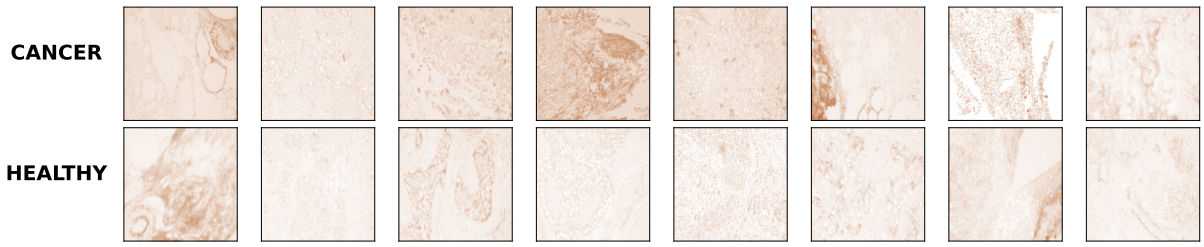


Figure 29 – Image samples from biglycan dataset.

The second dataset used was colorectal (KATHER et al., 2016), which contains 5,000 histological images, each 150 × 150 pixels. Slides were digitized using an Aperio ScanScope (Aperio/Leica Biosystems) at 20× magnification. The dataset included histological samples of human colorectal adenocarcinomas (primary tumors) from the pathology archive of the Institute of Pathology, University Medical Center Mannheim, Heidelberg University, Mannheim, Germany. It featured eight different tissue classes, with each category containing 625 images of hematoxylin and eosin (H&E) stained samples. Figure 30 shows image samples from the colorectal dataset for each class.

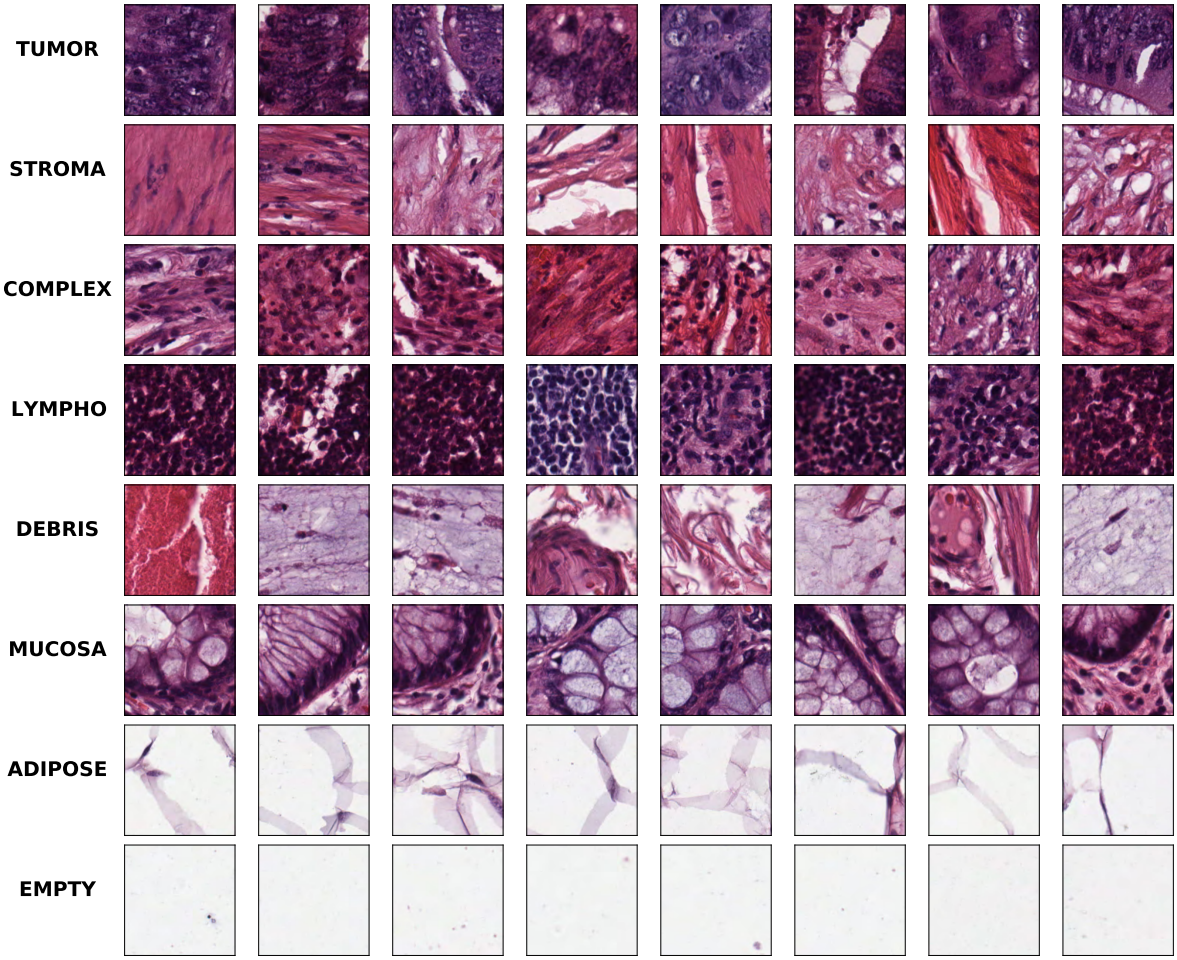
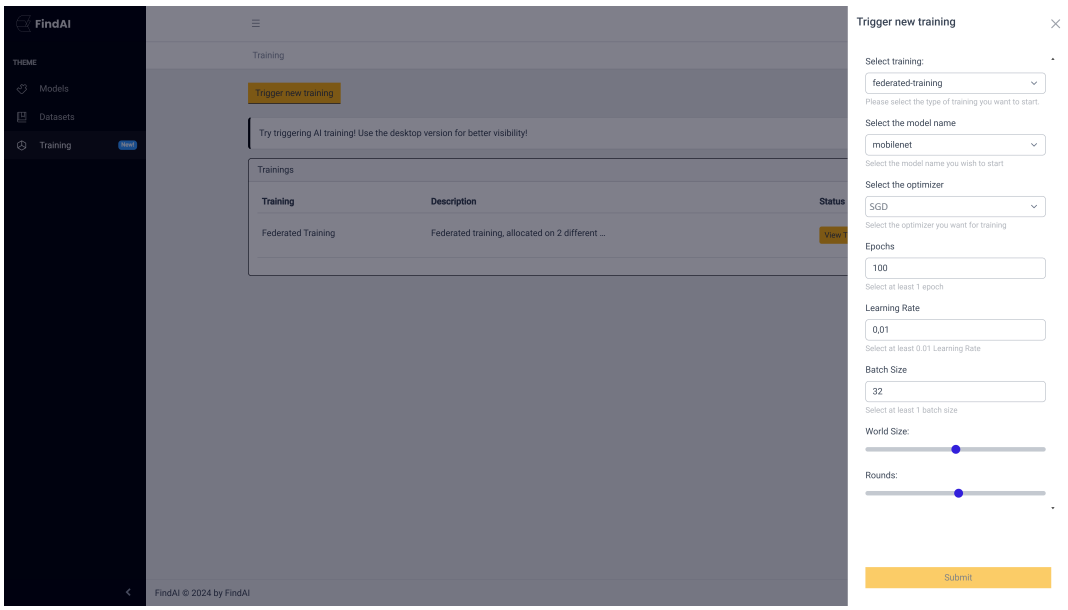


Figure 30 – Image samples from colorectal dataset.

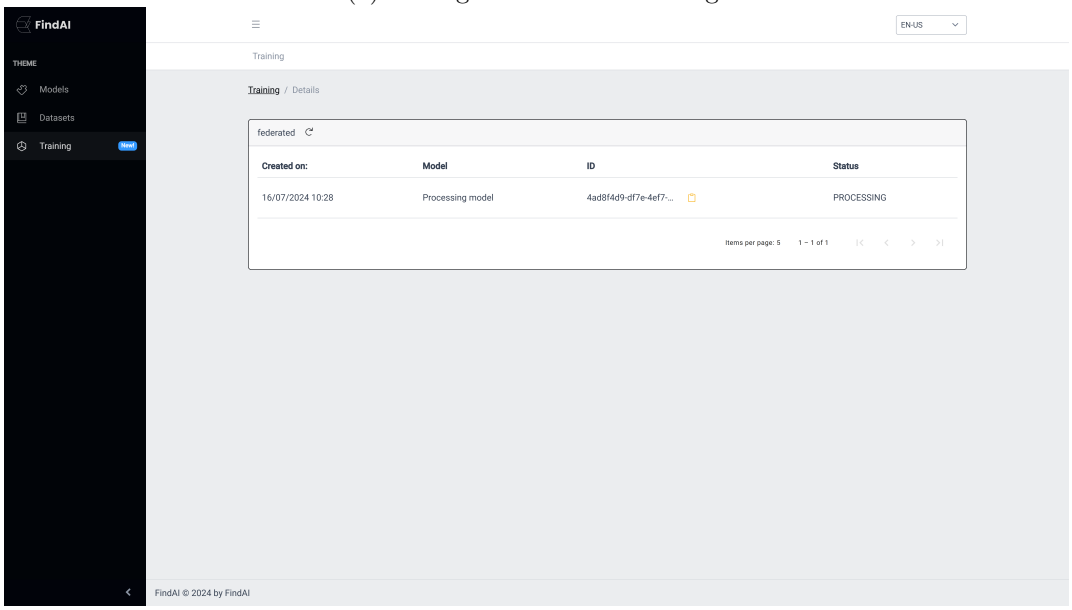
In this evaluation, we choose MobileNet-V2 (SANDLER et al., 2018) was chosen to evaluate FL because of its efficiency in mobile and edge devices where computational and memory resources are limited. Its lightweight design, featuring depth-wise separable convolutions, significantly reduces the parameters and computational load, making it ideal for FL in resource-constrained environments. The compact and efficient architecture ensures effective training and deployment across diverse devices within our AIaaS platform, maintaining scalability and accessibility without sacrificing the model classification performance.

$\mathcal{RQ}1$: How can federated training be implemented seamlessly?

To answer $\mathcal{RQ}1$, we developed a web application using open source technologies. The front-end services were developed using the Angular framework, whereas backend services were implemented using Flask. Through our AIaaS GUI, we enable Internet Service Provider (ISP) managers or users to start training jobs effortlessly through the AIaaS architecture and streamline communication between the Edge Predictor Service and AIaaS management platform, as illustrated in Figure 31. This approach simplifies integration with frameworks such as Flower FL, facilitating the orchestration of distributed model training.



(a) Management model training.



(b) Training details.

Figure 31 – Screens of proposed web application.

Additionally, AIaaS centralized the management and monitoring of federated training through real-time dashboards and reports. These features enable administrators to track the progress, performance, and quality of training, even when they are distributed across multiple devices. The automated and optimized configuration provided by AIaaS dynamically adjusts the model requirements and training processes for diverse devices, making FL accessible and efficient within the AIaaS environment, regardless of device capabilities.

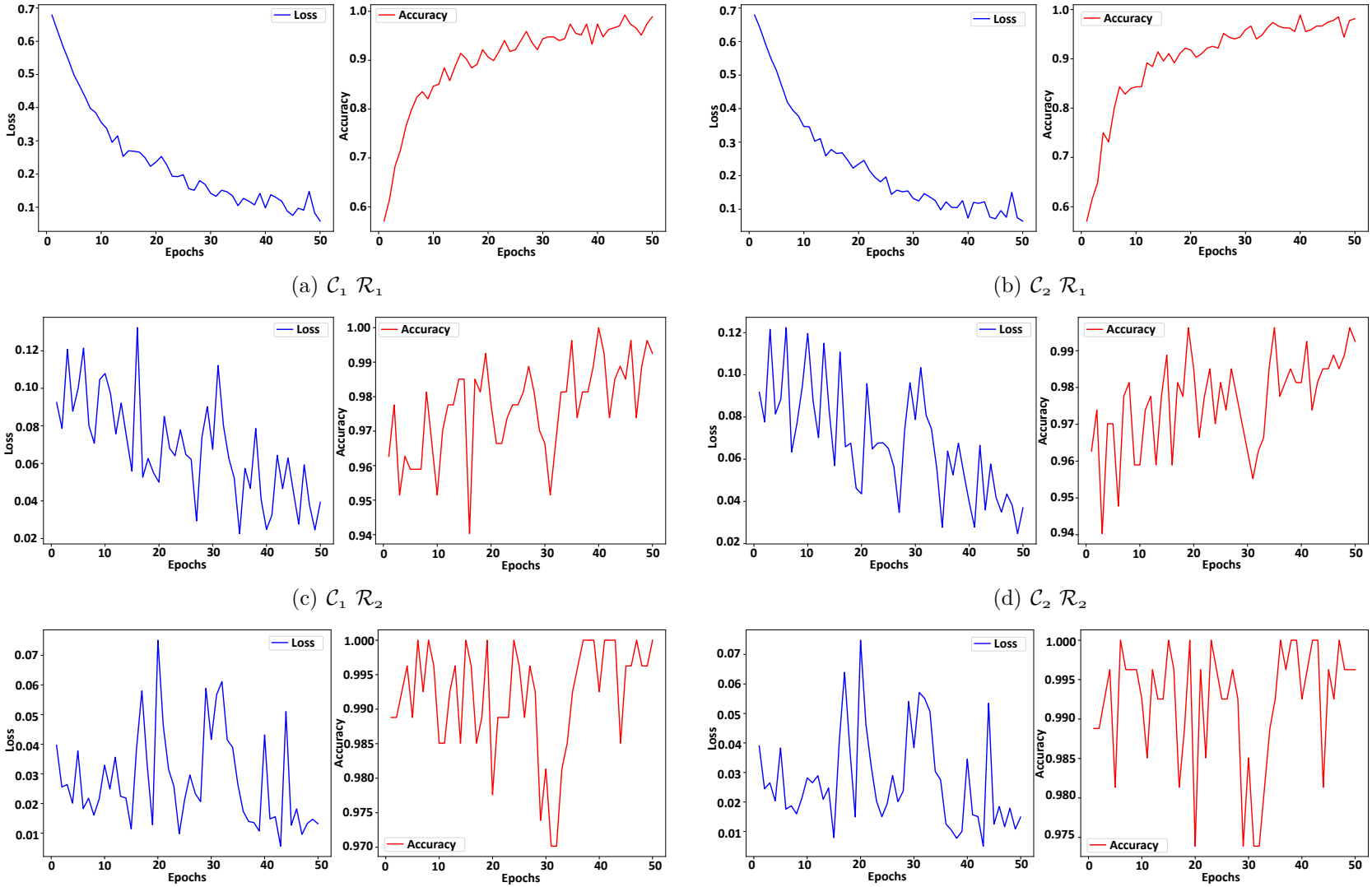
$\mathcal{RQ}2$: How does the integration of FL into the AIaaS architecture impact the classification performance across different datasets?

To answer $\mathcal{RQ}2$, we evaluated the integration of FL into AIaaS and the impact on classification performance across biglycan and colorectal datasets. Initially, the datasets are randomly partitioned into three subsets: 80% for training, 10% for validation, and 10% for testing. Also, we resized all images to 224 × 224 pixels. We implemented the FL training in the AIaaS architecture with a WS of three, involving two clients ($\mathcal{C}_1$ and $\mathcal{C}_2$) and one server. This configuration was designed to facilitate efficient model training and aggregation in a distributed environment. In addition, the training configurations for each dataset were defined by considering distinct parameters to optimize the model performance based on its characteristics and requirements. We set the hyperparameters according to Barbosa et al. (2024) and Nanni, Ghidoni e Brahnam (2020) for the biglycan and colorectal datasets, respectively. Table 14 lists the training configurations used in the experiment.

Table 14 – Training configurations for each dataset. The hyperparameters were defined by Barbosa et al. (2024) and Nanni, Ghidoni e Brahnam (2020).

Dataset	**Epochs**	**Batch Size**	**Learning Rate**	**Optimizer**
Biglycan	50	32	0.0001	SGD
Colorectal	50	32	0.001	SGD

Figures 32 and 33 show the loss and accuracy values for both clients when training the biglycan and colorectal datasets, respectively. The loss curves (in blue) indicate the convergence behavior and stability of the model training on each client's local dataset. Generally, a decrease in loss over epochs indicates effective learning and model optimization. The accuracy (in red) shows the performance of the model, reflecting how well the local models generalize to their respective datasets. By comparing these metrics, we can assess the consistency of the training process for different clients and evaluate the performance of the global model.

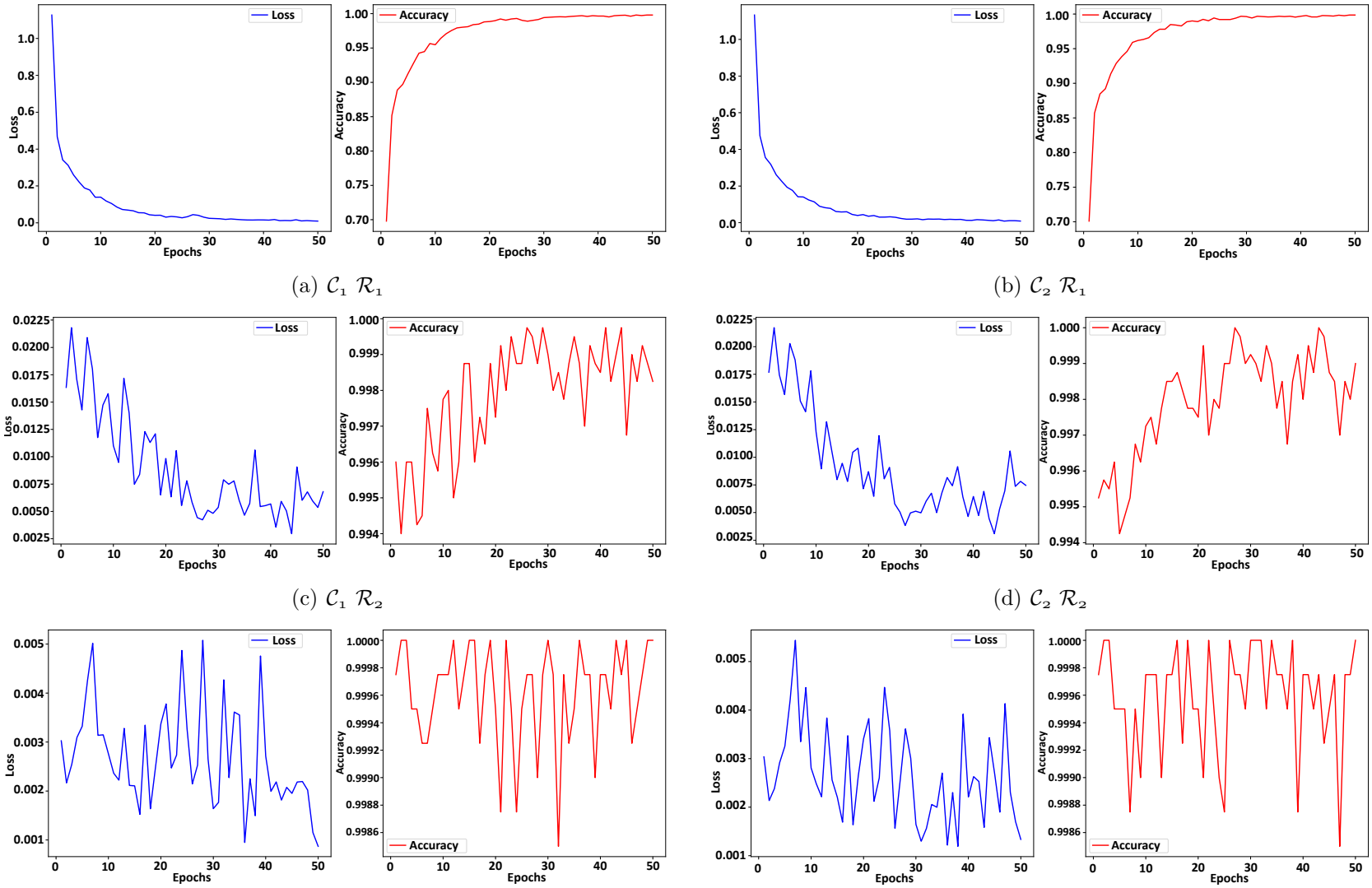


(a) $\mathcal{C}_1$ $\mathcal{R}_1$ (b) $\mathcal{C}_2$ $\mathcal{R}_1$

(c) $\mathcal{C}_1$ $\mathcal{R}_2$ (d) $\mathcal{C}_2$ $\mathcal{R}_2$

(e) $\mathcal{C}_1$ $\mathcal{R}_3$ (f) $\mathcal{C}_2$ $\mathcal{R}_3$

Figure 32 – Evolution of loss and accuracy values during training of biglycan dataset considering WS three.



(a) $\mathcal{C}_1$ $\mathcal{R}_1$

(b) $\mathcal{C}_2$ $\mathcal{R}_1$

(c) $\mathcal{C}_1$ $\mathcal{R}_2$

(d) $\mathcal{C}_2$ $\mathcal{R}_2$

(e) $\mathcal{C}_1$ $\mathcal{R}_3$

(f) $\mathcal{C}_2$ $\mathcal{R}_3$

Figure 33 – Evolution of loss and accuracy values during training of colorectal dataset considering WS three.

The classification performance obtained for each dataset is summarized in Tables 15 and 16 indicating the round, classification metrics, and training time.

Table 15 – Classification performance - biglycan dataset.

Round	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)	Training Time
1	61.76	58.83	56.91	56.33	7.33 s
2	100	100	100	100	84.75 s
3	97.06	96.56	97.56	96.97	149.97 s
Server	**100**	**100**	**100**	**100**	**217.61 s**

Table 16 – Classification performance - colorectal dataset.

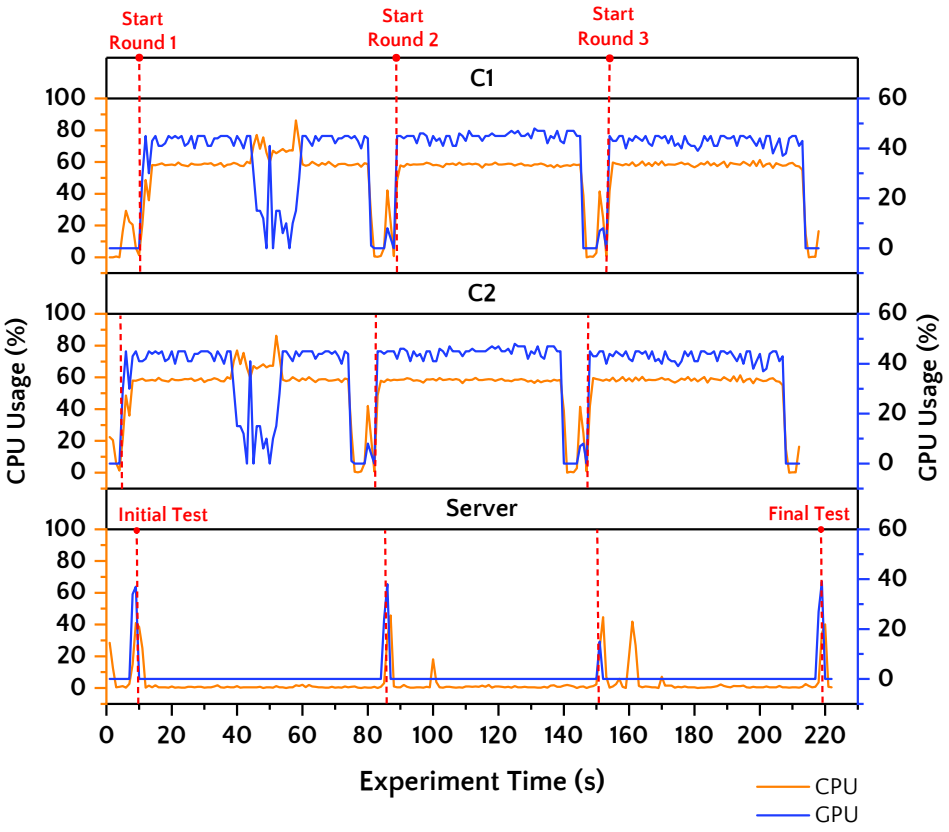
Round	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)	Training Time
1	11.90	98.30	11.90	74.70	33.71 s
2	94.70	94.84	94.70	94.74	1035.66 s
3	95.30	95.51	95.30	95.34	2056.50 s
Server	**96.40**	**96.44**	**96.40**	**96.38**	**3075.65 s**

The results obtained for each evaluated dataset demonstrate consistent improvements across the rounds, indicating the model's ability to learn and generalize effectively with each FL round. The server achieves a high classification performance because it aggregates the knowledge from multiple distributed nodes, and these nodes send their updated model weights back to the server, where they are combined to form a global model. This process leverages the data across nodes, helping the server model to learn more effectively and improve its performance. Thus, the server performance, as shown in the results, tends to be strong, often outperforming individual nodes.

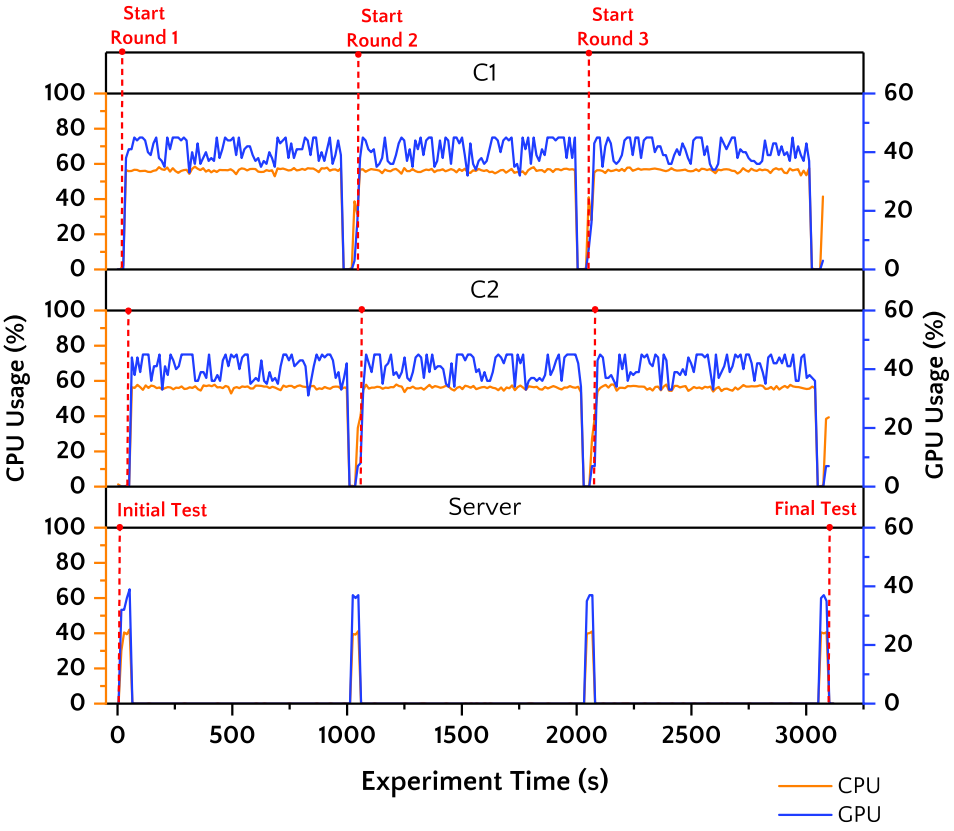
$\mathcal{RQ}3$: How does the integration of FL into the AIaaS architecture influence efficiency and scalability in resource-constrained environments?

