الجمهـوريـة الجـزائريـة الديـمقراطيـة الشـعبيـة
PEOPLE'S DEMOCRATIC REPUBLIC OF ALGERIA
وزارة التعـليـم العـالـي و البـحث العلـمي
MINISTRY OF HIGHER EDUCATION AND SCIENTIFIC RESEARCH
جامعـة عمّـار ثليـجـي بالأغـواط
AMAR TELIDJI UNIVERSITY OF LAGHOUAT

كـليـة العلـوم
FACULTY OF SCIENCES
DEPARTMENT OF COMPUTER SCIENCE

Master's Thesis

Field: Mathematics and Computer Science

Specialty: Computer Science

Option: Networks,Distributed Systems and Applications
Option: Information Systems and Decision-Making

Submitted By:
Ghebache Yasmin Meriem & Boussebci Fatima Zohra

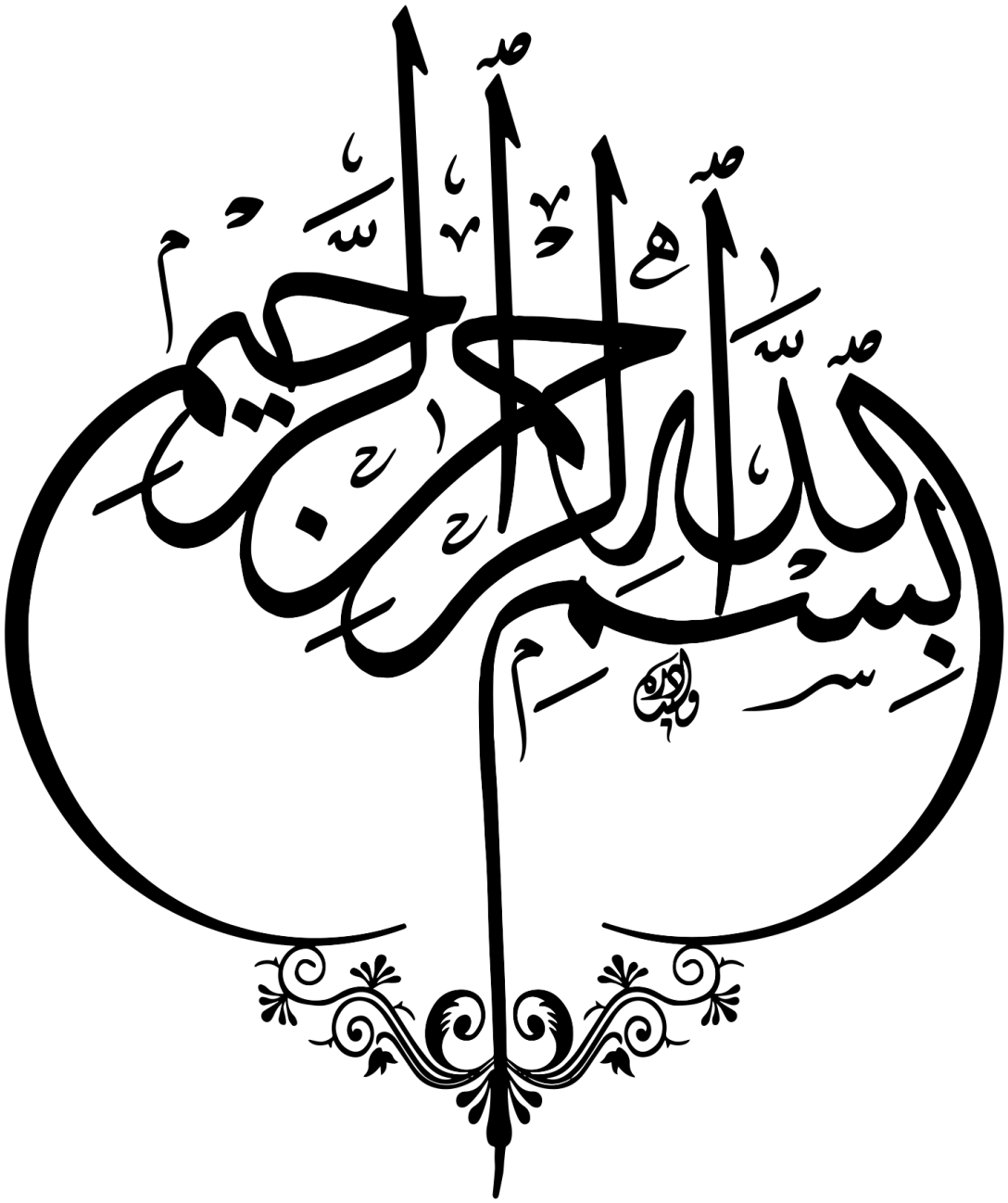
A Machine Learning-based System to Translate Algerian Sign Language to Speech

---

Jury Members:

Dr.	Tahar BENDOUMA	President
Dr.	Younes GUELLOUMA	Examiner
Dr.	Saida Sarra BOUDOUH	Examiner
Dr.	Benameur ZIANI	Supervisor

Academic Year 2024/2025


بسم الله الرحمن الرحيم

Dedications

With heartfelt emotion and endless thanks,
I would like to dedicate this thesis,

To my loving mother and father,
Your love has been my greatest comfort, and your support my steady foundation. You stood by me with patience, strength, and unwavering belief.
Every step I've taken was guided by your wisdom and lifted by your care.
This achievement is a reflection of everything you've given me .
It is as much yours as it is mine

To my dear sisters,
for their boundless love, and continuous support even in the moments when the path felt uncertain. Your presence has always been a comforting reminder that I am never alone.
And to my youngest sister,
Sara as you take on the baccalaureate exams, know that I am proud of you.

To my beloved grandmother and aunts,
thank you for your care, concern, and constant presence throughout the years.
When I was far from home, your support and kindness meant more than words can express.

To my cherished friends,
From last-minute panics, side-eye jokes, and unexpected laugh breaks you turned this academic rollercoaster into a ride worth remembering. Thank you for being the best kind of distraction, and the best kind of support.

And finally, to my partner in this thesis and my friend,
Thank you for sharing this journey with me. Your presence throughout this process contributed to seeing it through. I appreciate the moments we worked together and the support you provided along the way.

And to myself,
For holding on when things got tough, for showing up even when it felt impossible.
"There were pages turned with the bridges burned, everything you lose is a step you take."
Taylor Swift, YOYOK

Yasmin Meriem Ghebache

Dedications

With heartfelt gratitude and profound appreciation,
I would like to dedicate this thesis,

To my beloved parents,
whose endless love, sacrifices, and unwavering faith in me have been the foundation of everything I have achieved thank you for teaching me resilience, humility, and the value of hard work.

To my dear siblings, who have cheered me on through every challenge and reminded me to keep moving forward even in moments of doubt.
To my cherished grandparents on my mother's side, whose wisdom, prayers, and quiet strength have always been a comforting presence in my life.
To my extended family and cousins, thank you for your encouragement, love, and belief in me, especially during uncertain times.
Your presence has always been a comforting reminder that I am never alone on this journey.

To my dear uncle and aunt, For their unwavering support, encouragement, and generosity throughout the course of this work. Your guidance and belief in me made this journey not only possible but deeply meaningful.

To my friends, whose companionship, shared struggles, and joyful moments made university life not only endurable but memorable.

To my beloved grandparents, whose memory, love, and quiet strength have guided me throughout this journey. This work honors the pride you would have felt and the inspiration you still give me.
And finally, to my partner in this thesis and my friend, thank you for sharing this journey with me. Your support and collaboration were invaluable every step of the way.

To Aladin yahia And to everyone who, in big or small ways, offered guidance, kindness, or encouragement this work is also yours.

From the bottom of my heart, thank you for walking beside me on this journey.
And to myself,
For holding on when things got tough, for showing up even when it felt impossible.
"Happiness can be found, even in the darkest of times, if one only remembers to turn on the light." Albus Dumbledore

Fatima Zohra Boussebci

Acknowledgments

First and foremost, no volume of words is enough to express praise to Allah Almighty. All praise is due to Allah, the Lord of the Worlds.

We would like to express our sincere gratitude to our supervisor, Dr. Benameur ZIANI, for his invaluable guidance, continuous support, and encouragement throughout the course of this research. His insightful advice and availability were essential to the successful completion of this work.

We also extend our heartfelt thanks to several professors who supported and guided us during the preparation of this thesis. In particular, we would like to thank Mr. Chaker Abdelaziz Kerrache and Dr. Reguig Mourad for their constructive feedback and assistance during key phases of the project. Their expertise and dedication had a significant impact on our progress

Moreover, we would like to express our deep appreciation to all the teachers who have contributed to our education and personal growth over the past years.

Finally, we thank all those who, directly or indirectly, contributed to the realization of this work.

Our sincere thanks go to the committee members : Prof. Tahar BENDOUMA, Dr. Younes GUELLOUMA , and Dr. Saida Sarra Boudouhe, for accepting to review our work,

Ghebache Yasmin Meriem & Boussebci Fatima Zohra, June 2025

Abstract

Abstract

This thesis presents the development of a real-time Algerian Darija sign language translation tool, which was thought to facilitate the communication of hard-of-hearing and deaf communities.The said tool recognizes Arabic letter hand movements and converts them into verbal Darija through pre-recorded sound files. Implemented in Python, the system combines MediaPipe's hand- tracking library with a convolutional neural network (CNN) that has been trained on a bespoke gesture dataset. Designed to run offline on a Raspberry Pi Zero 2W , it uses pre-recorded Darija audio files to vocalize recognized gestures. Testing showed high accuracy under the specified environment and the assigned hardware Its ease of use and modularity allow for future expansion, with possibilities for mobile integration . This study adds to the body of assistive technology by offering a low-cost and contextually relevant solution that facilitates communication between sign language users and nonsigners in daily communication.

Keywords : ALGSL, Hand gestures, Machine Learning, Deep Learning, Speech translation, AI, Deaf and Hearing-Impaired, Transfer Learning, Raspberry Pi, Reel-time, voice output.

Un système basé sur l'apprentissage automatique pour traduire la langue des signes algérienne en parole
Résumé

Cette thèse présente le développement d'un outil de traduction en temps réel de la langue des signes algérienne (Darija), conçu pour faciliter la communication des personnes sourdes et malentendantes. Cet outil reconnaît les mouvements des mains représentant les lettres arabes et les convertit en paroles du dialecte algérien à l'aide de fichiers audio préenregistrés.

Développé en Python, le système combine la bibliothèque de suivi des mains de MediaPipe avec un réseau de neurones convolutif (CNN) entraîné sur un jeu de données gestuelles spécialement conçu. Il est conçu pour fonctionner hors ligne sur un Raspberry Pi Zero 2W, en utilisant des fichiers audio en darija pour vocaliser les gestes reconnus.

Les tests ont montré une grande précision dans des conditions spécifiques. Sa facilité d'utilisation et sa modularité permettent d'envisager des évolutions futures, notamment une intégration mobile. Cette étude constitue une contribution aux technologies d'assistance, en proposant une solution à faible coût, adaptée au contexte local, qui facilite la communication entre les utilisateurs de la langue des signes et les non-signeurs dans la vie quotidienne

Mots Clés : ALGSL, Gestes de la main, Apprentissage automatique, Apprentissage profond, Traduction vocale, AI, Sourds et malentendants.

نظام ترجمة لغة الإشارة الجزائرية إلى الكلام باستخدام تقنيات التعلم الآلي

ملخص

تتناول هذه المذكرة تطوير نظام لترجمة لغة الإشارة الجزائرية (الدارجة) إلى كلام منطوق بشكل فوري، وقد تم تصميمه خصيصًا لتسهيل التواصل مع فئة الصم وضعاف السمع. يعتمد هذا النظام على التعرف على إشارات اليد التي تمثل الحروف العربية، ثم تحويلها إلى كلمات منطوقة باللهجة الجزائرية باستخدام ملفات صوتية مسجلة مسبقًا.

