



People's Democratic Republic of Algeria
Ministry of Higher Education and Scientific Research
University of Mohamed Khider – BISKRA
Faculty of Exact Sciences, Science of Nature and Life
Computer Science Department

Number Of Order :SIOD01/M2/2024

Dissertation

Presented to obtain the academic master's degree in

Computer Science

Option: Information Systems, Optimization, and Decision (SIOD)

Title

Recognition of Hand Gestures Using Deep Learning Techniques

Presented by :
DJELLA Selma

Defended On 11/06/2024 By the Jury Composed Of :

Dr.Benameur Sabrina	MCB	President
Dr.Babahenini Sara	MAC	Supervisor
Dr.Abdelli Adel	MAC	Examiner

Academic year2023-2024

Acknowledgements

Alright, first and foremost, praise be to Allah for the joy of accomplishment, and praise be to Allah at the beginning and the end. I would like to express my sincere gratitude to Allah Almighty for His abundant blessings, wise guidance, and limitless generosity that have accompanied me throughout my academic journey. Without His divine support, I would not have been able to achieve this distinguished accomplishment.

*To my supervisor **Dr. Sara Babahenini**, my sincere gratitude and utmost respect for all the efforts you have made to help me achieve my goals and develop my skills. Your gentle touch and valuable guidance were not only helpful but also a source of inspiration. Thank you for the support, and I hope to always live up to your expectations.*

*To **my father**, who has illuminated my paths, guided my way, and been my role model every step of the way, and to **my loving mother** and her daily prayers for me.*

*To **my sister rand**, brothers, friends, and those who stood by me and supported me throughout my educational journey.*

*To **my secret favorite person**, thank you for you for your encouragement. I will never forget your wise words and continuous support.*

To all who guided, directed, and motivated me, I dedicate this humble work and the fruit of my efforts to all of you.

DJELLA SELMA

Abstract

by DJELLA Selma

Hand gestures play a vital role in human communication, helping to enhance non-verbal communication and clarify meanings. These gestures improve message understanding and direct attention, making conversations more dynamic and interactive. Additionally, hand gestures express emotions and situations clearly and tangibly, enriching human interaction and making communication more expressive and understandable.

These hand gestures vary significantly and encompass classifications. Hand gestures are complex in a way that cannot be overlooked, as their meanings and uses can change depending on culture and social context. In this study, systems based on machine learning algorithms and deep learning were proposed and utilized to analyze hand gestures.

We used comprehensive databases containing diverse sets of images of hand gestures to develop and evaluate the system. Through leveraging machine learning and deep learning, our system aims to provide accurate classification and comprehensive analysis of hand gestures. The system can accurately identify patterns and expressions in hand gestures by analyzing the images and extracting distinctive features. This information can be used by researchers and specialists in various fields, such as human interaction, non-verbal communication, and smart user interface development.

Keywords: Hand gesture, Classification, Deep learning, CNN

Résumé

La main joue un rôle crucial dans notre interaction quotidienne et la communication humaine, et toute blessure ou changement dans les gestes manuels peut avoir un impact significatif sur notre capacité à communiquer et à comprendre les autres. Une technique utilisée pour évaluer les gestes manuels est la détection de mouvement, qui enregistre les mouvements et les positions de la main au fil du temps. Les signaux de détection de mouvement sont des outils essentiels dans l'étude de l'interaction humaine et de la communication non verbale.

Détecter et classer différents types de gestes manuels est un défi complexe qui nécessite l'application de méthodes avancées. Dans cette étude, nous présentons un système qui utilise des algorithmes d'apprentissage automatique et d'apprentissage en profondeur pour relever ce défi. Le système se compose d'une seule étape principale : "la détection des types de gestes manuels". Pour développer et évaluer le système, nous utilisons une base de données complète contenant une variété de gestes manuels. En exploitant la puissance de l'apprentissage automatique et de l'apprentissage en profondeur, notre système vise à fournir une compréhension précise et détaillée des gestes manuels et à offrir des solutions efficaces pour améliorer la communication et l'interaction humaine. Grâce à l'analyse des signaux de détection de mouvement, le système peut aider les chercheurs et les professionnels à comprendre en profondeur le langage corporel et à renforcer la communication humaine.

Mots clés: détection des types de gestes manuels, les gestes manuels, Deep Learning, classification des gestes de la main.

ملخص

اليد تلعب دورًا حيويًا في تفاعلنا اليومي والاتصال البشري، وأي إصابة أو تغيير في الأيماءات اليدوية يمكن أن يؤثر بشكل كبير على قدرتنا على التواصل وفهم الآخرين. تقنية تُستخدم لتقييم الأيماءات اليدوية هي استشعار الحركة، الذي يسجل حركات ومواقف اليد عبر الزمن. إشارات استشعار الحركة تعد أداة أساسية في دراسات التفاعل البشري وفهم اللغة الجسدية.

اكتشاف وتصنيف أنواع مختلفة من الأيماءات اليدوية هو تحدي معقد يتطلب تطبيق أساليب متقدمة. في هذه الدراسة، نقدم نظامًا يستخدم خوارزميات التعلم الآلي والتعلم العميق لمواجهة هذا التحدي. يتألف النظام من مرحلة واحدة رئيسية: "اكتشاف أنواع الأيماءات اليدوية". لتطوير وتقييم النظام، نستخدم قاعدة بيانات شاملة تحتوي على مجموعة متنوعة من الأيماءات اليدوية. من خلال استغلال قوة التعلم الآلي والتعلم العميق، يهدف نظامنا إلى توفير فهم دقيق ومفصل للأيماءات اليدوية وتوفير حلول فعالة لتحسين التواصل والتفاعل البشري. من خلال تحليل إشارات استشعار الحركة، يمكن للنظام مساعدة الباحثين والمهنيين في فهم عميق للغة الجسدية وتعزيز الاتصال البشري.

الكلمات المفتاحية: كشف أنواع الأيماءات اليدوية، الإيماءات اليدوية، التعلم العميق، تصنيف الإيماءات اليدوية.

Contents

1	Hand Gestures Recognition	1
1.1	Introduction	2
1.2	The Role of the Hand in Human Gesture	3
1.2.1	Anatomy of the hand	3
1.2.2	Anthropometry of the hand	5
1.2.3	The Biomechanics of the Hand	6
1.2.4	Hand gestures and positions	9
1.3	Applications of hand gesture recognition	12
1.3.1	Virtual Reality and Augmented Reality	12
1.3.2	Robotics and Telepresence	12
1.3.3	Desktop and Tablet PC Applications	15
1.3.4	Gaming	15
1.3.5	Education	16
1.3.6	Clinical and health	18
1.4	Devices for Gesture Interaction and Acquisition	19
1.4.1	Contact-Based Gesture Recognition Technologies	20
1.4.2	Vision-Based Devices and Their Varied Sensor Technologies	22
1.4.3	Advantages and disadvantages of both technologies	23
1.5	Characterizing gestures	24
1.5.1	Gesture interfaces with matrix	24
1.5.2	Sign language	24
1.6	Conclusion	29
2	Convolutional Neural Networks for hand recognition	30
2.1	Introduction	31
2.2	Artificial Intelligence	32
2.3	Deep Learning and Its Architectures	33
2.3.1	Convolutional Neural Networks (CNN)	34
2.3.2	Recurrent Neural Networks (RNN)	38
2.3.3	Transfer Learning	40
2.4	Related work	42
2.4.1	Hand gesture recognition based on VGG16 and Alexnet	43
2.4.2	Hand gesture Classification Model Using EMG Signals with CNN and Traditional Classification Methods	44

2.4.3	Hand gesture recognition based on ensemble-based Convolutional Neural Network	45
2.5	Conclusion	46
3	System Design	47
3.1	Introduction	48
3.2	Proposed Approach for Hand Gesture Detection	49
3.2.1	Preprocessing Phase	50
3.2.2	Feature extraction Phase	52
3.2.3	Proposed CNN Architectures	54
3.2.4	proposed approach for hand gesture detection	55
3.3	Conclusion	57
4	Implementation and Results	58
4.1	Introduction	59
4.2	Implementation and Evaluation Tools	60
4.2.1	Python language	60
4.2.2	Kaggle	61
4.2.3	Keras	61
4.2.4	OpenCv	62
4.2.5	Tensorflow	62
4.2.6	Numpy	63
4.2.7	Matplotlib	63
4.3	Performance Evaluation Metrics in Classification Tasks	64
4.3.1	Accuracy	64
4.3.2	Precision	65
4.3.3	Recall	65
4.3.4	F1-score	65
4.3.5	Confusion Matrix	65
4.3.6	ROC	66
4.4	Hand Gesture detection model implementation	66
4.4.1	Dataset description	66
4.4.2	Data splitting	66
4.4.3	Preprocessing phase	67
4.4.4	Feature extraction phase and Model Training	69
4.4.5	Experiments and results	71
4.5	Creating a Model for Detecting Hand Gestures	75

4.5.1	Preprocessing phase	75
4.5.2	Feature extraction phase	76
4.5.3	Model Learning	77
4.5.4	Experiments and results	77
4.6	System deployment	79
4.6.1	Predicting Image Classes from a File Path	79
4.6.2	Displaying and Predicting Image Classes from a Test Set	80
4.6.3	Visualizing Image Predictions from Test Set Using Indexing	82
4.7	Comparison and discussion	83
4.7.1	Model Comparison: Performance and Effectiveness	83
4.7.2	Proposed Model vs. Related Work	84
4.8	Conclusion	85

List of Figures

1	Anatomy of the hand[23].	3
2	The anatomy of the wrist bones [28]	4
3	Metacarpals bones [3].	4
4	Phalanges [22].	5
5	Flexion/Extension and Abduction/Abduction angles of the index finger [12].	7
6	Manipulation Gestures.	10
7	Deictic Gesture	10
8	Examples of Semaphore Gestures [27]	11
9	system setup with the Kinect sensor	13
10	Prototype robot with Kinect sensor and a typical interaction scene [71].	14
11	Performing hand gesture to control the virtual game control a. rubik’s cub game implementation; b. rubik’s cub game [84].	16
12	Television control using hand gestures	17
13	Surgeon using gestix to browse medical images [75].	18
14	hand gesture controlled wheelchair [50].	19
15	(a) CyberGlove II. (b) IGS-190 body suit [48]	21
16	Sensor-based data glove(used sign Language Recognition translate sign language gestures into text or speech) [68].	21
17	Vision-Based Devices (a) PTZ camera, (b)Senz3D sensor [40]	23
18	(1).The sign [uncle] in LS (Algerian Sign Language) is represented (2). The sign [aunt] in LS is shown (3). The signs [hello] LS and [welcome] LS are depicted (4). The sign [disgusting] in LS is also included [35].	25
19	finger-spelling [47].	26
20	The schema of artificial intelligence.	32
21	Deep Learning Architectures [20].	34
22	A Typical Convolutional Neural Networks	35
23	Types of Pooling Layers [81].	36
24	Recurrent Neural Networks.	39
25	Difference between the traditional learning process of two differ- ent tasks and transfer learning. In transfer learning, task 1 is fully trained using a large dataset, and task 2 uses the knowledge ob- tained from task 1 to improve accuracy and learn faster [51].	41

26	VGG16 Architecture.	42
27	ResNet50 Architecture	42
28	Proposed method for hand gesture recognition based on VGG16 and AlexNet.	43
29	Visual representation of the methodology.	50
30	The preprocessing steps to be followed.	51
31	Feature extraction in local image image regions.	52
32	The network architecture of CNN [61].	55
33	Feature Extraction with Pre-trained CNN Models Workflow.	56
34	The Python version utilized on Kaggle.	60
35	Kaggle logo.	61
36	keras logo.	62
37	OpenCV logo.	62
38	TensorFlow logo.	63
39	Numpy logo.	63
40	Visualization Data by Matplotlib [39].	64
41	Confusion Matrix.	66
42	complex-valued, MIMO beat signal for preprocessing.	68
43	Example of an input preprocessed image.	68
44	Loading Data.	68
45	Define CNN Model for SAR Image Classification.	69
46	Defining the CNN model architecture.	70
47	Early stopping callback.	70
48	Analyzing Model Performance and Optimization Strategies.	72
49	Confusion Matrix for the model.	73
50	Receiver Operating Characteristic.	74
51	Complex Image Conversion to RGB Format.	75
52	Data Initialization Using Data Augmentation Techniques.	76
53	Proposed features extractor function using VGG16.	76
54	Improving Model Performance with Early Stopping and Data Augmentation.	77
55	ROC Curves for all classes.	78
56	Confusion Matrix.	79
57	Predicting image class from any link.	80
58	Predicting image class using test set index and displaying full image.	81
59	Predict of gesture ok.	81

60 Index-based Image Prediction and Display in Test Set. 82
61 predict of gesture fist. 83

List of Tables

1	Exploring hand attributes and mobility	6
2	Comparison between Contact-devices and Vision-devices.	24
3	Recently related work for Hand gesture recognition	44
4	Comparing CNN models for gesture recognition on multiple datasets.	45
5	Hand Gesture Recognition Dataset.	67
6	Number of samples per class in both training and testing sets.	67
7	Comparing our proposed model with related work.	84

General introduction

Hand gestures have become essential today due to their effective non-verbal communication, supported by communication technologies. Their cultural integration contributes to deeper understanding, aiding communication in noisy environments. They also find wide applications in modern technologies like signal recognition and motion control.