Initially, we conducted experiments using the Biglycan dataset, which is relatively small, with a fixed WS of three. This choice was made to establish the baseline performance and resource utilization profile. We then applied the same WS of three to the colorectal dataset to maintain consistency in our initial evaluations.

We analyzed the impact of edge resources CPU and GPU on handling the AIaaS training job triggered by $\mathcal{C}_1$ and $\mathcal{C}_2$ and the server for both Biglycan (Fig. 34(a)), and Colorectal cancer (Figure 34(b)) datasets. The charts (Figure 34) highlights the beginning of each round and the results of the initial and final tests. We observed that the CPU and GPU loads increased at the start of each round, reflecting the computational overhead, and decreased at the end of each round, indicating the task completion. This pattern was consistent across both datasets, highlighting the resource demands during the distributed training and testing.



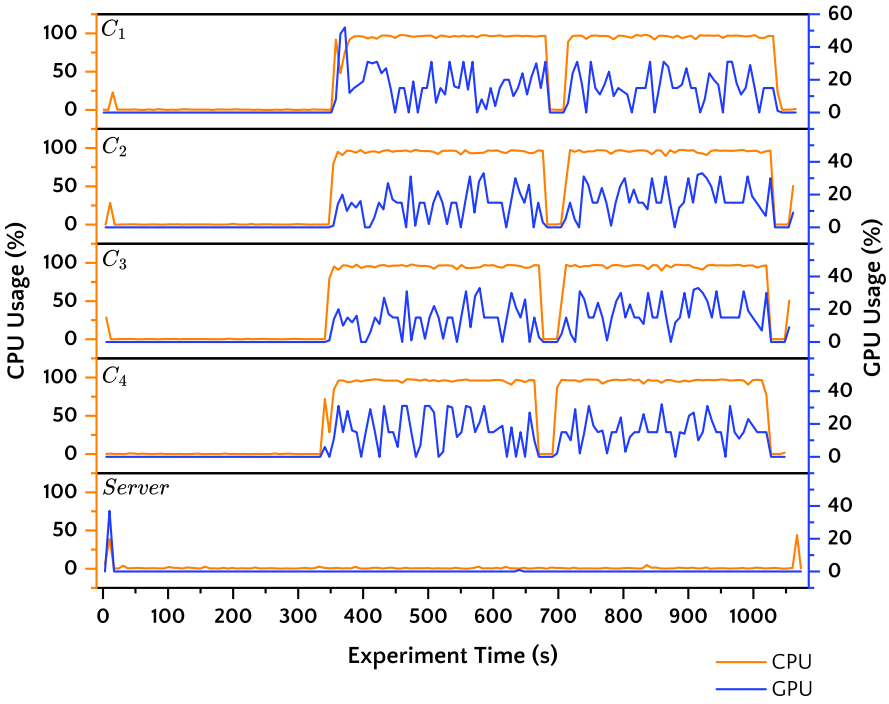
(a) Biglycan.



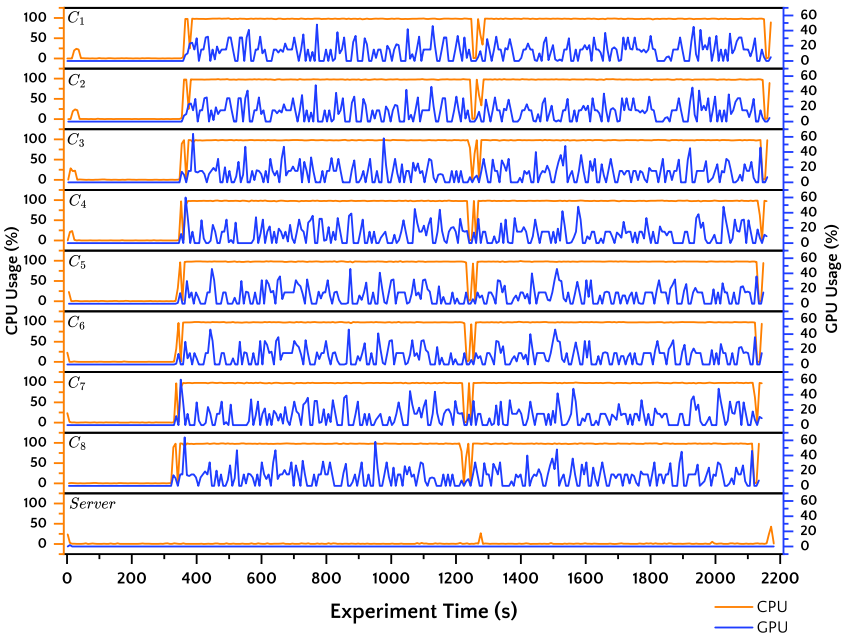
(b) Colorectal.

Figure 34 – CPU and GPU usage considering WS = 3 for each dataset evaluated.

Following these initial evaluations, we hypothesized that varying WS might yield different results, particularly in terms of classification performance and resource efficiency. Given that the biglycan dataset is smaller, it serves as an ideal candidate for this exploratory analysis. We increased WS from five and nine to observe any potential improvements in classification accuracy or changes in computational cost. Figure 35 shows the CPU and GPU consumption considering WS values of five and nine.



(a) WS = 5.



(b) WS = 9.

Figure 35 – CPU and GPU usage considering Biglycan dataset considering different WSs.

The results demonstrate that, while we set an increasing WS parameter in the AIaaS GUI, this influenced CPU and GPU resource consumption and classification performance, as summarized in Table 17. This important finding guided our decision to maintain a WS of three for the colorectal dataset because increasing it would have resulted in unnecessary computational overhead without enhancing the model classification.

Table 17 – Classification performance considering different WS and biglycan dataset.

WS	Round	Accuracy (%)	Precision (%)	Recall (%)	F1-Score	Training Time
3	1	61.76	58.83	59.91	56.33	7.33 s
	2	100	100	100	100	84.75 s
	3	97.06	96.56	97.56	96.97	149.97 s
	Server	**100**	**100**	**100**	**100**	**217.61 s**
5	1	61.76	58.83	56.91	56.33	7.73 s
	2	88.24	88.89	86.45	87.35	354.49 s
	3	95.59	95.70	95.08	95.37	710.37 s
	Server	**98.22**	**98.78**	**98.48**	**98.53**	**1064.2 s**
9	1	48.53	43.13	44.04	43.10	8.42 s
	2	82.35	83.28	79.68	80.68	358.45 s
	3	92.65	92.59	92.01	92.28	1262.5 s
	Server	**94.12**	**94.51**	**93.23**	**93.78**	**2161.71 s**

By answering $\mathcal{RQ}3$, our findings suggest that the integration of FL into the AIaaS architecture enhances both the efficiency and scalability in resource-constrained environments. By leveraging FL, the AIaaS architecture enables distributed model training directly on edge devices, thereby mitigating the need for extensive data transfer to centralized servers. This distributed approach optimizes the use of local computational resources and reduces the associated overhead on central infrastructure. Additionally, the AIaaS architecture balances computational efficiency with model performance, making it well suited for deployment in diverse and resource-limited environments without unnecessary escalating resource consumption.

$\mathcal{RQ}4$: How can different AI paradigms benefit from AIaaS architecture?

Our experimental results indicate that the AIaaS architecture enables an effortless way for an ISP manager or user to handle AI models the life-cycle, manages diverse datasets, and employing various training strategies, leading to an improved classification performance and addressing $\mathcal{RQ}_4$. This scalability ensures advanced computer vision capabilities in resource-constrained environments without sacrificing the performance. Additionally, AIaaS enables cloud-based training and updates by leveraging resources beyond local device capabilities. Our implementation of FL within AIaaS proved effective in enhancing model performance while maintaining data privacy, optimizing training, and reducing local resource overhead.

As illustrated in Figure 36 particularly the relationship between the main core modules in the proposed architecture. On the left side, we can see that a user can require a training

job in the AIaaS architecture. On the right side of the figure, the AIaaS architecture commissions computing resources for a distributed training scenario, which triggers a training job for clients passing a formatted specification containing many AI parameters such as the learning rate, epochs, round, and dataset directory.

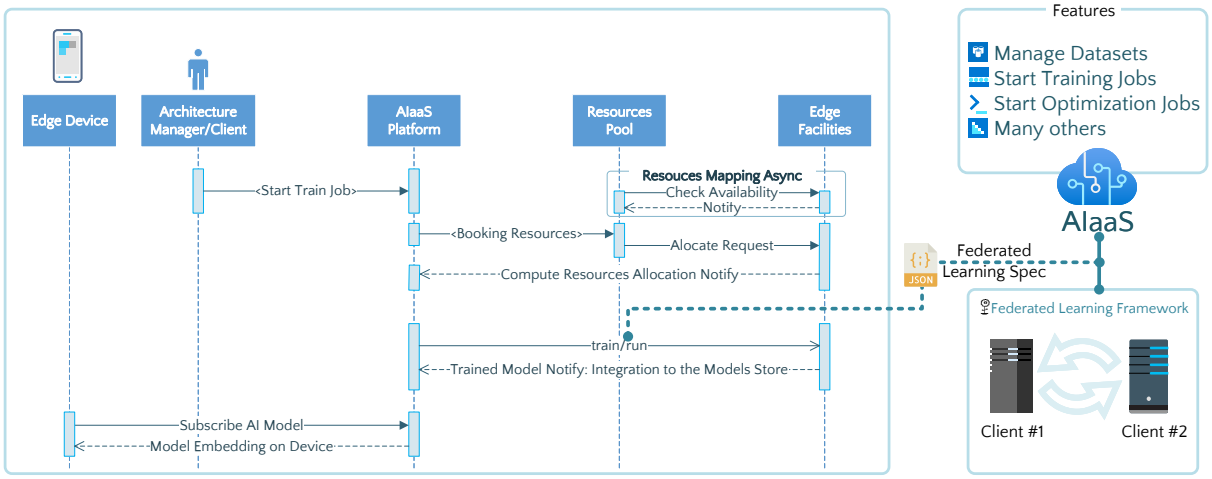


Figure 36 – AIaaS triggering different learning jobs.

Furthermore, AIaaS enables cloud-based training and updating of computer vision models by leveraging computational resources that are unavailable on local devices. Our implementation of FL within the AIaaS architecture demonstrated its effectiveness in enhancing model performance. The FL approach enables for continuous model improvements while maintaining data privacy, as shown by the management of model updates and resource utilization in our experiments. This capability highlights the advantage of AIaaS in optimizing model training and updates, while minimizing the overhead on local resources.

We compared our best results obtained with FL with other state-of-the-art methods in the literature for the same Biglycan and Colorectal datasets. As shown in Table 18, while we measured the resource consumption demanded by the AIaaS training job, the best score of the AI model, in which training clients uploaded to the Model Store, was either superior or very close to the best state-of-the-art techniques reported in the literature.

Table 18 – Comparison with literature.

Dataset	Work	Training Environment	Edge Device Training	Privacy	Accuracy (%)
Biglycan	Silva Neto et al. (2023)	Centralized	○	○	93
	Barbosa et al. (2024)	Centralized	○	○	97.06
	Ma'touq e Alnuman (2024)	Centralized	○	○	97.62
	Our Approach	**Federated**	●	●	**100**
Colorectal	Kather et al. (2016)	Centralized	○	○	87.40
	Nanni, Ghidoni e Brahnam (2020)	Centralized	○	○	97.60
	Paladini et al. (2021)	Centralized	○	○	96.16
	Dif et al. (2021)	Centralized	○	○	92.62
	Ghosh et al. (2021)	Centralized	○	○	92.83
	Arthi et al. (2022)	Federated	○	●	96
	Our Approach	**Federated**	●	●	**96.40**

Based on these experiments, we confirmed that the AIaaS architecture provides a scalable framework for deploying and managing advanced AI models in different deployment setups. Our results demonstrate the capability of the architecture to enhance classification performance, optimize resource utilization, and maintain efficient operation even in resource-constrained settings. Additionally, FL integrated into the AIaaS architecture facilitates distributed model training and updates, which improves both model accuracy and operational efficiency while minimizing data transfer and maintaining privacy. These findings demonstrate the capability of our AIaaS architecture to address the challenges associated with deploying computer-vision applications. In Table 19, we summarize the $\mathcal{RQ}s$ and findings.

Table 19 – Research questions and summary findings for experiments evaluating the platform, model training, and dataset management.

Research Question ($\mathcal{RQ}$)	**Finding**
$\mathcal{RQ}1$ How can federated training be implemented seamlessly?	• AIaaS architecture provides an efficient framework for retrieving and utilizing cognitive services. • Enabling seamless communication between edge devices and AI Model Store.
$\mathcal{RQ}2$ How does the integration of FL into the AIaaS architecture impact the classification performance across different datasets?	• Improving each classification performance metric throughout the FL rounds. • Allowing an effective distributed model training, leveraging data from multiple nodes to enhance the global model's performance.

Table 19: (continued).

$\mathcal{RQ}3$ How does the integration of FL into the AIaaS architecture influence efficiency and scalability in resource-constrained environments?	• Enabling distributed model training directly on edge devices. • Optimizing local computational resources, and reducing overhead on central infrastructure, as demonstrated by the efficient management of CPU and GPU usage.
$\mathcal{RQ}4$ How can different AI paradigms benefit from AIaaS architecture?	• This provides a flexible and scalable environment that supports various AI paradigms. • Enables efficient deployment and management of models across heterogeneous devices while balancing computational efficiency with model performance.

6.2 Evaluating the Model Optimizer and Model Validation

In this section, we present the results of our research obtained from the experiments to investigate the model optimizer and model optimization functionalities of our AIaaS architecture. This features enables the architecture administrator or model subscriber to fine-tune models using well-established approaches, such as grid search, random search, Bayesian optimization, and evolutionary strategies (BISCHL et al., 2023). By leveraging these optimization techniques, the architecture can effectively adapt and optimize AI models to suit specific tasks and applications.

Specifically, we propose a hybrid model using a GA (HOLLAND, 1992) and Residual CNN (HE et al., 2016a) to predict leukemia using microscopic images available in ALL-IDB2 dataset. We chose to focus on Acute Lymphoblastic Leukemia (ALL) because of its clinical significance and its impact on public health. ALL is the most common cancer in children, and despite advancements in treatment, diagnostic challenges persist (FORCE; AL., 2019) (SCHWALBE; WAHL, 2020).

Prior works focused on applying AI techniques to support ALL diagnosis based on image. For instance, several studies have been proposed for classifying ALL in mi-

croscopy images with hand-crafted feature extraction (i.e., using a non-automated user-based process) (SCOTTI, 2005; SINGHAL; SINGH, 2016; RODRIGUES et al., 2016; DE FARIA; RODRIGUES; MARI, 2018). In addition, others approaches proposed evolutionary strategies to optimize conventional classifiers (SAHLOL et al., 2017; SAHLOL; ABDELDAIM; HASSANIEN, 2019; SAHLOL; KOLLMANNSBERGER; EWEES, 2020).

Although most of the previous works achieve an accuracy rate above 90%, they depend on the use of a proper segmentation process and handcrafted feature extraction. To overcome this limitation, strategies based on deep learning, specifically CNN, have been proposed to classify ALL in microscopy images and produce better results than all classical techniques (VOGADO et al., 2018; DAS; MEHER, 2021; CLARO et al., 2022).

Automatic feature extraction is the main strength of previous studies based on deep learning. However, these studies do not investigate the appropriate choice of training-relevant hyperparameters or propose a mechanism such as our AIaaS, which can significantly impact the classification performance. Thus, to overcome this gap, we proposed an approach based on GA to identify the optimal hyperparameters in a broad search space that considers a uniform distribution, data augmentation strategy, and training based on fine-tuning. Also, our method does not need the segmentation process, it is more robust to the intensity variations in the images, and it is also suitable to deal with the lack of training data for approaches based on deep learning.

The goal of these experiments was to answer the following Research Questions ($\mathcal{RQ}$).

- ❑ $\mathcal{RQ}1$: What are the best values of hyperparameters (dropout, learning rate and momentum coefficient), which bring the highest classification performance?
- ❑ $\mathcal{RQ}2$: How much does the hyperparameter optimization increase the performance of ResNet-50 V2, considering fine-tuning approach?
- ❑ $\mathcal{RQ}3$: Considering metrics derived from confusion matrix, what is the most suitable approach for leukocytes image classification: training without GA or with GA?
- ❑ $\mathcal{RQ}4$: In terms of consumed time for training and the classification performance, considering GA, random search, and Bayesian optimal parameter finding methods, which one is the best approach?

We aimed to assess the performance of a deep residual CNN architecture for classifying ALL in microscopy images. Specifically, we aim to identify the optimal hyperparameter combination that enhances classification performance. Figure. 37 illustrates the general scheme of the proposed approach and outlines the steps involved in the evaluation process.

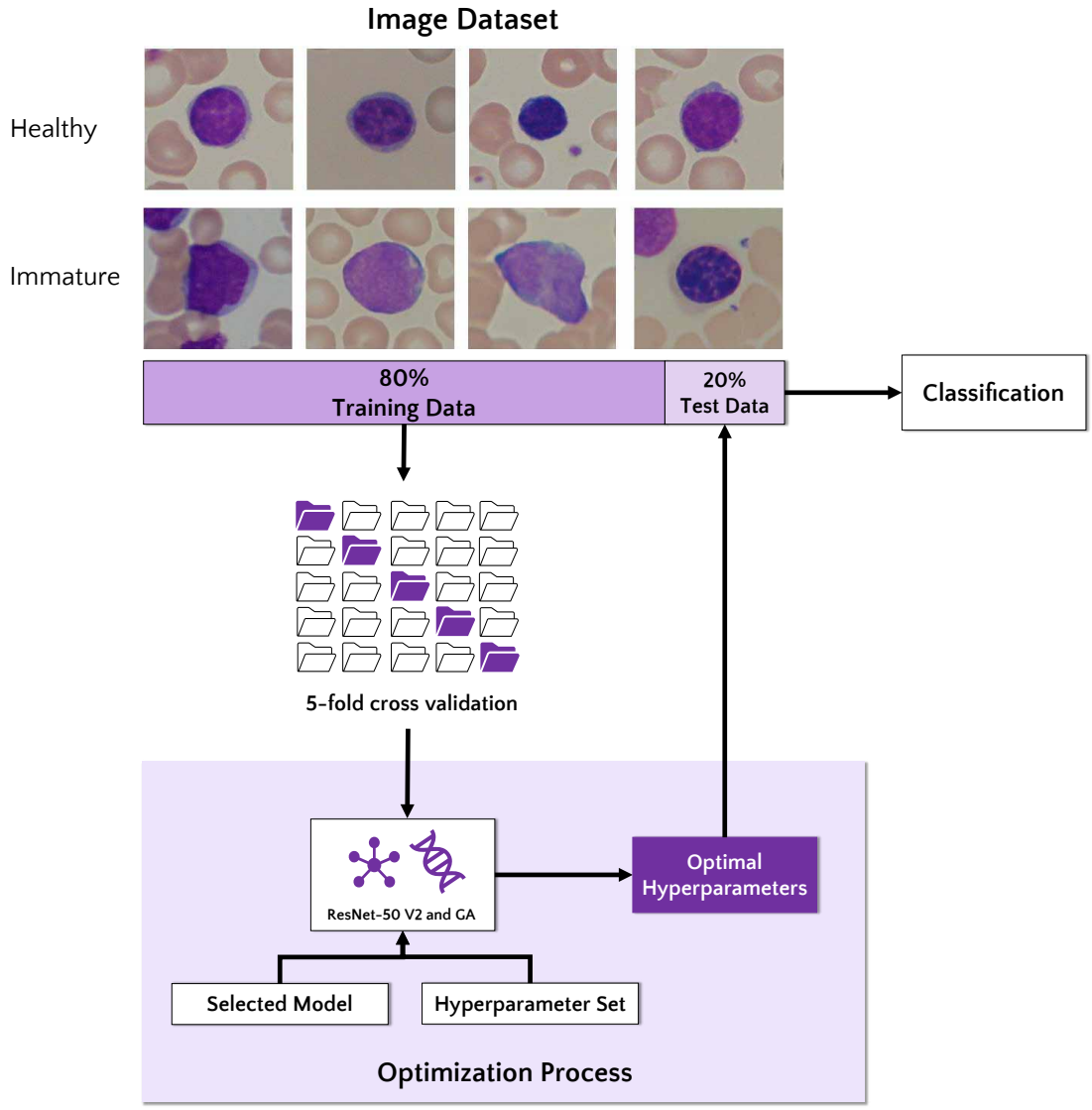


Figure 37 – Proposed method to evaluate the Models Optimizer of AIaaS.

6.2.1 Proposed Genetic Algorithm

GA is an evolutionary algorithm inspired by natural biological evolution. It consists of a population with n individuals and employs bio-inspired operations like selection, crossover, and mutation (HOLLAND, 1992). GA is well-suited for combinatorial optimization problems and has been shown to outperform other algorithms like Simulated Annealing and Tabu Search in terms of solution quality (YOUSSEF; SAIT; ADICHE, 2001). We chose GA over other heuristic algorithms for hyperparameter CNN evolution because GA can incorporate domain-specific knowledge in all optimization phases (YOUSSEF; SAIT; ADICHE, 2001), which is essential for hyperparameter combination in deep learning.

The conventional GA includes three main steps, as shown in Algorithm 1 and explained in the next subsections. Initially, n chromosomes are randomly generated according to

the specific encoding method. Then, new offspring are continuously generated from the existing population and combined with the older generation to form a new generation. Finally, there is a population evolution process in which all individuals enter an iterative competition generating a new population composed of the survivors generated from the crossover of survivors and mutation operation.

Algorithm 1 GA Framework for Hyperparameter Optimization

1: **Input:** the image dataset $\mathcal{D}$, size of population p, the number of survival individuals per evolution generation k, the number of evolutionary generations G, and the feasible hyperparameter space $\mathcal{H}$.
2: **Encoding:** the gene includes three elements: dropout rate (d), learning rate (l), and momentum coefficient (μ).
3: **Population initialization:** n chromosomes are randomly selected from feasible solution space and the fitness f_i and evolutionary probability p_i values ($i = 1, \ldots, n$) of individuals are evaluated.
4: **for** $g = 1, \ldots, G$ **do**
5: **Selection:** linear ranking and tournament ($t = 2$).
6: **Crossover**: two individuals are selected according to fitness value.
7: **Mutation:** applying mutation operator.
8: **end for**
9: **Output:** In G^{th} generation, the individual with the largest fitness is the optimal solution in the hyperparameter space $\mathcal{H}$.

6.2.1.1 Population Initialization

Initialization is the process of randomly selecting candidate solutions in the search space. In this study, we defined the search space size as an input from the hyperparameter set. Also, we use a uniform distribution to ensure the random distribution of candidates in the search space. There is a particular interval for all hyperparameters in which the gene can assume values within the defined range, limiting the search space of the GA.

During population initialization, hyperparameters are defined, and a random value is chosen within the pre-defined interval to encode the chromosome. We must include the information of dropout, learning rate, and momentum coefficient for the chromosome encoding in an array. Each chromosome is composed of three genes, and each gene encodes a real value for its respective hyperparameter. The structure of the chromosome is shown in Figure 38.

Droupout	Learning Rate	Momentum
0.4047	0.0012	0.8058

Figure 38 – Chromosome representation for hyperparameter optimization.

6.2.1.2 Selection

The selection is based on fitness, i.e., the fittest individual is selected to participate in the reproduction process. In this operator, the current generation members with the highest fitness values are the most likely to generate the next population. In this study, we applied linear ranking with tournament selection:

- **Linear Ranking:** It is based on the classification of individuals according to fitness, being the probability of selecting an individual depends solely on fitness. For this, the worst individual gets a rating of 1; the second-worst individual gets a rating of 2. This idea repeats successively until the best individual receives a rating of n (corresponding to the number of chromosomes in the population). Thus, based on their rating ($rank_i$), each individual i has the probability P_i to be selected from a population of n individuals, as defined in Equation 8 (KUMAR et al., 2012).

$$P_i = \frac{rank_i}{n \times (n-1)} \tag{8}$$

- **Tournament:** The tournament selection (MILLER; GOLDBERG, 1996) aims to select a set of k individuals randomly, which will be sorted according to their relative fitness. Afterward, the fittest individual is chosen for reproduction. This process is repeated several times for the entire population, and the probability P_i of each individual being selected is expressed in Equation 9.

$$\mathcal{P}_i = \begin{cases} \mathcal{C}_{n-1}^{k-1}, & \text{if } i \in [1, n-k-1], \\ 0, & \text{if } i \in [n-k, 1]. \end{cases} \tag{9}$$

Figure 39 shows the selection method based on linear ranking and tournament applied in this study. We applied the linear ranking to select half of the best chromosomes, sorting the highest accuracy and lowest loss at the top of the population. Then, the selection with a tournament ($t = 2$) is performed considering the best-ranked chromosomes.