تم تنفيذ هذا النظام باستخدام لغة البرمجة "Python"، حيث يجمع بين مكتبة تتبع اليدين الخاصة بـ "MediaPipe" وهي مكتبة لمعالجة الصور وتتبع حركة اليد، وشبكة عصبية التفافية(CNN) تم تدريبها على مجموعة بيانات تحتوي على إشارات مخصصة تم جمعها لهذا الغرض.

وقد صُمم هذا النظام ليعمل دون الحاجة إلى اتصال بالإنترنت على حاسوب صغير من نوع "Raspberry Pi zero 2w"، حيث يقوم بتشغيل الملفات الصوتية المُسجلة مسبقًا باللهجة الجزائرية عند التعرّف على كل حركة.

أظهرت التجارب فعالية عالية للنظام من حيث الدقة، وذلك ضمن بيئة تشغيل محددة وعلى العتاد المخصّص له. كما يتميز بسهولة الاستخدام وبنية مرنة تسمح بتطويره مستقبلًا، سواء من خلال دمجه في الهواتف الذكية . تُشكّل هذه الدراسة إضافة نوعية إلى مجال التقنيات المساعدة، حيث تقترح حلًا منخفض التكلفة ومناسبًا للواقع المحلي، مما يسهم في تسهيل التواصل اليومي بين مستخدمي لغة الإشارة وغير الناطقين بها.

الكلمات المفتاحية : الترجمة الآنية، لغة الإشارة الجزائرية ،(ALGSL) إشارات اليد،التعلم الآلي، التعلم العميق، ترجمة الإشارات إلى صوت، الذكاء الاصطناعي ،(AI) الصم وضعاف السمع، التعلم بالنقل، CNN، Pi، Raspberry الإخراج الصوتية

Contents

List of Figures

List of Tables

List of Scripts

Glossary

AI Artificial Intelligence. 15

ASL Algerian Sign Language. 15, 17

CNN Convolutional Neural Network. 15

CV Computer Vision. 15

DL Deep Learning. 15

FPS Frames Per Second. 15

GPU Graphics Processing Unit. 15

HGR Hand Gesture Recognition. 15

ML Machine Learning. 15

OS Operating System. 15

Raspberry Pi A low-cost, single-board computer. 15, 34

SBC Single Board computer. 15

SL Sign Language. 15, 17

SLR Sign Language Recognition. 15

TensorFlow An open-source machine learning framework. 15

TL Transfer Learning. 15

TTS Text-to-Speech. 15

Glossary of Terms

Accuracy is a performance metric which means the proportion of correct predictions made by the model, typically expressed as a percentage.. 15

Batch Size The number of training examples that are handled before the model updates its internal parameters.. 15

Dataset A methodically structured collection of data, such as pictures representing hand gestures, used to train and evaluate a model.. 15

EfficientNet B0 is a compact and efficient deep learning model designed for image classification. It balances accuracy and speed using a method that scales depth, width, and resolution together, making it ideal for use on mobile or low-power devices.. 15

Epoch One pass through the entire training data set. Train models typically continue with several epochs to increase learning. . 15

Fine-Tuning The process of adjusting a pre-trained model's weights on a new, related task to improve performance without training from scratch.. 15

Gesture A specific hand shape or movement that represents a letter or word in sign language.. 15

Keras A high-level neural networks API written in Python and capable of running on top of TensorFlow. 15

Landmarks Key points on the hand detected using computer vision techniques to represent gestures. 15

Loss A measure that shows how far the model's predictions deviate from actual labels. The smaller the loss, the higher the model's performance during training. 15

MediaPipe A framework developed by Google for building multimodal (e.g., video, audio) perception pipelines. 15

Model A statistical construct developed in training capable of predicting from novel data.. 15

Overfitting When a model learns the training data too well, including noise and subtle details such that it does not perform well on new, unseen data.. 15

Prediction Output produced by the model from input data, e.g., a letter corresponding to a particular hand movement.. 15

preprocessing Preparing raw input data (like images) to make them suitable for effective use by a machine learning algorithm.. 15

PyGame A Python library used for writing video games, here used to play audio files. 15

Raspberry Pi A small, low-priced computer used in embedded systems and educational environments ideally matched for the implementation of light AI applications.. 15

Real-Time Processing Real-Time Processing is the capacity of the system to handle and react to information in real time or with little delay.. 15

Training The process of teaching a machine learning model to learn patterns by feeding it data and adjusting its internal variables.. 15

Underfitting is when a model is too simple, unable to learn the underlying patterns in the data, resulting in poor performance on both the training and test datasets.. 15

Validation A training phase where part of the data is used to check the performance of the model on unseen instances. Generalization refers to the capacity of a model to make effective predictions from unseen data.. 15

Chapter 1

Introduction

1.1 Context

Communication is essential for human interaction. It allows individuals to convey ideas, articulate emotions, and establish social connections. For those who are deaf or have speech difficulties, sign language (SL)acts as their main communication method. SL utilize hand movements to express meaning. In Algeria, the most commonly spoken dialect for daily communication is Algerian Darija. However, there currently exists no all-encompassing assistive technology that can translate Algerian Sign LanguageASL into real-time spoken language, which complicates communication and creates challenges for individuals who are deaf or hard of hearing.

This thesis introduces a real-time translation tool for ASL, which was created using single board computer (sbc) and deep learning methods in Python. Our tool identifies hand gestures that correspond to Arabic letters and converts them into spoken Darija through pre-recorded audio clips. The goal of this system is to empower individuals who are deaf or hard-of-hearing by facilitating effortless communication with those who do not sign in everyday situations.

Figure 1.1: Arabic Sign Language Alphabet.

1.2 Motivation

The inspiration for this work stems from an experience that has left an indelible mark in my life both academically and personally. I had a deaf best friend in school. In most cases, communication with her was hard not for lack of desire to do so, but for the simple fact that we did not possess the ability to understand one another to the best of our capabilities. I saw firsthand how hard it was for her to be able to communicate well in spaces not designed to meet her needs. It became clear to me how isolating daily life is for those who use sign language when most others around them do not. This made me conscious of the communication gaps still present, not just in schools but in day-to-day social life. It also showed me the value of inclusive technologies that have the potential to bring human beings closer to each other. With this project, I wish to humbly yet significantly play my part in making communication possible, especially for those individuals whose voices are too often not heard.

1.3 Problem Statement and Objective

Even though the importance of SL is recognized, it remains largely unfamiliar to many individuals. Gaining fluency in this form of communication requires dedication, time, and regular practice and commitments that many in Algerian society are reluctant to undertake.[1].This absence of comprehension and involvement leads to a communication gap between hearing individuals and those who are deaf or hard of hearing, often causing social isolation. Deaf individuals typically must adapt to conform to the hearing world's expectations, while society mostly does not provide the necessary accommodations for them.[2]

To address this challenge, we propose developing a cost-effective embedded system designed to translate (ASL) into verbal communication. Our approach, named "Smart Pin," employs a Raspberry Pi zero 2W sbc paired with a camera to recognize hand gestures. Utilizing deep learning and transfer learning techniques, the system can detect and comprehend dynamic hand movements in real-time and convert them into spoken words.

The primary goal of this project is to provide a portable, user-friendly, and effective communication tool that empowers people who are deaf or hard of hearing. By minimizing their dependence on interpreters or the goodwill of others, this system fosters inclusivity and encourages more spontaneous interactions with individuals who can hear.

1.4 Organization of the Thesis

The organization of this thesis is structured as follows :

- **Chapter two** (2) Presents foundational background necessary to understand the design and implementation of our approach.
- **Chapter three** (3) Presents Literature review of existing systems developed for sign language recognition.
- **Chapter four** (4) Presents the architecture and system design of SignPin.
- **Chapter five** (5) Presents the software implementation and development and the outcome of our experimentsaspects.
- **Chapter six** (6) Presents general conclusion and the future work of our appoach.

Chapter 2

Background

2.1 Introduction

Chapter 2 provides the foundational background necessary to understand the design and implementation of our hand gesture recognition (hgr) system for asl, specifically adapted to Algerian Darija. It begins with an overview of hand gesture recognition techniques and highlights the transition from traditional methods to modern deep learning-based approaches. It then explores the key challenges encountered in gesture recognition, including variability in gesture execution, visual similarity between signs, and environmental constraints. To address these challenges, the chapter also discusses the role of deep learning and transfer learning in enhancing recognition accuracy with limited data. Finally, it introduces the concept of single board computer(acrshortsbc), which serve as the backbone of our real-time, low-power deployment platform. This comprehensive background sets the stage for the technical details and implementation strategies presented in the following chapters.

2.2 Hand Gesture Recognition

Hand gesture recognition enables machines to interpret human hand movements as a form of input, allowing for more natural and intuitive interaction. Early systems typically relied on methods such as color segmentation, sensor-equipped gloves, or background subtraction. However, these approaches often struggle in real-world environments due to variations in lighting conditions, hand shapes, and backgrounds.

Recent advancements in computer vision and deep learning have significantly improved the robustness and scalability of gesture recognition. Modern systems now commonly use landmark-based models to detect and track key points of the hand in real time. For example, frameworks like MediaPipe can extract 21 hand landmarks per frame, providing rich spatial information that can be used for accurate gesture classification.

Despite these improvements, hand gesture recognition still faces notable challenges—especially in real-time applications involving complex gesture sets, such as full alphabets or sign languages. These challenges include ensuring consistent detection regardless of user differences, managing the absence of contextual cues like facial expressions or head pose, and maintaining high accuracy and low latency in gesture classification.[3][4] [5]

2.2.1 Variability in Gesture Execution

Different users naturally perform hand gestures in varying ways. Variations in hand size, finger alignment, gesture speed, and camera distance can affect detection accuracy. This variation necessitates training data that reflects diverse user profiles to ensure the model generalizes well in real-world applications[4].

2.2.2 Visually Similar Gestures

There are slight differences in finger positioning for various Arabic letters, particularly those belonging to the finger-spelling alphabet. For instance, the main distinction between the signs for أ (alif) and ض (ḍād) lies in the placement of the thumb; in أ, the thumb is raised, whereas in ض, it is set to the side. These subtle differences give rise to the low interclass variance problem, which is a common issue in the grouping of static gestures. [6][3].

(a) Hand gesture for the Arabic letter أ

(b) Hand gesture for the Arabic letter ض

Figure 2.1: Examples of visually similar Arabic hand gestures

2.2.3 Inconsistent Gesture Presentation

Because the system processes video frame by frame, not all frames represent a complete gesture. Users may start, hold, or release signs inconsistently. Transitional gestures or frames captured in mid-motion often confuse the classifier and generate false outputs. To mitigate this, prediction smoothing or multi-frame validation strategies must be employed [7]

2.2.4 Occlusion and Poor Orientation

The accuracy of gesture recognition decreases when hands are partially occluded, turned at poor angles, or overlap with the face or body. Side angles, finger overlap, and far distances from the webcam lead to incomplete landmark data. Since this system uses only a single frontal camera, it lacks multi-view perspectives that might otherwise help resolve occlusion errors [8][9]

2.2.5 Environmental Conditions

Although MediaPipe's hand-tracking is robust to many environmental changes, factors like extreme lighting, shadows, and background clutter still cause degradation in tracking accuracy especially in low-cost webcam environments [5][4]. These factors are typical in deployment contexts such as schools, homes, or internet cafes in Algeria, and must be accounted for during development and testing.

2.2.6 Real-Time Performance Constraints

To ensure the system works in real time, it must balance accuracy and speed. Deeper CNNs offer better accuracy but require more computation. On the other hand, lightweight models like MobileNet are faster but may be less precise [10][11]. This trade-off makes it essential to optimize the inference pipeline to run efficiently on general-purpose devices without GPU acceleration.

2.2.7 Dataset Limitations

A key limitation in developing this system is the absence of a publicly available dataset for Algerian Sign Language, particularly one compatible with Darija phonetics. which introduces challenges in terms of:

- Small sample size
- Limited signer diversity
- Restricted recording settings

These issues can lead to overfitting and reduce the model's ability to generalize [9][8][11].