Classifying hand gestures plays a vital role in several cases, including human interaction with technology, signal recognition, motion control, and cultural interaction for understanding ideas and meanings effectively. The task of classifying gestures becomes relatively complex due to cultural diversity, contextual variations, requiring accuracy and deep understanding of non-verbal movements and expressions.

Researchers are driven to develop automated techniques for gesture classification due to the need for enhancing human-computer interaction systems reliant on hand gestures. Powerful tools for analyzing large amounts of hand gesture data include advanced machine learning techniques like deep neural networks for precise classification and understanding underlying meanings. These tools also encompass image processing techniques for image enhancement and feature extraction, probabilistic and statistical methods for data distribution analysis and pattern recognition, and mathematical classification algorithms for data analysis. Together, these tools aid in accurate and effective understanding and classification of hand gestures, contributing to the development of various applications in human-computer interaction and behavioral analysis fields.

My thesis introduces a comprehensive system for detecting and classifying hand gestures, aiming to leverage machine learning and deep learning algorithms in identifying various types of hand gestures. The system aims to achieve an accurate and thorough understanding of these gestures, contributing to developing effective applications in human-computer interaction and advanced human behavior recognition.

Where the first chapter presents an introduction to the importance of hand gesture recognition in human-computer interaction and reviews the applications and challenges in this field. Then chapter two explains the role of Convolutional Neural Networks (CNNs) in hand gesture recognition, detailing feature extraction

and performance improvement. Then chapter three presents the proposed system design, including the architecture and techniques for data collection and processing to achieve accurate gesture recognition. Finally chapter four discusses the system implementation and test results, analyzing performance, challenges, and successes in hand gesture recognition.

The dedication and professionalism evident in research and studies to improve the lives of individuals with speech and hearing impairments are truly commendable. These studies strive to develop interactive technological tools and systems to facilitate communication and enhance understanding of sign language and hand gestures. They highlight the wide-ranging opportunities provided by creating a comprehensive and interactive educational environment that fosters communication and understanding among all segments of society.

1 Hand Gestures Recognition

1.1 Introduction

In an era where human-computer interaction is deemed indispensable, understanding hand gestures and movements emerges as a pivotal element in shaping modern interaction systems. This first chapter of our thesis spotlights fundamental aspects of the hand, delving into its intricate anatomy, human dimensions, and associated biomechanics. This comprehensive exploration aims to open up a deeper understanding of the hand's vital role in facilitating seamless interaction between humans and computers.

The chapter doesn't merely focus on anatomical aspects but goes beyond to scrutinize intricate details of hand movements and postures. It presents advanced classifications derived from cutting-edge research in this dynamic field, emphasizing the significance of deciphering the language of the hand in the context of human-computer interaction. Meticulous examinations of key studies by esteemed researchers such as Maria Karam, M.C. Shraefel, and Kendon contribute to laying a robust foundation for advanced studies in this burgeoning domain.

As deep learning techniques continue to evolve, there is a growing interest in seamlessly integrating these advanced concepts into diverse applications. The chapter strategically highlights various application domains, transcending traditional boundaries. From virtual reality and robotics to desktop and tablet PC applications, games, sign language interpretation, television control, and gesture-to-speech technology, each domain represents a unique facet of the expanding possibilities within the field of hand gesture recognition.

This chapter provides a careful overview of the devices used in gesture sensing and interaction, emphasizing joint efforts to enhance the efficacy of human-computer interaction. It envisions a future where the intricate language of the hand seamlessly integrates with cutting-edge technologies, reshaping how we interact with and navigate the digital world.

1.2 The Role of the Hand in Human Gesture

1.2.1 Anatomy of the hand

Through thesis on the study of the Human Hand Anatomy-Based Prosthetic Hand [13] we have reached, that the human hand is composed of several key components, including carpal bones (wrist bones), metacarpal bones (palm bones), and phalanges (finger bones) as shown in Figure 1. Each finger typically consists of four bones: the metacarpal bone, proximal phalanx, middle phalanx, and distal phalanx. The thumb may have one less phalanx than the other fingers. This arrangement provides the hand with its intricate structure and flexibility, enabling a wide range of movements and functions.

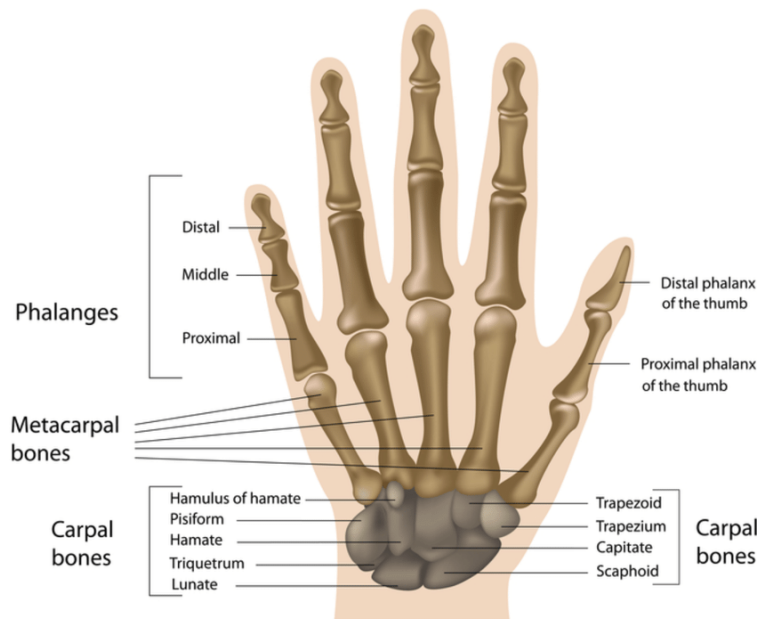


Figure 1: Anatomy of the hand[23].

- **Carpal:** is the area of the hand that connects to the forearm and contains a set of bones interconnected with each other. The wrist consists of eight bones that play a vital role in the movement and support of the hand (see in Figure 2).

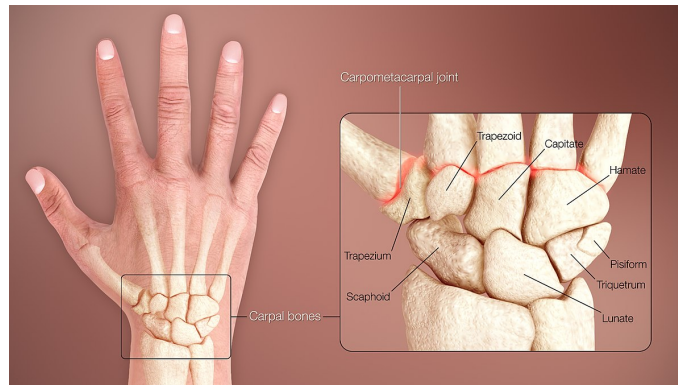


Figure 2: The anatomy of the wrist bones [28]

- **The metacarpals:** are the bones located between the carpal bones (wrist) and the phalanges (fingers) in the hand. There are five metacarpal bones in the human hand, corresponding to the five fingers. These bones constitute the midsection of the hand and are essential for the structure and mobility of the fingers (see Figure 3).

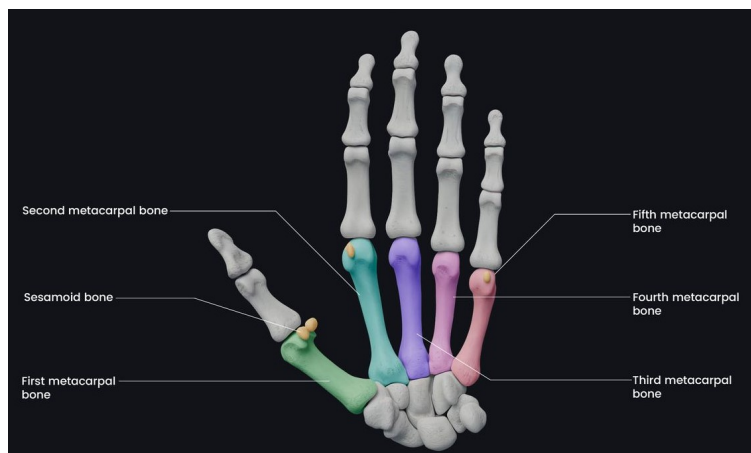


Figure 3: Metacarpals bones [3].

- **The phalanges:** are the bones of the fingers and toes. Phalanges consist of three main parts: the proximal phalanx, the intermediate phalanx (if present), and the distal phalanx. These bones form the structure of the fingers and toes, enabling their movement and flexibility. Each finger of the human hand typically has three phalanges, except for the thumb, which has two see Figure 4.

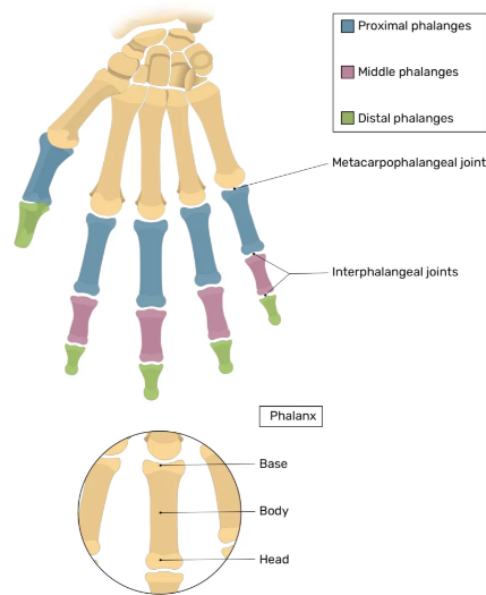


Figure 4: Phalanges [22].

1.2.2 Anthropometry of the hand

Anthropometry is the art of measuring the proportions of the human body, practiced in the fields of human engineering and criminology. The word originates from Greek, where "anthropos" signifies human, and "metron" denotes measurement. When applied to the hand, it involves calculating various ratios between the palm, fingers, and phalanges [11]. Examination of Hand Size and Joint Mobility in Pianists [77] This study aims to understand the extent of variation in hand size and joint mobility among professional pianists. Twenty dimensions of hand size, 17 ranges of active movement, and 11 features of passive joint mobility were tested on both hands of 127 male pianists aged between 17 and 63, and 111 female pianists aged between 16 and 64. The table 1 below illustrates the key findings:

Variables	Results
Male vs. Female Hand Size	The male hand shows significantly higher mean values in all hand-size variables, except for fingertip prominence 3–5.
Finger Spans with and without Thumb	All mean values of finger spans with the thumb were greater in men, and a similar trend was observed for finger spans without the thumb.
Ranges of Active Movement	Generally, ranges of active movement were higher in women.
Passive Joint Mobility	Significantly higher mean values for all characteristics were observed in women, with one exception having significantly higher values for the left hand.
Correlation with Age	A weak negative correlation was found between joint mobility and age, calculated only for male pianists.

Table 1: Exploring hand attributes and mobility

1.2.3 The Biomechanics of the Hand

Biomechanics of the hand can be defined, based on the two texts, as the study of the relationship between mechanical and biological forces occurring in the structure and function of the hand [77]. This study deals with how the mechanical components of the hand - such as joints and muscles - interact with environmental forces and different tasks. This includes understanding how the size and shape of objects grasped by the hand affect the forces the hand needs to apply and how design and function can be improved to avoid hand injuries and reduce fatigue during various tasks.

The anatomical details of the joints in the fingers and hand relate to the number of degrees of freedom for each joint :

- **Interphalangeal joints (IPD and IPP):** Each of these joints has only one degree of freedom: flexion and extension movement.