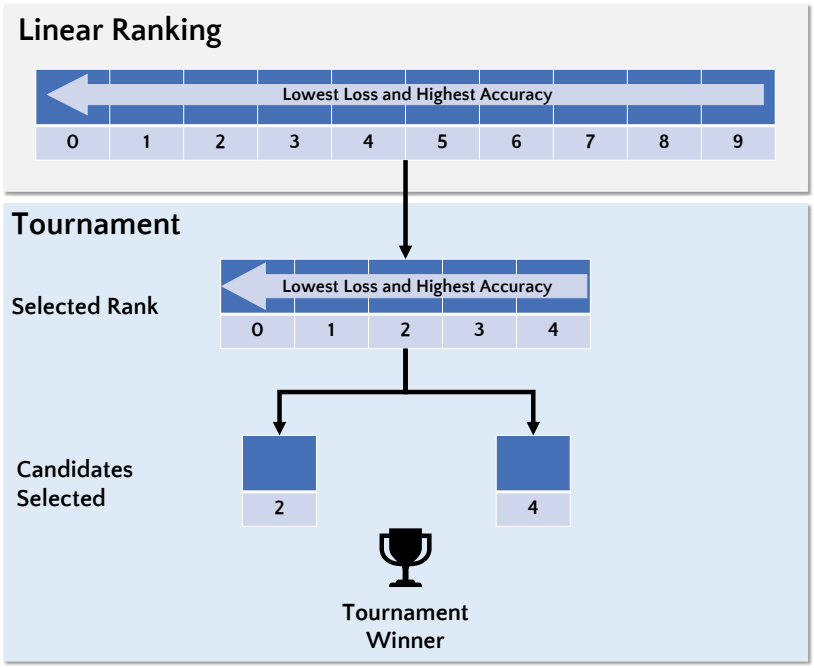


Figure 39 – Selection based on linear ranking and tournament.

6.2.1.3 Crossover and Mutation

The crossover operator is used to generate new individuals by recombining the genes of parent chromosomes. In this study, we used a single-point crossover. For each iteration in the GA process, one point was randomly selected. For example, as illustrated in Figure 40, to generate a "child 1", the random crossover point is index 1. Thus, index 0 and 1 from "parent 1" will be selected as head, and index 2 from "parent 2" will be chosen as tail. After, a similar process occurs to generate a "child 2".

Also, we applied the mutation operator with a rate of 30%, which is initiated after the crossover process by randomly modifying one bit of an individual's chromosome to generate a child.

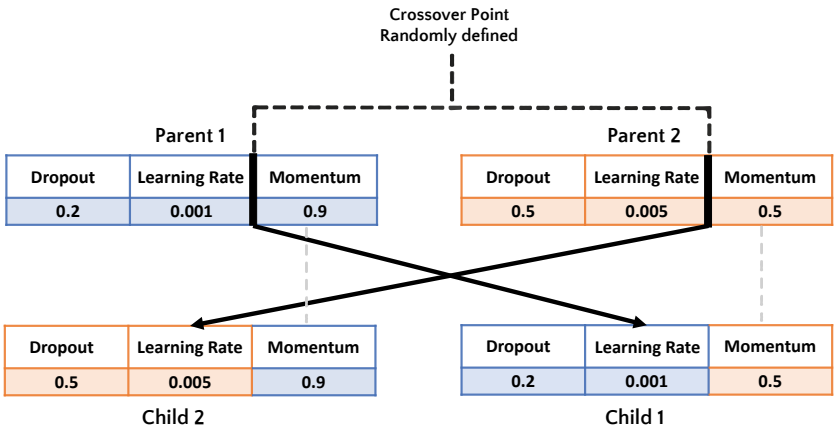


Figure 40 – Example of single-point crossover.

6.2.1.4 Fitness Evaluation

We considered the objective function as the loss-function defined by $\mathcal{L}(W)$. Equation 10 show that $\mathcal{L}(W)$ is computed over a set of training samples X_j considering the tuned weights W, parameters $f(x_j)$, and the known classes y_j, where j represents the classes lymphocytes immature and healthy.

$$\mathcal{L}(W) = \frac{1}{n} \sum_{j=1}^{N} \ell(y_j, f(x_j; W)) \tag{10}$$

In this way, to minimize $\mathcal{L}(W)$, we applied the SGD (LECUN et al., 1998) optimization algorithm with hyperparameters (dropout, learning rate, and momentum) optimized through GA. Consequently, when we minimize the loss-function, the accuracy (Equation 11) is maximized.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{11}$$

6.2.1.5 Image Dataset and CNN Architecture

The ALL-IDB2 dataset[1], is composed of 260 images categorized into healthy (130 images) and immature (130 images) classes. The images were obtained from the Department of Information Technology, Universitá degli Studi di Milano (LABATI; PIURI; SCOTTI, 2011) and are in JPG format with a 24-bit color depth and a size of 2592 × 1944 pixels. Figure 41 shows some images from the dataset for both classes, healthy and immature.

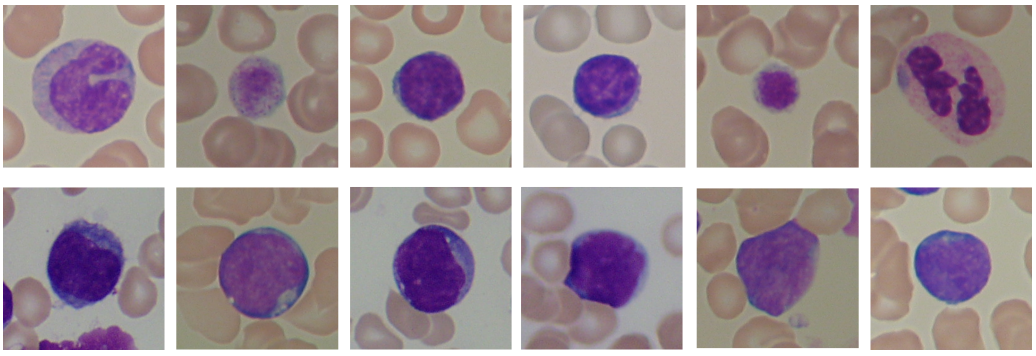


Figure 41 – Image instances from the ALL-IDB2 dataset showing healthy (top) and immature (bottom) lymphocytes.

This experiment focused on evaluating the performance of ResNet-50 V2 (HE et al., 2016b), an upgraded version of ResNet-50 (HE et al., 2016a), which performed well in the ILSVRC 2015 challenge. The improvement comes from using a pre-activation variant of the residual block, enabling gradients to flow more effectively through shortcut connections to earlier layers. The convolutional block (Figure 42b) comprises batch normalization, ReLU activation, and convolution with a kernel of size k.

[1] Available in: <https://homes.di.unimi.it/scotti/all/>

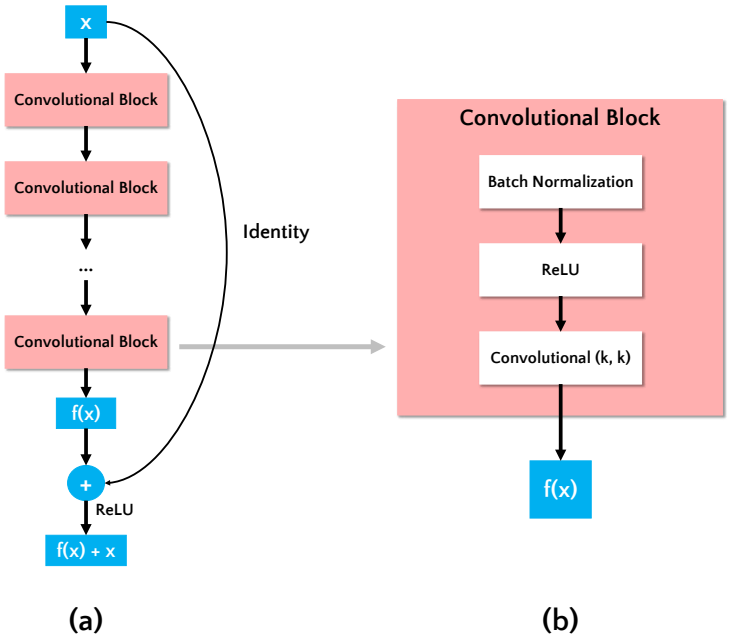


Figure 42 – (a) Residual Block. (b) Detail of a convolutional block inside the Residual Block.

We adapted the size of the images of ALL-IDB dataset for the input of ResNet-50 V2, resizing the images to 224 × 224 pixels using bilinear interpolation. We arranged the dataset into training and test sets: 80% for training (further split into 80% for training and 20% for internal validation using a stratified 5-fold cross-validation method (DEVIJVER; KITTLER, 1982)) and 20% for testing (external validation). We applied 5-fold internal cross-validation to all experiments and reported the results on an external test set.

Neural networks, such as ResNet, generally require a large amount of data during training to avoid overfitting. Given the reduced number of images in the dataset, we used data augmentation and transfer learning techniques to improve the network training (TAJBAKHSH et al., 2016; CLARO et al., 2020). Data augmentation enables an artificial increase in the training set by generating new samples of a given image under different variations without introducing labeling costs (KRIZHEVSKY; SUTSKEVER; HINTON, 2012). In our case study, we performed data augmentation using only vertical and horizontal flips.

Additionally, we applied transfer learning, that is, we used pre-trained 2012 ImageNet weights to initialize the network (DENG et al., 2009). This strategy enables the use of low-level features learned from larger datasets, which are better than those learned using a network trained from scratch in a smaller dataset. For comparison, the ImageNet dataset contains approximately 1.2 million images divided into 1000 classes. In this sequence, we used our small dataset to fine-tune the network weights to our problem, that is, we initialized all convolutional layers with weights from the pre-trained model, and fine-tuning was performed only in the deeper layers (PONTI et al., 2017). Finally. we report our results in terms of average accuracy, precision, recall, and F1-score.

6.2.2 Experiments

The experiments were conducted using the Google Colaboratory cloud service with a machine Intel(R) Xeon(R) 2.20GHz processor, 12 GB RAM, and a GPU NVIDIA Tesla T4. The experiments were programmed using Python (version 3.6) and Keras 2.0 (CHOLLET et al., 2015) deep learning framework. The hyperparameter optimization algorithms random search and Bayesian optimization were drawn from the KerasTuner library (O'MALLEY et al., 2019), version 1.1.0.

Below we describe each experiment carried out to answer the four research questions raised in this case study.

> **$\mathcal{RQ}1$: What are the best values of hyperparameters (dropout, learning rate and momentum coefficient), which bring the highest classification performance?**

To answer $\mathcal{RQ}1$, we first evaluated the impact of training without hyperparameter optimization in order to understand the classification performance when default parameters were used. We trained the CNN SGD optimizer with a learning rate of 0.01, momentum of 1.0, batch size of eight, and 100 epochs. The values of the learning rate and momentum were defined according to the literature (PONTI et al., 2017).

The results obtained for the test set are presented in Table 20 and demonstrate that using the default values generates poor results, indicating that these hyperparameters need to be well adjusted for the CNN to achieve a good performance. Figure 43 shows the behavior of the accuracy and loss during the training phase (considering the fifth iteration of the k-fold), which generates noise values that result in underfitting.

Table 20 – Classification results considering training without GA using the test set.

Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
50.00	25.00	50.00	33.00

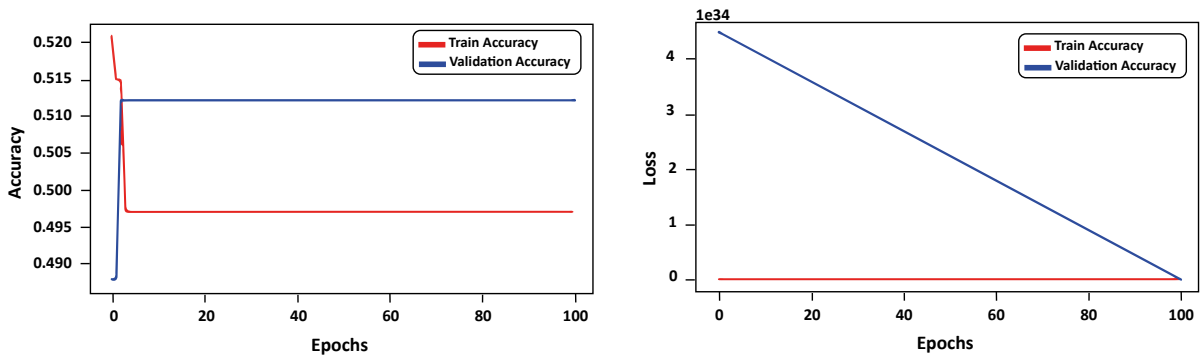


Figure 43 – Evolution of accuracy and loss values considering the training and validation set.

Drawing from the repercussions of the absence of hyperparameter optimization, we conducted experiments aimed at showing the impact of hyperparameter tuning of the ResNet-50 V2 deep CNN. Our approach involved the utilization of GA to optimize nuanced parameters, such as the dropout rate, learning rate, and momentum coefficient within the pre-trained ResNet-50 V2 architecture.

The hyperparameter optimization procedure for each cross-validation fold required approximately 78 minutes to run. The GA configuration encompassed a mutation rate of 30%, a population size of ten individuals, and five generations. For each generational iteration, CNN was trained over 20 epochs. Although a higher population size and an increased maximum generation count typically yield enhanced performance, such an approach requires substantial time and computational resources. To overcome this limitation, we applied the same setting adopted by (KILICARSLAN; CELIK; SAHIN, 2021) to optimize a deep CNN using GA.

Table 21 presents the hyperparameter space search results evaluated in this study. We searched all hyperparameters by considering a uniform distribution. In addition, Table 22 presents the best values for each hyperparameter returned by the GA for answering $\mathcal{RQ}1$.

Table 21 – Hyperparameter search space used for optimization.

Hyperparameter	**Value**
Dropout	x ∈ [0.0, 0.5]
Learning Rate	x ∈ [0.005, 0.01]
Momentum	x ∈ [0.0, 1.0]

Table 22 – Hyperparameter optimized with GA.

	Hyperparameter		
Fold	**Dropout**	**Learning Rate**	**Momentum**
1	0.01326	0.00179	0.15547
2	0.01489	0.00187	0.15547
3	0.01326	0.00187	0.15547
4	0.01489	0.00187	0.09274
5	0.01326	0.00179	0.15547

$\mathcal{RQ}2$: How much does the hyperparameter optimization increase the performance of ResNet-50 V2, considering fine-tuning approach?

To answer $\mathcal{RQ}2$, we analyzed the results obtained for each hyperparameter setting (HS) obtained from 5-fold internal cross-validation. In addition, we trained ResNet-50 V2 using the optimized hyperparameters obtained by the GA with 100 epochs. As shown in Table 23, GA hyperparameter optimization significantly improves the classification performance. Hence, in response to research question $\mathcal{RQ}2$, we observed that

the optimization yielded a substantial accuracy improvement of 48.46 percentage points, increasing the accuracy level from 50.00% to 98.46%.

Table 23 – Classification results considering training with five different hyperparameters setting from Table 22 applied on the test set.

HS	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
1	100.00	100.00	100.00	100.00
2	98.08	98.15	98.08	98.08
3	96.15	96.43	96.15	96.15
4	100.00	100.00	100.00	100.00
5	98.08	98.15	98.08	98.08
Average	**98.46 ± 0.01**	**98.55 ± 0.01**	**98.46 ± 0.01**	**98.46 ± 0.01**

$\mathcal{RQ}3$: Considering metrics derived from confusion matrix, what is the most suitable approach for leukocytes image classification: training without GA or with GA?

By answering $\mathcal{RQ}3$, our results demonstrate that training with GA is the most suitable approach for leukocyte image classification, significantly improving classification performance. To assess the learning behavior during the training phase, Figure 44 shows the variations in the loss and accuracy metrics for the second iteration of the k-fold procedure. It is noteworthy to highlight the diminished loss function values. This trend indicates that the training process avoided overfitting the data, thereby preserving the capacity of the deep residual CNN for generalization.

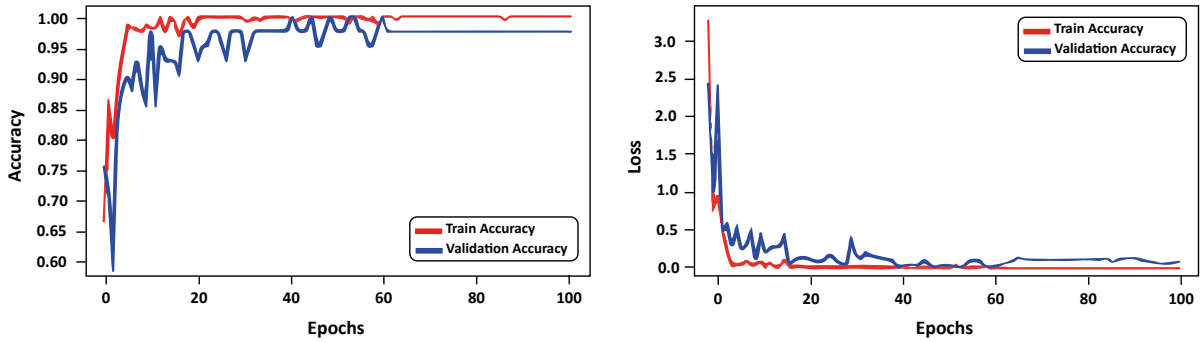


Figure 44 – Evolution of accuracy and loss values considering the training and validation set and GA hyperparameter optimization.

Furthermore, Table 24 presents the confusion matrix for ResNet-50 V2 optimized by GA. It can be seen that our approach correctly classified 97.69% of leukemia images and did not require preprocessing or segmentation processes (commonly used in state-of-the-art techniques).

Table 24 – Confusion matrix for ResNet-50 V2 optimized with GA averaged over the five hyperparameters.

	Healthy	**Leukemia**
Healthy	99.23%	0.77%
Leukemia	2.31%	97.69%

Our results demonstrates that defining optimal hyperparameters manually requires considerable domain expertise, and manual preconfigurations limit the feasible solution space to miss out on the better set of hyperparameters. It is a relief that the GA can automatically search for the optimal solution in a large space without a manual setting. Thus, automatic selection by GA reduces the domain expertise requirements in deep learning for researchers and helps some non-expert researchers to define the best hyperparameters as a module of our AIaaS architecture to obtain high-performance classifiers.

$\mathcal{RQ}4$: In terms of consumed time for training and the classification performance, considering GA, random search, and Bayesian optimal parameter finding methods, which one is the best approach?

To answer $\mathcal{RQ}4$, we evaluated the impact of optimization using random search and Bayesian optimization approaches to compare their performances. These approaches were chosen because they are widely used in CNN hyperparameter optimization (MOREIRA et al., 2020; NISHIO et al., 2020; SALEEM; POTGIETER; ARIF, 2020; DA SILVA; RODRIGUES; MARI, 2020; DA ROCHA; RODRIGUES; MARI, 2020; HIZUKURI et al., 2021; KAUR; AGGARWAL; RANI, 2021).

For a fair comparison, we considered the same search space evaluated in the GA hyperparameter optimization (see Table 21). The relevant hyperparameters configurations derived from random search and Bayesian optimization for each fold are documented in Tables 25 and 26 respectively.

Table 25 – Hyperparameter optimized with random search.

	Hyperparameter		
Fold	**Dropout**	**Learning Rate**	**Momentum**
1	0.31971	0.00519	0.40853
2	0.31971	0.00519	0.40853
3	0.31971	0.00519	0.40853
4	0.27407	0.00533	0.49753
5	0.31971	0.00519	0.40853

Table 26 – Hyperparameter optimized with Bayesian optimization.

	Hyperparameter		
Fold	**Dropout**	**Learning Rate**	**Momentum**
1	0.0	0.005	0.0
2	0.00624	0.005	0.0
3	0.00610	0.005	0.0
4	0.86178	0.00944	0.35184
5	0.5	0.005	0.0

We observed that the values returned by the Bayesian optimization exhibited a more comprehensive exploration of the search space, discerning various potential optimal solutions. Given the optimized hyperparameters obtained by the random search and Bayesian optimization, we trained ResNet-50 V2 with 100 epochs. As shown in Table 27, random search optimization achieved an accuracy of 62.31%.

However, this result is only better than the classification without hyperparameter optimization. In contrast, Bayesian optimization performed better than random search obtained an accuracy of 80.39% as shown in Table 28. Furthermore, when considering the results obtained in hyperparameter settings 2 and 3, we observed that the accuracy is very close to the results obtained with GA.

Table 27 – Classification results on the test set considering training with five different hyperparameters setting obtained from random search.

HS	**Accuracy (%)**	**Precision (%)**	**Recall (%)**	**F1-Score (%)**
1	71.15	71.44	71.15	71.06
2	51.92	75.49	51.92	37.47
3	50.00	25.00	50.00	33.00
4	71.15	71.97	71.15	70.88
5	67.31	80.23	67.31	63.40
Average	**62.31 ± 0.10**	**64.83 ± 0.22**	**62.31 ± 0.10**	**55.16 ± 0.18**

Table 28 – Classification results on the test set considering training with five different hyperparameters setting obtained from Bayesian optimization.

HS	**Accuracy (%)**	**Precision (%)**	**Recall (%)**	**F1-Score (%)**
1	92.31	92.56	92.31	92.30
2	98.08	98.15	98.08	98.08
3	98.08	98.15	98.08	98.08
4	57.69	68.84	57.69	50.35
5	55.77	76.53	55.77	45.01
Average	**80.39 ± 0.21**	**86.85 ± 0.13**	**80.39 ± 0.21**	**76.76 ± 0.26**

Finally, as shown in the Table 29, we compared the the hyperparameter optimization methods in terms of classification performance metrics and optimization time.We observed

that the GA improved leukemia image classification significantly, although it requires more time to perform the optimization. This result demonstrates that the random search and Bayesian optimization approaches do not explore the search space distribution as well as the GA. Simultaneously, the GA can incorporate domain-specific knowledge in all optimization phases, guaranteeing gains in all classification performance metrics. In addition, once GA obtains the best hyperparameters, the training time becomes similar to that of training with a manual configuration, which requires approximately 10 minutes.

Table 29 – Average classification and optimization time for each optimization method.

Method	**Accuracy (%)**	**Precision (%)**	**Recall (%)**	**F1-Score (%)**	**Optimization Time**
Random Search	62.31	64.83	62.31	55.16	22min 10s
Bayesian	80.39	86.85	80.39	76.76	22min 36s
GA	**98.46**	**98.55**	**98.46**	**98.46**	78min 45s

We also compared our best result obtained with GA hyperparameter optimization with other state-of-the-art methods in the literature for the same ALL-IDB2 dataset. As shown in Table 30 our best score is higher than that of the best state-of-the-art technique reported in the literature.

Table 30 – Comparison with literature.

Method	**Accuracy (%)**
Rodrigues et al. (2016)	85.00
Singhal e Singh (2016)	93.84
Sahlol et al. (2017)	91.80
Sus e Oliveira (2017)	94.60
de Faria, Rodrigues e Mari (2018)	97.22
Sahlol, Abdeldaim e Hassanien (2019)	95.23
Sahlol, Kollmannsberger e Ewees (2020)	96.11
Das e Meher (2021)	97.18
Our proposal	**98.46**

The results obtained with this case study showcases the potential of AIaaS for medical image analysis, using ALL classification as a proof of concept for the optimization module within the AIaaS architecture. By employing ResNet-50 V2 and GA optimization, we achieved a remarkable accuracy of 98.46%, surpassing existing benchmarks. We established that hyperparameter tuning based on GA significantly enhances CNN performance compared to non-optimized and alternative optimization techniques, shedding light on optimal parameter configurations and their impact.

In addition, the integration of hyperparameter optimization and transfer learning techniques not only improved the classification performance but also paved the way for

the utility of AIaaS in streamlining medical diagnosis using advanced machine learning methodologies. In Table 31, we summarize all Research Questions ($\mathcal{RQ}s$) and findings.

Table 31 – Research questions and summary findings for experiments evaluating the model optimizer and model validation.

Research Question ($\mathcal{RQ}$)	**Finding**
$\mathcal{RQ}1$ What are the best values of hyperparameters (dropout, learning rate and momentum coefficient), which bring the highest classification performance?	• Values presented in Table 22.
$\mathcal{RQ}2$ How much does the hyperparameter optimization increase the performance of ResNet-50 V2, considering fine-tuning approach?	• Improvement from 50% to 98.46%.
$\mathcal{RQ}3$ Considering metrics derived from confusion matrix, what is the most suitable approach for leukocytes image classification: training without GA or with GA?	• Training with GA.
$\mathcal{RQ}4$ In terms of consumed time for training and the classification performance, considering GA, random search, and Bayesian optimal parameter finding methods, which one is the best approach?	• GA.

Our findings are in line with the aim of the AIaaS Architecture, which is to provide users and applications with easily accessible intelligent solutions and resources. By incorporating hyperparameter optimization, users can take advantage of efficient AIaaS

facilities and create high-performing classifiers without needing extensive manual configuration or specialized knowledge.

Chapter 7

Conclusion

This chapter summarizes the findings of this thesis and highlights its contributions to the state-of-the-art. Our approach shares the common goals of streamlining and automating the lifecycle of AI models, but we achieve these objectives from different perspectives. Whereas Machine learning operations (MLOps) are a set of practices that aim to ensure that ML models can be reliably updated, versioned, and maintained in production, our AIaaS focuses on providing a cloud-based service model where users can access and deploy AI capabilities, such as model training, inference, and updates, without managing the underlying infrastructure. It abstracts the complexity of AI deployment, making it accessible to users with varying expertise levels, and emphasizes scalability, resource efficiency, and ease of use for deploying AI applications across diverse environments, such as edge devices.

The rapid growth of AI applications across various aspects of daily life highlights its transformative potential. With global investments in AI projected, the need for efficient deployment and delivery mechanisms for AI services has become more pressing. However, current manual and context-specific approaches to deploying AI services are limited in scalability and personalization. This challenge has led to the proposed of AIaaS Architecture, which offers on-demand AI-based solutions to users and applications.

Although AI applications can benefit from Automated Machine Learning (AutoML), this approach focuses specifically on automating the development of ML models by simplifying tasks such as feature selection, model selection, hyperparameter tuning, and validation. This reduces the need for manual intervention in the model building process. However, the AIaaS architecture goes further by offering a complete service framework that also supports the integration of AutoML, enabling seamless deployment, scalability, and management of AI models across various platforms and devices.

The main goal of this research is to explore and confirm the viability of providing AI capabilities through the AIaaS Architecture, particularly in the field of Computer Vision.