2.3 Deep Learning Techniques

Deep learning is a subfield of machine learning that has demonstrated significant success in solving complex problems in various domains such as image classification, speech recognition, and natural language processing. It functions by stacking multiple layers of non-linear processing units known as artificial neurons into deep neural network architectures. These layers allow the system to automatically learn hierarchical representations from raw data, gradually transforming low-level features into abstract patterns in deeper layers [7] [10][9][6][12] Among the most successful architectures in deep learning are Convolutional Neural Networks (CNNs), which are particularly effective for image-based tasks. CNNs are inspired by the human visual cortex and consist of layers that perform convolution (to detect local spatial patterns), pooling (to reduce dimensionality while preserving essential information), and fully connected layers for classification. This architecture enables models to capture and exploit spatial hierarchies within visual data, making CNNs especially suitable for tasks like gesture recognition and sign language interpretation [13]; [14].

However, training deep networks from scratch is computationally expensive and demands access to large, labeled datasets. This presents a notable challenge in domains such as Arabic Sign Language recognition, where data is scarce and manual collection is labor-intensive.

To address this issue, transfer learning is applied. Transfer learning is a technique where a model pre-trained on a large and diverse dataset (e.g., ImageNet, with over 1.2 million labeled images across 1,000 categories) is adapted to a new but related task [15]. Instead of building a model from scratch, we leverage the learned parameters and fine-tune the network on a smaller target dataset. This approach reduces computational costs and improves performance, especially in low-resource settings. .

Figure 2.2: Diffrence from training from scratch and Transfer Learning .[16]

2.4 Single-board computer

A Single-Board Computer (SBC) is a complete computer built on a single printed circuit board. It integrates a microprocessor, memory, input/output interfaces, and storage, all in one compact design. Unlike traditional desktop computers, SBCs are compact, low-cost, and energy efficient, which makes them widely used in embedded systems, educational environments, and rapid prototyping [17] Main Functionalities of an SBC:

- Processing: Executes instructions using its built-in central processing unit (CPU).
- Memory and Storage: Contains RAM and usually uses microSD cards or flash storage to hold the operating system and files.
- Connectivity: Offers ports such as USB, HDMI, Ethernet, and often includes wireless technologies like Wi-Fi and Bluetooth [17]
- General-Purpose Input/Output (GPIO): Provides digital pins to connect and control external components such as sensors, LEDs, and motors [18]

- Operating System Support: Supports lightweight operating systems, mainly Linux-based, optimized for ARM architecture [19]
- Multimedia: Can process and display audio and video content through HDMI or audio jacks.
- Application Flexibility: Widely used in Internet of Things (IoT) devices, robotics, home automation, AI experiments, and low-cost computing platforms [20].

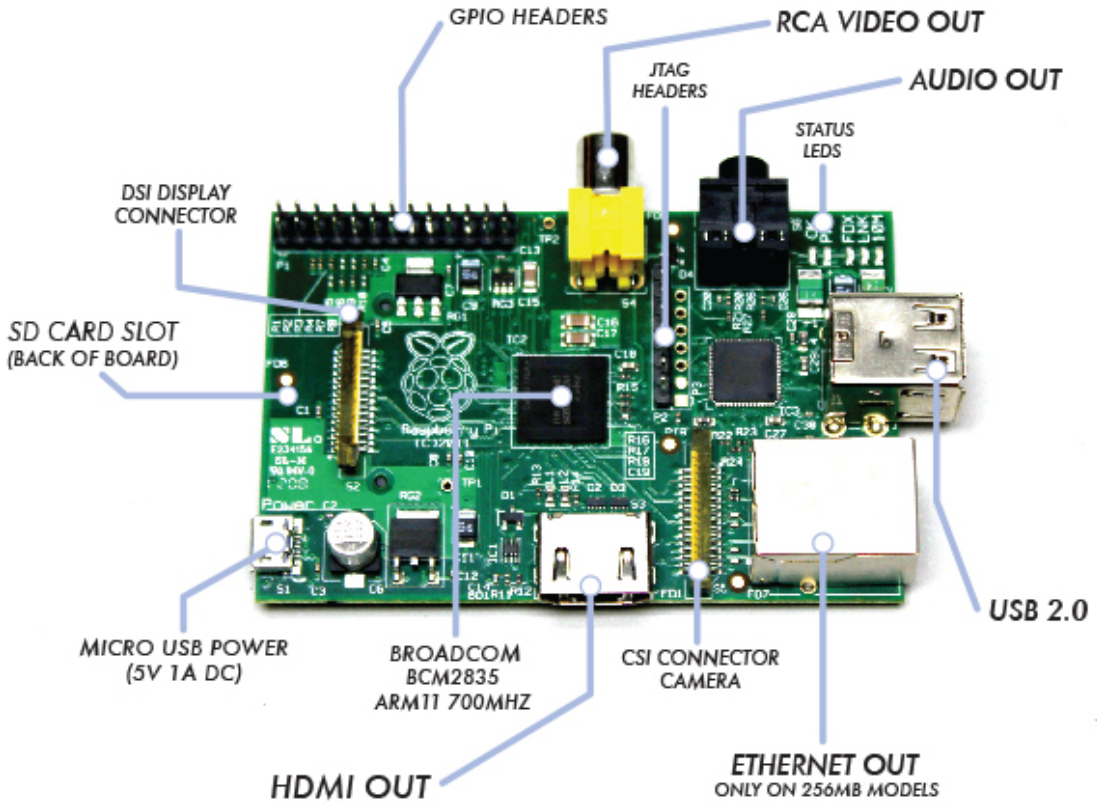


Figure 2.3: Single Board computer: raspberry pi .[21]

Chapter 3

Literature Review

3.1 Introduction

Hand gesture recognition has been an active area of research in computer vision and human-computer interaction. Sign language recognition, as a specific application, presents unique challenges due to the complexity, cultural variation, and gesture ambiguity inherent to the task. This chapter reviews key literature and systems related to hand gesture-based sign language recognition, with a focus on Arabic scripts, deep learning methods, and resource-efficient implementations.

Chapter 3 presents a literature review of existing systems developed for sign language recognition. The reviewed works are grouped into two main categories. The first category includes systems based on wearable tools such as smart gloves, wrist sensors, or motion tracking devices. These approaches rely on direct physical interaction to capture hand movements. The second category focuses on vision-based and application-level systems that use cameras and deep learning models for gesture recognition without requiring the user to wear any equipment. This review aims to highlight the strengths and limitations of each approach and to position our SmartPi project within this research context.

3.2 Vision-based SLR approaches

1. Lokesh Kumar Viswavarapu [7]presents the development of a real-time American Sign Language (ASL) fingerspelling recognition system using deep convolutional neural networks (CNNs). The solution was built in Python and employs a combination of image processing techniques and machine learning models to recognize static hand gestures captured via webcam. Key components include blink detection to trigger recognition, skin-color-based hand segmentation using convex hull, and gesture classification through an ensemble of MobileNet CNNs trained with transfer learning. To enhance accuracy and user interaction, the system integrates head pose estimation as a feedback mechanism, allowing users to confirm or reject predictions through natural head movements. The ensemble approach demonstrated improved performance over individual models, achieving up to 92% classification accuracy. Overall, the project contributes a practical and efficient solution for gesture-based communication, with potential for future enhancements in dynamic gesture recognition and language generation.

2. Mohamed Amine Mahroug and Abdelhafid Zeghidour [34] present a real-time recognition system for Algerian Sign Language (LSA) using deep learning techniques, with a

focus on enhancing communication for the deaf community in Algeria. Motivated by the lack of available LSA datasets and the underrepresentation of this language in computational research, the authors collected a custom dataset (SIGAL) through fieldwork in deaf schools. They developed a system based on autoencoder and attention mechanisms to interpret video sequences of LSA signs. The study thoroughly addresses challenges in data preprocessing, including key-frame selection and augmentation, and evaluates model performance using metrics such as accuracy and F1-score. The system aims to bridge the communication gap between deaf and hearing individuals by enabling fast, accurate, and culturally sensitive sign recognition. The project's practical outcomes include not only a trained model but also a user-friendly interface and a business model to support future deployment and scalability.

3. Alghamdi et al. [10] propose a real-time Arabic Sign Language (ArSL) recognition system based on deep learning. The authors use a lightweight convolutional neural network to classify static hand gestures representing Arabic letters. The system demonstrates high accuracy while maintaining real-time performance, making it suitable for practical applications. Their approach leverages hand landmark detection and a streamlined CNN architecture to ensure fast inference even on resource-limited devices. While the study achieves promising results with Modern Standard Arabic, it does not specifically address regional dialects such as Algerian Darija, highlighting a gap that the present work aims to fill.

3.3 Sensor-based and wearable approaches

1. Hassan Moin et al.[35] In their work on assistive technology, researchers proposed a smart glove designed to translate American Sign Language (ASL) into digital text in real time, offering a practical and affordable solution to enhance communication accessibility. The glove is equipped with stretchable strain sensors placed along the fingers and thumb, allowing it to detect the distinct postures of the hands associated with each letter of the ASL alphabet. By converting these sensor readings into binary codes, the system successfully maps gestures to corresponding characters, which are then transmitted wirelessly to external devices. Notably, the glove is constructed from low-cost, commercially available materials, making it both affordable and accessible for broader use. The study demonstrated reliable letter-level recognition and highlighted the glove's potential for integration into mobile and wearable platforms. This contribution is particularly relevant in the context of sign language interpretation technologies, as it emphasizes simplicity, cost-effectiveness, and the potential for real-time, on-the-go deployment.

2. The SignAloud Gloves project by Navid Azodi and Thomas Pryor [36] introduces a wearable system designed to translate American Sign Language (ASL) into spoken English in real time, with the aim of bridging the communication gap between Deaf individuals and nonsigners. Developed by researchers at the University of Washington, the system uses sensor-equipped gloves that track hand movements, gestures, and positions with high precision. These data are processed using machine learning algorithms trained to recognize specific ASL signs. Once identified, the gestures are converted into corresponding words or phrases and output as speech through a connected device. The system emphasizes portability, real-time processing, and user comfort, demonstrating strong potential for everyday use in educational, professional, and public settings. By combining wearable

technology with machine learning, SignAloud offers a practical step toward inclusive, speech-enabled communication tools for the Deaf and hard-of-hearing community.

3.4 Other Work

Muhammad Al-Qurishi et al. [37] provides a comprehensive review of the current state of sign language recognition (SLR) using deep learning and machine learning techniques. Explores key methodologies, benchmark datasets, and open challenges in building reliable SLR systems. The authors critically evaluate both vision-based and sensor-based input modalities, emphasizing that hybrid approaches tend to outperform unimodal systems. Despite recent advancements enabling the recognition of continuous sign language in near real-time, the study notes that existing models still lack the generalization necessary for robust, large-scale deployment. The paper also introduces a proposed framework for SLR development and underscores the importance of high-quality, domain-specific datasets. By identifying common architectural approaches, assessing their strengths, and discussing the ongoing limitations in dataset availability and cross-linguistic adaptability, the work lays a valuable foundation for future research in the field of accessible communication technologies.

3.5 Identified Gaps

Despite considerable progress in the field of Sign Language Recognition (SLR), a review of current literature highlights several critical gaps that limit the inclusivity and real-world impact of these technologies—especially in regional and resource-constrained contexts. This project addresses the following key issues:

- **Lack of Support for Regional Dialects:** Most existing SLR systems focus on widely documented languages such as American Sign Language (ASL) or Modern Standard Arabic (MSA), with little to no support for regional variations like Algerian Darija. This neglect creates linguistic bias and limits accessibility for native users of underrepresented dialects.
- **Dependence on High-End Hardware:** Many systems achieve high performance but are designed for powerful computing environments. Few are optimized for deployment on low-cost, portable platforms like the Raspberry Pi, making them inaccessible to users in schools, households, or institutions with limited resources.
- **Limited Output Modalities:** The majority of SLR tools output recognized gestures as plain text. However, this format may not be suitable for young users, individuals with low literacy, or those who benefit from multimodal interaction. Audio or visual feedback is rarely implemented but can significantly improve accessibility and user engagement.
- **Poor Real-World Robustness:** Systems often perform well under controlled lab conditions but fail in dynamic environments. Real-world challenges such as varied lighting, background clutter, and hand occlusion reduce the reliability of gesture recognition. There is a need for systems that maintain accuracy in uncontrolled settings.
- **Insufficient User Interaction Design:** Features like blink detection or head pose estimation are explored in academic research but are rarely integrated into lightweight,

practical applications. Enhancing system responsiveness with intuitive interaction methods remains an untapped opportunity.