- **Metacarpophalangeal joints (MCP):** These joints have two degrees of freedom, which include flexion/extension and abduction/adduction, except for the thumb's MCP joint, which has only one degree of freedom (flexion/extension).
- **Thumb's Carpometacarpal joint (CMC):** This joint has three degrees of freedom, including flexion/extension, abduction/adduction, and a pseudo-rotation due to incongruity between the carpal bones and the base of the thumb's metacarpal, along with the relaxation of the connecting ligaments.

Biomechanical constraints on finger movements, where certain postures are not achievable. For example, the flexion/extension angle of the first phalanges of the four fingers ranges from 110° to 15° . Constraints can be either static or dynamic. Static constraints represent limits on the possible angular movements of the joints, whether in abduction/adduction or in flexion/extension. Figure 5 provides an example of flexion and extension angles $\theta_{f/e}^{IPD}$, $\theta_{f/e}^{IPP}$ and $\theta_{f/e}^{MCP}$, in addition to abduction/adduction angle, $\theta_{a/a}^{MCP}$ for the thumb of the hand.

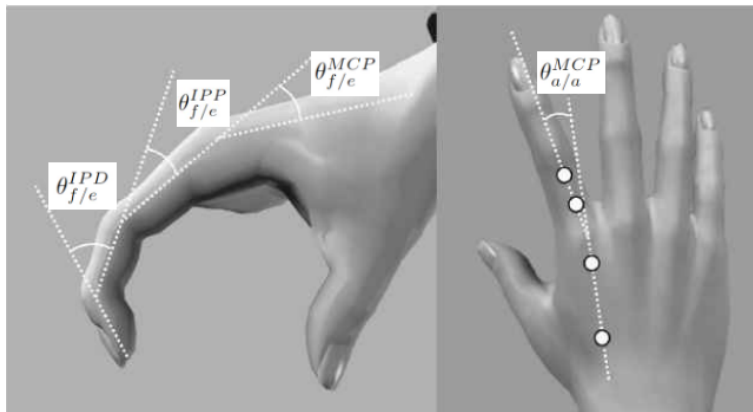


Figure 5: Flexion/Extension and Abduction/Adduction angles of the index finger [12].

The dynamic constraints represent the interdependencies between the degrees of freedom of the finger joints.

For the four fingers other than the thumb, these are:

- The relationship between the flexions of the proximal phalanx and the middle phalanx (see Equation 1):

$$\theta_{f/e}^{\text{IPD}} = \frac{2}{3}\theta_{f/e}^{\text{IPP}} \quad (1)$$

- The interdependence between flexion/extension and abduction/adduction of each phalanx is crucial. Indeed, the greater the angle of flexion/extension, the more limited the abduction or adduction of the fingers becomes. The maximum value of the abduction/adduction angle of the phalanx, given that the flexion/extension angle is not zero is determined as show in Equation 2:

$$\left| \theta_{a/a}^{\text{MCP}} \right| \leq \left(1 - \frac{1}{\theta_{f/e \text{ max}}^{\text{MCP}}} \right) \theta_{f/e}^{\text{MCP}} \times \theta_{a/a}^{\text{MCP}} \quad (2)$$

$\theta_{f/e \text{ max}}^{\text{MCP}}$: is the maximum value of the flexion/extension angle of the phalanx.

- The relationship between the flexions/extensions of adjacent finger phalanges for example, the flexion of the index or ring finger phalanx leads to the flexion of the middle finger phalanx (see Equation 3).

$$\theta_{f/e}^{\text{MCP Majeur}} \geq \max \left(\begin{array}{c} \theta_{f/e}^{\text{MCP Index}} - 25^\circ \\ \theta_{f/e}^{\text{MCP Annulaire}} - 45^\circ \\ \theta_{f/e \text{ min}}^{\text{MCP Majeur}} \end{array} \right) \quad (3)$$

$\theta_{f/e \text{ min}}^{\text{MCP Majeur}}$: is the minimum value of the flexion/extension angle of the middle phalanx of the middle finger.

Due to its unique structure, the movements of the thumb are subject to various constraints. For example, the angle of thumb flexion at the carpometacarpal joint ranges from -80 degrees to 25 degrees. Additionally, dynamic constraints link the flexion/extension and abduction/adduction of the first two thumb joints (see Equation 4).

$$\theta_{f/e}^{\text{MCP}} = 2 \times \left(\theta_{f/e}^{\text{CMC}} - \frac{\pi}{6} \right); \theta_{a/a}^{\text{MCP}} = \frac{7}{6} \times \theta_{a/a}^{\text{CMC}} \quad (4)$$

1.2.4 Hand gestures and positions

In general, gestures can be defined as a kinetic language that speaks without words, where movements of the hands and fingers are considered an effective means of non-verbal expression. Gestures can stand alone or accompany verbal communication to convey thoughts and feelings. Gestures are characterized by the ability to add a human dimension to understanding, allowing individuals to convey a message easily through hand movements that instantly convey emotions. Manual gestures can range from simplicity, such as a simple hand signal for confirmation, to complex expressions that require precise coordination to convey more intricate meanings. The use of hand gestures is an effective way to add depth and expression to communication, evident in message delivery and daily interaction [15].

To better understand gestures and postures of hand by [Lin et al 2000]:

- **Hand positions:** refers to the static or fixed position of the hand at a specific moment, without any movement. For example, when you extend your hand in front of you and hold it in a particular place without any motion, this is considered a "hand posture." This term is used to indicate the static position of the hand without considering any changes in location or movement.
- **Hand gestures:** Hand movement (gestures) represents the change in the hand's position over time. It denotes the dynamic motion of the hand or fingers within a short period. For instance, raising your hand to wave goodbye, this is considered "hand movement." This term emphasizes the dynamic change in position or state of the hand rather than stability, as in hand posture.

Many known hand gestures can be classified and categorized based on context and usage. Here are some common classifications for hand gestures:

- **Classification of hand gestures by Maria Karam and M.C. Schraefel:** Maria Karam and M.C. Schraefel presented a classification of gesture patterns, specifically those involving the use of the hand and arm, aiming to summarize various interaction types described in scientific literature [26]. Here are five main gesture patterns they introduced:
 - * **Manipulation Gestures:** The goal of these gestures is to control an entity by establishing a close relationship between the gesture's

movement and the manipulated entity (see Figure 6).



Figure 6: Manipulation Gestures.

- * **Deictic Gestures:** Used to express the identity or location of an object, often in virtual environments. These gestures can be considered implicit in other gestures, for instance, when pointing with a finger towards an object to manipulate (see Figure 7).



Figure 7: Deictic Gesture

- * **Semaphoric Gestures:** these gestures are part of any gestural system based on a conventional catalog of static or dynamic gestures. For example, the static gesture for the "OK" sign or the dynamic hand gesture for saying "goodbye" (see Figures 8a 8b, and 8c).

This classification provides an overview of how gestures are used in human-computer interactions. It helps understand how gestures can be categorized based on intention and usage in various contexts, which can be valuable in designing user interfaces and interactive experiences.

- **Classification of hand gestures by kendon:** The classification proposed by Kendon reflects a spectrum that synthesizes various reflections on gestural language, illustrating a progressive evolution in the

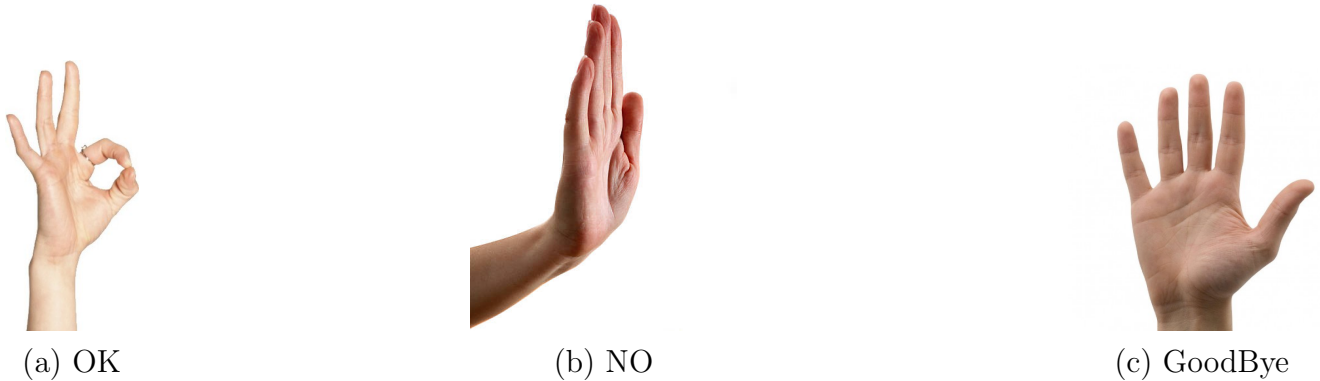


Figure 8: Examples of Semaphoric Gestures [27]

complexity and richness of gestural messages when moving from left to right [37].

- * **Spontaneous Gesticulation:** Encompasses all gestures that naturally occur and accompany speech. For example using hands naturally during the conversation, such as moving them to clarify a point or express specific emotions.
- * **Language slotted Gestures:** These gestures reflect linguistic translation but do not constitute a complete language. For example, gestures are used in sports officiating, where referees use specific gestures to signal the decision made.
- * **Pantomime:** Gestures aiming to provide a gestural image of an object or event.

Types of Iconic Gestures:

- **Spatiographic:** Show spatial relationships between objects.
- **Pictographic:** Evoke an object's shape, size, or surface state.
- **Kinegraphic:** Represent the movement of something or someone.
- * **Emblems:** Gestures that complement or substitute for words with a fixed societal meaning. For example, raising the middle finger indicates disrespect or greeting in certain cultures.
- * **Sign Languages:** The only ones considered genuine languages possess rules for formation, arrangement, and the ability to gener-

ate new lexical items. For example, American Sign Language or other sign languages are used by deaf communities.

In summary, these categories comprehensively explain the diversity of gestures and gestural language, ranging from spontaneous expressions and sub-language to symbolic pantomime, socially agreed-upon emblems, and complete sign languages.

1.3 Applications of hand gesture recognition

The use of computer vision in recognizing hand gestures and postures has led to the development of diverse applications. Examples include interpreting sign language, integration in robotics, and utilization in various fields such as virtual reality, gaming healthcare, and augmented reality [27]. Given the diverse potential applications of gesture recognition, we provide an overview of some application domains that benefit from gesture interactions :

1.3.1 Virtual Reality and Augmented Reality

Gesture usage in virtual and augmented reality applications has experienced widespread adoption in the field of computing. Virtual reality interactions rely on gestures to achieve a realistic interaction with virtual objects using hands, whether for 3D display interactions [57] or simulating 3D interactions on 2D displays [65].

For example as shown in Figure 9 system consists of an Alienware computer with a powerful processor and NVIDIA graphics card, and we have a Kinect device for tracking hand movements [62]. The setup allows interaction with a large LCD screen, where the user can control using hand gestures while standing two meters away from the screen.

1.3.2 Robotics and Telepresence

Telepresence and telerobotic applications are often found in space exploration and military research projects. The gestures used for interacting with and controlling robots are similar to those used in fully-immersed virtual reality interactions. However, in this context, the environments are often real. Operators receive live video feeds from cameras mounted on the robot [19]. In

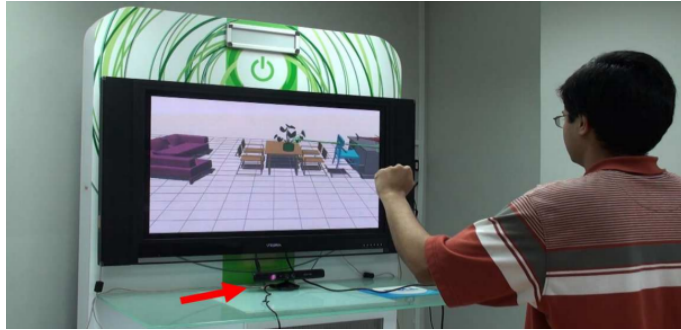


Figure 9: system setup with the Kinect sensor [62].

this setting, gestures can control the robot’s hand and arm movements to reach and interact with real objects, as well as navigate through the actual world.

Waldherr et al. [52] have introduced a template-based hand gesture recognition system designed for a mobile robot. This system incorporates gestures that instruct the robot to stop or follow, and it includes rudimentary pointing capabilities. However, it’s noteworthy that the gesture system relies on a color-based tracker, leading to certain limitations pertaining to the types of clothing and contrast with the background.