We prepared a literature review of previous studies that have moved towards providing AIaaS. We have highlighted significant advances in AIaaS Architecture concerning its peers. After recognizing that the state of the art has gaps, we evaluated the suitability of AIaaS Architecture to support computer vision applications. This thesis introduces and validates the AIaaS Architecture, showing its potential for Computer Vision applications. Our hypothesis and its implications are corroborated by the experiments reported and discussed in this thesis.

Initially, the **Edge Predictor Service Evaluation** demonstrated the AIaaS capabilities to deliver predictive services on edge devices, focusing on metrics such as response time, energy consumption, and prediction classification performance. The experimental results demonstrated the successful integration of AI models into low-cost devices, making this technology accessible in resource-limited regions.

The architecture enables state-of-the-art AI solutions, enhancing the diagnostic accuracy for critical diseases, such as cancer and COVID-19. Moreover, the AI Model Store opens new opportunities for monetizing specialized models, fostering a new generation of AI applications. Additionally, we carried out edge intelligence using the AIaaS architecture to empower last-mile clients to retrieve cognitive services from the AI Model Store. In addition, we found that embedding pretrained CNNs in our edge device through cognitive service acquisition on the AI Model Store leads to predictions with low computational overhead and latency compared to HPC.

The **Service Management Evaluation** demonstrates how the AIaaS architecture efficiently manages the entire lifecycle of AI models, including training, validation, testing, and fine-tuning. We functionally explored and evaluated the suitability of the AIaaS architecture to enable both ISP managers and users to effortlessly initiate AI training jobs. In the state of the art, managing the lifecycle of AI models is resource-intensive and requires significant development and computational resources, with no standardization of how these services can be delivered to end users. We assessed the computational overhead imposed by training jobs on the resource pool and validated the feasibility of the architecture using an FL use case.

We investigated the model optimizer and the optimization functionalities within the architecture. It enables administrators or model subscribers to fine-tune models using well-established techniques, such as grid search, random search, Bayesian optimization, and evolutionary strategies. By leveraging these advanced optimization methods, the architecture adapts and refines AI models to meet specific tasks and application requirements.

Our AIaaS architecture also impacts mobile network ecosystem such as Beyond 5G (B5G) and 6G. By leveraging AIaaS, these next-generation networks can enhance their efficiency and adaptability, allowing for more intelligent and dynamic management of network resources. The proposed AIaaS enables the design of an evolution proposal

aligned with 3rd Generation Partnership Project (3GPP) for NWDAF enabling different players to access or offer third-party cognitive services in a mobile network through a northern interface on NWDAF. The ability to deploy and manage AI models as a service enables more flexible and scalable solutions, aligning with the demands of complex network infrastructures and providing network optimization, security, and user experience.

Finally, this thesis introduces new ideas for standardizing the AIaaS architecture to seamlessly deliver an intelligent application experience to the industry, applications, and users. By proposing and validating a new AIaaS, this study significantly advances this field by offering a comprehensive solution for managing and deploying AI services. This research also delves into the specific challenges of applying AIaaS in edge prediction contexts, highlighting the effectiveness of the architecture in addressing these issues. Overall, the contributions of this thesis mark a significant step forward in AIaaS, particularly in the field of computer vision. The proposed architecture provides a new concept for future research and development in this domain. These findings emphasize the potential of AIaaS to support the management and deployment of AI models, paving the way for more efficient and adaptable solutions.

7.1 List of Contributions

Journal publications:

- **Rodrigues Moreira et al. (2024)** Deep Learning based Image Classification for Embedded Devices: A Systematic Review. Neurocomputing **(under review)**.
- **Rodrigues Moreira et al. (2023)** An Artificial Intelligence-as-a-Service Architecture for deep learning model embodiment on low-cost devices: A case study of COVID-19 diagnosis. Applied Soft Computing.
- **Rodrigues et al. (2022)** Optimizing a Deep Residual Neural Network with Genetic Algorithm for Acute Lymphoblastic Leukemia Classification. Journal of Digital Imaging.

Conference publications:

- **Rodrigues Moreira et al. (2024b)** Towards Cognitive Service Delivery on B5G through AIaaS Architecture. In: Anais do IV Workshop de Redes 6G.
- **Rodrigues Moreira et al. (2024a)** Maximizing the Power of Cognitive Services with an AI-as-a-Service Architecture for Seamless Delivery. In: 2024 IEEE 13th International Conference on Cloud Networking (CloudNet) (IEEE CloudNet 2024).

- ❑ **Rodrigues Moreira et al. (2024)** Enabling Intelligence on edge through an Artificial Intelligence as a Service Architecture. In: 2024 IEEE 13th International Conference on Cloud Networking (CloudNet) (IEEE CloudNet 2024).

Co-authoring:

- ❑ **Barbosa et al. (2024)** Optimization and Learning Rate Influence on Breast Cancer Image Classification. Proceedings of the 19th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISAPP).
- ❑ **Moreira, Rodrigues Moreira e Oliveira Silva (2023)** An intelligent network monitoring approach for online classification of Darknet traffic. Computers and Electrical Engineering.
- ❑ **Cardoso et al. (2023)** Improving Sickle Cell Disease Classification: A Fusion of Conventional Classifiers, Segmented Images, and Convolutional Neural Networks. In: Anais do XX Encontro Nacional de Inteligência Artificial e Computacional (ENIAC 2023).
- ❑ **Moreira et al. (2022)** VINEVI: A Virtualized Network Vision Architecture for Smart Monitoring of Heterogeneous Applications and Infrastructures. In: Advanced Information Networking and Applications. AINA 2022. Lecture Notes in Networks and Systems, vol 449. Springer.
- ❑ **Moreira et al. (2022)** AgroLens: A low-cost and green-friendly Smart Farm Architecture to support real-time leaf disease diagnostics. Internet of Things.
- ❑ **Antico, Rodrigues Moreira e Moreira (2022)** Evaluating the Potential of Federated Learning for Maize Leaf Disease Prediction. In: Anais do XIX Encontro Nacional de Inteligência Artificial e Computacional (ENIAC 2022).

7.2 Future Work

In future work, several directions and developments will emerge. One significant avenue is the further expansion of the AIaaS architecture to support a wider variety of AI models and services. This could involve integrating advanced ML algorithms, such as transformers or graph neural networks, to enhance the versatility and applicability of the architecture.

Another future work is a detailed analysis of computational costs across a broader range of devices, including emerging hardware technologies such as quantum computers, neuromorphic chips, and optical computing systems. This expanded evaluation could

offer a more comprehensive understanding of the trade-offs associated with deploying deep learning models on different platforms. Additionally, exploring advanced model optimization techniques, such as meta-learning or neural architecture search, could lead to improvements in the training efficiency and model deployment. Efforts to further reduce computational and memory overheads for large models would also be relevant.

We plan to enhance the edge intelligent layer, notably the incorporation of additional authentication methods and expansion of the AI Model Store. Further investigations could fully integrate the AIaaS architecture with standardized edge frameworks, including 3GPP and European Telecommunications Standards Institute (ETSI). We are also working to extend the deployment of our architecture to broader network segments beyond mobile networks. Additionally, we aimed to explore the potential of delivering cognitive services to a wider range of embedded devices, vendors, and vertical industries, thereby enhancing the applicability of AIaaS across various domains and different computer vision tasks.

Bibliography

3GPP. *5G System; Network Data Analytics Services; Stage 3*. France, 2023.

AGGARWAL, P.; MISHRA, N. K.; FATIMAH, B.; SINGH, P.; GUPTA, A.; JOSHI, S. D. COVID-19 image classification using deep learning: Advances, challenges and opportunities. **Computers in Biology and Medicine**, p. 105350, 2022. ISSN 0010-4825. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0010482522001421>.

AGHI, D.; MAZZIA, V.; CHIABERGE, M. Local Motion Planner for Autonomous Navigation in Vineyards with a RGB-D Camera-Based Algorithm and Deep Learning Synergy. **Machines**, v. 8, n. 2, 2020. ISSN 2075-1702. Disponível em: <https://www.mdpi.com/2075-1702/8/2/27>.

AHMED, I.; JEON, G.; PICCIALLI, F. From artificial intelligence to explainable artificial intelligence in industry 4.0: A survey on what, how, and where. **IEEE Transactions on Industrial Informatics**, v. 18, n. 8, p. 5031–5042, 2022.

ALAM, M. U.; RAHMANI, R. Federated Semi-Supervised Multi-Task Learning to Detect COVID-19 and Lungs Segmentation Marking Using Chest Radiography Images and Raspberry Pi Devices: An Internet of Medical Things Application. **Sensors**, v. 21, n. 15, 2021. ISSN 1424-8220. Disponível em: <https://www.mdpi.com/1424-8220/21/15/5025>.

ALAMEH, M.; ABBASS, Y.; IBRAHIM, A.; VALLE, M. Smart Tactile Sensing Systems Based on Embedded CNN Implementations. **Micromachines**, v. 11, n. 1, 2020. ISSN 2072-666X. Disponível em: <https://www.mdpi.com/2072-666X/11/1/103>.

ALBANESE, A.; NARDELLO, M.; BRUNELLI, D. Low-power deep learning edge computing platform for resource constrained lightweight compact UAVs. **Sustainable Computing: Informatics and Systems**, v. 34, p. 100725, 2022. ISSN 2210-5379. Disponível em: <https://www.sciencedirect.com/science/article/pii/S2210537922000609>.

ALHOSANI, K.; ALHASHMI, S. M. Opportunities, challenges, and benefits of AI innovation in government services: a review. **Discover Artificial Intelligence**, v. 4, n. 1, p. 18, Mar 2024. ISSN 2731-0809. Disponível em: <https://doi.org/10.1007/s44163-024-00111-w>.

ALI, A. Q.; SULTAN, A. B. M.; GHANI, A. A. A.; ZULZALIL, H. A Systematic Mapping Study on the Customization Solutions of Software as a Service Applications. **IEEE Access**, v. 7, p. 88196–88217, 2019.

ALIPPI, C.; DISABATO, S.; ROVERI, M. Moving Convolutional Neural Networks to Embedded Systems: The AlexNet and VGG-16 Case. In: **2018 17th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)**. Porto, Portugal: IEEE, 2018. p. 212–223.

AMATO, G.; CARRARA, F.; FALCHI, F.; GENNARO, C.; VAIRO, C. Facial-based Intrusion Detection System with Deep Learning in Embedded Devices. In: **Proceedings of the 2018 International Conference on Sensors, Signal and Image Processing**. New York, NY, USA: Association for Computing Machinery, 2018. (SSIP '18), p. 64–68. ISBN 9781450366205. Disponível em: <https://doi.org/10.1145/3290589.3290598>.

AMIN, M. I.; HAFEEZ, M. A.; TOUSEEF, R.; AWAIS, Q. Person Identification with Masked Face and Thumb Images under Pandemic of COVID-19. In: **2021 7th International Conference on Control, Instrumentation and Automation (ICCIA)**. Tabriz, Iran: IEEE, 2021. p. 1–4.

ANTICO, T.; RODRIGUES MOREIRA, L. F.; MOREIRA, R. Evaluating the Potential of Federated Learning for Maize Leaf Disease Prediction. In: **Anais do XIX Encontro Nacional de Inteligência Artificial e Computacional**. Porto Alegre, RS, Brasil: SBC, 2022. p. 282–293. ISSN 2763-9061. Disponível em: <https://sol.sbc.org.br/index.php/eniac/article/view/22789>.

ARMBRUST, M.; FOX, A.; GRIFFITH, R.; JOSEPH, A. D.; KATZ, R.; KONWINSKI, A.; LEE, G.; PATTERSON, D.; RABKIN, A.; STOICA, I.; ZAHARIA, M. A View of Cloud Computing. **Commun. ACM**, Association for Computing Machinery, New York, NY, USA, v. 53, n. 4, p. 50–58, 2010. ISSN 0001-0782. Disponível em: <https://doi.org/10.1145/1721654.1721672>.

ARREDONDO-VELáZQUEZ, M.; DIAZ-CARMONA, J.; TORRES-HUITZIL, C.; BARRANCO-GUTIéRREZ, A.-I.; PADILLA-MEDINA, A.; PRADO-OLIVAREZ, J. A streaming accelerator of Convolutional Neural Networks for resource-limited applications. **IEICE Electronics Express**, v. 16, n. 23, p. 20190633–20190633, 2019.

ARTHI, N. T.; MUBIN, K. E.; RAHMAN, J.; RAFI, G.; SHEJA, T. T.; REZA, M. T.; ALAM, M. A. Decentralized Federated Learning and Deep Learning Leveraging XAI-Based Approach to Classify Colorectal Cancer. In: **2022 IEEE Asia-Pacific Conference on Computer Science and Data Engineering (CSDE)**. Gold Coast, Australia: IEEE, 2022. p. 1–6.

BACCIU, D.; AKARMAZYAN, S.; ARMENGAUD, E.; BACCO, M.; BRAVOS, G.; CALANDRA, C.; CARLINI, E.; CARTA, A.; CASSARà, P.; COPPOLA, M.; DAVALAS, C.; DAZZI, P.; DEGENNARO, M. C.; SARLI, D. D.; DOBAJ, J.; GALLICCHIO, C.; GIRBAL, S.; GOTTA, A.; GROPPO, R.; LOMONACO, V.; MACHER, G.; MAZZEI, D.; MENCAGLI, G.; MICHAIL, D.; MICHELI, A.; PEROGLIO, R.; PETRONI, S.; POTENZA, R.; POURDANESH, F.; SARDIANOS, C.; TSERPES, K.; TAGLIABÓ, F.; VALTL, J.; VARLAMIS, I.; VELEDAR, O. TEACHING - Trustworthy autonomous cyber-physical applications through human-centred intelligence. In: **2021 IEEE International Conference on Omni-Layer Intelligent Systems (COINS)**. Barcelona, Spain: IEEE, 2021. p. 1–6.

BACCOUR, E.; ALLAHHAM, M. S.; ERBAD, A.; MOHAMED, A.; HUSSEIN, A. R.; HAMDI, M. Zero Touch Realization of Pervasive Artificial Intelligence as a Service in 6G Networks. **IEEE Communications Magazine**, v. 61, n. 2, p. 110–116, 2023.

BACKES, A. R. Pap-smear image classification by using a fusion of texture features. In: **2022 35th SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)**. Natal, Brazil: IEEE, 2022. v. 1, p. 139–144.

BACKES, A. R.; SÁ JUNIOR, J. J. d. M. **Introdução à Visão Computacional Usando MATLAB**. Rio de Janeiro: Alta Books Editora, 2016. ISBN 978-85-508-0023-3.

BALDOMINOS, A.; ALBACETE, E.; SAEZ, Y.; ISASI, P. A scalable machine learning online service for big data real-time analysis. In: **2014 IEEE Symposium on Computational Intelligence in Big Data (CIBD)**. Orlando, FL, USA: IEEE, 2014. p. 1–8.

BARAS, N.; ZIOUZIOS, D.; DASYGENIS, M.; TSANAKTSIDIS, C. A cloud based smart recycling bin for waste classification. In: **2020 9th International Conference on Modern Circuits and Systems Technologies (MOCAST)**. Bremen, Germany: IEEE, 2020. p. 1–4.

BARBOSA, G.; MOREIRA, L.; SOUSA, P. M. de; MOREIRA, R.; BACKES, A. Optimization and Learning Rate Influence on Breast Cancer Image Classification. In: INSTICC. **Proceedings of the 19th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications - Volume 3: VISAPP**. Rome, Italy: SciTePress, 2024. p. 792–799. ISBN 978-989-758-679-8.

BENGIO, Y. Practical recommendations for gradient-based training of deep architectures. In: **Neural Networks: Tricks of the Trade: Second Edition**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012. p. 437–478. ISBN 978-3-642-35289-8. Disponível em: <https://doi.org/10.1007/978-3-642-35289-8_26>.

BERGMANN, M. A.; MOREIRA, L. F. R.; KROHLING, B.; SILVEIRA, T. L. T.; JUNG, C. R.; TANG, J.; FEITOSA, M. V.; GOMES, R. L. B.; SOARES, B. N. An Approach Based on LiDAR and Spherical Images for Automated Vegetation Inspection in Urban Power Distribution Lines. **IEEE Access**, v. 12, p. 105119–105130, 2024.

BERGSTRA, J. S.; BARDENET, R.; BENGIO, Y.; KéGL, B. Algorithms for hyper-parameter optimization. In: SHAWE-TAYLOR, J.; ZEMEL, R. S.; BARTLETT, P. L.; PEREIRA, F.; WEINBERGER, K. Q. (Ed.). **Advances in Neural Information Processing Systems 24**. Curran Associates, Inc., 2011. p. 2546–2554. Disponível em: <http://papers.nips.cc/paper/4443-algorithms-for-hyper-parameter-optimization.pdf>.

BEUTEL, D. J.; TOPAL, T.; MATHUR, A.; QIU, X.; PARCOLLET, T.; LANE, N. D. Flower: A Friendly Federated Learning Research Framework. **CoRR**, abs/2007.14390, 2020. Disponível em: <https://arxiv.org/abs/2007.14390>.

BHATTACHARYA, S.; Reddy Maddikunta, P. K.; PHAM, Q.-V.; GADEKALLU, T. R.; Krishnan S, S. R.; CHOWDHARY, C. L.; ALAZAB, M.; Jalil Piran, M. Deep learning and medical image processing for coronavirus (COVID-19) pandemic: A survey. **Sustainable Cities and Society**, v. 65, p. 102589, 2021. ISSN 2210-6707. Disponível em: <https://www.sciencedirect.com/science/article/pii/S2210670720308076>.

BIANCHINI, M.; SCARSELLI, F. On the Complexity of Neural Network Classifiers: A Comparison Between Shallow and Deep Architectures. **IEEE Transactions on Neural Networks and Learning Systems**, v. 25, n. 8, p. 1553–1565, 2014.

BISCHL, B.; BINDER, M.; LANG, M.; PIELOK, T.; RICHTER, J.; COORS, S.; THOMAS, J.; ULLMANN, T.; BECKER, M.; BOULESTEIX, A.-L.; DENG, D.; LINDAUER, M. Hyperparameter optimization: Foundations, algorithms, best practices, and open challenges. **WIREs Data Mining and Knowledge Discovery**, v. 13, n. 2, p. e1484, 2023. Disponível em: <https://wires.onlinelibrary.wiley.com/doi/abs/10.1002/widm.1484>.

BLOTT, M.; PREUSSER, T. B.; FRASER, N. J.; GAMBARDELLA, G.; O'BRIEN, K.; UMUROGLU, Y.; LEESER, M.; VISSERS, K. FINN-R: An End-to-End Deep-Learning Framework for Fast Exploration of Quantized Neural Networks. **ACM Transactions on Reconfigurable Technology and Systems**, Association for Computing Machinery, New York, NY, USA, v. 11, n. 3, dec 2018. ISSN 1936-7406. Disponível em: <https://doi.org/10.1145/3242897>.

BOAG, S.; DUBE, P.; MAGHRAOUI, K. E.; HERTA, B.; HUMMER, W.; JAYARAM, K. R.; KHALAF, R.; MUTHUSAMY, V.; KALANTAR, M.; VERMA, A. Dependability in a Multi-tenant Multi-framework Deep Learning as-a-Service Platform. In: **2018 48th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W)**. Luxembourg, Luxembourg: IEEE, 2018. p. 43–46.

BRELAND, D. S.; SKRIUBAKKEN, S. B.; DAYAL, A.; JHA, A.; YALAVARTHY, P. K.; CENKERAMADDI, L. R. Deep Learning-Based Sign Language Digits Recognition From Thermal Images With Edge Computing System. **IEEE Sensors Journal**, v. 21, n. 9, p. 10445–10453, 2021.

BRILHANTE, D. d. S.; MANJARRES, J. C.; MOREIRA, R.; VEIGA, L. de O.; REZENDE, J. F. de; MüLLER, F.; KLAUTAU, A.; MENDES, L. L.; FIGUEIREDO, F. A. P. de. A Literature Survey on AI-Aided Beamforming and Beam Management for 5G and 6G Systems. **Sensors**, v. 23, n. 9, 2023. ISSN 1424-8220. Disponível em: <https://www.mdpi.com/1424-8220/23/9/4359>.

BUSHRA, K. F.; AHAMED, M. A.; AHMAD, M. Automated detection of COVID-19 from X-ray images using CNN and Android mobile. **Research on Biomedical Engineering**, v. 37, n. 3, p. 545–552, Sep 2021. ISSN 2446-4740. Disponível em: <https://doi.org/10.1007/s42600-021-00163-2>.

CARDOSO, V.; RODRIGUES MOREIRA, L. F.; MOREIRA, R.; MARI, J. F. Improving Sickle Cell Disease Classification: A Fusion of Conventional Classifiers, Segmented Images, and Convolutional Neural Networks. In: **Anais do XX Encontro Nacional de Inteligência Artificial e Computacional**. Porto Alegre, RS, Brasil: SBC, 2023. ISSN 2763-9061.

CARO, V. D.; BANO, S.; MACHUMILANE, A.; GOTTA, A.; CASSARà, P.; CARTA, A.; SEMOLA, R.; SARDIANOS, C.; CHRONIS, C.; VARLAMIS, I.; TSERPES, K.; LOMONACO, V.; GALLICCHIO, C.; BACCIU, D. Ai-as-a-service toolkit for human-centered intelligence in autonomous driving. In: **2022 IEEE International Conference**

on Pervasive Computing and Communications Workshops and other Affiliated Events (PerCom Workshops). Pisa, Italy: IEEE, 2022. p. 91–93.

CASTILLO, E. A.; AHMADINIA, A. Distributed Deep Convolutional Neural Network For Smart Camera Image Recognition. In: **Proceedings of the 11th International Conference on Distributed Smart Cameras**. New York, NY, USA: Association for Computing Machinery, 2017. (ICDSC 2017), p. 169–173. ISBN 9781450354875. Disponível em: <https://doi.org/10.1145/3131885.3131935>.

CASTRO, D.; HICKSON, S.; BETTADAPURA, V.; THOMAZ, E.; ABOWD, G.; CHRISTENSEN, H.; ESSA, I. Predicting daily activities from egocentric images using deep learning. In: **Proceedings of the 2015 ACM International Symposium on Wearable Computers**. New York, NY, USA: Association for Computing Machinery, 2015. (ISWC '15), p. 75–82. ISBN 9781450335782. Disponível em: <https://doi.org/10.1145/2802083.2808398>.

CERAR, G.; HRIBAR, J. Machine learning operations model store: Optimizing model selection for ai as a service. In: **2023 International Balkan Conference on Communications and Networking (BalkanCom)**. İstanbul, Turkiye: IEEE, 2023. p. 1–5.

CHAKRABORTY, S. K.; A., S.; DUBEY, K.; JAT, D.; CHANDEL, N. S.; POTDAR, R.; RAO, N. G.; KUMAR, D. Development of an optimally designed real-time automatic citrus fruit grading–sorting machine leveraging computer vision-based adaptive deep learning model. **Engineering Applications of Artificial Intelligence**, v. 120, p. 105826, 2023. ISSN 0952-1976. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0952197623000106>.

CHAN, A.; EZELL, C.; KAUFMANN, M.; WEI, K.; HAMMOND, L.; BRADLEY, H.; BLUEMKE, E.; RAJKUMAR, N.; KRUEGER, D.; KOLT, N.; HEIM, L.; ANDERLJUNG, M. Visibility into AI Agents. In: **Proceedings of the 2024 ACM Conference on Fairness, Accountability, and Transparency**. New York, NY, USA: Association for Computing Machinery, 2024. (FAccT '24), p. 958–973. ISBN 9798400704505. Disponível em: <https://doi.org/10.1145/3630106.3658948>.

CHAN, S.; STONE, T.; SZETO, K. P.; CHAN, K. H. PredictionIO: A Distributed Machine Learning Server for Practical Software Development. In: **Proceedings of the 22nd ACM International Conference on Information & Knowledge Management**. New York, NY, USA: Association for Computing Machinery, 2013. (CIKM '13), p. 2493–2496. ISBN 9781450322638.

CHANG, Z.; LIU, S.; XIONG, X.; CAI, Z.; TU, G. A Survey of Recent Advances in Edge-Computing-Powered Artificial Intelligence of Things. **IEEE Internet of Things Journal**, v. 8, n. 18, p. 13849–13875, 2021.

CHEN, A. T.-Y.; BIGLARI-ABHARI, M.; WANG, K. I.-K.; BOUZERDOUM, A.; TIVIVE, F. H. C. Convolutional neural network acceleration with hardware/software codesign. **Applied Intelligence**, v. 48, n. 5, p. 1288–1301, May 2018. ISSN 1573-7497. Disponível em: <https://doi.org/10.1007/s10489-017-1007-z>.

CHEN, F.; LI, S.; HAN, J.; REN, F.; YANG, Z. Review of Lightweight Deep Convolutional Neural Networks. **Archives of Computational Methods in Engineering**, v. 31, n. 4, p. 1915–1937, May 2024. ISSN 1886-1784. Disponível em: <https://doi.org/10.1007/s11831-023-10032-z>.

CHEN, Z.; YANG, J.; CHEN, L.; JIAO, H. Garbage classification system based on improved ShuffleNet v2. **Resources, Conservation and Recycling**, v. 178, p. 106090, 2022. ISSN 0921-3449. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0921344921006984>.

CHOLLET, F. et al. **Keras**. 2015. <https://keras.io>.

CHOUDHARI, O.; CHOPADE, M.; CHOPDE, S.; DABHADKAR, S.; INGALE, V. HARDWARE ACCELERATOR: IMPLEMENTATION OF CNN ON FPGA FOR DIGIT RECOGNITION. In: **2020 24th International Symposium on VLSI Design and Test (VDAT)**. Bhubaneswar, India: IEEE, 2020. p. 1–6.

CHOWDHURY, M. E. H.; RAHMAN, T.; KHANDAKAR, A.; MAZHAR, R.; KADIR, M. A.; MAHBUB, Z. B.; ISLAM, K. R.; KHAN, M. S.; IQBAL, A.; EMADI, N. A.; REAZ, M. B. I.; ISLAM, M. T. Can AI Help in Screening Viral and COVID-19 Pneumonia? **IEEE Access**, v. 8, p. 132665–132676, 2020.

CLARO, M.; VOGADO, L.; VERAS, R.; SANTANA, A.; TAVARES, J.; SANTOS, J.; MACHADO, V. Convolution Neural Network Models for Acute Leukemia Diagnosis. In: **2020 International Conference on Systems, Signals and Image Processing (IWSSIP)**. Niteroi, Brazil: IEEE, 2020. p. 63–68.