This thesis aims to fill these gaps by developing an inclusive, real-time SLR system tailored to Algerian Darija. It emphasizes regional language support, low-cost deployment, multimodal feedback, and robust performance in everyday conditions—offering a more accessible and practical solution for diverse users.

3.5.1 Comparison against Existing Systems

Unlike Viswavarapu's ASL system [7], which requires facial and eye data, ours uses hand gestures only, reducing complexity and hardware dependency.

Compared to Alsulaiman et al. [8], who used LSTM for dynamic gestures, our static letter approach prioritizes simplicity and real-time performance.

Alghamdi et al. [10] and Lussier et al. [11] demonstrated the effective use of lightweight CNNs for Arabic or multilingual sign systems, which aligns with our use of a CNN + MediaPipe combo.

Our system distinguishes itself by focusing on Darija, which has not been covered in any of the referenced works

3.6 Conclusion

The studies covered in this chapter show the great strides that have been achieved in the field of sign language recognition through the application of deep learning and computer vision. Various systems have been suggested, many of which have achieved outstanding accuracy in static and dynamic sign recognition. These advances have yet to be translated to all linguistic and technological platforms.

Among the recurring themes observed within literature is that of the under-representation of regional sign languages, such as Algerian Darija. With American Sign Language (ASL) and Modern Standard Arabic (MSA) being the central concern of sign language recognition thus far, restricting our attention to these two languages overlooks and disenfranchises regional dialects. This disregard is especially pronounced within the Arabic speaking world of North Africa, where local dialects deviate significantly from official standards of language but form the foundation of day-to-day communication for deaf communities.

Moreover, even with significant attempts at improving the accuracy of models and response times, the majority of current systems are still highly dependent on high-performance computing facilities, rendering them less suitable for use in low-resource settings. This is a significant barrier to the implementation of this technology in settings such as public schools, rural areas, or homes that do not have access to advanced technology.

There is a further shortage in the field of user experience design for SLR systems. Existing approaches do not heavily emphasize the presentation of recognizable signs using text only, which may not suit everyone effectively, particularly children, individuals with poor literacy, or those who can be more effectively supported by sensory-rich feedback systems. The potential for multi-modal systems, with auditory or visual elements, is untapped despite the promise of more natural and engaging user interactions.

In short, though individual research works have started exploring alternative modes of interaction, like head movement and blink detection, their integration into lightweight deployable systems is uncommon. Consequently, there is still no demand for user-friendly and intuitive

interaction loops that achieve optimal recognition accuracy without adding additional technical complexity.

In light of the results of the study, this research seeks to remedy some of the identified weaknesses by creating a real-time sign language recognition system for Algerian Darija. The system to be created is optimized for effective use on affordable hardware such as the Raspberry Pi and is designed to incorporate multimedia feedback (video and audio) to enhance accessibility. Through a technical efficiency and contextual relevance orientation, the contribution of this work is to a practical and integrative SLR approach that is more closely in line with the needs of real-world use…

Chapter 4

SignPin:A Tool to Translate Algerian Sign Language into speech

4.1 Introduction

This chapter describes the design stage of a tool designed to translate Algerian Sign Language into spoken language, using a camera, a Raspberry Pi, and deep learning techniques. This phase serves as a vital link between the analysis of requirements and the actual execution of the solution. We outline the software and hardware architecture through several UML diagrams, which include a use case diagram, class diagram, activity diagram, and a diagram illustrating the gesture recognition model. The aim is to offer a clear and structured representation of the internal workings of the proposed tool, highlighting the interactions among components, data flows, and the key steps of the translation process, from image capture to the synthesis of the final voice output.

4.2 System Architecture

Prior to showcasing our SmartPin design, our approach focuses on creating a solution that prioritizes user needs by utilizing transfer learning and computer vision to translate Algerian Sign Language into spoken words. This system is constructed around a Raspberry Pi paired with a camera, allowing it to operate independently without requiring internet access. We have pinpointed crucial features and functionalities that emphasize accessibility, ease of use, and real-time performance. These design choices were made after thorough evaluation of the system requirements and a clear understanding of the needs of sign language users. SmartPi is engineered to process images locally and produce the respective voice output, fostering swift and effective communication assistance.

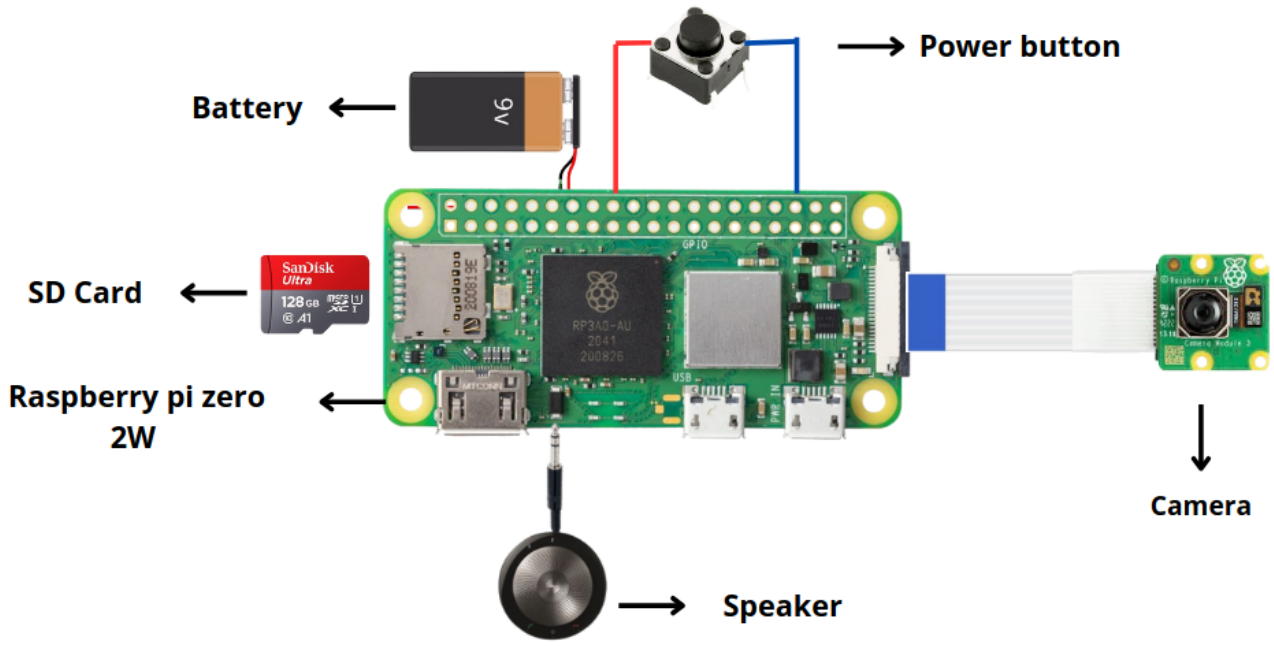


Figure 4.1: SignPin Architectur.

4.2.1 SignPin Composant

A comprehensive overview of all the hardware elements utilized in the SmartPin system is provided below, starting with the image capture module (camera), followed by the processing unit (Raspberry Pi), and concluding with the output module (speaker).

4.2.1.1 Raspberry pi zero 2w

The Raspberry Pi Zero 2 W is a compact and affordable single-board computer ideal for IoT, embedded systems, and educational projects. It features a quad-core processor, 512MB RAM, built-in Wi-Fi and Bluetooth, and supports camera and GPIO interfaces, making it powerful yet energy-efficient.[43]

Table 4.1: Technical Specifications of the Raspberry Pi Zero 2 W

Component	**Specification**
Processor	Quad-core ARM Cortex-A53 @ 1 GHz
Memory	512 MB LPDDR2 SDRAM
Wireless	2.4 GHz Wi-Fi, Bluetooth 4.2
USB	1 × micro USB OTG
Display	Mini HDMI port
Camera Interface	CSI camera connector
GPIO	40-pin GPIO header (unpopulated)
Power Supply	5V / 2.5A via micro-USB
Dimensions	65 mm × 30 mm

4.2.1.2 Speaker

A speaker is a component of electronic devices that transforms electrical signals into sound waves that can be heard, which is essential for gadgets such as smartphones, computers, and TVs. It is made up of a diaphragm, a voice coil, a magnet, and an enclosure. When the electrical signal interacts with the magnet, it causes the coil to move the diaphragm, producing sound waves. Aspects such as materials and design affect the frequency response and overall sound quality. Speakers come in various sizes and types, ranging from compact built-in models to elaborate multi-speaker systems for superior audio experiences. They play a vital role in providing immersive sound experiences, enhancing multimedia content, and improving user satisfaction with electronic devices.[40]

4.2.1.3 Power button

A power button is a control found on electronic devices that allows users to switch them on or off, usually positioned on the front panel, side, or touchscreen interface. It begins the process of starting up or shutting down the device, and may offer extra features such as sleep mode or restart. Power buttons can be physical, capacitive touch, or virtual, designed for easy and intuitive use.[41]

4.2.1.4 Battery

An external battery is a portable energy storage device designed to power other devices via a USB port. It provides a stable output voltage (typically 5V), and its capacity, measured in mAh (milliampere-hours), determines its runtime. For the Raspberry Pi Zero 2 W, a battery between 5000 and 10000 mAh with a 5V/2A output is ideal.[39]

4.2.1.5 Camera

The Raspberry Pi Camera Module v5 is a compact, high-quality camera designed specifically for Raspberry Pi boards. It features a 64MP sensor, autofocus capability, and supports high-resolution photo and video capture. Ideal for computer vision and AI applications, it connects via the CSI interface.[38]

4.3 SignPin System Design

A UML diagram is a way to visualize systems and software using Unified Modeling Language (UML). Software engineers create UML diagrams to understand the designs, code architecture, and proposed implementation of complex software systems.[45]

4.3.1 Use Case diagram

A use case diagram is used to represent the functional requirements of a system. It shows how different types of users (called actors) interact with the system to achieve specific goals (called use cases)[46]

Scenario

In a typical usage scenario, a deaf or hard-of-hearing person initiates an interaction with the system by making a hand gesture in front of the camera. The system captures this gesture from a live video feed, serving as the input for the translation process. Using MediaPipe's hand tracking technology, the system detects and extracts subtle hand gestures, then passes them to a pre-trained convolutional neural network (CNN). The model processes this data to classify the gesture into its corresponding Arabic letter. Once classification is complete, the system retrieves the pre-recorded colloquial audio file and plays it through a loudspeaker. This interaction enables the user to communicate effectively with non-speakers of sign language, providing real-time audio that facilitates communication between sign language and spoken language in a culturally appropriate manner. [46]

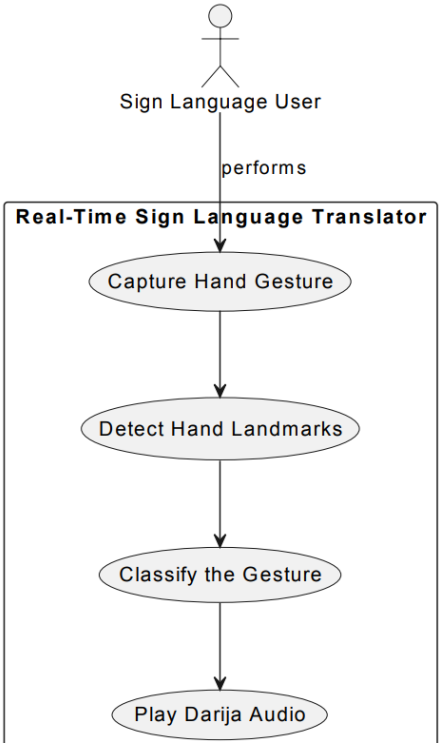


Figure 4.2: Use Case diagram.