In the research paper [70], Van den Bergh and colleagues introduced an interactive system for real-time hand gesture recognition using a Time-of-Flight (ToF) camera. The system utilizes both depth images captured by the ToF camera and color images from the RGB camera for Haarlet-based hand gesture classification. Similar ToF-based systems are also discussed in the literature: Breuer et al. [5] fit an articulated hand model to depth data, although not in real-time; Kollorz et al. [31] use depth information to segment the hand and classify the 2D silhouette of the hand; and Soutschek et al. [63] employ a similar approach in a medical application.

This research introduces a real-time system for the robot’s interaction with hand gestures. Implemented on an interactive robot, the system employs a depth sensor (integrated with a color camera) to detect hand movements, translating them into interaction commands and 3D pointing directions. The hand gesture recognition algorithms outlined in [70] are adapted and implemented on the Kinect device, allowing for gesture detection irrespective of hand orientation,

particularly beneficial during pointing. The system is expanded to extract the 3D pointing direction, enabling the robot to define its goal on a map. The interaction system is integrated into a prototype robot using ROS, as illustrated in Figure 10.

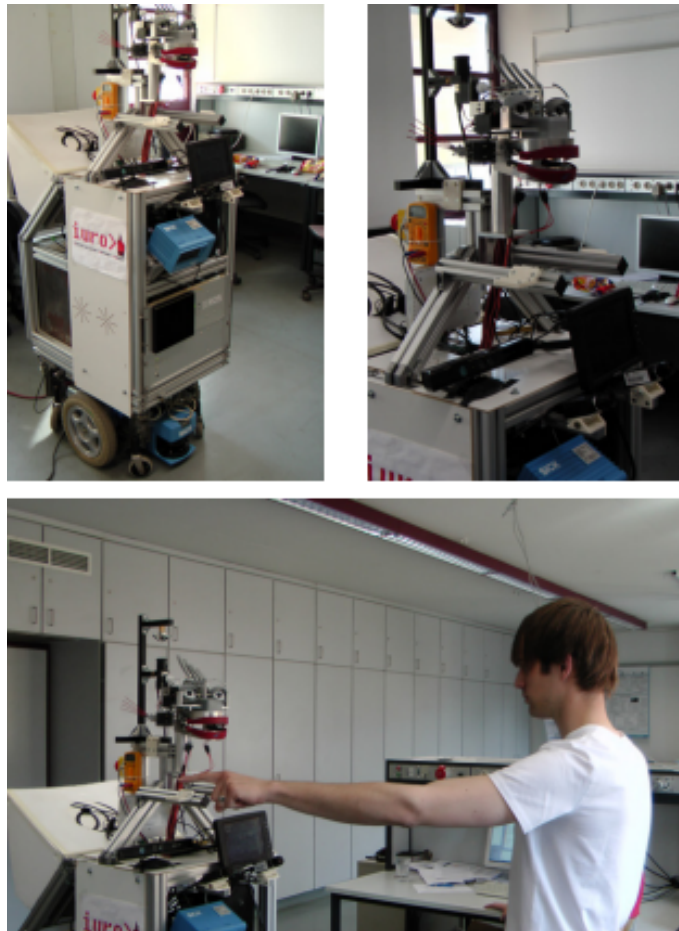


Figure 10: Prototype robot with Kinect sensor and a typical interaction scene [71].

A demonstration scenario showcases the robot following a human pointing directions and actively seeking a new person for additional interaction and guidance. The main contributions include:

- Real-time, orientation-invariant pointing direction detection and gesture recognition, enhancing the naturalness of human-robot interaction.
- Implementation of pointing and gesture detection on the cost-effective Kinect sensor.
- Application of 3D pointing gestures for directed robotic exploration tasks without prior map knowledge.

1.3.3 Desktop and Tablet PC Applications

In desktop computing applications, gestures can serve as an alternative interaction method to the mouse and keyboard [66]. Many gestures used in desktop computing tasks involve manipulating graphics or annotating and editing documents using pen-based gestures [60].

1.3.4 Gaming

The use of gestures in computer games. Researchers, led by Freeman et al. [17], mention monitoring the player's hand or body position to control the movement and orientation of interactive game objects, such as controlling a car's movement. On the other hand, Konrad et al. [32] conducted research where they used gestures to control the movement of avatars in a virtual world. PlayStation 2 participated in this domain by introducing the Eye Toy, a camera capable of tracking hand movements for interactive games [6]. This approach illustrates how gestures can be an effective means of controlling interactive elements within computer games, adding a new dimension of interaction and excitement to the gaming experience (see Figure 11).

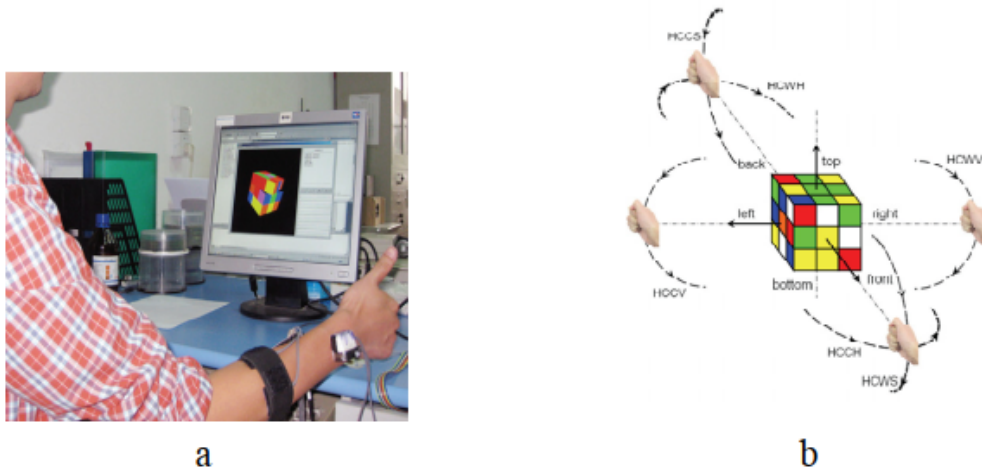


Figure 11: Performing hand gesture to control the virtual game control
 a. rubik's cub game implementation; b. rubik's cub game [84].

1.3.5 Education

- **Sign language recognition** Sign language represents a significant category of communicative gestures. Given its highly structured nature, it serves as an excellent testbed for vision algorithms [69]. Simultaneously, it can be an effective means to assist individuals with disabilities in interacting with computers. Sign language for the deaf, such as American Sign Language, is an example that has received substantial attention in the gesture literature [41] [64] [74] [78].
- **Television Control** Another application for hand postures and gestures involves controlling television devices (Joseph, 1999). Freeman (1995) developed a system for television control using hand gestures. By employing an open hand, users can change channels, turn the television on and off, adjust volume levels, and mute the sound. (see Figure 12a 12b 12c 12d 12e)

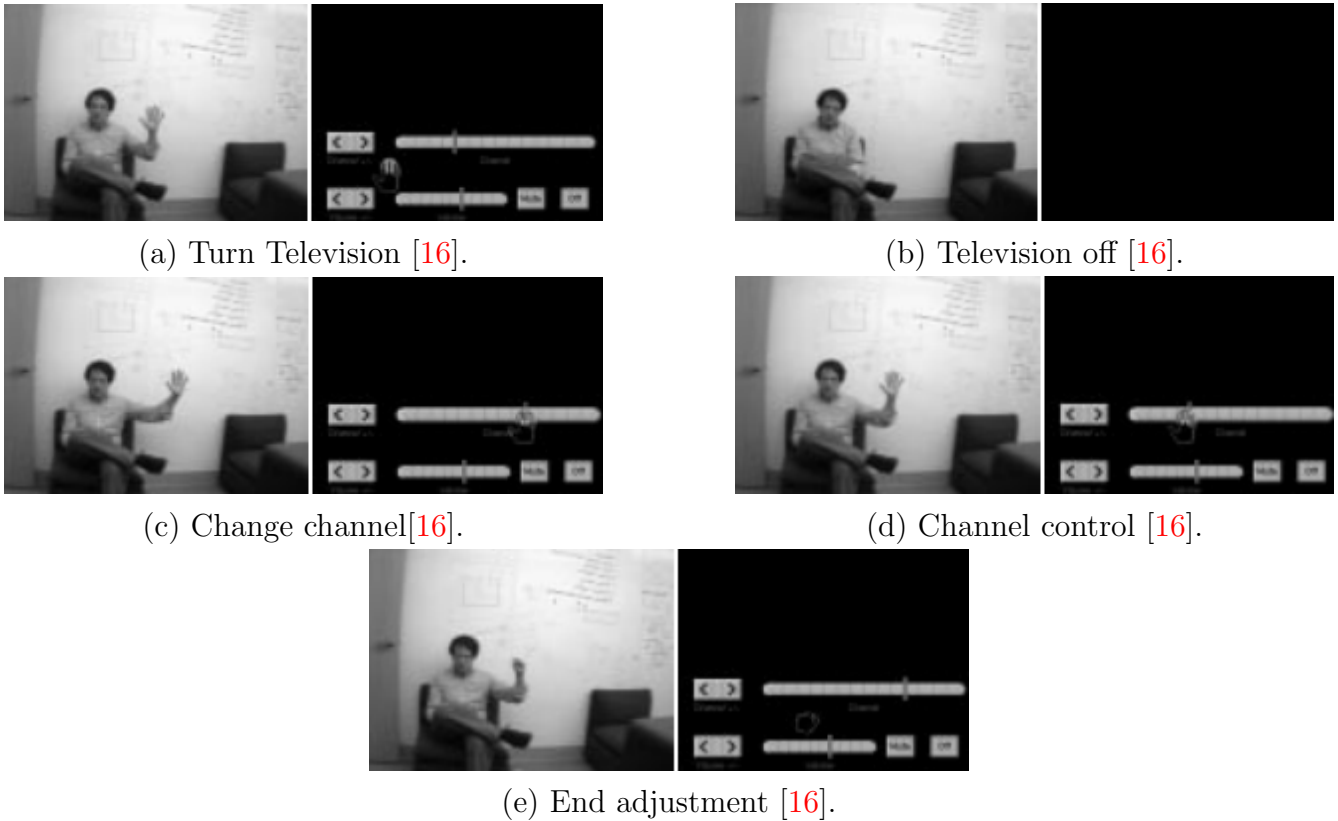


Figure 12: Television control using hand gestures

- Gesture-to-speech** In the realm of technology, Gesture-to-speech stands as a groundbreaking application, seamlessly transforming hand gestures into articulate speech. Pioneered by Joseph in 1999, this innovative system serves as a bridge for the hearing-impaired, enabling them to communicate with their surroundings via computers effortlessly 12c. A notable advancement, introduced by Fels in 1993 and 1998, is the Glove Talk system interface. This interface ingeniously links a speech synthesizer with a data glove device, employing neural networks to map intricate hand gestures to expressive speech. This technological synergy not only breaks down communication barriers for the hearing-impaired but also fosters easy interaction with others, eliminating the need for proficiency in sign language.

1.3.6 Clinical and health

The first system application for hand gesture control in the field of medical systems demonstrates significant benefits in improving hygiene and preventing infection spread in healthcare environments. Hand gestures can be utilized to enhance resource distribution in hospitals, interact with medical tools, control imaging displays, and provide effective support for users with disabilities during rehabilitation processes. For example, the "Face MOUSE" system illustrates how surgeons can control the movement of the laparoscope using facial gestures, offering an alternative to traditional control devices. Additionally, the "Gestix" system demonstrates how surgeons can naturally navigate through MRI images using hand gestures, meeting the requirements for immediate and natural usage [75] (see Figure13).



Figure 13: Surgeon using gestix to browse medical images [75].

When individuals experience mobility impairments in the lower limbs, the use of devices like electric wheelchairs controlled by joysticks becomes essential for facilitating movement. However, control of these devices remains significantly limited, especially for patients with upper limb injuries. Therefore, the development of a more natural and comfortable human-computer interface for electric wheelchair control has garnered increasing attention [82]. Hand motion-based control interfaces are considered a valuable option to enable users

to control electric wheelchairs naturally and efficiently. The range of gestures used in wheelchair control includes hand movements and their components, with the system benefiting from information about the two or three-dimensional motion path, as well as the direction and positioning of the hands. In this context, a new system for wheelchair control using hand gestures has been developed, consisting of five main hand positions and three compound states. This system is characterized by minimal hand movements, user engagement detection, and a strategy based on human-centered principles (see Figure 14).