CLARO, M. L.; VERAS, R. de M.; SANTANA, A. M.; VOGADO, L. H. S.; Braz Junior, G.; MEDEIROS, F. N. de; TAVARES, J. M. R. Assessing the impact of data augmentation and a combination of CNNs on leukemia classification. **Information Sciences**, v. 609, p. 1010–1029, 2022. ISSN 0020-0255. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0020025522007484>.

COSTELLO, K.; RIMOL, M. **Gartner Identifies the Top 10 Strategic Technology Trends for 2020**. 2019. Disponível em: <https://www.gartner.com/en/newsroom/press-releases/2019-10-21-gartner-identifies-the-top-10-strategic-technology-trends-for-2020>.

CURTIN, B. H.; MATTHEWS, S. J. Deep Learning for Inexpensive Image Classification of Wildlife on the Raspberry Pi. In: **2019 IEEE 10th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)**. New York, NY, USA: IEEE, 2019. p. 0082–0087.

DA ROCHA, E. L.; RODRIGUES, L. F.; MARI, J. F. Maize leaf disease classification using convolutional neural networks and hyperparameter optimization. In: **XVI Workshop de Visão Computacional**. Porto Alegre, RS, Brasil: SBC, 2020. p. 104–110. ISSN 0000-0000. Disponível em: <https://sol.sbc.org.br/index.php/wvc/article/view/13489>.

DA SILVA, M. V.; RODRIGUES, L.; MARI, J. F. Optimizing data augmentation policies for convolutional neural networks based on classification of sickle cells. In: **Anais do XVI Workshop de Visão Computacional**. Porto Alegre, RS, Brasil: SBC, 2020. p. 46–51. ISSN 0000-0000. Disponível em: <https://sol.sbc.org.br/index.php/wvc/article/view/13479>.

DA SILVA, M. V.; SILVA, L. H. F. P.; JUNIOR, J. D. D.; ESCARPINATI, M. C.; BACKES, A. R.; MARI, J. F. Generating synthetic multispectral images using neural style transfer: A study with application in channel alignment. **Computers and Electronics**

in Agriculture, v. 206, p. 107668, 2023. ISSN 0168-1699. Disponível em: <https://www.sciencedirect.com/science/article/pii/S016816992300056X>.

DANOPOULOS, D.; KACHRIS, C.; SOUDRIS, D. Acceleration of image classification with Caffe framework using FPGA. In: **2018 7th International Conference on Modern Circuits and Systems Technologies (MOCAST)**. Thessaloniki, Greece: IEEE, 2018. p. 1–4.

DAS, P. K.; MEHER, S. An efficient deep Convolutional Neural Network based detection and classification of Acute Lymphoblastic Leukemia. **Expert Systems with Applications**, v. 183, p. 115311, 2021. ISSN 0957-4174. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0957417421007405>.

DE FARIA, L. C.; RODRIGUES, L. F.; MARI, J. F. Cell classification using handcrafted features and bag of visual words. In: **2018 Workshop de Visão Computacional (WVC)**. Ilhéus, BA: WVC, 2018. p. 68–73. ISBN 978-85-7455-514-0.

DENG, J.; DONG, W.; SOCHER, R.; L, L.; LI, K.; FEI-FEI, L. Imagenet: A large-scale hierarchical image database. In: **2009 IEEE Conference on Computer Vision and Pattern Recognition**. Miami, FL, USA: IEEE, 2009. p. 248–255. ISSN 1063-6919.

DEVIJVER, P. A.; KITTLER, J. **Pattern recognition: A statistical approach**. London: Prentice Hall, 1982.

DHOUIBI, M.; SALEM, A. K. B.; SAOUD, S. B. CNN for object recognition implementation on FPGA using PYNQ framework. In: **2020 IEEE Eighth International Conference on Communications and Networking (ComNet)**. Hammamet, Tunisia: IEEE, 2020. p. 1–6.

DIF, N.; ATTAOUI, M. O.; ELBERRICHI, Z.; LEBBAH, M.; AZZAG, H. Transfer learning from synthetic labels for histopathological images classification. **Applied Intelligence**, Apr 2021. ISSN 1573-7497. Disponível em: <https://doi.org/10.1007/s10489-021-02425-z>.

DING, C.; LIAO, S.; WANG, Y.; LI, Z.; LIU, N.; ZHUO, Y.; WANG, C.; QIAN, X.; BAI, Y.; YUAN, G.; MA, X.; ZHANG, Y.; TANG, J.; QIU, Q.; LIN, X.; YUAN, B. CirCNN: accelerating and compressing deep neural networks using block-circulant weight matrices. In: **Proceedings of the 50th Annual IEEE/ACM International Symposium on Microarchitecture**. New York, NY, USA: Association for Computing Machinery, 2017. (MICRO-50 '17), p. 395–408. ISBN 9781450349529. Disponível em: <https://doi.org/10.1145/3123939.3124552>.

DISABATO, S.; ROVERI, M. Incremental On-Device Tiny Machine Learning. In: **Proceedings of the 2nd International Workshop on Challenges in Artificial Intelligence and Machine Learning for Internet of Things**. New York, NY, USA: Association for Computing Machinery, 2020. (AIChallengeIoT '20), p. 7–13. ISBN 9781450381345. Disponível em: <https://doi.org/10.1145/3417313.3429378>.

ELSAADOUNY, M.; BAROWSKI, J.; ROLFES, I. FPGA Based Accelerator for Buried Objects Identification. In: **2020 43rd International Conference on Telecommunications and Signal Processing (TSP)**. Milan, Italy: IEEE, 2020. p. 559–562.

ELSHAWI, R.; SAKR, S.; TALIA, D.; TRUNFIO, P. Big Data Systems Meet Machine Learning Challenges: Towards Big Data Science as a Service. **Big Data Research**, v. 14, p. 1–11, 2018. ISSN 2214-5796. Disponível em: <https://www.sciencedirect.com/science/article/pii/S2214579617303957>.

FACELI, K.; LORENA, A. C.; GAMA, J.; CARVALHO, A. C. P. d. L. F. d. **Inteligência Artificial: uma abordagem de Aprendizado de Máquina**. Rio de Janeiro: LTC, 2011.

FAN, H.; WANG, G.; FERIANC, M.; NIU, X.; LUK, W. Static Block Floating-Point Quantization for Convolutional Neural Networks on FPGA. In: **2019 International Conference on Field-Programmable Technology (ICFPT)**. Tianjin, China: IEEE, 2019. p. 28–35.

FEBRIANA, A.; MUCHTAR, K.; DAWOOD, R.; LIN, C.-Y. USK-COFFEE Dataset: A Multi-Class Green Arabica Coffee Bean Dataset for Deep Learning. In: **2022 IEEE International Conference on Cybernetics and Computational Intelligence (CyberneticsCom)**. Malang, Indonesia: IEEE, 2022. p. 469–473.

FIORETTI, L.; CROCE, C. L.; SIVIERO, A.; CLEMMONS, E. **The Global Impact of Artificial Intelligence on the Economy and Jobs: AI will Steer 3.5% of GDP in 2030**. 2024. Https://www.idc.com/getdoc.jsp?containerId=US51057924.

FORCE, L. M.; AL. et. The global burden of childhood and adolescent cancer in 2017: an analysis of the Global Burden of Disease Study 2017. **The Lancet Oncology**, v. 20, n. 9, p. 1211–1225, 2019. ISSN 1470-2045.

FORTUNA, C.; MUŠIĆ, D.; CERAR, G.; ČAMPA, A.; KAPSALIS, P.; MOHORČIČ, M. **On-Premise Artificial Intelligence as a Service for Small and Medium Size Setups**. Cham: Springer International Publishing, 2023. 53–73 p. ISBN 978-3-031-29301-6. Disponível em: <https://doi.org/10.1007/978-3-031-29301-6_3>.

FUJII, T.; SATO, S.; NAKAHARA, H. A threshold neuron pruning for a binarized deep neural network on an FPGA. **IEICE TRANSACTIONS on Information and Systems**, The Institute of Electronics, Information and Communication Engineers, v. 101, n. 2, p. 376–386, 2018.

FUJII, T.; SATO, S.; NAKAHARA, H.; MOTOMURA, M. An fpga realization of a deep convolutional neural network using a threshold neuron pruning. In: WONG, S.; BECK, A. C.; BERTELS, K.; CARRO, L. (Ed.). **Applied Reconfigurable Computing**. Cham: Springer International Publishing, 2017. p. 268–280.

FUKUSHIMA, K. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. **Biological Cybernetics**, v. 36, n. 4, p. 193–202, Apr 1980. ISSN 1432-0770. Disponível em: <https://doi.org/10.1007/BF00344251>.

GAO, Z.; YAO, Y.; WEI, X.; YAN, T.; ZENG, S.; GE, G.; WANG, Y.; ULLAH, A.; REVIRIEGO, P. Reliability evaluation of FPGA based pruned neural networks. **Microelectronics Reliability**, v. 130, p. 114498, 2022. ISSN 0026-2714. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0026271422000221>.

GHOSH, S.; BANDYOPADHYAY, A.; SAHAY, S.; GHOSH, R.; KUNDU, I.; SANTOSH, K. Colorectal Histology Tumor Detection Using Ensemble Deep Neural Network. **Engineering Applications of Artificial Intelligence**, v. 100, p. 104202, 2021. ISSN 0952-1976. Disponível em: <https://www.sciencedirect.com/science/article/pii/S095219762100049X>.

GKONIS, P. K.; NOMIKOS, N.; TRAKADAS, P.; SARAKIS, L.; XYLOURIS, G.; MASIP-BRUIN, X.; MARTRAT, J. Leveraging network data analytics function and machine learning for data collection, resource optimization, security and privacy in 6g networks. **IEEE Access**, v. 12, p. 21320–21336, 2024.

GOEL, A.; AGHAJANZADEH, S.; TUNG, C.; CHEN, S.-H.; THIRUVATHUKAL, G. K.; LU, Y.-H. Modular Neural Networks for Low-Power Image Classification on Embedded Devices. **ACM Trans. Des. Autom. Electron. Syst.**, Association for Computing Machinery, New York, NY, USA, v. 26, n. 1, oct 2020. ISSN 1084-4309. Disponível em: <https://doi.org/10.1145/3408062>.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. Deep Learning. Book in preparation for MIT Press. 2016. Disponível em: <http://www.deeplearningbook.org>.

GRECO, A.; SAGGESE, A.; VENTO, M.; VIGILANTE, V. A Convolutional Neural Network for Gender Recognition Optimizing the Accuracy/Speed Tradeoff. **IEEE Access**, v. 8, p. 130771–130781, 2020.

GUAN, H.; YAP, P.-T.; BOZOKI, A.; LIU, M. Federated learning for medical image analysis: A survey. **Pattern Recognition**, v. 151, p. 110424, 2024. ISSN 0031-3203. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0031320324001754>.

GUNTUPALLI, N.; RUDRAMALLA, V. Artificial intelligence as a service: Providing integrity and confidentiality. In: MORUSUPALLI, R.; DANDIBHOTLA, T. S.; ATLURI, V. V.; WINDRIDGE, D.; LINGRAS, P.; KOMATI, V. R. (Ed.). **Multi-disciplinary Trends in Artificial Intelligence**. Cham: Springer Nature Switzerland, 2023. p. 309–315. ISBN 978-3-031-36402-0.

HADIDI, R.; CAO, J.; WOODWARD, M.; RYOO, M. S.; KIM, H. Real-Time Image Recognition Using Collaborative IoT Devices. In: **Proceedings of the 1st on Reproducible Quality-Efficient Systems Tournament on Co-Designing Pareto-Efficient Deep Learning**. New York, NY, USA: Association for Computing Machinery, 2018. (ReQuEST '18). ISBN 9781450359238. Disponível em: <https://doi.org/10.1145/3229762.3229765>.

HAJIPOUR, V.; HEKMAT, S.; AMINI, M. A value-oriented Artificial Intelligence-as-a-Service business plan using integrated tools and services. **Decision Analytics Journal**, p. 100302, 2023. ISSN 2772-6622. Disponível em: <https://www.sciencedirect.com/science/article/pii/S277266222300142X>.

HAN, S.; LIU, X.; MAO, H.; PU, J.; PEDRAM, A.; HOROWITZ, M. A.; DALLY, W. J. EIE: Efficient Inference Engine on Compressed Deep Neural Network. **SIGARCH Comput. Archit. News**, Association for Computing Machinery, New York, NY, USA, v. 44, n. 3, p. 243–254, jun 2016. ISSN 0163-5964. Disponível em: <https://doi.org/10.1145/3007787.3001163>.

HANHIROVA, J.; KäMäRäINEN, T.; SEPPäLä, S.; SIEKKINEN, M.; HIRVISALO, V.; YLä-JääSKI, A. Latency and throughput characterization of convolutional neural networks for mobile computer vision. In: **Proceedings of the 9th ACM Multimedia Systems Conference**. New York, NY, USA: Association for Computing Machinery, 2018. (MMSys '18), p. 204–215. ISBN 9781450351928. Disponível em: <https://doi.org/10.1145/3204949.3204975>.

HARRIS, B.; BAE, I.; EGGER, B. Architectures and algorithms for on-device user customization of CNNs. **Integration**, v. 67, p. 121–133, 2019. ISSN 0167-9260. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0167926018302700>.

HE, K.; ZHANG, X.; REN, S.; SUN, J. Deep Residual Learning for Image Recognition. In: **2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**. Las Vegas, NV, USA: IEEE, 2016. p. 770–778.

_____. Identity mappings in deep residual networks. In: LEIBE, B.; MATAS, J.; SEBE, N.; WELLING, M. (Ed.). **Computer Vision – ECCV 2016**. Cham: Springer International Publishing, 2016. p. 630–645. ISBN 978-3-319-46493-0.

HELALY, R.; HAJJAJI, M. A.; M'SAHLI, F.; MTIBAA, A. Deep Convolution Neural Network Implementation for Emotion Recognition System. In: **2020 20th International Conference on Sciences and Techniques of Automatic Control and Computer Engineering (STA)**. Monastir, Tunisia: IEEE, 2020. p. 261–265.

HIZUKURI, A.; NAKAYAMA, R.; NARA, M.; SUZUKI, M.; NAMBA, K. Computer-Aided Diagnosis Scheme for Distinguishing Between Benign and Malignant Masses on Breast DCE-MRI Images Using Deep Convolutional Neural Network with Bayesian Optimization. **Journal of Digital Imaging**, v. 34, n. 1, p. 116–123, Feb 2021. ISSN 1618-727X. Disponível em: <https://doi.org/10.1007/s10278-020-00394-2>.

HOLLAND, J. H. **Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence**. Cambridge, MA, USA: MIT Press, 1992. ISBN 0262082136.

HOSNY, K. M.; DARWISH, M. M.; LI, K.; SALAH, A. COVID-19 diagnosis from CT scans and chest X-ray images using low-cost Raspberry Pi. **PLOS ONE**, Public Library of Science, v. 16, n. 5, p. 1–18, 05 2021. Disponível em: <https://doi.org/10.1371/journal.pone.0250688>.

HOSSAIN, D.; IMTIAZ, M. H.; GHOSH, T.; BHASKAR, V.; SAZONOV, E. Real-Time Food Intake Monitoring Using Wearable Egocnetric Camera. In: **2020 42nd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)**. Montreal, QC, Canada: IEEE, 2020. p. 4191–4195.

HOU, D.; HOU, M. R.; HOU, J. On-device Subspace Learning Chest X-Ray Screening. In: **2020 IEEE International Conference on Consumer Electronics (ICCE)**. Las Vegas, NV, USA: IEEE, 2020. p. 1–5.

HOU, W.; LIU, L.; ZHANG, H.; SUN, H.; ZHENG, N. DFSNet: Dividing-fuse deep neural networks with searching strategy for distributed DNN architecture. **Neurocomputing**, v. 483, p. 488–500, 2022. ISSN 0925-2312. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0925231221016076>.

HOWARD, A. G.; ZHU, M.; CHEN, B.; KALENICHENKO, D.; WANG, W.; WEYAND, T.; ANDREETTO, M.; ADAM, H. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. **CoRR**, abs/1704.04861, 2017. Disponível em: <http://arxiv.org/abs/1704.04861>.

HUANG, G.; LIU, Z.; WEINBERGER, K. Q. Densely Connected Convolutional Networks. **CoRR**, abs/1608.06993, 2016. Disponível em: <http://arxiv.org/abs/1608.06993>.

HUSSAIN, F. **Internet of Everything**. Cham: Springer International Publishing, 2017. 1–11 p. ISBN 978-3-319-55405-1. Disponível em: <https://doi.org/10.1007/978-3-319-55405-1_1>.

IANDOLA, F. N.; HAN, S.; MOSKEWICZ, M. W.; ASHRAF, K.; DALLY, W. J.; KEUTZER, K. **SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size**. 2016.

ISHTEYAQ, I.; MUZAFFAR, K.; SHAFI, N.; ALATHBAH, M. A. Unleashing the Power of Tomorrow: Exploration of Next Frontier With 6G Networks and Cutting Edge Technologies. **IEEE Access**, v. 12, p. 29445–29463, 2024.

ISLAM, M. M.; KARRAY, F.; ALHAJJ, R.; ZENG, J. A Review on Deep Learning Techniques for the Diagnosis of Novel Coronavirus (COVID-19). **IEEE Access**, v. 9, p. 30551–30572, 2021.

ISLAMY, F.; MUCHTAR, K.; ARNIA, F.; DAWOOD, R.; FEBRIANA, A.; ELWIREHARDJA, G. N.; PARDAMEAN, B. Performance evaluation of coffee bean binary classification through deep learning techniques. In: **Innovative Technologies in Intelligent Systems and Industrial Applications**. Cham: Springer Nature Switzerland, 2023. p. 311–321. ISBN 978-3-031-29078-7.

ISMAIL, L.; BUYYA, R. Artificial Intelligence Applications and Self-Learning 6G Networks for Smart Cities Digital Ecosystems: Taxonomy, Challenges, and Future Directions. **Sensors**, v. 22, n. 15, 2022. ISSN 1424-8220. Disponível em: <https://www.mdpi.com/1424-8220/22/15/5750>.

JAFFRI, A.; SICULAR, S. **What's New in Artificial Intelligence from the 2023 Gartner Hype Cycle**. 2023. Disponível em: <https://www.gartner.com/en/articles/what-s-new-in-artificial-intelligence-from-the-2023-gartner-hype-cycle>.

JAVADI, S. A.; CLOETE, R.; COBBE, J.; LEE, M. S. A.; SINGH, J. Monitoring Misuse for Accountable 'Artificial Intelligence as a Service'. In: **Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society**. New York, NY, USA: Association for Computing Machinery, 2020. (AIES '20), p. 300–306. ISBN 9781450371100. Disponível em: <https://doi.org/10.1145/3375627.3375873>.

JAYAKODI, N. K.; BELAKARIA, S.; DESHWAL, A.; DOPPA, J. R. Design and Optimization of Energy-Accuracy Tradeoff Networks for Mobile Platforms via Pretrained Deep Models. **ACM Trans. Embed. Comput. Syst.**, Association for Computing Machinery, New York, NY, USA, v. 19, n. 1, feb 2020. ISSN 1539-9087. Disponível em: <https://doi.org/10.1145/3366636>.

JIANG, B.; CHEN, J.; LIU, Y. Single-shot pruning and quantization for hardware-friendly neural network acceleration. **Engineering Applications of Artificial Intelligence**, v. 126, p. 106816, 2023. ISSN 0952-1976. Disponível em: <https://www.sciencedirect.com/science/article/pii/S095219762301000X>.

JIAO, L.; LUO, C.; CAO, W.; ZHOU, X.; WANG, L. Accelerating low bit-width convolutional neural networks with embedded FPGA. In: **2017 27th International Conference on Field Programmable Logic and Applications (FPL)**. Ghent, Belgium: IEEE, 2017. p. 1–4.

KALGAONKAR, P.; EL-SHARKAWY, M. Image Classification with CondenseNeXt for ARM-Based Computing Platforms. In: **2021 IEEE International IOT, Electronics and Mechatronics Conference (IEMTRONICS)**. Toronto, ON, Canada: IEEE, 2021. p. 1–6.

KANG, D.; KIM, E.; BAE, I.; EGGER, B.; HA, S. C-GOOD: C-code Generation Framework for Optimized On-device Deep Learning. In: **2018 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)**. San Diego, CA, USA: IEEE, 2018. p. 1–8.

KANG, W.; KIM, D.; PARK, J. DMS: Dynamic Model Scaling for Quality-Aware Deep Learning Inference in Mobile and Embedded Devices. **IEEE Access**, v. 7, p. 168048–168059, 2019.

KATHER, J. N.; WEIS, C.-A.; BIANCONI, F.; MELCHERS, S. M.; SCHAD, L. R.; GAISER, T.; MARX, A.; ZÖLLNER, F. G. Multi-class texture analysis in colorectal cancer histology. **Scientific Reports**, v. 6, n. 1, p. 27988, Jun 2016. ISSN 2045-2322. Disponível em: <https://doi.org/10.1038/srep27988>.

KAUR, S.; AGGARWAL, H.; RANI, R. Diagnosis of Parkinson's disease using deep CNN with transfer learning and data augmentation. **Multimedia Tools and Applications**, v. 80, n. 7, p. 10113–10139, Mar 2021. ISSN 1573-7721. Disponível em: <https://doi.org/10.1007/s11042-020-10114-1>.

KAVIS, M. **Architecting the Cloud: Design Decisions for Cloud Computing Service Models (SaaS, PaaS, and IaaS)**. New Jersey: Wiley, 2014. (Wiley CIO). ISBN 9781118826461.

KHAN, A.; SOHAIL, A.; ZAHOORA, U.; QURESHI, A. S. A survey of the recent architectures of deep convolutional neural networks. **Artificial Intelligence Review**, v. 53, n. 8, p. 5455–5516, Dec 2020. ISSN 1573-7462. Disponível em: <https://doi.org/10.1007/s10462-020-09825-6>.

KHAN, M.; MEHRAN, M. T.; HAQ, Z. U.; ULLAH, Z.; NAQVI, S. R.; IHSAN, M.; ABBASS, H. Applications of artificial intelligence in COVID-19 pandemic: A comprehensive review. **Expert Systems with Applications**, v. 185, p. 115695, 2021. ISSN 0957-4174. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0957417421010794>.

KHASOGGI, B.; ERMATITA, E.; SAHMIN, S. Efficient mobilenet architecture as image recognition on mobile and embedded devices. **Indonesian Journal of Electrical Engineering and Computer Science**, v. 16, n. 1, p. 389–394, 2019.

KHOSHAVI, N.; MAGHSOUDLOO, M.; ROOHI, A.; SARGOLZAEI, S.; BI, Y. HARDeNN: Hardware-assisted attack-resilient deep neural network architectures. **Microprocessors and Microsystems**, v. 95, p. 104710, 2022. ISSN 0141-9331. Disponível em: <https://www.sciencedirect.com/science/article/pii/S014193312200240X>.

KHUDHAIR, A. B.; GHANI, R. F. IoT Based Smart Video Surveillance System Using Convolutional Neural Network. In: **2020 6th International Engineering Conference "Sustainable Technology and Development" (IEC)**. Erbil, Iraq: IEEE, 2020. p. 163–168.

KILICARSLAN, S.; CELIK, M.; SAHIN Şafak. Hybrid models based on genetic algorithm and deep learning algorithms for nutritional Anemia disease classification. **Biomedical Signal Processing and Control**, v. 63, p. 102231, 2021. ISSN 1746-8094. Disponível em: <https://www.sciencedirect.com/science/article/pii/S174680942030361X>.

KIM, W.; LEE, Y.-K.; JUNG, W.-S.; YOO, D.; KIM, D.-H.; JO, K.-H. An Adaptive Batch-Image Based Driver Status Monitoring System on a Lightweight GPU-Equipped SBC. **IEEE Access**, v. 8, p. 206074–206087, 2020.

KINGMA, D. P.; BA, J. **Adam: A Method for Stochastic Optimization**. 2017.

KONEČNÝ, J.; MCMAHAN, H. B.; RAMAGE, D.; RICHTÁRIK, P. Federated Optimization: Distributed Machine Learning for On-Device Intelligence. **CoRR**, abs/1610.02527, 2016. Disponível em: <http://arxiv.org/abs/1610.02527>.

KRIZHEVSKY, A.; SUTSKEVER, I.; HINTON, G. E. Imagenet classification with deep convolutional neural networks. In: PEREIRA, F.; BURGES, C. J. C.; BOTTOU, L.; WEINBERGER, K. Q. (Ed.). **Advances in Neural Information Processing Systems 25**. Curran Associates, Inc., 2012. p. 1097–1105. Disponível em: <http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf>.

KUMAR, R. et al. Blending roulette wheel selection & rank selection in genetic algorithms. **International Journal of Machine Learning and Computing**, IACSIT Press, v. 2, n. 4, p. 365–370, 2012.

KYRKOU, C.; THEOCHARIDES, T. EmergencyNet: Efficient Aerial Image Classification for Drone-Based Emergency Monitoring Using Atrous Convolutional Feature Fusion. **IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing**, v. 13, p. 1687–1699, 2020.

LABATI, R. D.; PIURI, V.; SCOTTI, F. All-IDB: The acute lymphoblastic leukemia image database for image processing. In: **2011 18th IEEE International Conference on Image Processing**. Brussels, Belgium: IEEE, 2011. p. 2045–2048.

LECUN, Y.; BENGIO, Y.; HINTON, G. Deep learning. **Nature**, v. 521, n. 7553, p. 436–444, May 2015. ISSN 1476-4687. Disponível em: <https://doi.org/10.1038/nature14539>.

LECUN, Y.; BOTTOU, L.; BENGIO, Y.; HAFFNER, P. Gradient-based learning applied to document recognition. **Proceedings of the IEEE**, v. 86, n. 11, p. 2278–2324, Nov 1998. ISSN 0018-9219.

LEE, J. R.; WANG, L.; WONG, A. EmotionNet Nano: An Efficient Deep Convolutional Neural Network Design for Real-Time Facial Expression Recognition. **Frontiers in Artificial Intelligence**, v. 3, 2021. ISSN 2624-8212. Disponível em: <https://www.frontiersin.org/articles/10.3389/frai.2020.609673>.