4.3.2 Class Diagram

A class diagram is a static structure diagram that describes the structure of a system by showing its classes, attributes, methods, and the relationships between objects[46]

This system implements a real-time sign language translation framework that converts manual gestures into audible speech output. The architecture follows a modular pipeline design with three core functional components: input processing, gesture recognition, and output generation.

The input stage utilizes computer vision techniques through MediaPipe's hand tracking solution. This subsystem analyzes video frames to detect and extract precise anatomical landmarks representing hand configurations and movements. These quantitative measurements serve as the foundation for subsequent interpretation.

At the heart of the system, a machine learning classifier processes the extracted hand landmark data. The model, trained on annotated sign language datasets, performs pattern recognition to map observed gestures to their linguistic equivalents. This classification process employs

learned statistical representations rather than rigid rules, enabling robust interpretation of natural signing variations.

For output generation, the system incorporates an audio synthesis module that converts recognized signs into verbal communication. A lookup mechanism retrieves pre-recorded or synthesized speech segments corresponding to identified gestures.

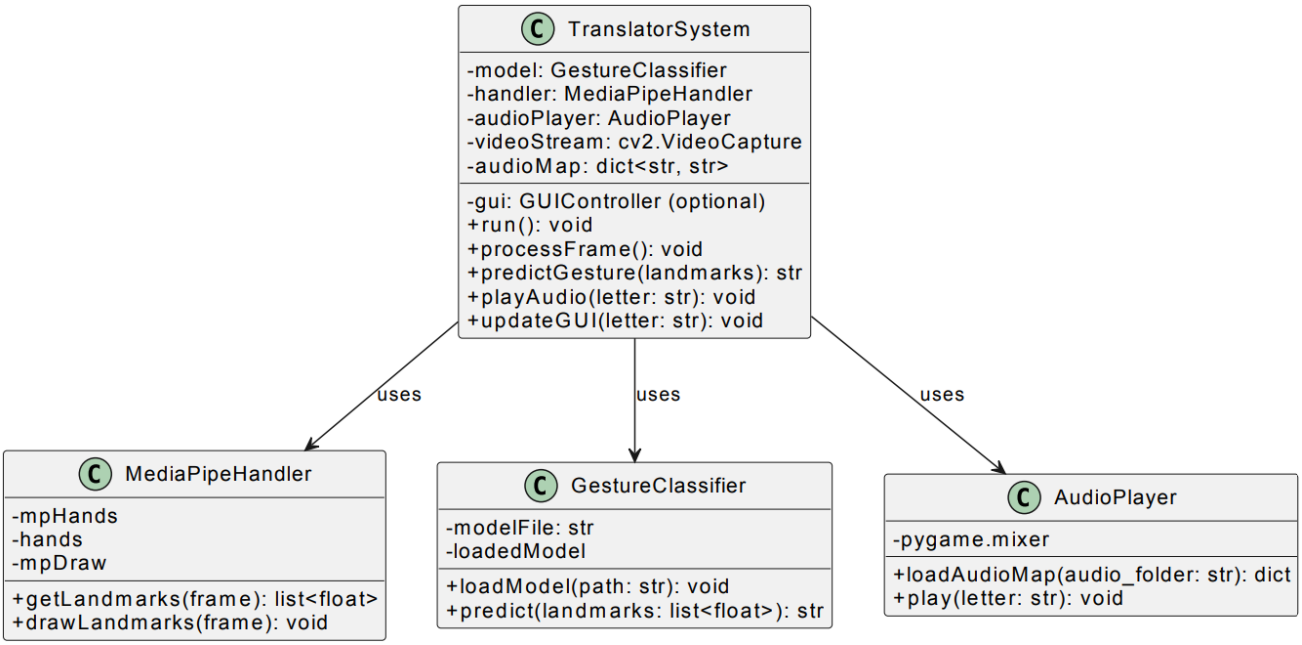


Figure 4.3: Class diagram.

4.3.3 Activity Diagram

Activity diagram is a type of diagram used to model the workflow or process of a system. It shows the sequence of activities, decisions, parallel processes, and the overall flow from start to finish.[46]

Scenario

In this operational scenario, the system is first initialized, loading all necessary components, including the camera, gesture recognition model, and audio playback module. Once powered on, the system starts to continuously capture video frames from the camera. Each frame is processed in real time, with the MediaPipe framework identifying and extracting the user's hand landmarks. These landmarks are then passed to a pretrained convolutional neural network, which classifies the observed hand gesture into a corresponding Arabic letter. Once the letter is identified, the system locates the associated colloquial audio file and plays it using PyGame, providing immediate verbal feedback. This entire cycle operates in a loop, ensuring continuous recognition as long as the system remains active. The scenario reflects the tool's ability to efficiently provide real-time interpretation, making it suitable for dynamic, everyday use by members of the deaf and hard-of-hearing community in Algeria.

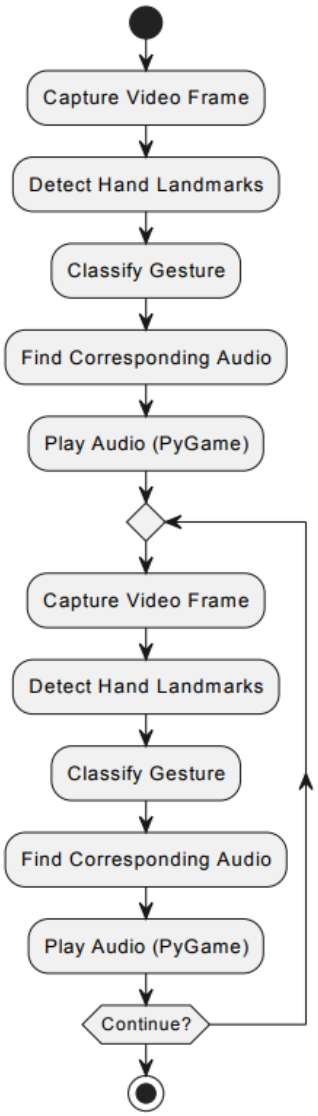


Figure 4.4: Activity diagram.

4.4 Conclusion

In conclusion, the system design presented in this chapter outlines a practical and accessible solution to translating the Algerian Sign Language into spoken Darija. By combining a Raspberry Pi, camera, and deep learning techniques, the tool operates in real time without requiring internet access. Each component from hand detection to gesture classification and audio playback was selected to ensure efficiency, cultural relevance, and ease of use. The UML diagrams provided a clear representation of the system's functionality and structure. The use case diagram captured user interaction, the class diagram clarified the relationships between components, and the activity diagram illustrated the continuous processing flow. Together, these elements reflect a modular and scalable design that supports future improvements while remaining focused on the needs of the Deaf community in Algeria…

Chapter 5

Implementation

5.1 Introduction

In the previous chapter, we explored the design of the system from a hardware point of view, highlighting the components utilized and how they work together. This chapter builds upon that base by concentrating on the software implementation and development aspects of the real-time Algerian Darija sign language translator. It starts by introducing and elucidating the primary technologies and libraries employed Python, MediaPipe, TensorFlow, and PyGame chosen for their suitability with low-resource hardware and real-time embedded applications. Next, it offers a thorough overview of the dataset's creation and preprocessing, the training of a gesture recognition model leveraging transfer learning, the software integration of the detection and prediction pipeline, and the assessment of the system's real-time functionality. The implementation is designed to operate efficiently on devices like the Raspberry Pi, ensuring it is accessible and responsive for practical applications.

5.2 Used Software

In order to create an informative and timely real-time translating system for Algerian Darija sign language, the necessary software libraries were chosen with care. The selected libraries:Python, MediaPipe, TensorFlow, and PyGame, were chosen not just for their superior functionalities in machine learning and real-time processing, but also for their stress-free implementation on Raspberry Pi OS Bullseye. This chapter starts off by mentioning the choice of Raspberry Pi OS Bullseye as the im plementation platform. This particular release was used because it is reliable, supports modern Python environments, and is compatible with lightweight machine learning libraries for ARM-based processors.ronments, and is compatible with lightweight machine learning libraries which are specifically designed for ARM-based processors. Next, this section explains each library and its specific purpose in the system as a whole, these resources form the basis of the implementation pipeline, facilitating activities such as hand landmark detection, gesture recognition, model deployment, and user interaction. Their efficiency and lightness of operation guarantee unproblematic operation in resource-constrained devices, such as the Raspberry Pi.

5.2.1 Operating System: Raspberry Pi OS Bullseye

Raspberry Pi OS Bullseye is a Debian-based operating system forRaspberry Pi hardware. It was launched in November 2021, taking over from the aging Buster and being overtaken by the

newer Bookworm. Bullseye represents a trade-off between stability, compatibility, and having recent software libraries available and is thus appropriate for real-time embedded systems.[22]

5.2.1.1 Why Bullseye Over Bookworm or Buster

Compared to Buster: Bullseye has more modern Python environments, better GPU support, and more compatibility with recent libraries like TensorFlow and MediaPipe. Buster's older packages make it difficult to install and run modern machine learning packages without lots of workarounds or building by hand. [22]

Comparing to Bookworm: Although there is one year difference (Bookworm was created on Debian 12), Bookworm introduces major changes like the Wayland display server and glibc version updates that are incompatible with many real-time libraries and tools. Packages like MediaPipe and TensorFlow are not yet officially supported or require involved manual installs on Bookworm. [23][24]

Bullseye's Edge: Bullseye is currently the most stable and well-supported OS for Raspberry Pi machine learning projects. It provides a tested platform to execute lightweight, real-time AI applications, especially on devices with resource limitations like the Raspberry Pi 3 or 4.[22]

5.2.2 Python: The Core Programming Language

Python is an interpreted, high-level language that supports good-looking syntax and excellent support for modular and object-oriented programming paradigms. It has become a standard option both in research and industrial settings for work ranging from web development to scientific computing.[25]

Within the scope of this project, the primary development language used is Python. Python's readability and simplicity accelerate prototyping, and its compact environment namely libraries like NumPy, OpenCV, TensorFlow, and MediaPipe makes it especially fantastic for machine learning and real-time computer vision use cases. Moreover, Python's support on Raspberry Pi hardware enables effortless integration with underlying system peripherals, such as GPIOs, cameras, and sensors, which are essential to real-time embedded use cases

5.2.3 Key Libraries

To aid in robust gesture detection and user input, the system leverages three exclusive libraries: MediaPipe for real-time hand tracking, TensorFlow for gesture recognition, and PyGame for UI programming. All these libraries contribute something unique to the performance, accuracy, and responsiveness of the application on a resource-constrained device such as the Raspberry Pi.

5.2.3.1 MediaPipe: Hand Landmark Detection

MediaPipe, a Google-developed open-source cross-platform framework utilized to build multimodal perception pipelines, contains pre-existing models and pipelines for operations such as face detection, pose estimation, and hand tracking. In this solution, MediaPipe is used for real-time detection and extraction of 21 hand landmarks from camera input. These 21 landmarks, which are the significant points such as fingertips and joints, form the basis of dynamic hand gesture recognition.Its high accuracy and speed are suitable for real-time embedded applications.[26]

5.2.3.2 TensorFlow: Gesture Recognition Model

TensorFlow is an open-source and Google-created library for machine learning that is targeted specifically at deep learning applications. It allows flexible models to be produced with optimized deployment on a wide range of platforms, including embedded devices and mobiles.