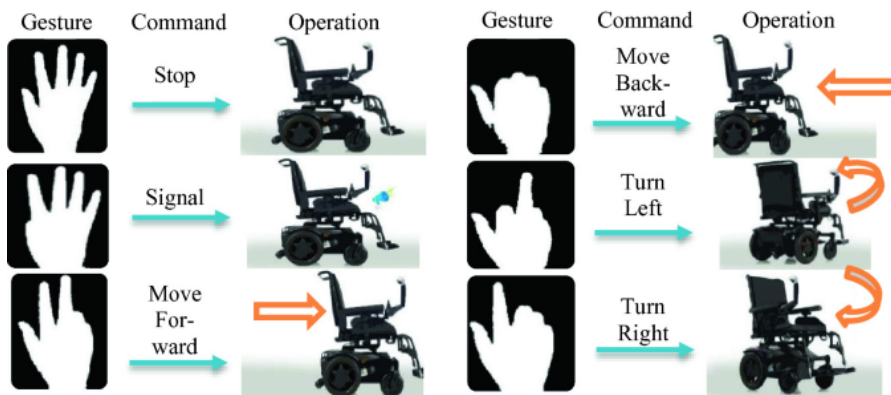


Figure 14: hand gesture controlled wheelchair [50].

1.4 Devices for Gesture Interaction and Acquisition

Technologies enabling hand gesture recognition refer to technical means that facilitate the implementation of accurate and precise systems for recognizing hand movements. This progress has been made possible due to advancements in modern technology. These technologies can be broadly classified into two main categories based on the type of sensors used. The first category includes devices that rely on direct contact with the hand, while the second category involves devices utilizing vision-based technologies to monitor and interpret hand movements. The text compares these two categories to highlight differences and challenges in implementing hand gesture recognition systems using these technologies [40].

1.4.1 Contact-Based Gesture Recognition Technologies

Contact-based gesture recognition devices rely on genuine user-device interaction to enhance recognition precision. Utilizing electromechanical devices, they capture gesture data through an array of motion sensors connected to the computer. Signal processing and pattern matching techniques are employed to analyze and categorize these data, enabling accurate identification of various gestures [40].

Some of the primary technologies in gesture recognition involve equipped gloves, facilitating precise tracking of hand movements. These technologies also encompass multi-touch screens, allowing direct interaction with devices through touch. Additionally, they rely on speed sensors to measure body movements and include tracking devices like control sticks, commonly used for remote gaming interactions. These devices may incorporate various sensors, ranging from multi-touch screens to three-axis speed sensors, providing a comprehensive interactive experience.

This type of device can be classified into five categories:

- **Magnetic Measurement Technology:** This technology relies on measuring changes in the artificial magnetic field to monitor and analyze the movement of objects. It utilizes magnetic trackers to achieve precision in motion tracking and is used in areas such as game control and interactive technologies.
- **Ultra-Sonic Technology:** This technology uses ultrasonic waves to monitor the movement of objects. It is effective in low-light environments or those containing magnetic obstacles and is used in applications such as indoor tracking and virtual reality games.
- **Mechanical Devices Technology:** This includes the use of wearable devices to record movements, such as the IGS-190 body suit and CyberGlove II gloves which are shown in Figure 15. It relies on mechanical trackers or inertial sensors to accurately record user movements.

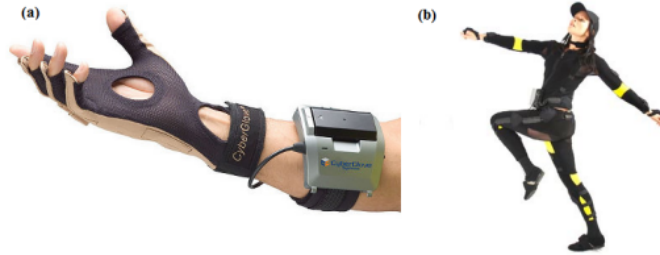


Figure 15: (a) CyberGlove II. (b) IGS-190 body suit [48]

This glove as seen in Figure 16, is a combination of sensory devices, such as flex sensors, accelerometers, and gyroscopes, to capture and interpret hand motion data. This technology allows for effective interaction between the user and the computer through hand gestures, enabling precise determination of palm and finger coordinates, as well as hand orientation and configuration. The use of mechanical trackers and sensors places it within the broader classification of mechanical devices used for motion tracking and interaction [46].

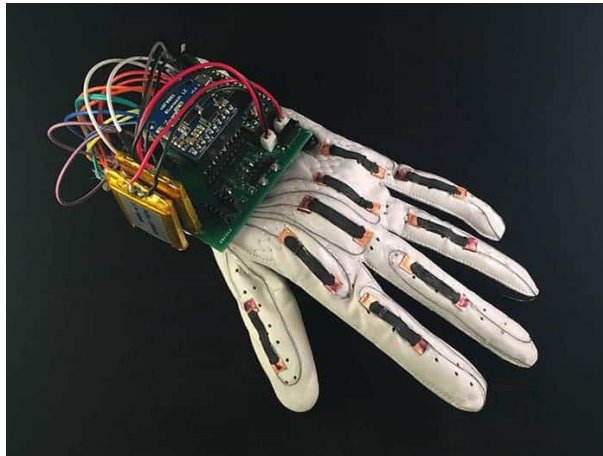


Figure 16: Sensor-based data glove(used sign Language Recognition translate sign language gestures into text or speech) [68].

- **Inertial Devices Technology:** This technology uses acceleration and rotation sensors to monitor the movement of objects. It is effective in motion tracking and is used in control devices and motion interaction technologies.
- **Haptics Technology:** This technology relies on measuring forces, vibrations, or movements performed by the user to achieve tactile interaction and

simulate the sense of touch in technology. It is used in touch screens and non-contact interface technologies.

1.4.2 Vision-Based Devices and Their Varied Sensor Technologies

Vision-based devices for gesture recognition rely on the use of one or multiple cameras to capture a video sequence. The recorded video is processed using image processing and artificial intelligence techniques to recognize and interpret gestures. The visual appearance of the hand can significantly vary due to its deformable nature (with over 25 degrees of freedom), different camera viewpoints, scales, lighting conditions, and variations in gesture execution (speed and structure) [33].

One of the advantages of these devices is their ability to capture a wide range of gestures, enhancing user interaction with systems and applications. Different types of vision sensors can be classified :

- **Infrared Cameras:** Utilize infrared radiation to capture an image, providing a silhouette of the body, hand, or object.
- **Color Cameras:** The most common type due to their cost-effectiveness, may possess interesting features such as fish-eye lenses for wide-angle vision.
- **PTZ Cameras (Pan-Tilt-Zoom):** Capable of rotating horizontally (panning), vertically (tilting), and manually or automatically zooming. They can focus on a specific object and track it within the camera's field of view (see Figure 17 (a)).
- **Depth Cameras:** Able to capture depth information, some devices integrate multiple sensor types, such as Kinect 2 and Senz3D, which capture both color and depth information (see Figure 17 (b)).
- **Stereo Cameras:** Feature two or more lenses to simulate human vision and capture 3D information.
- **Body Markers:** Used to enhance recognition accuracy, they can be passive (reflect light only) or active (such as LEDs). In these systems, each camera delivers 2D frames with marker positions from its perspective. A preprocessing step is typically required to interpret views and positions in a 3D space.

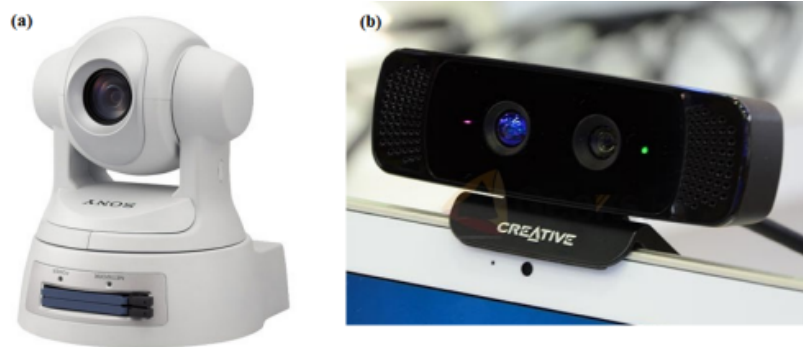


Figure 17: Vision-Based Devices (a) PTZ camera, (b) Senz3D sensor [40]

1.4.3 Advantages and disadvantages of both technologies

The mentioned technologies have advantages and disadvantages that must be considered when choosing the most suitable for our system [45]. On one hand, user cooperation is required for contact-based devices. Users need to wear some kind of clothes or devices while performing gestures, which can be uncomfortable for extended periods and may restrict user movements. However, in return, they provide more precise information and less complexity in implementation. Specifically, these techniques yield good results in simulated environments, but their use may not be practical in real-world scenarios due to their invasiveness and uncontrolled context. Additionally, the contact nature of these devices can lead to allergies due to physical contact with certain materials and a risk of cancer due to magnetic radiation. On the other hand, vision-based devices require a complex configuration and suffer from occlusion problems more than contact-based ones. Nevertheless, they are more user-friendly and, therefore, more likely to be used in the long run. For these reasons, vision-based systems have become more common in recent years. This discussion has been summarized in Table 2, which presents a comparison of the most relevant points for each type of technology.

critierion	contact-devices	vision-devices
User cooperation	Yes	No
User intrusive	Yes	No
Precise	Yes/No	No/yes
Flexible to configure	Yes	No
Flexible to use	No	Yes
Occlusion problem	No(Yes)	yes
Health issues	Yes(No)	No

Table 2: Comparison between Contact-devices and Vision-devices.

1.5 Characterizing gestures

1.5.1 Gesture interfaces with matrix

refer to a technology that enables devices to sense and analyze human movements using a matrix-based approach. In this context, user gestures and movements are captured and interpreted by representing the captured space as a matrix or grid [24].

For example, Converting Data into a Matrix : The captured data (images or depth data) are transformed into a matrix representation. Each element in the matrix represents a specific point in the captured space

1.5.2 Sign language

Sign Language is the means of expression used by deaf or hard-of-hearing communities to communicate among themselves. Sign Language is a genuine language in its own right, complete with vocabulary and syntax, making it the most evolved form of gestural communication (Cuxac, 2000; McNeil, 1992). Expressing sentences in Sign Language goes beyond gestures produced by the hands alone; the entire body can contribute to conveying a sentence. Three main parts come into play: the hands, the head (facial expressions and gaze), and the torso. In this study, our focus is solely on the hands; hence, the other parts of the body will not be addressed later on. Each sentence consists of a sequence of hand movements known as signs, which are arranged according to a syntax governed by spatial and temporal logic. Our primary interest lies in interpreting the signs and the relationships established between them, and these will be precisely described in the following sections [7].

- **Algerian Sign Language (LSA):** Based on the information provided, it can be said that Algerian Sign Language (LSA) is considered the primary language for the deaf community in Algeria. The language possesses a set of Standard Signs (SS) recognized by deaf communities to express various words, forming a comprehensible dictionary for those familiar with it [35]. Structurally, Algerian Sign Language includes Structures of Great Iconicity (SGI) that emerge from iconic representation. These structures enable the representation of size, shape, as well as the simulation of personal or dynamic situations.

In the process of expression, Algerian Sign Language involves the use of hands, face, gaze, and the entire body. Phrases are constructed using parameters such as hand configuration, articulation location, movement, orientation, contact, arrangement, and non-manual behavior.

The language is influenced by the vocal Arabic language in certain aspects of sign formation, as evidenced in some visual representations of words. Additionally, there is an influence from French Sign Language (LSF) and the Arabic Sign Dictionary in the structure of signs

On the technical side, a deficiency in the processing of sign languages in Arab countries in general. There is a need for technological development and language documentation, utilizing techniques such as XML-based gesture encoding.

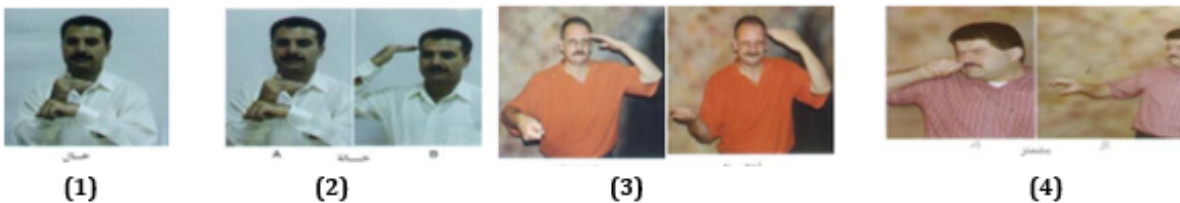


Figure 18: (1).The sign [uncle] in LS (Algerian Sign Language) is represented (2). The sign [aunt] in LS is shown (3). The signs [hello] LS and [welcome] LS are depicted (4). The sign [disgusting] in LS is also included [35].