LEE, S.; LEE, J. Compressed Learning of Deep Neural Networks for OpenCL-Capable Embedded Systems. **Applied Sciences**, v. 9, n. 8, 2019. ISSN 2076-3417. Disponível em: <https://www.mdpi.com/2076-3417/9/8/1669>.

LEE, S.; NIRJON, S. Neuro.ZERO: a zero-energy neural network accelerator for embedded sensing and inference systems. In: **Proceedings of the 17th Conference on Embedded Networked Sensor Systems**. New York, NY, USA: Association for Computing Machinery, 2019. (SenSys '19), p. 138–152. ISBN 9781450369503. Disponível em: <https://doi.org/10.1145/3356250.3360030>.

_____. Fast and scalable in-memory deep multitask learning via neural weight virtualization. In: **Proceedings of the 18th International Conference on Mobile Systems, Applications, and Services**. New York, NY, USA: Association for Computing Machinery, 2020. (MobiSys '20), p. 175–190. ISBN 9781450379540. Disponível em: <https://doi.org/10.1145/3386901.3388947>.

LEWICKI, K.; LEE, M. S. A.; COBBE, J.; SINGH, J. Out of Context: Investigating the Bias and Fairness Concerns of "Artificial Intelligence as a Service". In: **Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems**. New York, NY, USA: Association for Computing Machinery, 2023. (CHI '23). ISBN 9781450394215. Disponível em: <https://doi.org/10.1145/3544548.3581463>.

LI, G.; ZHANG, J.; ZHANG, M.; WU, R.; CAO, X.; LIU, W. Efficient depthwise separable convolution accelerator for classification and UAV object detection. **Neurocomputing**, v. 490, p. 1–16, 2022. ISSN 0925-2312. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0925231222002387>.

LI, H.; OTA, K.; DONG, M. Learning IoT in Edge: Deep Learning for the Internet of Things with Edge Computing. **IEEE Network**, v. 32, n. 1, p. 96–101, 2018.

LI, L. E.; CHEN, E.; HERMANN, J.; ZHANG, P.; WANG, L. Scaling Machine Learning as a Service. In: HARDGROVE, C.; DORARD, L.; THOMPSON, K.; DOUETTEAU, F. (Ed.). **Proceedings of The 3rd International Conference on Predictive Applications and APIs**. Boston, USA: PMLR, 2017. (Proceedings of Machine Learning Research, v. 67), p. 14–29. Disponível em: <https://proceedings.mlr.press/v67/li17a.html>.

LI, T.; SAHU, A. K.; ZAHEER, M.; SANJABI, M.; TALWALKAR, A.; SMITH, V. Federated Optimization in Heterogeneous Networks. In: DHILLON, I.; PAPAILIOPOULOS, D.; SZE, V. (Ed.). **Proceedings of Machine Learning and Systems**. Austin, TX, USA: MLSys Conference, 2020. v. 2, p. 429–450. Disponível em: <https://proceedings.mlsys.org/paper_files/paper/2020/file/1f5fe83998a09396ebe6477d9475ba0c-Paper.pdf>.

LI, X.; LI, C.; ZHU, D. COVID-MobileXpert: On-Device COVID-19 Patient Triage and Follow-up using Chest X-rays. In: **2020 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)**. Seoul, Korea (South): IEEE, 2020. p. 1063–1067.

LI, X.; LI, W.; YANG, Q.; YAN, W.; ZOMAYA, A. Y. Building an Online Defect Detection System for Large-scale Photovoltaic Plants. In: **Proceedings of the 6th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation**. New York, NY, USA: Association for Computing Machinery, 2019. (BuildSys '19), p. 253–262. ISBN 9781450370059. Disponível em: <https://doi.org/10.1145/3360322.3360835>.

LI, X.; YIN, Z.; XU, F.; ZHANG, F.; XU, G. Design and implementation of neural network computing framework on Zynq SoC embedded platform. **Procedia Computer Science**, v. 183, p. 512–518, 2021. ISSN 1877-0509. Proceedings of the 10th International Conference of Information and Communication Technology. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1877050921005676>.

LIANG, X.; ZHAO, X.; ZHAO, C.; JIANG, N.; TANG, M.; WANG, J. Task Decoupled Knowledge Distillation For Lightweight Face Detectors. In: **Proceedings of the 28th ACM International Conference on Multimedia**. New York, NY, USA: Association for Computing Machinery, 2020. (MM '20), p. 2184–2192. ISBN 9781450379885. Disponível em: <https://doi.org/10.1145/3394171.3414069>.

LIASHCHYNSKYI, P.; LIASHCHYNSKYI, P. Grid Search, Random Search, Genetic Algorithm: A Big Comparison for NAS. **CoRR**, abs/1912.06059, 2019. Disponível em: <http://arxiv.org/abs/1912.06059>.

LIN, Z.; YIH, M.; OTA, J. M.; OWENS, J. D.; MUYAN-ÖZçELIK, P. Benchmarking Deep Learning Frameworks and Investigating FPGA Deployment for Traffic Sign Classification and Detection. **IEEE Transactions on Intelligent Vehicles**, v. 4, n. 3, p. 385–395, 2019.

LINS, S.; PANDL, K. D.; TEIGELER, H.; THIEBES, S.; BAYER, C.; SUNYAEV, A. Artificial Intelligence as a Service. **Business & Information Systems Engineering**, v. 63, n. 4, p. 441–456, Aug 2021. ISSN 1867-0202. Disponível em: <https://doi.org/10.1007/s12599-021-00708-w>.

LIU, B.; LV, N.; GUO, Y.; LI, Y. Recent advances on federated learning: A systematic survey. **Neurocomputing**, v. 597, p. 128019, 2024. ISSN 0925-2312. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0925231224007902>.

LIU, J.; LIU, J.; DU, W.; LI, D. Performance Analysis and Characterization of Training Deep Learning Models on Mobile Device. In: **2019 IEEE 25th International Conference on Parallel and Distributed Systems (ICPADS)**. Tianjin, China: IEEE, 2019. p. 506–515.

LIU, S.; CHU, R. S. W.; WANG, X.; LUK, W. Optimizing CNN-Based Hyperspectral Image Classification on FPGAs. In: HOCHBERGER, C.; NELSON, B.; KOCH, A.; WOODS, R.; DINIZ, P. (Ed.). **Applied Reconfigurable Computing**. Cham: Springer International Publishing, 2019. p. 17–31. ISBN 978-3-030-17227-5.

LIU, X.; JIA, Z.; HOU, X.; FU, M.; MA, L.; SUN, Q. Real-time Marine Animal Images Classification by Embedded System Based on Mobilenet and Transfer Learning. In: **OCEANS 2019 - Marseille**. Marseille, France: IEEE, 2019. p. 1–5.

LO, S.-C.; LOU, S.-L.; LIN, J.-S.; FREEDMAN, M.; CHIEN, M.; MUN, S. Artificial convolution neural network techniques and applications for lung nodule detection. **IEEE Transactions on Medical Imaging**, v. 14, n. 4, p. 711–718, 1995.

LOMONACO, V.; CARO, V. D.; GALLICCHIO, C.; CARTA, A.; SARDIANOS, C.; VARLAMIS, I.; TSERPES, K.; COPPOLA, M.; MARMPENA, M.; POLITI, S.; SCHOITSCH, E.; BACCIU, D. AI-Toolkit: A Microservices Architecture for Low-Code Decentralized Machine Intelligence. In: **2023 IEEE International Conference on Acoustics, Speech, and Signal Processing Workshops (ICASSPW)**. Rhodes Island, Greece: IEEE, 2023. p. 1–5.

LONI, M.; DANESHTALAB, M.; SJöDIN, M. ADONN: Adaptive Design of Optimized Deep Neural Networks for Embedded Systems. In: **2018 21st Euromicro Conference on Digital System Design (DSD)**. Prague, Czech Republic: IEEE, 2018. p. 397–404.

LONI, M.; MAJD, A.; LONI, A.; DANESHTALAB, M.; SJöDIN, M.; TROUBITSYNA, E. Designing Compact Convolutional Neural Network for Embedded Stereo Vision Systems. In: **2018 IEEE 12th International Symposium on Embedded Multicore/Many-core Systems-on-Chip (MCSoC)**. Hanoi, Vietnam: IEEE, 2018. p. 244–251.

LUO, Y.; CAI, X.; QI, J.; GUO, D.; CHE, W. FPGA–accelerated CNN for real-time plant disease identification. **Computers and Electronics in Agriculture**, v. 207, p. 107715, 2023. ISSN 0168-1699. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0168169923001035>.

MA, B.; YANG, J.; WONG, F. K. Y.; WONG, A. K. C.; MA, T.; MENG, J.; ZHAO, Y.; WANG, Y.; LU, Q. Artificial intelligence in elderly healthcare: A scoping review. **Ageing Research Reviews**, v. 83, p. 101808, 2023. ISSN 1568-1637. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1568163722002501>.

MA, N.; ZHANG, X.; ZHENG, H.-T.; SUN, J. Shufflenet v2: Practical guidelines for efficient cnn architecture design. In: FERRARI, V.; HEBERT, M.; SMINCHISESCU, C.; WEISS, Y. (Ed.). **Computer Vision – ECCV 2018**. Cham: Springer International Publishing, 2018. p. 122–138. ISBN 978-3-030-01264-9.

MADADUM, H.; BECERIKLI, Y. FPGA-Based Optimized Convolutional Neural Network Framework for Handwritten Digit Recognition. In: **2019 1st International Informatics and Software Engineering Conference (UBMYK)**. Ankara, Turkey: IEEE, 2019. p. 1–6.

MAGHDED, H. S.; GHAFOOR, K. Z.; SADIQ, A. S.; CURRAN, K.; RAWAT, D. B.; RABIE, K. A Novel AI-enabled Framework to Diagnose Coronavirus COVID-19 using Smartphone Embedded Sensors: Design Study. In: **2020 IEEE 21st International Conference on Information Reuse and Integration for Data Science (IRI)**. Las Vegas, NV, USA: IEEE, 2020. p. 180–187.

MANVI, S. S.; KRISHNA SHYAM, G. Resource management for Infrastructure as a Service (IaaS) in cloud computing: A survey. **Journal of Network and Computer Applications**, v. 41, p. 424–440, 2014. ISSN 1084-8045. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1084804513002099>.

MARCO, V. S.; TAYLOR, B.; WANG, Z.; ELKHATIB, Y. Optimizing Deep Learning Inference on Embedded Systems Through Adaptive Model Selection. **ACM Trans. Embed. Comput. Syst.**, Association for Computing Machinery, New York, NY, USA, v. 19, n. 1, feb 2020. ISSN 1539-9087. Disponível em: <https://doi.org/10.1145/3371154>.

MARCOS, A. P.; RODOVALHO, N. L. S.; BACKES, A. R. Coffee leaf rust detection using genetic algorithm. In: **2019 XV Workshop de Visão Computacional (WVC)**. São Bernardo do Campo, Brazil: IEEE, 2019. p. 16–20.

MA'TOUQ, J.; ALNUMAN, N. Comparative analysis of features and classification techniques in breast cancer detection for biglycan biomarker images. **Cancer Biomarkers**, IOS Press, v. 40, p. 263–273, 2024. ISSN 1875-8592. 3-4. Disponível em: <https://doi.org/10.3233/CBM-230544>.

MAZOUZ, A.; BRIDGES, C. P. Automated Offline Design-Space Exploration and Online Design Reconfiguration for CNNs. In: **2020 IEEE Conference on Evolving and Adaptive Intelligent Systems (EAIS)**. Bari, Italy: IEEE, 2020. p. 1–9.

MCMAHAN, B.; MOORE, E.; RAMAGE, D.; HAMPSON, S.; ARCAS, B. A. y. Communication-Efficient Learning of Deep Networks from Decentralized Data. In: SINGH, A.; ZHU, J. (Ed.). **Proceedings of the 20th International Conference on Artificial Intelligence and Statistics**. PMLR, 2017. (Proceedings of Machine Learning Research, v. 54), p. 1273–1282. Disponível em: <https://proceedings.mlr.press/v54/mcmahan17a.html>.

MELLIT, A. An embedded solution for fault detection and diagnosis of photovoltaic modules using thermographic images and deep convolutional neural networks. **Engineering Applications of Artificial Intelligence**, v. 116, p. 105459, 2022. ISSN 0952-1976. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0952197622004493>.

MERLUZZI, M.; BORSOS, T.; RAJATHEVA, N.; BENCZÚR, A. A.; FARHADI, H.; YASSINE, T.; MÜECK, M. D.; BARMPOUNAKIS, S.; STRINATI, E. C.; DAMPAHALAGE, D.; DEMESTICHAS, P.; DUCANGE, P.; FILIPPOU, M. C.; BALTAR, L. G.; HARALDSON, J.; KARAÇAY, L.; KORPI, D.; LAMPROUSI, V.; MARCELLONI, F.; MOHAMMADI, J.; RAJAPAKSHA, N.; RENDA, A.; UUSITALO, M. A. The Hexa-X Project Vision on Artificial Intelligence and Machine Learning-Driven Communication and Computation Co-Design for 6G. **IEEE Access**, v. 11, p. 65620–65648, 2023.

MESSAOUD, S.; BOUAAFIA, S.; MARAOUI, A.; AMMARI, A. C.; KHRIJI, L.; MACHHOUT, M. Deep convolutional neural networks-based Hardware–Software on-chip system for computer vision application. **Computers & Electrical Engineering**, v. 98, p. 107671, 2022. ISSN 0045-7906. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0045790621005930>.

MILLER, B. L.; GOLDBERG, D. E. Genetic Algorithms, Selection Schemes, and the Varying Effects of Noise. **Evolutionary Computation**, v. 4, n. 2, p. 113–131, 1996. Disponível em: <https://doi.org/10.1162/evco.1996.4.2.113>.

MONDAL, S. Implementation of Human Face and Spoofing Detection Using Deep Learning on Embedded Hardware. In: **2020 11th International Conference on Computing, Communication and Networking Technologies (ICCCNT)**. Kharagpur, India: IEEE, 2020. p. 1–7.

MORADIAN, N.; OCHS, H. D.; SEDIKIES, C.; HAMBLIN, M. R.; CAMARGO, C. A.; MARTINEZ, J. A.; BIAMONTE, J. D.; ABDOLLAHI, M.; TORRES, P. J.; NIETO, J. J.; OGINO, S.; SEYMOUR, J. F.; ABRAHAM, A.; CAUDA, V.; GUPTA, S.; RAMAKRISHNA, S.; SELLKE, F. W.; SOROOSHIAN, A.; HAYES, A. W.; MARTINEZ-URBISTONDO, M.; GUPTA, M.; AZADBAKHT, L.; ESMAILLZADEH, A.; KELISHADI, R.; ESTEGHAMATI, A.; EMAM-DJOMEH, Z.; MAJDZADEH, R.; PALIT, P.; BADALI, H.; RAO, I.; SABOURY, A. A.; RAO, L. J. M.; AHMADIEH, H.; MONTAZERI, A.; FADINI, G. P.; PAULY, D.; THOMAS, S.; MOOSAVI-MOVAHED, A. A.; AGHAMOHAMMADI, A.; BEHMANESH, M.; RAHIMI-MOVAGHAR, V.; GHAVAMI, S.; MEHRAN, R.; UDDIN, L. Q.; HERRATH, M. V.; MOBASHER, B.; REZAEI, N. The urgent need for integrated science to fight COVID-19 pandemic and beyond. **Journal of Translational Medicine**, v. 18, n. 1, p. 205, May 2020. ISSN 1479-5876. Disponível em: <https://doi.org/10.1186/s12967-020-02364-2>.

MOREIRA, R.; CUNHA, H. G. V. O. da; RODRIGUES MOREIRA, L. F.; SILVA, F. d. O. VINEVI: A Virtualized Network Vision Architecture for Smart Monitoring of Heterogeneous Applications and Infrastructures. In: BAROLLI, L.; HUSSAIN, F.; ENOKIDO, T. (Ed.). **Advanced Information Networking and Applications**. Cham: Springer International Publishing, 2022. p. 529–541. ISBN 978-3-030-99584-3.

MOREIRA, R.; RODRIGUES, L. F.; ROSA, P. F.; AGUIAR, R. L.; SILVA, F. d. O. Packet Vision: a convolutional neural network approach for network traffic classification. In: **2020 33rd SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)**. Porto de Galinhas, Brazil: IEEE, 2020. p. 256–263.

MOREIRA, R.; RODRIGUES MOREIRA, L. F.; MUNHOZ, P. L. A.; LOPES, E. A.; RUAS, R. A. A. AgroLens: A low-cost and green-friendly Smart Farm Architecture to support real-time leaf disease diagnostics. **Internet of Things**, v. 19, p. 100570, 2022. ISSN 2542-6605. Disponível em: <https://www.sciencedirect.com/science/article/pii/S2542660522000634>.

MOREIRA, R.; RODRIGUES MOREIRA, L. F.; OLIVEIRA SILVA, F. de. An intelligent network monitoring approach for online classification of Darknet traffic. **Computers and Electrical Engineering**, v. 110, p. 108852, 2023. ISSN 0045-7906. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0045790623002768>.

MUMBELLI, J. D.; GUARNERI, G. A.; LOPES, Y. K.; CASANOVA, D.; TEIXEIRA, M. An application of Generative Adversarial Networks to improve automatic inspection in automotive manufacturing. **Applied Soft Computing**, v. 136, p. 110105, 2023. ISSN 1568-4946. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1568494623001230>.

NADAR, A.; HäRRI, J. Enhancing Network Data Analytics Functions: Integrating AIaaS with ML Model Provisioning. In: **2024 22nd Mediterranean Communication and Computer Networking Conference (MedComNet)**. Nice, France: IEEE, 2024. p. 1–4.

NAHAVANDI, D.; ALIZADEHSANI, R.; KHOSRAVI, A.; ACHARYA, U. R. Application of artificial intelligence in wearable devices: Opportunities and challenges. **Computer Methods and Programs in Biomedicine**, v. 213, p. 106541, 2022. ISSN 0169-2607. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0169260721006155>.

NAKAHARA, H.; FUJII, T.; SATO, S. A fully connected layer elimination for a binarizec convolutional neural network on an FPGA. In: **2017 27th International Conference on Field Programmable Logic and Applications (FPL)**. Ghent, Belgium: IEEE, 2017. p. 1–4.

NAKAJIMA, K.; MOSHNYAGA, V.; HASHIMOTO, K. A comparative study of conventional and CNN-based implementations of facial recognition on Raspberry-Pi. In: **2021 IEEE 19th World Symposium on Applied Machine Intelligence and Informatics (SAMI)**. Herl'any, Slovakia: IEEE, 2021. p. 000217–000222.

NANNI, L.; GHIDONI, S.; BRAHNAM, S. Ensemble of convolutional neural networks for bioimage classification. **Applied Computing and Informatics**, Emerald Publishing Limited, v. 17, n. 1, p. 19–35, Jul 2020. Disponível em: <https://doi.org/10.1016/j.aci.2018.06.002>.

NAPISA, K.; MABABANGLOOB, G. R.; LUBAG, M.; II, R. C.; REDILLAS, M. M. Explainable and Interpretable Artificial Intelligence as a Service for Green Smart Cities and Communities. In: **2023 IEEE 15th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment, and Management (HNICEM)**. Coron, Palawan, Philippines: IEEE, 2023. p. 1–5.

NISHIO, M.; NOGUCHI, S.; MATSUO, H.; MURAKAMI, T. Automatic classification between COVID-19 pneumonia, non-COVID-19 pneumonia, and the healthy on chest X-ray image: combination of data augmentation methods. **Scientific Reports**, v. 10, n. 1, p. 17532, Oct 2020. ISSN 2045-2322. Disponível em: <https://doi.org/10.1038/s41598-020-74539-2>.

O'GRADY, M.; LANGTON, D.; O'HARE, G. Edge computing: A tractable model for smart agriculture? **Artificial Intelligence in Agriculture**, v. 3, p. 42–51, 2019. ISSN 2589-7217. Disponível em: <https://www.sciencedirect.com/science/article/pii/S2589721719300339>.

OLIVEIRA, J.; ALMEIDA, J.; MACEDO, D.; NOGUEIRA, J. Um framework NWDAF para algoritmos de análise de dados de rede 5G e além. In: **Anais do IV Workshop de Redes 6G**. Porto Alegre, RS, Brasil: SBC, 2024. p. 9–14. ISSN 0000-0000. Disponível em: <https://sol.sbc.org.br/index.php/w6g/article/view/29774>.

O'MALLEY, T.; BURSZTEIN, E.; LONG, J.; CHOLLET, F.; JIN, H.; INVERNIZZI, L. et al. **KerasTuner**. 2019. <https://github.com/keras-team/keras-tuner>.

OYEDOTUN, O. K.; ISMAEIL, K. A.; AOUADA, D. Training very deep neural networks: Rethinking the role of skip connections. **Neurocomputing**, v. 441, p. 105–117, 2021. ISSN 0925-2312. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0925231221002332>.

PACHECO, R. G.; COUTO, R. S.; SIMEONE, O. On the impact of deep neural network calibration on adaptive edge offloading for image classification. **Journal of Network and Computer Applications**, v. 217, p. 103679, 2023. ISSN 1084-8045. Disponível em: <https://www.sciencedirect.com/science/article/pii/S108480452300098X>.

PACHóN, C. G.; BALLESTEROS, D. M.; RENZA, D. An efficient deep learning model using network pruning for fake banknote recognition. **Expert Systems with Applications**, v. 233, p. 120961, 2023. ISSN 0957-4174. Disponível em: <https://www.sciencedirect.com/science/article/pii/S095741742301463X>.

PALADINI, E.; VANTAGGIATO, E.; BOUGOURZI, F.; DISTANTE, C.; HADID, A.; TALEB-AHMED, A. Two Ensemble-CNN Approaches for Colorectal Cancer Tissue Type Classification. **Journal of Imaging**, v. 7, n. 3, 2021. ISSN 2313-433X. Disponível em: <https://www.mdpi.com/2313-433X/7/3/51>.

PALURU, N.; DAYAL, A.; JENSSEN, H. B.; SAKINIS, T.; CENKERAMADDI, L. R.; PRAKASH, J.; YALAVARTHY, P. K. Anam-Net: Anamorphic Depth Embedding-Based Lightweight CNN for Segmentation of Anomalies in COVID-19 Chest CT Images. **IEEE Transactions on Neural Networks and Learning Systems**, v. 32, n. 3, p. 932–946, 2021.

PAN, K.; ONG, Y.-S.; GONG, M.; LI, H.; QIN, A.; GAO, Y. Differential privacy in deep learning: A literature survey. **Neurocomputing**, v. 589, p. 127663, 2024. ISSN 0925-2312. Disponível em: <https://www.sciencedirect.com/science/article/pii/S092523122400434X>.

PANDL, K. D.; TEIGELER, H.; LINS, S.; THIEBES, S.; SUNYAEV, A. Drivers and inhibitors for organizations' intention to adopt artificial intelligence as a service. In: . Hawaii, USA: Proceedings of the 54th Hawaii International Conference on System Sciences, 2021.

PARASKEVOULAKOU, E.; KYRIAZIS, D. ML-FaaS: Toward Exploiting the Serverless Paradigm to Facilitate Machine Learning Functions as a Service. **IEEE Transactions on Network and Service Management**, v. 20, n. 3, p. 2110–2123, 2023.

PASZKE, A.; GROSS, S.; MASSA, F.; LERER, A.; BRADBURY, J.; CHANAN, G.; KILLEEN, T.; LIN, Z.; GIMELSHEIN, N.; ANTIGA, L.; DESMAISON, A.; KOPF, A.; YANG, E.; DEVITO, Z.; RAISON, M.; TEJANI, A.; CHILAMKURTHY, S.; STEINER, B.; FANG, L.; BAI, J.; CHINTALA, S. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In: WALLACH, H.; LAROCHELLE, H.; BEYGELZIMER, A.; ALCHé-BUC, F. d'; FOX, E.; GARNETT, R. (Ed.). **Advances in Neural Information Processing Systems 32**. Curran Associates, Inc., 2019. p. 8024–8035. Disponível em: <http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf>.

PENG, P.; MINGYU, Y.; WEISHENG, X. Running 8-bit dynamic fixed-point convolutional neural network on low-cost ARM platforms. In: **2017 Chinese Automation Congress (CAC)**. Jinan, China: IEEE, 2017. p. 4564–4568.

PEREIRA NETO, R.; SOUSA, P.; RODRIGUES MOREIRA, L. F.; GOD, P.; MARI, J. F. Enhancing green coffee quality assessment through deep learning. **Anais do Workshop de Visão Computacional (WVC)**, p. 84–89, 2023. ISSN 0000-0000.

PEREIRA, R. M.; BERTOLINI, D.; TEIXEIRA, L. O.; SILLA, C. N.; COSTA, Y. M. COVID-19 identification in chest X-ray images on flat and hierarchical classification scenarios. **Computer Methods and Programs in Biomedicine**, v. 194, p. 105532, 2020. ISSN 0169-2607. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0169260720309664>.

PHAM, T. D.; PARK, C.; NGUYEN, D. T.; BATCHULUUN, G.; PARK, K. R. Deep Learning-Based Fake-Banknote Detection for the Visually Impaired People Using Visible-Light Images Captured by Smartphone Cameras. **IEEE Access**, v. 8, p. 63144–63161, 2020.

PIVOTO, D. G. S.; REZENDE, T. T.; FACINA, M. S. P.; MOREIRA, R.; SILVA, F. d. O.; CARDOSO, K. V.; CORREA, S. L.; ARAUJO, A. V. D.; SILVA, R. S.; NETO, H. S.; TEJERINA, G. R. L.; ALBERTI, A. M. A Detailed Relevance Analysis of Enabling Technologies for 6G Architectures. **IEEE Access**, p. 1–1, 2023.

PIYASENA, D.; WICKRAMASINGHE, R.; PAUL, D.; LAM, S.-K.; WU, M. Reducing Dynamic Power in Streaming CNN Hardware Accelerators by Exploiting Computational Redundancies. In: **2019 29th International Conference on Field Programmable Logic and Applications (FPL)**. Barcelona, Spain: IEEE, 2019. p. 354–359.

POLYAK, B. Some methods of speeding up the convergence of iteration methods. **USSR Computational Mathematics and Mathematical Physics**, v. 4, n. 5, p. 1 – 17, 1964. ISSN 0041-5553.