Here, TensorFlow is used to develop a model that classifies gestures, using hand landmark data from MediaPipe. Once trained, TensorFlow stores models in the .h5 file format (Hierarchical Data Format version 5), in which the architecture of the model, weights, and training details are stored in a single file. This facilitates easy reuse, sharing, and deployment of the model in different environments. [27]

5.2.3.3 MobileNetV2 for Transfer Learning

MobileNetV2 is used as the base architecture via transfer learning for better accuracy while maintaining low computation costs. it's designed as a lightweight convolutional neural network for resource-constrained environments such as mobile devices and embedded systems. MobileNetV2 has been pre-trained on the ImageNet dataset and it shows a highly favorable trade-off between accuracy and execution speed. [28]

In this thesis, we use MobileNetV2 to extract useful features from input images. All of its convolutional layers are frozen, meaning they do not change during training. This approach reduces training time and avoids the need for high-end hardware.A new classification head is added and trained for the collected gesture dataset. This action minimizes training time and takes advantage of a small number of samples of data, which helps improve generalization ability that will be necessary for real-time computation on the Raspberry Pi.

5.2.3.4 Pygame

PyGame is a free and open-source Python library designed for writing multimedia applications such as video games. It provides modules for handling graphics, sound, and input devices, making it suitable for interactive systems.[29] PyGame is used in this project for playing audio. After the detection and classification of a hand gesture, PyGame plays an associated pre-recorded audio file in .wav format using our voice, we recorded the sound of the letters and saved in a folder named son-alph. After the system has predicted the right letter, the script automatically found and played the corresponding file. This gives immediate result to the user, helping to reinforce the identified gesture in real time. It also makes the system more interactive and useful, especially as an assistive technology for people with communication impairments.

5.3 Installation and Implementation

This part provides an in-depth overview of the technical implementation of an Algerian Darija sign language real-time translator. It describes the entire development process, from dataset construction to model training, software integration, and real-time execution pipeline implementation. The application depends on a set of technologies: Python5.2.2, MediaPipe5.2.3.1, TensorFlow5.2.3.2, and PyGame. We used a Raspberry Pi 4 for our implementation instead of the Raspberry Pi Zero 2W in prototyping so that we would have large processing power and memory for real-time gesture detection and interpretation. The integration of the selected hardware and software offers an lightweight platform ideal for responsive, real-time gesture recognition applications

5.3.1 Setting up the implementation Environment

The development process was initiated by installing and configuring the necessary software environment on the Raspberry Pi that is operating Raspberry Pi OS Bullseye. we used the official raspberry pi os imager that have been uploeded on our sd card

Figure 5.1: Raspberry Pi imager interface

5.3.2 Dataset Preparation

Due to the absence of publicly accessible datasets for Algerian Sign Language, we had to create a custom dataset to support the development and evaluation of our recognition system. This dataset focused on capturing static hand signs representing the Arabic alphabet, with the goal of enabling precise classification and real-time translation. The dataset was created by using a standard webcam to picture hand gestures in real time. In order to enhance the model's generalization capabilities, images were taken under various conditions.

- **Lighting conditions:** natural daylight, artificial indoor lighting, and low-light settings
- **User diversity:** several people provided samples of hand gestures
- **Angles and orientations:** gestures were recorded from diffrent point of view such as frontal, lateral, and angled perspectives

This diversity guarantees that the dataset represents real-world usage situations and improves the strength of the trained model.

All of the images were placed in a parent folder called gesture-dataset. This folder contained 28 subfolders. A folder for each letter of the Arabic alphabet. The names of the subfolders are the same for every Arabic letter (أ, ب, ت ...). Approximately 130 images of the target hand gesture are contained in each class folder. The images that are implicitly labeled with their folder name can be loaded easily using standard image classification pipelines.
To illustrate the variety of the dataset, Figure 5.2 presents a sample of images from several classes. These examples point out how each letter was imaged with variations in lighting, background, and hand orientation.

Figure 5.2: Sample image from the dataset

5.3.3 Training Configuration

The training process was conducted on a personal computer utilizing Python and TensorFlow the Table 5.1 represente the specifications of it. The objective was to categorize static images of hand gestures that correspond to the 28 letters of the Arabic alphabet We performed several versions of training in order to identify the one with which we obtained the best performance. These included training with and without transfer learning, trying 2 different kind of transfer learning EffectNet and MobileNet In this section, we explain each experiment and the script used.

Table 5.1: Hardware Specifications Used for Training

Component	**Specification**
CPU	Intel® Core™ i5-5300U × 4
RAM	8 GB
GPU	None (No dedicated GPU)

5.3.3.1 Training Without Transfer Learning

The initial test was a custom convolutional neural network (CNN) from scratch, not with any pre-trained model. It was to see how effectively the model could learn directly from our hand gesture dataset.

You can access the code folder on GitHub:
GitHub Folder: code1

Training Parameters

To summarize the training configuration5.2 resumes the main hyper-parameters used during the training phase.

Table 5.2: Training Parameters for Custom CNN Model

Parameter	Value
Image Size	96 × 96
Batch Size	16
Epochs	50
Optimizer	Adam
Learning Rate	0.001
Loss Function	Categorical Crossentropy
Early Stopping	Enabled (patience = 5)

5.3.3.2 Training with Transfer Learning (MobileNetV2)

MobileNetV2 base was used with the classification head (top) removed. The entire convolutional layer in the base model was frozen to maintain the learned features. A new classification head constructed from scratch included:

- A Global Average Pooling layer
- Two Dropout layers (30 percent) to reduce overfitting
- A Dense layer with 128 neurons and ReLU activation
- A final Dense output layer with softmax activation, matching the number of gesture classes (28).

You can access the code folder on GitHub: GitHub Folder: code2

5.3.3.3 Training Parameters

We used the same hyper-parameters in table 5.2 for this training

The evaluation of training was conducted by tracking validation accuracy and loss. To avoid overfitting and preserve the model that performed the best, early stopping was utilized.

The completed trained model has been stored with the name **gesture-model.h5** the meaning of an h5 file is explained in the Tenserflow definition 5.2.3.2, and a **JSON** file named **class-name.json** was created to associate class labels with their corresponding folder names.

5.3.3.4 Transfer Learning with EfficientNetB0

This experiment reused the MobileNetV2 model but included early stopping to prevent overfitting. The training process was halted automatically if the validation loss stopped improving after a fixed number of epochs. You can access the code folder here:

GitHub Folder: code3

5.3.3.5 Training Parameters

We used the same hyper-parameters in table 5.2 for this training

5.3.4 Real-Time Gesture Processing Pipeline

In this section, we break down the real-time prediction script**??** we developed for Arabic Sign Language recognition. The script operates in a loop that takes frames from the webcam and processes them in real-time to recognize and voice the hand gestures

1. **Capture video frame:** that is done with OpenCV
2. **Hand landmarks detection:** MediaPipe detect and tracks the landmarks on the captured frame see Figure 5.3
3. **Landmark Normalization** in order to make these landmarks suitable for classification
4. **Gesture Classification:** A pre-trained CNN model classifies the hand gesture based on the landmark input.
5. **Audio Output:** PyGame plays pre-recorded audio file (using our own voices) that corresponds to the predicted letter.

To help reduce false positives, this system employed frame-level smoothing. This means that the same predicted label must be consistent for a certain number of consecutive frames (in our experience usually between 5) to be confirmed. This makes the predictions more stable, especially in systems that detect moving gestures.

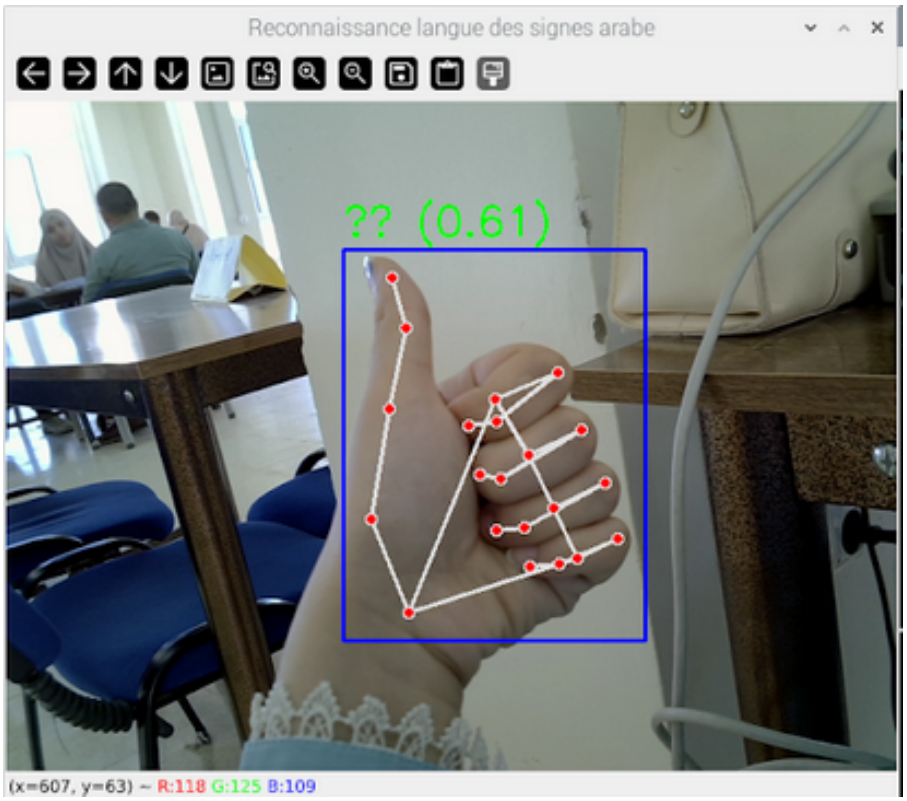


Figure 5.3: The testing Hand landmarks detection interface

5.3.5 Prediction Code

The following Python script was used to test all three trained models. The only required modification was updating the `.h5` file path (line 11), which corresponds to the trained model

to be evaluated. Each model is saved in the `.h5` format is (Hierarchical Data Format version 5), which stores the full architecture, trained weights, and training configuration in a single file[24]. This design makes it easy to load and test different models using the same prediction pipeline.

You can access the code folder here:
GitHub Folder: code4

5.4 Results and Discussion

This section shares the results of our work with the Algerian Sign Language (Darija) translation system. We focused on improving gesture recognition accuracy and responsiveness. All testing took place in real-time conditions using a live webcam feed.

As explained in Chapter 5, the system was set up to process video in real-time, detect hand gestures with Mediapipe, classify them using a deep learning model trained on a dataset, and play the corresponding audio for each detected letter. To achieve this, we explored three different training scenarios: training a CNN from scratch, using transfer learning with EfficientNet, and transfer learning with MobileNetV2. This chapter evaluates these approaches, examining their performance metrics and prediction stability over time.

5.4.1 Training a CNN without Transfer learning

5.4.1.1 Results And Evaluation

The CNN was initially trained for up to 50 epochs on our custom dataset without using transfer learning. However, early stopping caused the training to stop at the 7th epoch to avoid overfitting. As shown in Figure 6.1, the model achieved a high accuracy on the training set, exceeding 95%. On the other hand, the validation accuracy was between 58% and 70% before deteriorating in the final epoch.

At around the same time, the validation loss was growing exponentially towards the end of the training period as the training loss remained phenomenally low. This growing disparity was a strong indication of overfitting. The model is learning how to perform well on the training set but does not generalize to novel, unseen data..