- **French Sign Language (LSF):** [10] French Sign Language is a rich and unique system used for communication between the deaf and the hearing community. Its unique linguistic rules highlight its visual thinking and how the community interacts with it.

- **Official Recognition in France:** French Sign Language has been officially recognized as a self-contained language in the French educational system, requiring all relevant students to receive education in French Sign Language. Its learning and use are encouraged in exams and competitions.
- **Finger-spelling:** Finger-spelling is done with one hand in LSF, distinguishing it from British Sign Language family languages that use both hands (see Figure 19).



Figure 19: finger-spelling [47].

- * **Unique Linguistic Rules:** LSF features "three-dimensional" linguistic rules that allow expressing multiple ideas simultaneously, differing from the rules of French sign language.
- * **Uni and Multidirectional Verbs:** Uni and multidirectional verbs are used to specify the direction of communication, adding depth to expression.

- * **Signed French:** It involves using signs from LSF and ordering them according to the syntax of the French language, serving as a compromise for communication between French speakers who know signs but are not fluent in the syntax of LSF.
 - * **Unique Linguistic Order:** In LSF, the order is time first, then place, followed by the subject, and finally, the action, reflecting the visual thinking of the deaf.
 - * **Vocabulary and Sign Evolution:** The vocabulary is constantly evolving and enriching, reflecting the deaf community's discovery and access to specialized environments.
- **Interactions between both hands** A sign can involve either one or two hands, and the two hands interact differently. When both hands are involved in a sign, two cases arise [7].
 First Case: In this case, when both hands are involved in the sign, each hand has a specific role. One hand is known as the "dominant hand" and it describes the action itself, while the other hand, called the "non-dominant hand," serves as a reference point for this action. For example, if the sign represents the action of falling from a surface, the dominant hand (usually the right hand) will mimic the falling motion, while the non-dominant hand remains static and provides a reference point for this fall.
 Second Case: In this case, both hands are perfectly synchronized, with identical or symmetrical parameters. For example, if the sign represents the word "table," both hands will have identical formations and orientations, and they will move symmetrically along similar trajectories but in opposite directions. This reflects the complete synchronization between the movements to convey the intended concept.
 - **Different sign classes** Different sign categories represent an effective means of communication. In this context, signs can be divided into main categories, including Standard Signs, Specifiers, and Classifiers [2].
 Standard Signs are words with specific meanings, accurately linked to spoken languages, and play a significant role in motion recognition.
 Specifiers are used to describe objects, animals, or scenes characterized by strong iconic features based on hand shape, direction, and movement.
 Classifiers, similar to Specifiers in representation, function as pronouns and

can be used to represent the object referred to in sentences, contributing to accurate and effective communication.

1.6 Conclusion

In conclusion, hand gestures are fundamental elements that profoundly influence human-computer interaction. The intricate movements and postures of the hand form a complex language that bridges the gap between users and technology. Gesture recognition technologies have emerged as crucial tools in enhancing user experience and streamlining interactions with digital devices.

By leveraging the principles of machine learning and combining them with an understanding of human biomechanics and cognitive processes, researchers and developers can unlock new possibilities in gesture-based interfaces. However, there are challenges to overcome, such as refining algorithms to accurately interpret a wide range of gestures, ensuring robustness across diverse user populations, and integrating gesture recognition seamlessly into various applications and platforms.

Despite these challenges, the potential benefits of advanced gesture recognition systems are evident. With ongoing research and innovation, these systems have the potential to revolutionize user interfaces, making interactions more intuitive, efficient, and engaging. This evolution in gesture-based technology holds promise for transforming how we interact with digital environments, leading to enhanced user experiences and expanded capabilities in human-computer interaction.

2 Convolutional Neural Networks for hand recognition

2.1 Introduction

Artificial intelligence and machine learning plays a crucial role in hand gesture recognition technologies, which serve as a simple means of interacting with devices [43]. AI-powered algorithms have enabled sophisticated analysis and interpretation of hand movements, paving the way for intuitive control mechanisms and immersive user experiences. However, challenges persist in ensuring real-time accuracy and responsiveness of these systems, especially in complex environments or with diverse user inputs. Continuous efforts are underway to enhance the reliability and interpretability of artificial intelligence models in hand gesture recognition, aiming to create effective patterns across various applications and industries.

To address these challenges, researchers have used machine learning techniques, specifically Convolutional Neural Networks (CNNs), to develop robust and accurate models for classifying hand gesture images.

This chapter delves into using CNNs for classifying hand gestures. Hand gestures are crucial tools for interacting with devices and conveying information. CNNs can indeed be considered an effective solution for hand gesture recognition. Convolutional Neural Networks (CNNs) are considered an effective solution for hand gesture recognition, as they yield excellent results in image classification and accurately defining hand movements and gestures. This makes it a crucial and efficient choice for applications in hand gesture recognition [53].

In this chapter, we will focus on the fundamentals of CNNs, including their architecture, training process, and key components. We will discuss how CNNs can be applied to hand gesture recognition, providing insights into the benefits and challenges associated with this approach. Professionals, researchers, and developers in the field of hand gesture recognition can use the principles and applications of CNNs to enhance recognition accuracy and improve user experiences.

2.2 Artificial Intelligence

Artificial Intelligence (AI) is a field in computer science that focuses on developing systems and software capable of performing tasks that require intelligent thinking and decision-making. The goal of artificial intelligence is to create technological systems that can simulate and enhance human cognitive abilities, such as natural language understanding, learning from experience, analyzing big data, and making intelligent decisions.

Artificial Intelligence relies on advanced techniques such as artificial neural networks and machine learning (see Figure 20), enabling systems to adapt and improve automatically in response to changes in the environment or tasks. AI also enables systems to understand data, use it to make decisions and perform tasks intelligently without continuous human intervention.

Applications of Artificial Intelligence span a wide range of fields, including education, healthcare, technology, business, automation, machine translation, smart robotics, and more. Artificial Intelligence is considered an advanced and dynamic field that plays a crucial role in improving efficiency and advancing technology for a more sophisticated future [83].

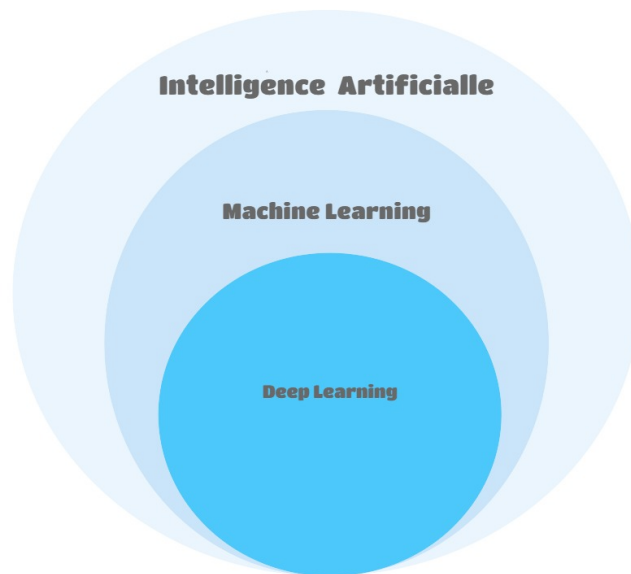


Figure 20: The schema of artificial intelligence.

2.3 Deep Learning and Its Architectures

Deep learning is an advanced approach in artificial intelligence, where computational models composed of multiple processing layers can learn data representations with multiple abstraction levels [36]. This method utilizes artificial neural networks to explore complex structures within large datasets. It employs the backpropagation algorithm to guide how the machine should adjust its internal parameters to compute representations at each layer from the representations in the previous layer.

Deep learning enables the discovery of intricate structures in high-dimensional data and finds application in various domains, such as image recognition, speech recognition, and interaction with large datasets. Overall, deep learning has made significant progress in solving problems that have challenged the artificial intelligence community for many years and is effective in uncovering complex structures in data, making it applicable in scientific, business, and government fields.

In deep learning architectures, deep neural networks are utilized to model and analyze data in a manner analogous to human thought processes. These networks rely on complex units known as artificial neurons, which resemble the neurons in the human brain.

Deep neural networks consist of multiple layers (see Figure 21), including hidden layers that allow for increased model accuracy. Utilizing deep neural networks is one of the prominent techniques in deep learning architectures, contributing effectively and accurately to modeling and analyzing advanced patterns in data [67].

Various deep learning architectures have demonstrated successful applications in diverse domains. Here are some well-known examples:

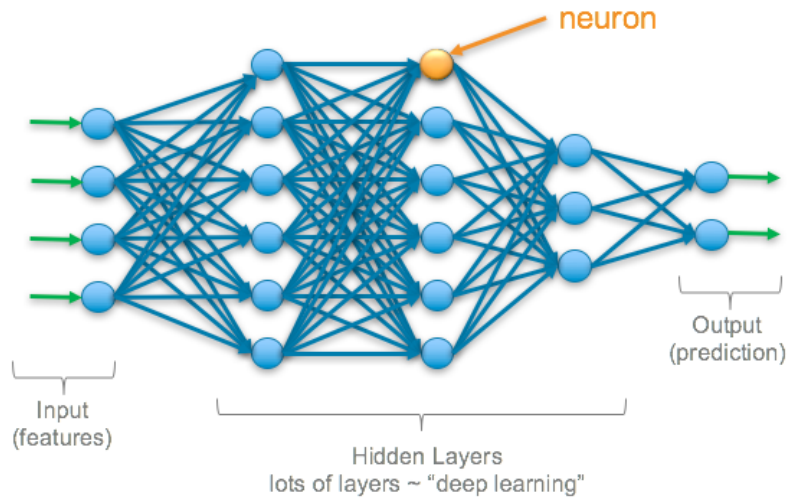


Figure 21: Deep Learning Architectures [20].

2.3.1 Convolutional Neural Networks (CNN)

Convolutional Neural Networks (CNNs) are powerful tools in the field of artificial intelligence, widely employed for image recognition tasks. Distinguished by their architecture, CNNs comprise sequences of convolutional and pooling layers, through which pertinent features are extracted from the original image. Subsequently, these features are utilized in one or more fully connected layers for prediction purposes. To employ CNNs for image recognition, prior training on a substantial dataset of labeled images containing the objects of interest is imperative. Throughout the training process, the CNN learns to associate the extracted features with the correct labels, facilitated by backpropagation and optimization. Once the training phase is completed, the network can be utilized for predictions on new images by passing the image through the network and selecting the label with the highest predicted probability [34].

The image 22 illustrates how the hand gesture representing the number three is processed using a deep neural network. The image is inputted into the inner layer, where it passes through the first hidden layer that applies a set of filters to the image to extract specific features like edges or wrinkles. The output of the first hidden layer then goes to the next hidden layer, where more filters are applied to extract features at a higher level. This process is repeated through several hidden layers until reaching the final hidden layer, which produces a set of features passed to the output layer. The output layer generates a probability distribution over three categories: ok gesture, number one, or number three. The

final figure reflects the category with the highest probability, which is the category the system classifies the image into. Essentially, the system trains itself to recognize different features in the image to distinguish between different categories by processing the image through different layers in the neural network.

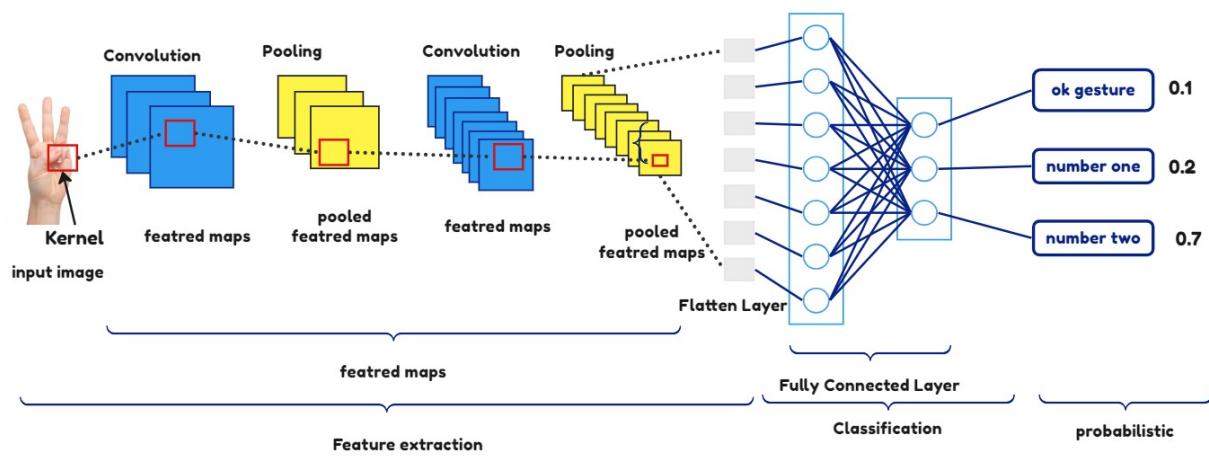


Figure 22: A Typical Convolutional Neural Networks .