PONTI, M. A.; RIBEIRO, L. S. F.; NAZARE, T. S.; BUI, T.; COLLOMOSSE, J. Everything You Wanted to Know about Deep Learning for Computer Vision but Were Afraid to Ask. In: **2017 30th SIBGRAPI Conference on Graphics, Patterns and Images Tutorials (SIBGRAPI-T)**. Niteroi, Brazil: IEEE, 2017. p. 17–41. ISSN 2474-0705.

QIN, Q.; REN, J.; YU, J.; WANG, H.; GAO, L.; ZHENG, J.; FENG, Y.; FANG, J.; WANG, Z. To Compress, or Not to Compress: Characterizing Deep Learning Model Compression for Embedded Inference. In: **2018 IEEE Intl Conf on Parallel & Distributed Processing with Applications, Ubiquitous Computing & Communications, Big Data & Cloud Computing, Social Computing & Networking, Sustainable Computing & Communications (ISPA/IUCC/BDCloud/SocialCom/SustainCom)**. Melbourne, VIC, Australia: IEEE, 2018. p. 729–736.

RAHMAN, T.; KHANDAKAR, A.; QIBLAWEY, Y.; TAHIR, A.; KIRANYAZ, S.; ABUL KASHEM, S. B.; ISLAM, M. T.; AL MAADEED, S.; ZUGHAIER, S. M.; KHAN, M. S.; CHOWDHURY, M. E. Exploring the effect of image enhancement techniques on COVID-19 detection using chest X-ray images. **Computers in Biology and Medicine**, v. 132, p. 104319, 2021. ISSN 0010-4825. Disponível em: <https://www.sciencedirect.com/science/article/pii/S001048252100113X>.

RANDHE, P. H.; DURBHA, S. S.; YOUNAN, N. H. Embedded high performance computing for on-board hyperspectral image classification. In: **2016 8th Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing (WHISPERS)**. Los Angeles, CA, USA: IEEE, 2016. p. 1–5.

RANGARAJAN, A. K.; RAMACHANDRAN, H. K. A preliminary analysis of AI based smartphone application for diagnosis of COVID-19 using chest X-ray images. **Expert Systems with Applications**, v. 183, p. 115401, 2021. ISSN 0957-4174. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0957417421008241>.

RAY, P. P.; DASH, D.; DE, D. Edge computing for Internet of Things: A survey, e-healthcare case study and future direction. **Journal of Network and Computer Applications**, v. 140, p. 1–22, 2019. ISSN 1084-8045. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1084804519301651>.

REDDI, S.; CHARLES, Z.; ZAHEER, M.; GARRETT, Z.; RUSH, K.; KONEčNý, J.; KUMAR, S.; MCMAHAN, H. B. **Adaptive Federated Optimization**. 2021. Disponível em: <https://arxiv.org/abs/2003.00295>.

RIBEIRO, M.; GROLINGER, K.; CAPRETZ, M. A. MLaaS: Machine Learning as a Service. In: **2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA)**. Miami, FL, USA: IEEE, 2015. p. 896–902.

RIZVI, S. T. H.; CABODI, G.; PATTI, D.; FRANCINI, G. GPGPU Accelerated Deep Object Classification on a Heterogeneous Mobile Platform. **Electronics**, v. 5, n. 4, 2016. ISSN 2079-9292. Disponível em: <https://www.mdpi.com/2079-9292/5/4/88>.

ROCHA, M. M. M.; LANDINI, G.; FLORINDO, J. B. Medical image classification using a combination of features from convolutional neural networks. **Multimedia Tools and Applications**, v. 82, n. 13, p. 19299–19322, May 2023. ISSN 1573-7721. Disponível em: <https://doi.org/10.1007/s11042-022-14206-y>.

RODRIGUES, L.; RODRIGUES, L.; SILVA, D. da; MARI, J. F. Evaluating Convolutional Neural Networks for COVID-19 classification in chest X-ray images. In: **Anais do XVI Workshop de Visão Computacional**. Porto Alegre, RS, Brasil: SBC, 2020. p. 52–57. ISSN 0000-0000. Disponível em: <https://sol.sbc.org.br/index.php/wvc/article/view/13480>.

RODRIGUES, L. F.; BACKES, A. R.; TRAVENÇOLO, B. A. N.; OLIVEIRA, G. M. B. de. Optimizing a Deep Residual Neural Network with Genetic Algorithm for Acute Lymphoblastic Leukemia Classification. **Journal of Digital Imaging**, v. 35, n. 3, p. 623–637, Jun 2022. ISSN 1618-727X. Disponível em: <https://doi.org/10.1007/s10278-022-00600-3>.

RODRIGUES, L. F.; NALDI, M. C.; MARI, J. F. Comparing convolutional neural networks and preprocessing techniques for hep-2 cell classification in immunofluorescence images. **Computers in Biology and Medicine**, v. 116, p. 103542, 2020. ISSN 0010-4825. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0010482519303993>.

RODRIGUES, L. F.; SILVA, J. H.; GONDIM, P. H. C. C.; MARI, J. F. Leukocytes classification in microscopy images for acute lymphoblastic leukemia identification. In: **XII Workshop de Visão Computacional**. Campo Grande, MS, Brazil: WVC, 2016. p. 68–73. ISSN 334-338.

RODRIGUES MOREIRA, L. F.; MOREIRA, R.; MARTINS, E. T.; JANSEN, V. F.; LIMA, Y. S.; RODRIGUES, L.; TRAVENÇOLO, B.; BACKES, A. Maximizing the power of cognitive services with an AI-as-a-Service architecture for seamless delivery. In: **2024 IEEE 13th International Conference on Cloud Networking (CloudNet) (IEEE CloudNet 2024)**. Rio de Janeiro, Brazil: IEEE, 2024. p. 8.

RODRIGUES MOREIRA, L. F.; MOREIRA, R.; SILVA, F. de O.; BACKES, A. R. Towards Cognitive Service Delivery on B5G through AIaaS Architecture. In: **Anais do IV Workshop de Redes 6G**. Porto Alegre, RS, Brasil: SBC, 2024. p. 1–8. ISSN 0000-0000. Disponível em: <https://sol.sbc.org.br/index.php/w6g/article/view/29773>.

RODRIGUES MOREIRA, L. F.; MOREIRA, R.; TRAVENÇOLO, B. A. N.; BACKES, A. R. An Artificial Intelligence-as-a-Service Architecture for deep learning model embodiment on low-cost devices: A case study of COVID-19 diagnosis. **Applied Soft Computing**, v. 134, p. 110014, 2023. ISSN 1568-4946. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1568494623000327>.

RODRIGUES MOREIRA, L. F.; SAAR, L.; MOREIRA, R.; RODRIGUES, L.; TRAVENÇOLO, B.; BACKES, A. Enabling intelligence on edge through an artificial intelligence as a service architecture. In: **2024 IEEE 13th International Conference on Cloud Networking (CloudNet) (IEEE CloudNet 2024)**. Rio de Janeiro, Brazil: IEEE, 2024. p. 8.

ROSER, M.; RITCHIE, H.; ORTIZ-OSPINA, E.; HASELL, J. Coronavirus disease (COVID-19)–Statistics and research. **Our World in data**, v. 4, 2020.

RUBIN, G. D.; RYERSON, C. J.; HARAMATI, L. B.; SVERZELLATI, N.; KANNE, J. P.; RAOOF, S.; SCHLUGER, N. W.; VOLPI, A.; YIM, J.-J.; MARTIN, I. B. K.; ANDERSON, D. J.; KONG, C.; ALTES, T.; BUSH, A.; DESAI, S. R.; GOLDIN, o.; GOO, J. M.; HUMBERT, M.; INOUE, Y.; KAUCZOR, H.-U.; LUO, F.; MAZZONE, P. J.; PROKOP, M.; REMY-JARDIN, M.; RICHELDI, L.; SCHAEFER-PROKOP, C. M.; TOMIYAMA, N.; WELLS, A. U.; LEUNG, A. N. The Role of Chest Imaging in Patient Management during the COVID-19 Pandemic: A Multinational Consensus Statement from the Fleischner Society. **Radiology**, v. 296, n. 1, p. 172–180, 2020. PMID: 32255413. Disponível em: <https://doi.org/10.1148/radiol.2020201365>.

RUSSAKOVSKY, O.; DENG, J.; SU, H.; KRAUSE, J.; SATHEESH, S.; MA, S.; HUANG, Z.; KARPATHY, A.; KHOSLA, A.; BERNSTEIN, M.; BERG, A. C.; FEI-FEI, L. ImageNet Large Scale Visual Recognition Challenge. **International Journal of Computer Vision**, v. 115, n. 3, p. 211–252, Dec 2015. ISSN 1573-1405. Disponível em: <https://doi.org/10.1007/s11263-015-0816-y>.

RUSSELL, S.; NORVIG, P. **Artificial Intelligence: A Modern Approach**. 3rd. ed. USA: Prentice Hall Press, 2009. ISBN 0136042597.

SAHLOL, A. T.; ABDELDAIM, A. M.; HASSANIEN, A. E. Automatic acute lymphoblastic leukemia classification model using social spider optimization algorithm. **Soft Computing**, v. 23, n. 15, p. 6345–6360, Aug 2019. ISSN 1433-7479. Disponível em: <https://doi.org/10.1007/s00500-018-3288-5>.

SAHLOL, A. T.; ISMAIL, F. H.; ABDELDAIM, A.; HASSANIEN, A. E. Elephant herd optimization with neural networks: A case study on acute Lymphoblastic Leukemia diagnosis. In: **2017 12th International Conference on Computer Engineering and Systems (ICCES)**. Cairo, Egypt: IEEE, 2017. p. 657–662.

SAHLOL, A. T.; KOLLMANNSBERGER, P.; EWEES, A. A. Efficient Classification of White Blood Cell Leukemia with Improved Swarm Optimization of Deep Features. **Scientific Reports**, v. 10, n. 1, p. 2536, Feb 2020. ISSN 2045-2322. Disponível em: <https://doi.org/10.1038/s41598-020-59215-9>.

SAIT, U.; K.V., G. L.; SHIVAKUMAR, S.; KUMAR, T.; BHAUMIK, R.; PRAJAPATI, S.; BHALLA, K.; CHAKRAPANI, A. A deep-learning based multimodal system for Covid-19 diagnosis using breathing sounds and chest X-ray images. **Applied**

Soft Computing, v. 109, p. 107522, 2021. ISSN 1568-4946. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1568494621004452>.

SAIT, U.; SHIVAKUMAR, S.; K.V., G. L.; KUMAR, T.; RAVISHANKAR, V. D.; BHALLA, K. A Mobile Application for Early Diagnosis of Pneumonia in the Rural context. In: **2019 IEEE Global Humanitarian Technology Conference (GHTC)**. Seattle, WA, USA: IEEE, 2019. p. 1–5.

SALEEM, M. H.; POTGIETER, J.; ARIF, K. M. Plant Disease Classification: A Comparative Evaluation of Convolutional Neural Networks and Deep Learning Optimizers. **Plants**, v. 9, n. 10, 2020. ISSN 2223-7747. Disponível em: <https://www.mdpi.com/2223-7747/9/10/1319>.

SALLAM, A.; ALAWI, A. E. B. Mobile-based Intelligent Skin Diseases Diagnosis System. In: **2019 First International Conference of Intelligent Computing and Engineering (ICOICE)**. Hadhramout, Yemen: IEEE, 2019. p. 1–6.

SANDLER, M.; HOWARD, A.; ZHU, M.; ZHMOGINOV, A.; CHEN, L.-C. MobileNetV2: Inverted Residuals and Linear Bottlenecks. In: **2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition**. Salt Lake City, UT, USA: IEEE, 2018. p. 4510–4520.

SANTOS, D. F. dos; FARIA, P. R. de; TRAVENÇOLO, B. A. N.; NASCIMENTO, M. Z. do. Automated detection of tumor regions from oral histological whole slide images using fully convolutional neural networks. **Biomedical Signal Processing and Control**, v. 69, p. 102921, 2021. ISSN 1746-8094. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1746809421005188>.

SARKER, I. H. AI-Based Modeling: Techniques, Applications and Research Issues Towards Automation, Intelligent and Smart Systems. **SN Computer Science**, v. 3, n. 2, p. 158, Feb 2022. ISSN 2661-8907. Disponível em: <https://doi.org/10.1007/s42979-022-01043-x>.

SCHWALBE, N.; WAHL, B. Artificial intelligence and the future of global health. **The Lancet**, v. 395, n. 10236, p. 1579–1586, 2020. ISSN 0140-6736. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0140673620302269>.

SCOTTI, F. Automatic morphological analysis for acute leukemia identification in peripheral blood microscope images. In: **CIMSA. 2005 IEEE International Conference on Computational Intelligence for Measurement Systems and Applications, 2005.** Messian, Italy: IEEE, 2005. p. 96–101.

SEIDERER, A.; DIETZ, M.; ASLAN, I.; ANDRé, E. Enabling Privacy with Transfer Learning for Image Classification DNNs on Mobile Devices. In: **Proceedings of the 4th EAI International Conference on Smart Objects and Technologies for Social Good**. New York, NY, USA: Association for Computing Machinery, 2018. (Goodtechs '18), p. 25–30. ISBN 9781450365819. Disponível em: <https://doi.org/10.1145/3284869.3284893>.

SHAH, F.; ANWAR, A.; HAQ, I. ul; ALSALMAN, H.; HUSSAIN, S.; AL-HADHRAMI, S. Artificial Intelligence as a Service for Immoral Content Detection and Eradication. **Scientific Programming**, Hindawi, v. 2022, p. 6825228, Jan 2022. ISSN 1058-9244. Disponível em: <https://doi.org/10.1155/2022/6825228>.

SHARFSTEIN, J. M.; BECKER, S. J.; MELLO, M. M. Diagnostic Testing for the Novel Coronavirus. **JAMA**, v. 323, n. 15, p. 1437–1438, 04 2020.

SHARMA, M.; TOMAR, A.; HAZRA, A. Edge Computing for Industry 5.0: Fundamental, Applications, and Research Challenges. **IEEE Internet of Things Journal**, v. 11, n. 11, p. 19070–19093, 2024.

SHI, H.; HAN, X.; JIANG, N.; CAO, Y.; ALWALID, O.; GU, J.; FAN, Y.; ZHENG, C. Radiological findings from 81 patients with COVID-19 pneumonia in Wuhan, China: a descriptive study. **The Lancet Infectious Diseases**, v. 20, n. 4, p. 425–434, 2020. ISSN 1473-3099. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1473309920300864>.

SHI, W.; CAO, J.; ZHANG, Q.; LI, Y.; XU, L. Edge computing: Vision and challenges. **IEEE Internet of Things Journal**, v. 3, n. 5, p. 637–646, 2016.

SILVA, C. F.; SIEBRA, C. A. An investigation on the use of convolutional neural network for image classification in embedded systems. In: **2017 IEEE Latin American Conference on Computational Intelligence (LA-CCI)**. Arequipa, Peru: IEEE, 2017. p. 1–6.

SILVA NETO, P. C. d.; KNUST, R.; BARBOSA, J. L. V.; LEINDECKER, A. P. T.; SAVARIS, R. F. Breast cancer dataset with biomarker Biglycan. **Data in Brief**, v. 47, p. 108978, 2023. ISSN 2352-3409. Disponível em: <https://www.sciencedirect.com/science/article/pii/S2352340923000963>.

SILVA, R. R.; ESCARPINATI, M. C.; BACKES, A. R. Sugarcane crop line detection from UAV images using genetic algorithm and Radon transform. **Signal, Image and Video Processing**, v. 15, n. 8, p. 1723–1730, Nov 2021. ISSN 1863-1711. Disponível em: <https://doi.org/10.1007/s11760-021-01908-3>.

SINGH, R.; GILL, S. S. Edge AI: A survey. **Internet of Things and Cyber-Physical Systems**, v. 3, p. 71–92, 2023. ISSN 2667-3452. Disponível em: <https://www.sciencedirect.com/science/article/pii/S2667345223000196>.

SINGH, S.; WU, Y.; GNS, M. R.; JOSHI, K.; BARNAGHI, P.; KANAGARATHINAM, M. R. **AI in wireless for beyond 5G networks**. Boca Raton: CRC Press, 2023.

SINGHAL, V.; SINGH, P. Texture features for the detection of acute lymphoblastic leukemia. In: SATAPATHY, S. C.; JOSHI, A.; MODI, N.; PATHAK, N. (Ed.). **Proceedings of International Conference on ICT for Sustainable Development**. Singapore: Springer Singapore, 2016. p. 535–543. ISBN 978-981-10-0135-2.

SIRAWATTANANON, C.; MUANGNAK, N.; PUKDEE, W. Designing of IoT-based Smart Waste Sorting System with Image-based Deep Learning Applications. In: **2021 18th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON)**. Chiang Mai, Thailand: IEEE, 2021. p. 383–387.

SONG, M.; XING, X.; DUAN, Y.; COHEN, J.; MOU, J. Will artificial intelligence replace human customer service? The impact of communication quality and privacy risks on adoption intention. **Journal of Retailing and Consumer Services**, v. 66, p. 102900,

2022. ISSN 0969-6989. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0969698921004665>.

SOOM, J.; PATTANAIK, V.; LEIER, M.; TUHTAN, J. A. Environmentally adaptive fish or no-fish classification for river video fish counters using high-performance desktop and embedded hardware. **Ecological Informatics**, v. 72, p. 101817, 2022. ISSN 1574-9541. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1574954122002679>.

SOUSA, D. J. de; CARDOSO, M. A.; BISCH, P. M.; LOPES, F. J. P.; TRAVENÇOLO, B. A. N. Automated standardization of images of Drosophila embryos. **Journal of Visual Communication and Image Representation**, v. 71, p. 102758, 2020. ISSN 1047-3203. Disponível em: <http://www.sciencedirect.com/science/article/pii/S1047320320300080>.

SOUSA, R.; PEREIRA, M.; KWON, Y.; KIM, T.; JUNG, N.; KIM, C. S.; FRANK, M.; ARAUJO, G. Tensor slicing and optimization for multicore NPUs. **Journal of Parallel and Distributed Computing**, v. 175, p. 66–79, 2023. ISSN 0743-7315. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0743731522002532>.

SOUSA REIS, V. C. de; FERREIRA, I. M.; DURVAL, M. C.; ANTUNES, R. C.; BACKES, A. R. Measuring water holding capacity in pork meat images using deep learning. **Meat Science**, v. 200, p. 109159, 2023. ISSN 0309-1740. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0309174023000657>.

SRIVASTAVA, N.; HINTON, G.; KRIZHEVSKY, A.; SUTSKEVER, I.; SALAKHUTDINOV, R. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. **J. Mach. Learn. Res.**, JMLR.org, v. 15, n. 1, p. 1929–1958, jan. 2014. ISSN 1532-4435.

STEINBRENER, J.; DESAI, P. Comparison of Deep Learning Architectures on Embedded Devices and Generalized Fixed-point Conversion Algorithm. In: **2019 5th Experiment International Conference (exp.at'19)**. Funchal, Portugal: IEEE, 2019. p. 343–347.

SUS, J. F. L.; OLIVEIRA, L. F. Leukocyte segmentation and classification using computational vision. In: **XIII Workshop de Visão Computacional**. Natal, RN, Brazil: WVC, 2017. p. 153–157.

SÜZEN, A. A.; DUMAN, B.; ŞEN, B. Benchmark Analysis of Jetson TX2, Jetson Nano and Raspberry PI using Deep-CNN. In: **2020 International Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA)**. Ankara, Turkey: IEEE, 2020. p. 1–5.

SZELISKI, R. **Computer Vision: Algorithms and Applications**. 2nd. ed. Berlin, Heidelberg: Springer-Verlag, 2022. ISBN 978-3-030-34372-9.

TAJBAKHSH, N.; SHIN, J. Y.; GURUDU, S. R.; HURST, R. T.; KENDALL, C. B.; GOTWAY, M. B.; LIANG, J. Convolutional Neural Networks for Medical Image Analysis: Full Training or Fine Tuning? **IEEE Transactions on Medical Imaging**, v. 35, n. 5, p. 1299–1312, 2016.

TAN, M.; LE, Q. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. In: CHAUDHURI, K.; SALAKHUTDINOV, R. (Ed.). **Proceedings of the 36th International Conference on Machine Learning**. PMLR, 2019. (Proceedings of Machine Learning Research, v. 97), p. 6105–6114. Disponível em: <https://proceedings.mlr.press/v97/tan19a.html>.

TAYLOR, B.; MARCO, V. S.; WOLFF, W.; ELKHATIB, Y.; WANG, Z. Adaptive deep learning model selection on embedded systems. In: **Proceedings of the 19th ACM SIGPLAN/SIGBED International Conference on Languages, Compilers, and Tools for Embedded Systems**. New York, NY, USA: Association for Computing Machinery, 2018. (LCTES 2018), p. 31–43. ISBN 9781450358033. Disponível em: <https://doi.org/10.1145/3211332.3211336>.

TEO, T. H.; TAN, W. M.; TAN, Y. S. Tumour Detection using Convolutional Neural Network on a Lightweight Multi-Core Device. In: **2019 IEEE 13th International Symposium on Embedded Multicore/Many-core Systems-on-Chip (MCSoC)**. Singapore: IEEE, 2019. p. 87–92.

THAI, H.-T.; LE, K.-H.; NGUYEN, N. L.-T. Towards sustainable agriculture: A lightweight hybrid model and cloud-based collection of datasets for efficient leaf disease detection. **Future Generation Computer Systems**, v. 148, p. 488–500, 2023. ISSN 0167-739X. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0167739X23002388>.

THOKRAIRAK, S.; THIBUY, K.; JITNGERNMADAN, P. Valuable Waste Classification Modeling based on SSD-MobileNet. In: **2020 - 5th International Conference on Information Technology (InCIT)**. Chonburi, Thailand: IEEE, 2020. p. 228–232.

TRIVEDI, C.; BHATTACHARYA, P.; PRASAD, V. K.; PATEL, V.; SINGH, A.; TANWAR, S.; SHARMA, R.; ALUVALA, S.; PAU, G.; SHARMA, G. Explainable AI for Industry 5.0: Vision, Architecture, and Potential Directions. **IEEE Open Journal of Industry Applications**, v. 5, p. 177–208, 2024.

TRUEX, S.; LIU, L.; GURSOY, M. E.; YU, L.; WEI, W. Demystifying Membership Inference Attacks in Machine Learning as a Service. **IEEE Transactions on Services Computing**, v. 14, n. 6, p. 2073–2089, 2021.

TRUONG, L.-P.; PHAM, B.-D.; VU, Q.-H. A Mobile Deep Convolutional Neural Network Combined with Grad-CAM Visual Explanations for Real Time Tomato Quality Classification System. In: **2020 5th International Conference on Green Technology and Sustainable Development (GTSD)**. Ho Chi Minh City, Vietnam: IEEE, 2020. p. 321–325.

UKAEGBU, U.; TARTIBU, L.; LASEINDE, T.; OKWU, M.; OLAYODE, I. A deep learning algorithm for detection of potassium deficiency in a red grapevine and spraying actuation using a raspberry pi3. In: **2020 International Conference on Artificial Intelligence, Big Data, Computing and Data Communication Systems (icABCD)**. Durban, South Africa: IEEE, 2020. p. 1–6.

ULLAH, S.; KIM, D.-H. Lightweight Driver Behavior Identification Model with Sparse Learning on In-Vehicle CAN-BUS Sensor Data. **Sensors**, v. 20, n. 18, 2020. ISSN 1424-8220. Disponível em: <https://www.mdpi.com/1424-8220/20/18/5030>.

ULUTAS, H.; SAHIN, M. E.; KARAKUS, M. O. Application of a novel deep learning technique using CT images for COVID-19 diagnosis on embedded systems. **Alexandria Engineering Journal**, v. 74, p. 345–358, 2023. ISSN 1110-0168. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1110016823003964>.

VALDENEGRO-TORO, M. Real-time convolutional networks for sonar image classification in low-power embedded systems. **CoRR**, abs/1709.02153, 2017. Disponível em: <http://arxiv.org/abs/1709.02153>.

VELASCO-MONTERO, D.; FERNáNDEZ-BERNI, J.; CARMONA-GALáN, R.; RODRÍGUEZ-VÁZQUEZ, A. Optimum Selection of DNN Model and Framework for Edge Inference. **IEEE Access**, v. 6, p. 51680–51692, 2018.

VERMA, A.; AMIN, S. B.; NAEEM, M.; SAHA, M. Detecting COVID-19 from chest computed tomography scans using AI-driven android application. **Computers in Biology and Medicine**, v. 143, p. 105298, 2022. ISSN 0010-4825. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0010482522000907>.

VITA, F. de; NOCERA, G.; BRUNEO, D.; TOMASELLI, V.; GIACALONE, D.; DAS, S. K. Quantitative Analysis of Deep Leaf: a Plant Disease Detector on the Smart Edge. In: **2020 IEEE International Conference on Smart Computing (SMARTCOMP)**. Bologna, Italy: IEEE, 2020. p. 49–56.

VOGADO, L. H.; VERAS, R. M.; ARAUJO, F. H.; SILVA, R. R.; AIRES, K. R. Leukemia diagnosis in blood slides using transfer learning in cnns and svm for classification. **Engineering Applications of Artificial Intelligence**, v. 72, p. 415 – 422, 2018. ISSN 0952-1976. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0952197618301039>.

VREčA, J.; STURM, K. J. X.; GUNGL, E.; MERCHANT, F.; BIENTINESI, P.; LEUPERS, R.; BREZOčNIK, Z. Accelerating Deep Learning Inference in Constrained Embedded Devices Using Hardware Loops and a Dot Product Unit. **IEEE Access**, v. 8, p. 165913–165926, 2020.

WANG, X.; HAN, Y.; LEUNG, V. C. M.; NIYATO, D.; YAN, X.; CHEN, X. Convergence of Edge Computing and Deep Learning: A Comprehensive Survey. **IEEE Communications Surveys & Tutorials**, v. 22, n. 2, p. 869–904, 2020.

WEI, C.; CHEN, R.; GAO, Q. et al. FPGA-based hardware acceleration for CNNs developed using highlevel synthesis. **Optics and Precision Engineering**, v. 28, n. 5, p. 1212–1219, 2020.

WONG, A.; SHAFIEE, M. J.; JULES, M. S. MicronNet: A Highly Compact Deep Convolutional Neural Network Architecture for Real-Time Embedded Traffic Sign Classification. **IEEE Access**, v. 6, p. 59803–59810, 2018.