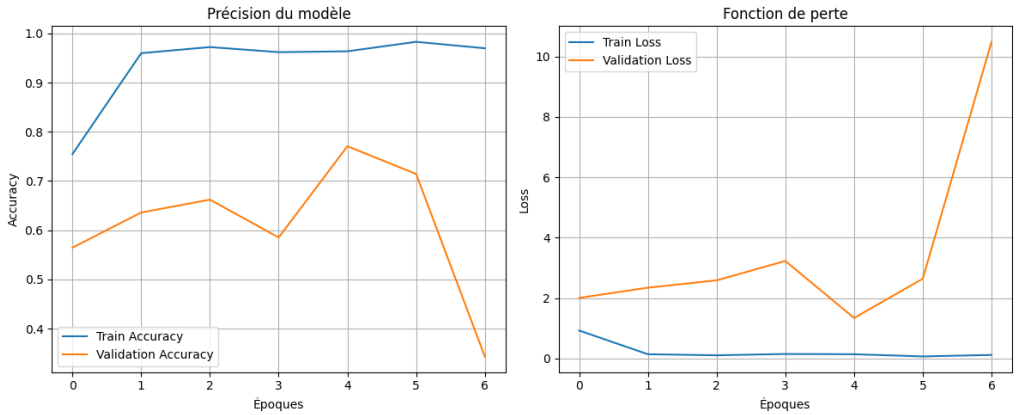


Figure 5.4: accuracy,loss and validation curves

The (Figure 5.5) shows the model's prediction confidence over time after the execution of the script**??**. Most predictions stayed around mid-to-low confidence levels, between 30% and 50%, with frequent and sudden changes. A brief segment, frames 250 to 300, showed relatively higher confidence levels, nearing 90%. However, the overall trend remained unstable and uncertain. The lack of sustained high confidence indicates that the model had trouble extracting reliable features or making consistent predictions, especially with different input conditions. These results show that training a CNN from scratch on a limited dataset is not enough for effective Arabic sign language recognition.

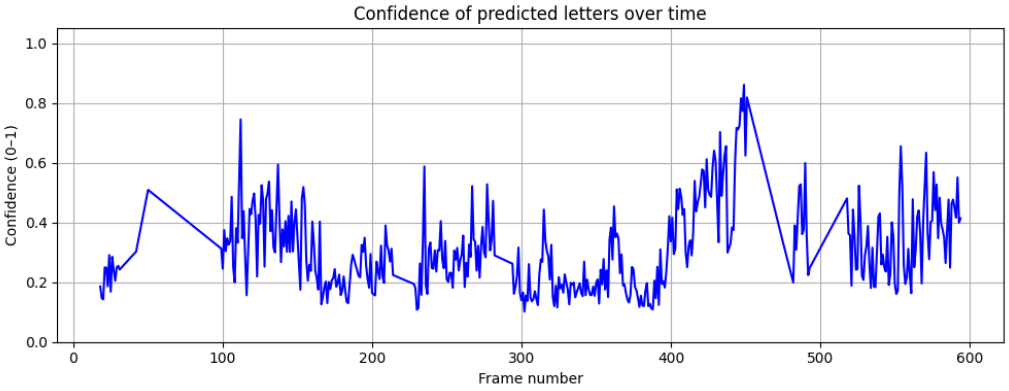


Figure 5.5: prediction confidence over time for the cnn model

5.4.1.2 Discussion

Training a CNN from scratch didn't work well for our application. The model overfitted quickly to the small training dataset, meaning it couldn't generalize the learning to new data. This led to wrong prediction . These findings highlight the deficiencies of using a randomly initialized network on a lim- dataset. Without access to a large and diverse dataset, the model cannot learn representative, Distinctive features for Arabic sign language recognition. This prompted exploration of transfer learning techniques, which are able to use pre-trained representations in order to improve performance and stability on small, domain-specific datasets.

5.4.2 Training With Transfer Learning(EfficientNet)

5.4.2.1 Results and Evalution

The EfficientNet architecture was used to train the model with transfer learning for up to 50 epochs. Early stopping was applied, however, and training halted after just a few epochs when there was no noticeable improvement in validation performance. As Figure 5.6 shows, the training and validation accuracy were both very low at approximately 3.5%, the equivalent of random guessing across the 28 output classes. Likewise, the loss values stayed approximately constant, with no significant drop across epochs. These results indicate that the model could not learn from the dataset

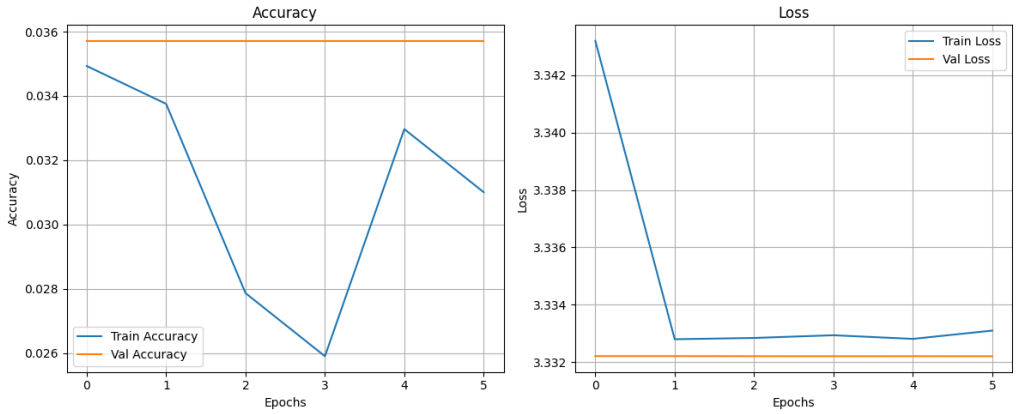


Figure 5.6: accuracy,loss and validation curves of the efficientNet training

The EfficientNet model did not produce notable results on our data. Training and validation accuracy were both low at around 3.5%. This is random guessing across the 28 classes. The loss values also did not show considerable decrease, which validates that the model did not learn meaningful patterns. The prediction confidence over time (Figure5.7) also supported this. Most of the predictions were extremely uncertain, hovering around the mean of 0.10, with a few random spikes. This means that this configuration was not ideal for detecting Arabic sign language in our setting.

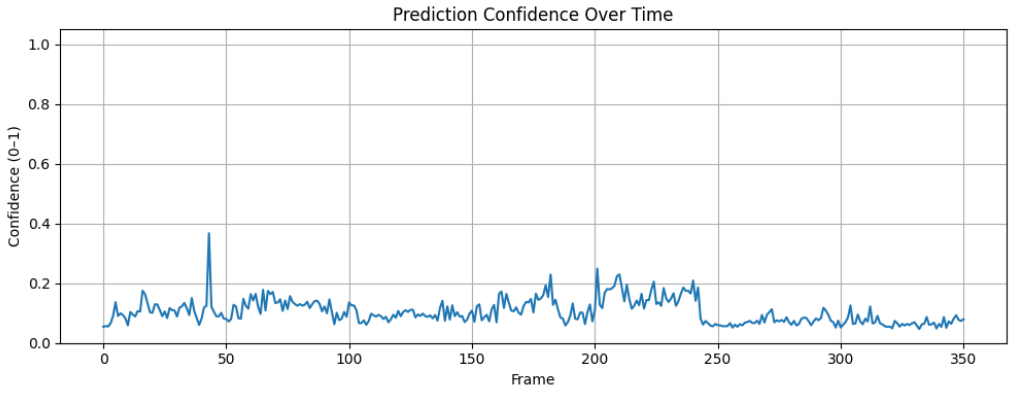


Figure 5.7: prediction confidence over time for the EfficientNet model

5.4.2.2 Discussion

This poor performance may be caused by various reasons, such as incompatible input preprocessing, poor data augmentation, or lack of adequate good fine-tuning of the EfficientNet layers. The result obviously shows that this configuration was not suitable for our sign language dataset

5.4.3 Training With Transfer Learning (MobileNetV2)

5.4.3.1 Results and Evalution

In this experiment, the model was trained using transfer learning with MobileNetV2, which had been pre-trained on the ImageNet dataset. Training was configured for a maximum of 50 epochs, but thanks to early stopping, the process was halted after 30 epochs once the

validation performance stopped improving, helping to avoid overfitting Figure 5.8 shows the training results

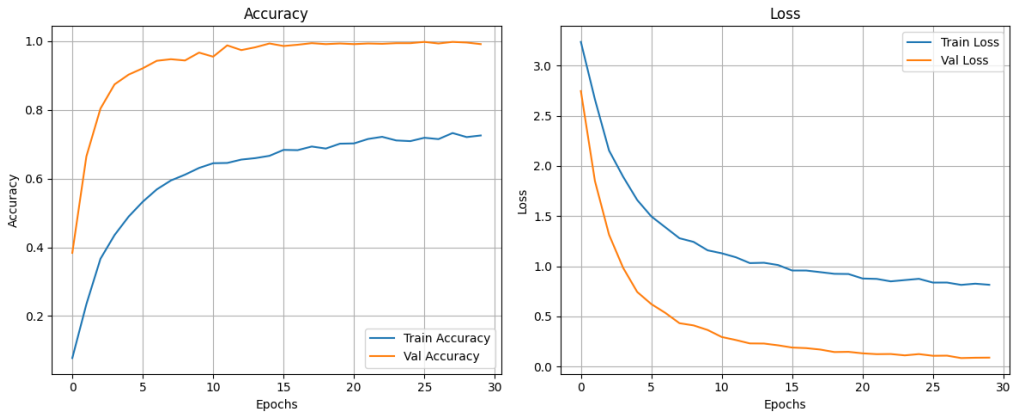


Figure 5.8: accuracy,loss and validation curves of the MobileNet model

This configuration produced the best results of the three different experimental conditions. The model exhibited excellent training accuracy and even and consistent validation accuracy during until the training session. The curves of loss were balanced and in a declining trend with no huge gap between the validation and training losses, which points to impressive generalization to unseen data. Furthermore, the prediction confidence graph over time (Figure 5.9) also supports this finding. Although confidence values varied with different input frames, the model usually made confident predictions (over 80%) and generally did not overestimate its confidence. Confidence changes reflect how the model responds to different inputs, depending on their complexity. Interestingly, the model did not provide consistently low-confidence predictions, which means that it copes well with most frames. Overall, these results show that the model is robust and viable for deployment in the real world.

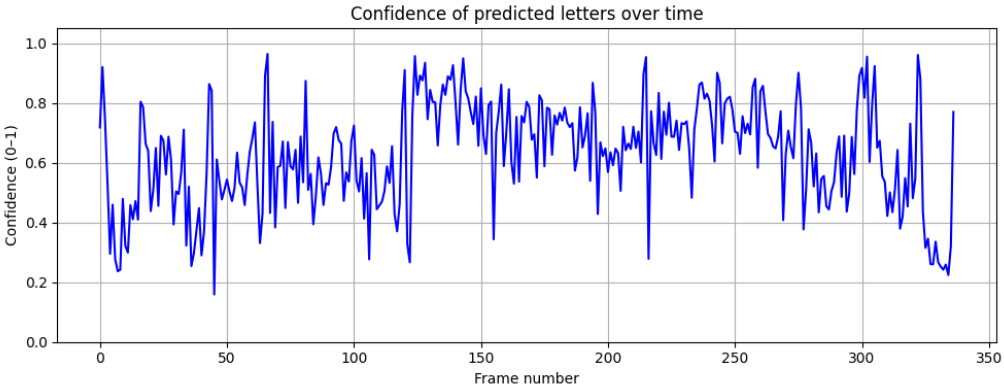


Figure 5.9: prediction confidence over time for MobileNet model

5.4.3.2 Discussion

These results prove that MobileNetV2, with early stopping , works well for Arabic sign language gesture recognition in our self-created dataset. As a result of the transfer learning,

the pre-trained model was able to use learned features from ImageNet and thus learn faster and obtain good performance without overfitting. This can be observed in fig 5.9

The prediction confidence over time (Figure 5.8) sThe confidence levels are mostly steady, with some high peaks and short drops. These changes show that the model can handle different levels of input difficulty while still making reliable predictions overall..

5.5 Conclusion

This chapter outlines the development and evaluation of a real-time Algerian Darija sign language translator, designed for low-resource environments like the Raspberry Pi. The system was built using lightweight, open-source tools including Python, TensorFlow, MediaPipe, and PyGame. A custom dataset of static Arabic alphabet hand gestures was collected under varied lighting and across multiple users to enhance model generalization.

Training was conducted using both custom CNNs and transfer learning models. While CNNs showed high training accuracy, they overfitted and performed poorly on new data. EfficientNetB0 also failed to learn relevant features, likely due to dataset limitations or preprocessing issues. MobileNetV2, however, delivered stable, accurate performance and proved well-suited for real-time recognition on embedded devices.