So the Convolutional Neural Networks (CNNs) consist of three types of layers: convolutional layers, pooling layers, and fully connected layers. When these layers are stacked together, they form what is known as a CNN architecture. Simplifying it further, a CNN architecture can be thought of as a set of layers stacked together to recognize patterns in images based on the patterns they have been trained on :

A. Convolutional Layer: A layer in Convolutional Neural Networks (CNNs) used to perform convolution operations on incoming data [54]. The convolutional layer typically employs a set of filters (kernels) to analyze features present in the data, aiding in extracting local patterns and important information. Convolution is applied across the spatial dimension of the data to produce a two-dimensional activation map that represents the presence of the feature in each region. The convolutional layer is a fundamental component in CNN design, significantly contributing to the model’s ability to extract features and enhance performance

in image classification and recognition tasks. To determine the size of the output after passing through convolutional layers, we commonly employ this formula:

$$\frac{(V - R) + 2Z}{S + 1}$$

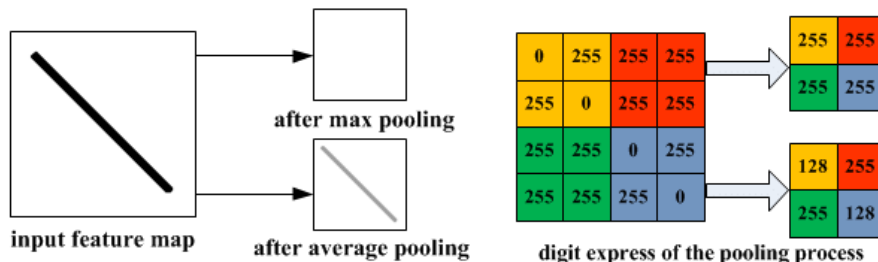
Input Size (V): The dimensions of the input volume (e.g., height × width × depth).

Filter Size (R): The size of the receptive field.

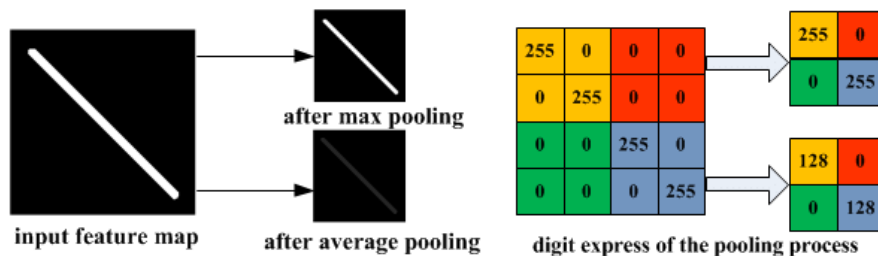
Padding (Z): The amount of zero padding added.

Stride (S): The step size for moving the filter across the input.

B. Pooling Layer: The pooling layer in deep neural networks is an essential component used to reduce the spatial dimensions of information in feature maps generated by convolution layers. The function of the pooling layer is to aggregate a group of values into fewer values, thereby reducing the data size and computational complexity in subsequent layers, which helps improve the model’s efficiency and reduces computational costs [18].



(a) Illustration of max pooling drawback



(b) Illustration of average pooling drawback

Figure 23: Types of Pooling Layers [81].

C. Fully Connected Layer: Fully Connected Layers in neural networks are

layers that connect every neuron in the previous layer to every neuron in the current layer [4]. In the context of deep convolutional neural networks (CNNs), Fully Connected Layers are typically placed at the end of the network to integrate the extracted features from previous layers and learn complex patterns, enabling the network to make final predictions. In deep neural networks, Fully Connected Layers comprise most of the network parameters, as each element in the previous layer is connected to each element in the current layer. These layers represent the more complex relationships between features extracted from earlier layers, aiding in improving the network's performance in tasks that require a deep understanding of data, such as classification, object recognition in images, and others.

D. Activation Function Layer: The Activation Function Layer in neural networks is the layer responsible for applying activation functions to the outputs of the previous layer in the network. The activation function determines whether the output from a given neuron should be propagated to the next layer or not, based on the computed value of weights and inputs [58].

In other words, the Activation Function Layer is the layer that introduces non-linearity to the neural network, aiding in learning and representing complexities in the data. Common activation functions include Sigmoid, Tanh, ReLU, and many other functions mentioned in the previous text.

In the complete training process, the Activation Function Layer is included between the layers of the neural network to enhance the network's ability to learn and represent.

E. Output Layer The output layer in the Convolutional Neural Network (CNN) is the final layer in the model [29]. This layer generates the final predictions or the desired output based on the inputs. The structure of this layer depends on the specific task; in classification tasks, softmax activation is commonly used to generate probabilities for each class, while linear activation function is often employed in regression tasks. The goal of the output layer is to guide the model towards making the final decision regarding the corresponding input class or generating the required final values.

2.3.2 Recurrent Neural Networks (RNN)

Recurrent Neural Networks (RNNs) are a type of neural network architecture that excels in processing sequential and temporal data. Unlike traditional feed-forward neural networks, RNNs have loops within their structure, allowing them to maintain a memory of past information while processing new inputs. This memory-like capability enables RNNs to capture dependencies and patterns over time, making them particularly well-suited for tasks such as natural language processing, speech recognition, and time series analysis [30].

The primary characteristic of RNNs is their capability to manage sequential data by transmitting information over time. In each time step, an RNN takes input and modifies its internal hidden state, acting as a memory of previous occurrences. This hidden state is subsequently utilized to produce an output (refer to Figure 24) and is also utilized as input for the subsequent time step, enabling the network to integrate contextual details from prior steps [85].

The recurrent nature of RNNs enables them to model long-term dependencies and context within a sequence, making them powerful tools for tasks such as language modeling, machine translation, sentiment analysis, and speech recognition. However, standard RNNs can suffer from the vanishing gradient problem, where gradients diminish as they propagate back in time, limiting the network's ability to capture long-term dependencies.

To address this issue, various types of recurrent neural networks have been developed, such as Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) [1]. These types incorporate specialized gating mechanisms that help in selectively retaining and updating information. These gated recurrent neural networks have proven effective in mitigating the vanishing gradient problem and have become popular choices for a wide range of applications.

Overall, recurrent neural networks and their variants have led to significant advancements in the field of sequential data processing. Their ability to model temporal dependencies and capture context over time has made them essential tools for tasks involving sequences, providing valuable insights and contributing to major breakthroughs across a wide range of domains.

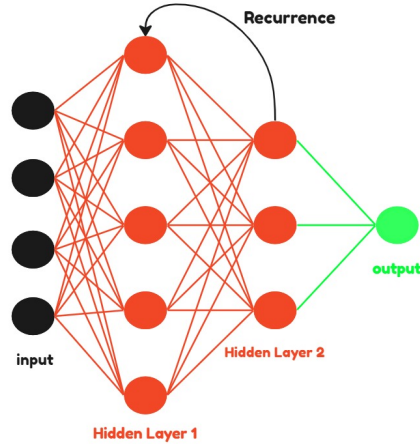


Figure 24: Recurrent Neural Networks.

Adjusting factors like the learning rate, batch size, epochs, early stopping criteria, and choice of optimizer plays a pivotal role in optimizing the training process of an artificial neural network (ANN) for precise predictions or classifications:

- A. **Learning rate:** The learning rate determines how quickly the model adjusts its parameters during training. A high learning rate may cause the model to overshoot the optimal parameters, while a low learning rate may result in slow convergence or suboptimal results.
- B. **Batch size:** The batch size refers to the number of training samples used in each forward and backward pass during training. Larger batch size can result in faster convergence, but it may also require more memory and computing resources.
- C. **Epochs:** The number of epochs is the number of times the model goes through the entire training dataset during training. Increasing the number of epochs can improve the accuracy of the model, but it may also increase the risk of overfitting the training data.
- D. **Early stopping:** Early stopping is a regularization technique used to prevent overfitting. It stops the training process early based on a predefined criterion. This callback monitors the validation loss and stops the training process if it does not improve for a specified number of epochs.

E. **Optimizer:** The optimizer is the algorithm used to update the model parameters during training. For example, Adam optimizer, SGD,...etc

2.3.3 Transfer Learning

Transfer Learning is a technique within the field of machine learning where a system or model utilizes knowledge gained from solving one task (the source task) to improve its performance in a related or different task (the target task), without starting the learning process from scratch. It involves leveraging pre-trained models, which have been trained on extensive and diverse datasets, and applying them as a starting point for learning new tasks or enhancing the model's performance in new tasks by adjusting certain layers or updating the model. Transfer Learning enables more efficient utilization of resources, reduces the need for extensive labeled data collection, and accelerates the development of machine learning models across various domains and applications [51].

Transfer Learning involves utilizing knowledge or expertise gained from solving one task to improve the performance of another related task, transferring part of the knowledge or the trained model from one task to another rather than starting the training process from scratch. On the other hand, traditional learning is a process where a machine learning model is trained on a specific task using a large and specific dataset tailored for that task.

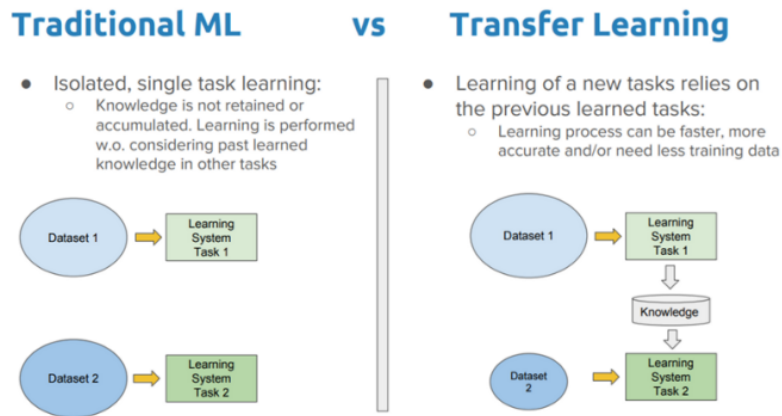


Figure 25: Difference between the traditional learning process of two different tasks and transfer learning. In transfer learning, task 1 is fully trained using a large dataset, and task 2 uses the knowledge obtained from task 1 to improve accuracy and learn faster [51].