WU, E.; ZHANG, X.; BERMAN, D.; CHO, I.; THENDEAN, J. Compute-Efficient Neural-Network Acceleration. In: **Proceedings of the 2019 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays**. New York, NY, USA: Association for Computing Machinery, 2019. (FPGA '19), p. 191–200. ISBN 9781450361378. Disponível em: <https://doi.org/10.1145/3289602.3293925>.

WU, M.; HE, W.; LIN, Y.; LI, L.; LIANG, S.; ZHOU, X. Waste Sorting System Using Binarized Neural Network. In: **2020 International Conference on Intelligent Computing, Automation and Systems (ICICAS)**. Chongqing, China: IEEE, 2020. p. 313–317.

WU, Z.; SHEN, C.; VAN DEN HENGEL, A. Wider or Deeper: Revisiting the ResNet Model for Visual Recognition. **Pattern Recognition**, v. 90, p. 119–133, 2019. ISSN 0031-3203. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0031320319300135>.

XIE, Y.; XU, E.; BOWE, B.; AL-ALY, Z. Long-term cardiovascular outcomes of COVID-19. **Nature Medicine**, Feb 2022. ISSN 1546-170X. Disponível em: <https://doi.org/10.1038/s41591-022-01689-3>.

XU, X.; CAULFIELD, S.; AMARO, J.; FALCAO, G.; MOLONEY, D. 1.2 Watt Classification of 3D Voxel Based Point-clouds using a CNN on a Neural Compute Stick. **Neurocomputing**, v. 393, p. 165–174, 2020. ISSN 0925-2312. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0925231219310549>.

YANG, Q.; LIU, Y.; CHEN, T.; TONG, Y. Federated Machine Learning: Concept and Applications. **ACM Trans. Intell. Syst. Technol.**, Association for Computing Machinery, New York, NY, USA, v. 10, n. 2, jan 2019. ISSN 2157-6904. Disponível em: <https://doi.org/10.1145/3298981>.

YANG, Y.; YANG, R.; PAN, L.; MA, J.; ZHU, Y.; DIAO, T.; ZHANG, L. A lightweight deep learning algorithm for inspection of laser welding defects on safety vent of power battery. **Computers in Industry**, v. 123, p. 103306, 2020. ISSN 0166-3615. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0166361520305406>.

YANG, Z.; ZHANG, S.; LI, C.; WANG, M.; YANG, J.; ZHANG, M. Joint heterogeneity-aware personalized federated search for energy efficient battery-powered edge computing. **Future Generation Computer Systems**, v. 146, p. 178–194, 2023. ISSN 0167-739X. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0167739X23001644>.

YAO, T.; WANG, J.; WAN, M.; XIN, Z.; WANG, Y.; CAO, R.; LI, S.; CHI, X. VenusAI: An artificial intelligence platform for scientific discovery on supercomputers. **Journal of Systems Architecture**, v. 128, p. 102550, 2022. ISSN 1383-7621. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1383762122001059>.

YONEKAWA, H.; NAKAHARA, H. On-Chip Memory Based Binarized Convolutional Deep Neural Network Applying Batch Normalization Free Technique on an FPGA. In: **2017 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)**. Lake Buena Vista, FL, USA: IEEE, 2017. p. 98–105.

YOUNIS, K. S.; AYYAD, W.; AL-AJLONY, A. Embedded system implementation for material recognition using deep learning. In: **2017 IEEE Jordan Conference on Applied Electrical Engineering and Computing Technologies (AEECT)**. Aqaba, Jordan: IEEE, 2017. p. 1–6.

YOUSSEF, H.; SAIT, S. M.; ADICHE, H. Evolutionary algorithms, simulated annealing and tabu search: a comparative study. **Engineering Applications of Artificial Intelligence**, v. 14, n. 2, p. 167 – 181, 2001. ISSN 0952-1976. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0952197600000658>.

ZAMAN, K. S.; REAZ, M. B. I.; BAKAR, A. A. A.; BHUIYAN, M. A. S.; ARSAD, N.; MOKHTAR, M. H. H. B.; Md Ali, S. H. Minimum signed digit approximation for faster and more efficient convolutional neural network computation on embedded devices. **Engineering Science and Technology, an International Journal**, v. 36, p. 101153, 2022. ISSN 2215-0986. Disponível em: <https://www.sciencedirect.com/science/article/pii/S2215098622000611>.

ZHANG, P.; GUO, Z.; ULLAH, S.; MELAGRAKI, G.; AFANTITIS, A.; LYNCH, I. Nanotechnology and artificial intelligence to enable sustainable and precision agriculture. **Nature Plants**, v. 7, n. 7, p. 864–876, 2021. ISSN 2055-0278. Disponível em: <https://doi.org/10.1038/s41477-021-00946-6>.

ZHANG, W.; ZEADALLY, S.; LI, W.; ZHANG, H.; HOU, J.; LEUNG, V. C. M. Edge AI as a Service: Configurable Model Deployment and Delay-Energy Optimization With Result Quality Constraints. **IEEE Transactions on Cloud Computing**, v. 11, n. 2, p. 1954–1969, 2023.

ZHANG, X.; ZHOU, X.; LIN, M.; SUN, J. ShuffleNet: An Extremely Efficient Convolutional Neural Network for Mobile Devices. In: **2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition**. Salt Lake City, UT, USA: IEEE, 2018. p. 6848–6856.

ZHAO, M.; LI, X.; ZHU, S.; ZHOU, L. A Method for Accelerating Convolutional Neural Networks Based on FPGA. In: **2019 4th International Conference on Communication and Information Systems (ICCIS)**. Wuhan, China: IEEE, 2019. p. 241–246.

ZHAO, X.; WANG, L.; ZHANG, Y.; HAN, X.; DEVECI, M.; PARMAR, M. A review of convolutional neural networks in computer vision. **Artificial Intelligence Review**, v. 57, n. 4, p. 99, Mar 2024. ISSN 1573-7462. Disponível em: <https://doi.org/10.1007/s10462-024-10721-6>.

ZHOU, J.; DAI, H.-N.; WANG, H. Lightweight Convolution Neural Networks for Mobile Edge Computing in Transportation Cyber Physical Systems. **ACM Trans. Intell. Syst. Technol.**, Association for Computing Machinery, New York, NY, USA, v. 10, n. 6, oct 2019. ISSN 2157-6904. Disponível em: <https://doi.org/10.1145/3339308>.

ZHOU, P.; YANG, X.-L.; WANG, X.-G.; HU, B.; ZHANG, L.; ZHANG, W.; SI, H.-R.; ZHU, Y.; LI, B.; HUANG, C.-L.; CHEN, H.-D.; CHEN, J.; LUO, Y.; GUO, H.; JIANG, R.-D.; LIU, M.-Q.; CHEN, Y.; SHEN, X.-R.; WANG, X.; ZHENG, X.-S.; ZHAO, K.; CHEN, Q.-J.; DENG, F.; LIU, L.-L.; YAN, B.; ZHAN, F.-X.; WANG, Y.-Y.; XIAO, G.-F.; SHI, Z.-L. A pneumonia outbreak associated with a new coronavirus of probable bat origin. **Nature**, v. 579, n. 7798, p. 270–273, 2020. ISSN 1476-4687. Disponível em: <https://doi.org/10.1038/s41586-020-2012-7>.

ZHOU, S. K.; GREENSPAN, H.; DAVATZIKOS, C.; DUNCAN, J. S.; GINNEKEN, B. V.; MADABHUSHI, A.; PRINCE, J. L.; RUECKERT, D.; SUMMERS, R. M. A Review of Deep Learning in Medical Imaging: Imaging Traits, Technology Trends, Case Studies

With Progress Highlights, and Future Promises. **Proceedings of the IEEE**, v. 109, n. 5, p. 820–838, 2021.

ZUALKERNAN, I. A.; DHOU, S.; JUDAS, J.; SAJUN, A. R.; GOMEZ, B. R.; HUSSAIN, L. A.; SAKHNINI, D. Towards an IoT-based Deep Learning Architecture for Camera Trap Image Classification. In: **2020 IEEE Global Conference on Artificial Intelligence and Internet of Things (GCAIoT)**. Dubai, United Arab Emirates: IEEE, 2020. p. 1–6.

Appendix

APPENDIX A

List of reviewed studies

Table 32 – List of reviewed studies: 111 works published between 2013 and 2023.

Field	Study	CNN	Prep.	DA	Embedded Task	Device	Metric
Agriculture	Aghi, Mazzia e Chiaberge (2020)	MobileNet	●	●	Training and Classification	Jackal UGV	Accuracy, Precision, Recall, and F1-Score
	Ukaegbu et al. (2020)	ResNet-50	○	○	Classification	Raspberry Pi	Accuracy
	Truong, Pham e Vu (2020)	MobileNet-V2	○	○	Classification	FPGA	Accuracy
	Vita et al. (2020)	Deep Leaf CNN	●	●	Classification	ARM Cortex	Accuracy, Precision, Recall, and F1-Score
	Thai, Le e Nguyen (2023)	Tiny-LeViT	○	○	Classification	Raspberry Pi and Jetson Nano	Accuracy, Precision, Recall, F1-Score, FLOP, Parameters, and Model Size
	Luo et al. (2023)	LiteCNN, AlexNet, GoogleNet, ResNet-50, DenseNet-40, ShuffleNet, MobileNet-V2	●	●	Training Classification	FPGA	Accuracy, Precision, Recall, F1-Score Prediction Time, and Model Size
	Chakraborty et al. (2023)	Proposed CNN, AlexNet, GoogLeNet	●	○	Classification	Raspberry Pi	Accuracy, Precision, Recall, F1-Score, and Prediction Time
Animal Classification	Curtin e Matthews (2019)	Proposed CNN	○	○	Classification	Raspberry Pi	Accuracy, Precision, and Recall
	Zualkernan et al. (2020)	Inception-V3, MobileNet-V2, ResNet-18, and DenseNet-121	○	○	Classification	Raspberry Pi	Accuracy, F1-Score, ROC Curve, and CPU utilization

Table 32: (continued).

Banknote Recognition	Pachón, Ballesteros e Renza (2023)	AlexNet, ResNet-18, and GoogLeNet	●	●	Classification	Jetson	Accuracy and Inference Time
	Pham et al. (2020)	Inception-V3, MobileNet-V2, ResNet-18, and DenseNet-121	○	○	Training and Classification	Jetson	Parameter Reduction, FLOP, and Accuracy
Daily Activity Analysis	Castro et al. (2015)	AlexNet	○	●	Classification	ARM Cortex	Accuracy
Dietary Monitoring	Hossain et al. (2020)	MobileNet	○	○	Classification	ARM Cortex	Accuracy, Precision, Sensitivity, Specificity, and F1-Score
Digit Recognition	Choudhari et al. (2020)	A extended LeNet-5 developed for FPGA	○	○	Classification	FPGA	Accuracy, Memory Utilization, DSP Utilization, Slice Utilization of FPGA, and Prediction Time
Driver Status Monitoring	Kim et al. (2020)	AlexNet, InceptionNet, ResNet, ShuffleNet, SqueezeNet, MobileNetV2, PydMobileNet	○	○	Training and Classification	Jetson	Accuracy
	Ullah e Kim (2020)	MobileNet	○	○	Classification	Jetson	Accuracy, Memory utilization, FLOP, and Inference Time
Edge/Mobile Computing	Castillo e Ahmadinia (2017)	Proposed CNN with 2 convolutional layers	○	○	Classification	Raspberry Pi	Inference Time

Table 32: (continued).

Seiderer et al. (2018)	Inception-V3 and MobileNet	○	○	Training and Classification	Nexus 4, Nexus 6P, Raspberry Pi, Odroid	Accuracy and Inference Time
Velasco-Montero et al. (2018)	Inception, ResNet-50, SqueezeNet, MobileNet, and SimpleNet	○	○	Classification	Raspberry Pi	Accuracy, Power Consumption, and Throughput
Khasoggi, Ermatita e Sahmin (2019)	MobileNet	○	○	Classification	Raspberry Pi	Accuracy, Power Consumption, and Throughput
Li et al. (2019)	Proposed CNN, AlexNet, VGG-16, Inception-V3, and ResNet-50	○	●	Classification	Drone	Accuracy
Pacheco, Couto e Simeone (2023)	MobileNetV2, ResNet-18, VGG-16, and ResNet-152	●	●	Training and Classification	Jetson	Accuracy
Yang et al. (2023)	AlexNet, VGG-16, ResNet-18, and MobileNet	○	○	Training and Classification	Raspberry Pi	Accuracy, Power Consumption, Inference Latency
Rizvi et al. (2016)	AlexNet, OverFeat and ResNet-34	●	●	Classification	Nvidia Shield Tablet K1	Accuracy
Kang, Kim e Park (2019)	VGG-16, ResNet-50, and ResNet-32	○	○	Training and Classification	Jetson	Accuracy, Energy Consumption, and Lattency
Zhou, Dai e Wang (2019)	Proposed Lightweight CNN, AlexNet, VGG-16, and FactorizationNet	●	●	Training and Classification	Jetson	Accuracy, Model Size, and Parameters

Table 32: (continued).

	Jayakodi et al. (2020)	LEANet-approach using VGG-6, VGG-10, and MobileNet	○	●	Training and Classification	Odroid	Accuracy and Energy Consumption
	Liu et al. (2019a)	VGG-16, ResNet-32, and ResNet-50	○	○	Classification	Jetson	Accuracy, Energy Consumption, Power Consumption, Memory Usage, and Throughput
Embedded Systems	Danopoulos, Kachris e Soudris (2018)	SqueezeNet	○	○	Training and Classification	FPGA	Inference Time
	Loni et al. (2018)	ResNet, DenseNet, and CNN-Arch 1	○	○	Classification	ARM Cortex	Accuracy, Inference Time, and Model Size
	Hanhirova et al. (2018)	Inception-V2 and MobileNet	○	○	Classification	Nokia Smartphone and Jetson	Inference Latency, and System Throughput
	Alippi, Disabato e Roveri (2018)	AlexNet and VGG-16	○	○	Training and Classification	FPGA and Jetson	Accuracy and Inference Time
	Hadidi et al. (2018)	AlexNet and VGG-16	○	○	Training and Classification	Raspberry PI and Jetson	Power consumption and Inference Time
	Blott et al. (2018)	Proposed CNN inspired by BinaryNet and VGG-16	○	○	Training and Classification	FPGA	Accuracy and Parameters
	Kang et al. (2018)	Darknet19 and DenseNet201	○	○	Training and Classification	Jetson and Odroid	Memory Usage
	Fujii, Sato e Nakahara (2018)	Binarized VGG-11	○	○	Training and Classification	FPGA	Accuracy and Latency

Table 32: (continued).

Silva e Siebra (2017)	Proposed CNN inspired by AlexNet	○	○	Classification	Raspberry PI	Time to load a model, Inference Time, Memory and CPU usage, Temperature, Accuracy, and F1-Score
Chen et al. (2018)	Proposed CNN	○	○	Classification	ARM Cortex	Accuracy and Execution Time
Taylor et al. (2018)	Inception, ResNet, and MobileNet	○	○	Classification	Jetson	Inference Time, Power Consumption, Accuracy, Precision, Recall, and F1-Score
Loni, Daneshtalab e Sjödin (2018)	Proposed CNNs using NAS	○	○	Training and Classification	FPGA and ARM Cortex	Accuracy and Loss
Madadum e Becerikli (2019)	LeNet-5	○	○	Training and Classification	FPGA	Accuracy and Training Time
Arredondo-Velázquez et al. (2019)	LeNet-5	○	○	Classification	FPGA	Power and Energy efficiency
Lee e Lee (2019)	LeNet-5, AlexNet, VGG-16, and ResNet-32	○	○	Training and Classification	Smartphone, Samsung Galaxy S7	Accuracy and Model Size
Qin et al. (2018)	Inception v1, Inception v2, Inception v3, Inception v4, ResNet-50, ResNet-101, ResNet-152, VGG-16, VGG-19, and MobileNet	○	○	Training and Classification	Jetson	Inference Time, Power/Energy Consumption, Accuracy, Precision, Recall, F1-Score

Table 32: (continued).

Zhao et al. (2019)	LeNet-5	○	○	Training and Classification	FPGA	Accuracy, Power Consumption, CPU Consumption, and Latency
Harris, Bae e Egger (2019)	LeNet-5	●	○	Training and Classification	ARM Cortex and Jetson	Accuracy and Processing Time
Piyasena et al. (2019)	VGG-16, AlexNet, CIFAR10-Quick, and LeNet	○	○	Training and Classification	FPGA	Accuracy, Power Consumption, and Latency
Wu et al. (2019)	MNIST-Net, CIFAR-Net, and GoogLeNet	○	○	Training and Classification	FPGA	Clock rate and Compute Efficiency
Lee e Nirjon (2020)	MobileNet and FaceNet	○	○	Training and Classification	Jetson	Accuracy, Memory Usage, and Total Weights
Xu et al. (2020)	3D Projection Network	○	○	Training and Classification	Neural Compute Stick	Accuracy, Training Time, Power Consumption, and Inference Time
Marco et al. (2020)	MobileNet-V1, Inception-V4, and ResNet-152	○	○	Training and Classification	Jetson	Inference Time, Energy Consumption, Accuracy, Precision, Recall, F1-Score

Table 32: (continued).

Vreča et al. (2020)	Simple CNN with 1 Convolutional Layer	O	O	Training and Classification	FPGA	Clock Cycle Count and Dynamic Instruction Count
Disabato e Roveri (2020)	ResNet-18 as feature extractor	O	O	Training and Classification	Raspberry Pi	Accuracy and Computing Time
Mazouz e Bridges (2020)	Proposed CNN	O	O	Classification	FPGA	Accuracy
Wei et al. (2020)	AlexNet	O	O	Classification	FPGA	Running Time and Accuracy
Kalgaonkar e El-Sharkawy (2021)	Proposed CNN	O	O	Classification	ARM Cortex	FLOP
Li et al. (2021)	AlexNet and Darknet-19	O	O	Classification	ARM Cortex	Number of Parameters, and FLOP
Goel et al. (2020)	ResNet and CondenseNet	O	O	Classification	Raspberry Pi	Energy Consumption, Accuracy, and Inference Time
Elsaadouny, Barowski e Rolfes (2020)	LeNet-5 and proposed CNN	O	O	Classification	ARM Cortex	Accuracy and Inference Time
Dhouibi, Salem e Saoud (2020)	Proposed CNN inspired by AlexNet	O	O	Classification	FPGA	Accuracy, Latency, Throughput and Energy Consumption
Süzen, Duman e Şen (2020)	Proposed CNN	O	O	Classification	Jetson and Raspberry Pi	Accuracy, Inference Time, Memory and CPU footprint, Power Consumption

Table 32: (continued).

	Steinbrener e Desai (2019)	AlexNet, GoogLeNet, and SqueezeNet	○	○	Classification	Raspberry Pi	Accuracy, Model Size, Number of Parameters, Precision, and Recall
	Fan et al. (2019)	ResNet-50, MobileNet-V2, Inception-V4, and VGG-16	○	○	Classification	FPGA	Accuracy, and Time for Quantization
	Jiao et al. (2017)	AlexNet and GoogLeNet	○	○	Classification	FPGA	Accuracy, Throughput, Power, and Parallel Hardware Metrics
	Albanese, Nardello e Brunelli (2022)	LeNet-5 and MobileNet-V2	○	○	Classification	Raspberry Pi	Accuracy, Precision, Recall, and F1-Score
Face Recognition	Lee, Wang e Wong (2021)	Proposed CNN	●	○	Classification	ARM Cortex	Accuracy and Power Consumption
	Amato et al. (2018)	CNNs as features extractor: VGG and ResNet-50 (classification with KNN	○	○	Classification	Raspberry Pi	Accuracy
	Greco et al. (2020)	Xception, SuffleNet, and SqueezeNet	●	●	Classification	Raspberry Pi	Accuracy and Latency
	Liang et al. (2020)	Proposed CNN	○	○	Classification	Raspberry Pi	Accuracy and Recall
	Nakajima, Moshnyaga e Hashimoto (2021)	Proposed CNN	●	○	Classification	Raspberry Pi	Accuracy and Prediction Time
	Helaly et al. (2020)	Xception	○	○	Classification	Raspberry Pi	Accuracy

Table 32: (continued).

	Mondal (2020)	Proposed CNN	●	●	Classification	Raspberry Pi	Accuracy, Precision, Recall, and F1-Score
Hand Gestures	Breland et al. (2021)	Proposed CNN	○	○	Classification	Raspberry Pi	Accuracy, Model Size, Number of Parameters
Hardware Accelerator	Fujii et al. (2017)	VGG-11	○	○	Training and Classification	FPGA Jetson	Accuracy
	Yonekawa e Nakahara (2017)	VGG-16	○	●	Classification	FPGA	Accuracy
	Nakahara, Fujii e Sato (2017)	VGG-11	○	○	Classification	FPGA	Accuracy
	Ding et al. (2017)	Proposed CNN	○	○	Classification	FPGA	Accuracy
	Peng, Mingyu e Weisheng (2017)	ZynqNet derived from SqueezeNet	○	○	Training and Classification	FPGA	Accuracy
	Lee e Nirjon (2019)	Proposed CNN	○	○	Classification	FPGA	Accuracy and Inference Time
	Li et al. (2022)	MobileNet-V2	○	○	Classification	FPGA	Accuracy
	Sousa et al. (2023)	Inception-V3, LeNet, MobileNet-V2, ResNet-50, and SqueezeNet	○	○	Training and Classification	NMP LG	Accuracy
	Jiang, Chen e Liu (2023)	MobileNet-V2	○	○	Training and Classification	Jetson	Accuracy and Model Size
	Khoshavi et al. (2022)	Proposed CNN inspired by BinaryNet and VGG-16	○	○	Training and Classification	FPGA	Accuracy
	Hou et al. (2022)	ResNet and MobileNet-V2	○	●	Training and Classification	Raspberry Pi	Accuracy and FLOP
	Zaman et al. (2022)	LeNet, MobileNet-v2, and EfficientNet	○	○	Training and Classification	FPGA	Accuracy
	Messaoud et al. (2022)	LeNet	○	○	Training and Classification	FPGA	Accuracy, Precision, Recall, and F1-Score
	Gao et al. (2022)	VGG-16	○	○	Training and Classification	FPGA	Accuracy

Table 32: (continued).

Hyperspectral Image	Randhe, Durbha e Younan (2016)	Caffe CNN	○	○	Training and Classification	Jetson	Accuracy
	Liu et al. (2019b)	CNN inspired by BASSNet	○	○	Classification	FPGA	Accuracy and Execution Time
Intelligent Vehicles	Lin et al. (2019)	ResNet	●	●	Classification	FPGA	Accuracy and Inference Time
Laser Welding Defects	Yang et al. (2020)	AlexNet, VGG-16, ResNet-50, ShuffleNet, MobileNet, and SqueezeNet.	○	○	Classification	Raspberry Pi	Accuracy, Loss, Inference Time. Model Size, FLOP, and Number of Parameters
Marine Aquaculture	Liu et al. (2019c)	MobileNetV2 and InceptionV3	○	○	Classification	ARM Cortex	Accuracy and Loss
	Soom et al. (2022)	CNN inspired by VGG-16	●	○	Classification	Raspberry Pi Jetson MediaTek Pumpkin	Accuracy,Precision, Recall, and F1-Score
Material Recognition	Younis, Ayyad e Al-Ajlony (2017)	VGG-16	○	●	Classification	Raspberry Pi	Accuracy
Medicine	Teo, Tan e Tan (2019)	AlexNet, Network-1, and Network-2	○	○	Classification	Raspberry Pi	Accuracy, FLOP, and Number of Parameters
	Hou, Hou e Hou (2020)	LeNet	○	○	Training and Classification	Raspberry Pi	Accuracy
	Amin et al. (2021)	Proposed CNN	○	○	Classification	Raspberry Pi	Accuracy and Inference Time
	Sallam e Alawi (2019)	MobileNet	●	●	Classification	Smartphone	Accuracy and Inference Time

Table 32: (continued).

	Rodrigues Moreira et al. (2023)	AlexNet, DenseNet, MobileNet, ResNet, ShuffleNet, and SqueezeNet	●	●	Classification	Raspberry Pi	Accuracy, Precision, Recall, F1-Score, Inference Time, Memory Usage, and Model Size
and Inference Time	Ulutas, Sahin e Karakus (2023)	COVIDNetCT, MobileNetV2, and ResNet-50	●	○	Training and Classification	Jetson	Accuracy, Precision, Recall, F1-Score,
Photovoltaic	Mellit (2022)	VGG-16 and ResNet50	●	●	Classification	Raspberry Pi	Accuracy, Precision, Recall, and F1-Score
Remote Sensing	Kyrkou e Theocharides (2020)	Proposed CNN	●	●	Classification	ARM Cortex	F1-Score, FLOP, Number of Parameters, and Memory Usage
Smart Waste Management	Chen et al. (2022)	GCNet based on ShuffleNet V2, MobileNet v2, and ResNet50	○	○	Classification	Raspberry Pi	Accuracy and Prediction Time
	Wu et al. (2020)	Binarized Neural Network	○	○	Classification	Jetson	Accuracy, Clock, Runtime, Throughput, Power Consumption, and Energy Efficiency
	Sirawattananon, Muangnak e Pukdee (2021)	ResNet-50	●	○	Classification	Raspberry Pi	Accuracy and Loss

Table 32: (continued).

	Thokrairak, Thibuy e Jitngernmadan (2020)	Proposed CNN	●	○	Classification	Raspberry Pi	Accuracy
	Baras et al. (2020)	ResNet-34	○	○	Classification	Raspberry Pi	Accuracy
Sonar Classification	Valdenegro-Toro (2017)	Small CNN inspired by SqueezeNet	○	○	Classification	Raspberry Pi	Accuracy
Surveillance	Khudhair e Ghani (2020)	MobileNet	○	○	Classification	Raspberry Pi	Accuracy, Precision, and Recall
Tactile Sensing Systems	Alameh et al. (2020)	Proposed CNN	○	○	Classification	Raspberry Pi	Accuracy, FLOP, Energy Consumption, Number of Parameters, and Energy Consumption
Traffic Recognition	Wong, Shafiee e Jules (2018)	Proposed CNN	●	●	Classification	Raspberry Pi	Accuracy, Inference Time, and Model Size