The final system integrates gesture detection, classification, and Darija audio playback in a real-time loop, enabling accessible communication. These results confirm the practicality of MobileNetV2 for lightweight gesture recognition and provide a solid base for future work on word and sentence-level translation.

Chapter 6

General Conclusion and Future work

This thesis has explored the design and implementation of a real-time sign language translation system adapted to the unique linguistic and cultural context of Algerian Darija. The motivation behind this project stems from the lack of accessible, localized communication tools for Deaf and hard-of-hearing individuals in Algeria an underserved population in both technological and educational domains. By combining computer vision, deep learning, and audio output mechanisms, the system provides a meaningful bridge between sign language users and the hearing community. Unlike many existing tools developed for ASL or Modern Standard Arabic, this system specifically addresses the Algerian dialect, offering spoken Darija output that aligns with the users' daily language.

The project relies on a camera and a Raspberry Pi for gesture capture and processing, employing MediaPipe to detect hand landmarks and a lightweight convolutional neural network (CNN) for gesture classification. Recognized letters are mapped to pre-recorded Darija audio files, which are played back in real time using PyGame. The system architecture, supported by UML diagrams, highlights a modular design that is both scalable and efficient. This architecture allows for straightforward updates and potential expansion toward dynamic gesture recognition or mobile deployment in future iterations.

In reviewing related work, this study positions itself among practical, real-world solutions by prioritizing affordability, offline functionality, and user accessibility. While previous research has made significant strides in sign language recognition, few systems offer support for regional dialects or are designed to function without internet access on low-resource hardware. This system contributes to that space by demonstrating that localized, culturally aware assistive technology is both feasible and impactful.

Though the current implementation focuses on isolated letter recognition, it establishes a strong technical and conceptual foundation for broader applications. Future work may include incorporating sequence models such as LSTMs or Transformers to handle word-level or phrase-based recognition, improving the dataset through community collaboration, and integrating the system into mobile or wearable platforms.

Ultimately, this project reflects a meaningful contribution to inclusive design and digital accessibility. It underscores the importance of adapting emerging technologies to local needs, and it offers a starting point for further innovations that support communication equity for Deaf individuals in Algeria and beyond.

6.1 Limitations

While the current version of the SignPin project demonstrates effective letter-level gesture recognition and real-time audio translation in Algerian Darija, several limitations remain, highlighting areas for improvement:

- There is no mobile connectivity; all processing occurs on the device, which prevents access to larger translation libraries or remote updates.
- The current version is designed for one-way communication only; individuals who do not use sign language cannot directly respond via the system.
- Audio output is limited to Algerian Darija; there is no support for other languages such as French or English, reducing accessibility in multilingual environments.
- The gesture dataset is limited in scope, lacking regional dialects of Algerian Sign Language and common conversational phrases.
- User personalization features, such as custom phrases or vocabulary expansion, are not available in the present version.
- No speech-to-text feature is integrated; Deaf users remain dependent on reading physical gestures rather than receiving spoken feedback in text format.

6.2 Future Perspectives

In future versions of the SignPin project, several enhancements are planned to broaden its capabilities and improve user experience:

- **Mobile Connectivity (Bluetooth/Wi-Fi):**
  - Enable connection to smartphones.
  - Allow use of larger translation dictionaries stored on the phone.
  - Support the creation of personalized phrases.
  - Facilitate vocabulary updates and UI customization via a companion mobile app.
  - Enable offline synchronization and user analytics for improved personalization and performance tracking.
- **Two-Way Communication via Voice Sensor:**
  - Integrate an inbuilt microphone and speech-to-text engine.
  - Convert speech from hearing individuals into text for display on the device or app.
  - Support real-time, two-way interaction to enhance independence in social and professional settings.
- **Expanded Gesture Recognition Dataset:**
  - Include regional dialects of Algerian Sign Language (Darija).

  - Add common conversational phrases and commands.
  - Improve context-awareness and responsiveness of the system.

- **Multilingual Audio Output:**
  - Provide audio output in multiple languages: Arabic, French, and English.
  - Increase accessibility for Algeria's multilingual population.
  - Enhance usability across diverse cultural and linguistic contexts.

Bibliography

[1] Karen Emmorey. *Language, Cognition, and the Brain: Insights from Sign Language Research*. Lawrence Erlbaum Associates, 2002.

[2] A. Hussain, S. Afridi, and M. Hussain. Sign language recognition using deep learning: Current trends and challenges. *Journal of Artificial Intelligence and Soft Computing Research*, 10(1):5–18, 2020.

[3] S. Mitra and T. Acharya. Gesture recognition: A survey. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 37(3):311–324, 2007.

[4] Camillo Lugaresi et al. Mediapipe: Cross-platform, customizable ml solutions for live and streaming media. `https://google.github.io/mediapipe/`, 2020.

[5] R. Wang and Q. Ji. A survey on hand gesture recognition. *Computer Vision and Image Understanding*, 160:103–117, 2017.

[6] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[7] Lokesh Kumar Viswavarapu. *Real-Time Finger Spelling American Sign Language Recognition Using Deep Convolutional Neural Networks*. Master's thesis, University of North Texas, Denton, TX, USA, 2018.

[8] M. Alsulaiman et al. Deep learning framework for dynamic arabic sign language recognition. *ISWA (International Semantic Web Applications)*, 2021.

[9] E. Tuba and A. Tuncer. Hand gesture recognition based on deep learning and hybrid features. *International Journal of Intelligent Technologies (IJIT)*, 2023.

[10] N.S. Alghamdi et al. Real-time arabic sign language alphabet recognition using deep learning. *IEEE Access*, 2021.

[11] N. Lussier et al. Mobile cross-lingual sign language translator. *ACM Transactions on Accessible Computing (TACCESS)*, 2023.

[12] Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural networks*, 61:85–117, 2015.

[13] Kunihiko Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological cybernetics*, 36(4):193–202, 1980.

[14] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[15] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. *IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.

[16] Inna Logunova. A guide to transfer learning. Serokell blog, November 2022. Accessed: 2025-06-28.

[17] H. Karvonen, T. Leppänen, and A. Ylisaukko-oja. Embedded development platforms: A comparative study of arduino, raspberry pi and beaglebone. In *Procedia Computer Science*, volume 13, pages 105–114, 2012.

[18] S. Pinto, J. Cabral, and A. Tavares. *Embedded Systems: Hardware, Software, and Applications*. Springer, 2019.

[19] E. Upton and G. Halfacree. *Raspberry Pi User Guide*. Wiley, 2nd edition, 2014.

[20] I. Ali, K. Munir, and M. Aftab. Internet of things applications: Recent advances and future challenges. *Journal of Network and Computer Applications*, 143:127–149, 2019.

[21] BINARYUPDATES.COM. Raspberry Pi Single Board Computer. `https://binaryupdates.com/raspberry-pi-single-board-computer/`, 2014. Accessed: 2025-06-28.

[22] Raspberry Pi Foundation. Raspberry pi os (bullseye), 2021. Accessed: 2025-06-05.

[23] Raspberry Pi Forum Users. Compatibility issues on bookworm, 2024. Accessed: 2025-06-05.

[24] GitHub Contributors. Issues with mediapipe and tensorflow on bookworm, 2024. Accessed: 2025-06-05.

[25] Mark Lutz. *Learning Python*. O'Reilly Media, 5th edition, 2013.

[26] Camillo Lugaresi, Jonathan Tang, Manu Puppala, Ameesh Makadia, Nikita Nasrabadi, Michael Chang, Rui Fan, Sen Lee, Wan Chang, Tim Dods, et al. Mediapipe: A framework for building perception pipelines. *arXiv preprint arXiv:1906.08172*, 2019.

[27] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. TensorFlow: Large-scale machine learning on heterogeneous systems. `https://www.tensorflow.org`, 2015. Software available from tensorflow.org.

[28] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4510–4520, 2018.

[29] Pete Shinners. *PyGame - Python Game Development*, 2001. Accessed: 2025-06-12.

[30] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In *Advances in Neural Information Processing Systems*, volume 27, pages 3320–3328, 2014.

[31] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. In *arXiv preprint arXiv:1704.04861*, 2017.

[32] L. García, L. Parra, J.M. Jimenez, J. Lloret, and P. Lorenz. Iot-based smart irrigation systems: An overview on the recent trends on sensors and iot systems for irrigation in precision agriculture. *Computers and Electronics in Agriculture*, 136:91–104, 2017.

[33] Derrick Spooner, George Silowash, Daniel Costa, and Michael Albrethsen. Navigating the insider threat tool landscape: Low-cost technical solutions to jump start an insider threat program. In *2018 IEEE Symposium on Security and Privacy Workshops (SPW)*, pages 219–224, San Francisco, CA, USA, 2018. IEEE.

[34] Mohamed Amine Mahroug and Abdelhafid Zeghidour. Deep-RSL: Deep Learning for Sign Language Recognition from a Video Sequence. Master's thesis, Université des Sciences et de la Technologie Houari Boumediene (USTHB), Algiers, Algeria, 2023.

[35] Hassan Moin, Chaojun Yang, Victor Leung, Howard Ho-Ching, Faezeh Abtahi, Xiao Zhang, Saba Emrani, Dinesh Bharadia, Shadi Dayeh, and Patrick P. Mercier. A low-cost smart glove for american sign language translation. In *2017 IEEE Sensors*, pages 1–3. IEEE, 2017.

[36] Navid Azodi and Thomas Pryor. Signaloud gloves: Translating sign language into speech. `https://www.washington.edu/news/2016/04/12/sign-language-translation-glove-wins-lemelson-mit-student-prize/`, 2016. Accessed: 2025-06-11.

[37] Muhammad Al-Qurishi, Farhan Ahmad, Ammar Mohammed, Ahmed Barnawi, Samee U. Khan, and Albert Zomaya. Deep learning for sign language recognition: Current techniques, benchmarks, and open issues. *IEEE Transactions on Artificial Intelligence*, 4(1):2–24, 2023.

[38] Raspberry pi camera module 3 datasheet. `https://www.raspberrypi.com/documentation/accessories/camera.html`, 2023. Accessed June 2025.

[39] Raspberry Pi Foundation. Power supply for raspberry pi zero 2 w, 2022. Accessed June 2025.

[40] Eric W. Healy, Srinivas Kothapally, Ian McCowan, and DeLiang Wang. Deep learning based speaker separation and dereverberation can generalize across different languages to improve intelligibility. *The Journal of the Acoustical Society of America*, 150(4):2526–2538, 2021.

[41] Zhanna Sarsenbayeva, Tommi Kinnunen, Markku Vainio, Juha Keskinen, and Jani Pekkanen. Methodological standards in accessibility research-plxbccr-on motor impairments: A survey. *ACM Computing Surveys*, 55(7):1–35, 2022.

[42] Technology Networks. Understanding battery types, components and the role of battery material testing in development. `https://www.technologynetworks.com/applied-sciences/articles/understanding-battery-types-components-and-the-role-of-battery-material-testing-in-` 2023.

[43] Raspberry Pi Foundation. Raspberry pi zero 2 w. https://www.raspberrypi.com/products/raspberry-pi-zero-2-w/, 2021. Accessed: 2025-06-02.

[44] TechTarget. Digital camera. https://www.techtarget.com/whatis/definition/digital-camera, n.d. Accessed: 2025-06-02.

[45] Grady Booch, James Rumbaugh, and Ivar Jacobson. *The Unified Modeling Language User Guide*. Addison-Wesley, Boston, 2nd edition, 2005.

[46] Martin Fowler. *UML Distilled: A Brief Guide to the Standard Object Modeling Language*. Addison-Wesley, Boston, 3rd edition, 2004.

[47] Sebastian Raschka and Vahid Mirjalili. *Python Machine Learning: Machine Learning and Deep Learning with Python, scikit-learn, and TensorFlow 2*. Packt Publishing, 3rd edition, 2019.

[48] Simon Richardson. *Programming the Raspberry Pi: Getting Started with Python*. McGraw-Hill Education, 2nd edition, 2016.