- VGG16** is a convolutional neural network model developed by K. Simonyan et al. [59] Over the years, VGG16 has maintained its reputation as an excellent vision model. The VGG16 Model has 16 Convolutional and Max Pooling layers, 3 Dense layers for the Fully-Connected layer, and an output layer of 1,000 nodes (Figure 26). This model was trained on the ImageNet dataset, specifically for the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) held in 2014. This dataset comprises millions of labeled images belonging to 1000 different classes. During the challenge, VGG16 demonstrated exceptional performance and emerged as one of the top-performing models. Its impressive results solidified its position as a leading architecture in the competition. It has been successfully applied to various deep neural network models for different tasks [25].
- ResNet50** is a deep neural network model, specifically belonging to the class of deep neural networks known as "Residual Neural Networks." It was developed by researchers at Microsoft Research in 2015. ResNet50 is characterized by its residual nature, which includes "residual units" that allow for deeper network training without the negative impact known as the vanishing gradient problem, a common issue in training deep networks. ResNet50 utilizes these units to skip data being sent to subsequent layers, enabling the network to learn more complex representations [9]. Thanks to

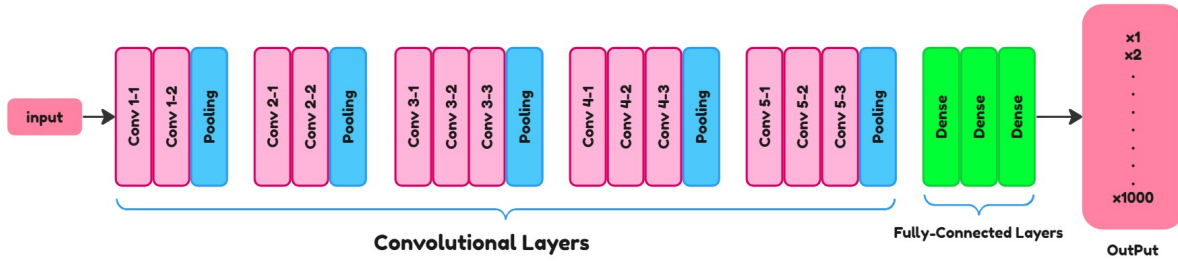


Figure 26: VGG16 Architecture.

its robust structure and ability to efficiently handle deep networks, ResNet50 has become one of the most popular and widely used models in deep learning. It is used in a variety of applications, including classification, object detection, image synthesis, machine translation, object recognition, and more. Here's the structure of ResNet50 with the number of layers as show in Figure 27:

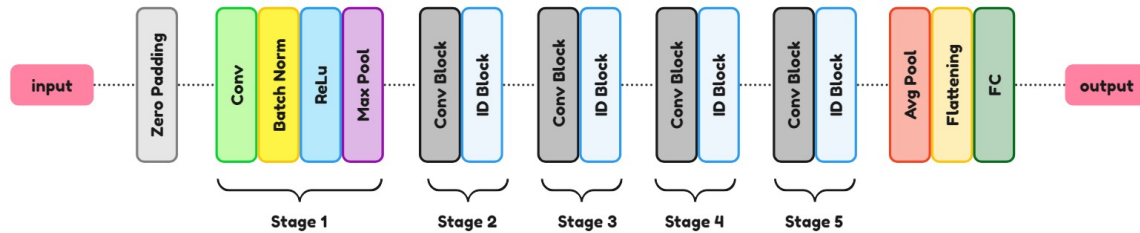


Figure 27: ResNet50 Architecture

2.4 Related work

The use of artificial intelligence (AI) tools for hand gesture analysis has been the subject of various studies. In this section, we will explore some earlier research that employed different deep-learning methods to recognize hand gestures.

2.4.1 Hand gesture recognition based on VGG16 and Alexnet

This research aims to develop a method for hand gesture recognition using transfer learning. The proposed method was utilized to make the system more robust and efficient by avoiding the use of skin color segmentation, bubble detection, skin area cropping, and extracting the skin center for unidirectional dynamic gestures. The model resulting from this research was tested on seven volunteers with different backgrounds and varying lighting conditions, achieving an accuracy of 93.09 %. This method seems to have shown success in achieving the intended goal of the research, which is to develop an effective system for hand gesture recognition based on computer vision on a sufficiently diverse and ample dataset containing images of hand gestures in various poses, backgrounds, and lighting conditions.

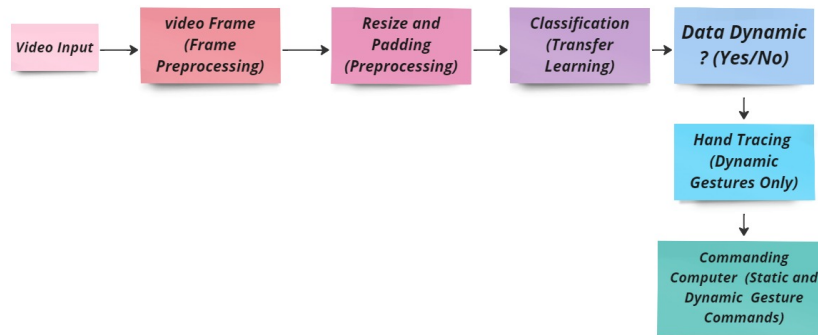


Figure 28: Proposed method for hand gesture recognition based on VGG16 and AlexNet.

2.4.2 Hand gesture Classification Model Using EMG Signals with CNN and Traditional Classification Methods

In this paper, a model for classifying hand gestures using electromyography (EMG) signals was developed, with a focus on utilizing CNN in addition to traditional methods in the classification process. The Ninapro DB2 database, a reliable source of EMG data related to hand gestures, was used. The paper included feature extraction from EMG signals and data segmentation using sliding window techniques. The performance of CNN-based classification models was compared with traditional methods such as SVM, and the models were evaluated using appropriate evaluation metrics. The results were discussed, highlighting the impact of data preprocessing on classification accuracy. This paper aims to provide new insights into hand gesture classification using EMG signals and to compare the effectiveness of traditional methods with CNN. The results show improvement in the performance of CNN-based models, indicating their potential effective use in classifying different hand gestures.

We will now present a comparison of our resume table (see table 3):

Approach	Paper	Feature Extraction	Type of dataset	Accuracy (%)
VGG16	S. Hussain et al. (2017) [21]	Not mentioned	11 gesture classes (8 Dynamic, 6 static)	93.09%
AlexNet				76.96%
CNN	K. Xing et al. (2018) [80]	sEMG	17 gesture classes	With preprocessing=82.16% Increase in accuracy with preprocessing=2.52% With preprocessing=82.16 Increase in accuracy with preprocessing=2.52% With preprocessing 83.23without preprocessing=77.52% Increase in accuracy with preprocessing=5.71%
		SVM		
		RF		

Table 3: Recently related work for Hand gesture recognition

2.4.3 Hand gesture recognition based on ensemble-based Convolutional Neural Network

The research focuses on developing an efficient gesture recognition system using ensemble-based CNN and transfer learning techniques. It stands out for its comprehensive attempt to improve hand gesture recognition techniques, employing a set of stages for image processing and hand region segmentation before inputting them into three parallel-trained neural network models. Additionally, the research explores the use of transfer learning to enhance the performance of the models used. The results demonstrate the effectiveness of the proposed system with recognition rates exceeding 99% in some cases, making it promising for real-time applications and gesture control systems in a variety of scenarios and practical applications. The results have been compiled into a summary table 4 for ease of reference and better understanding of the system’s performance.

Paper	Feature Ex- traction	Performance 1		Performance 2		Performance 3	
		dataset1	Accuracy (%)	dataset2	Accuracy (%)	dataset3	Accuracy (%)
A. Sen et al. (2022) [55]	VGG16-TL	10 ges- ture classes	99.65%	16 ges- tures classes	95.65%	14 ges- tures classes	99.30%
	AlexNet-TL		99.14%		94.00%		98.43%
	GoogLeNet-TL		99.60%		93.64%		97.60%
	VGGNet-like		99.65%		95.94%		99.53%
	AlexNet-like		99.36%		94.21%		98.53%
	GoogLeNet-like		99.50%		93.38%		97.20%

Table 4: Comparing CNN models for gesture recognition on multiple datasets.

2.5 Conclusion

In this chapter, we have discussed some of the fundamental aspects of our current work in hand gesture recognition, which include deep learning, and neural networks. Given their significant relevance to this field, Convolutional Neural Networks (CNNs) have been chosen as the primary focus of this chapter, particularly due to their connection to deep learning. Furthermore, we have provided a comprehensive overview of the specific issue we are addressing, namely hand gesture recognition, highlighting different techniques employed in this domain and relevant prior research.

The next chapter will explore the design of the system, the dataset used, its preparation, and potential models.

3 System Design

3.1 Introduction

A wide range of fields has witnessed significant advancements in modern technologies, such as image and signal processing, machine learning, and deep learning. These techniques have shown great effectiveness in supporting diagnostic, therapeutic, and research processes. This chapter highlights the design of the system used in our project, which focuses on hand gesture recognition using images as data, with the aim of utilizing these advanced techniques in classification and analysis. We will cover an overview of the methods used, detailed data description, discussions on data processing and feature extraction techniques, as well as analysis of the models used for classification purposes throughout this chapter.

3.2 Proposed Approach for Hand Gesture Detection

The proposed approach for recognizing hand gestures static or dynamic from images using CNN involves neural networks with multiple layers that extract patterns and distinctive features from the images (shown in Figure 29). The quality of the images used in this approach is crucial; they must be clear and high-resolution to ensure the system's ability to accurately recognize hand gestures.

The process of cleaning and enhancement is essential in this context, aiming to improve the image quality and remove noise and distortions, thus enhancing recognition accuracy and making the data more suitable for use.

The next step features are extracted from the processed images, and the role of these features is to represent the images in a simplified and distinctive manner, making it easier for the system to discern patterns and crucial details for gesture recognition.

Classification techniques are then used to classify the images and identify gestures based on the extracted features, which is a crucial part of gesture recognition.

Following that, the system is trained and tested using known datasets, helping evaluate its performance and improve its recognition capabilities.

All these processes contribute to enhancing the system's performance and increasing its accuracy, allowing it to predict gestures more accurately and effectively. The clinical value of this approach lies in its ability to be applied in various applications. Additionally, this approach can have significant benefits for individuals with auditory and visual impairments.

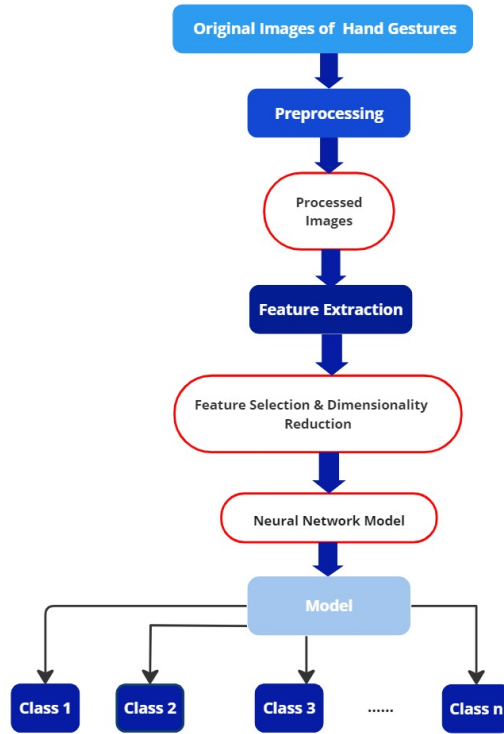


Figure 29: Visual representation of the methodology.

3.2.1 Preprocessing Phase

Image processing is considered one of the most essential steps to improve model performance. This stage consists of several phases, such as motion detection using background subtraction, setting a binary threshold for the detected hand region, segmenting it using perimeter-based region selection, and then resizing the images [56] then, techniques such as filtering, enhancement, segmentation, and classification can be used for image processing. In our study, a critical step in the image-preprocessing phase was implemented as an essential precursor to any subsequent image-processing steps. This initial step as depicted in Figure 30.

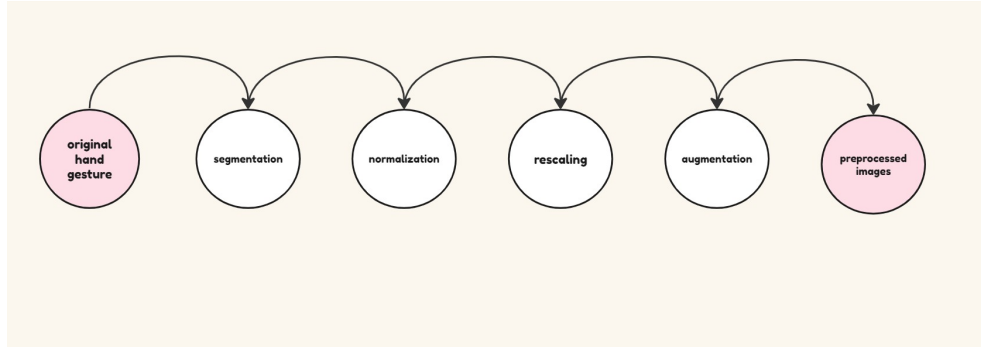


Figure 30: The preprocessing steps to be followed.

- **Segmentation** Image segmentation is a process used to extract the hand region from the background of a sign image. In the case of grayscale images in the database, the hand region is segmented from the enhanced image using Otsu’s method. This means that the algorithm determines the boundaries of the hand region in the image based on brightness and contrast analysis, and it uses a specific technique (Otsu’s method) to identify the boundaries automatically [45]. As for color images in database, the hand region is segmented from the background of the enhanced image using the skin color detection method in the YCbCr color space [8]. It is a fundamental step in analyzing hand gesture images and sign language.
- **Normalizatiom** Normalization is standardizing multiple data facets or sizes to make them consistent or similar within the context they are used, such as standardizing diverse responses that may be of different scales or effects, using techniques like Local Response Normalization (LRN) and L2 normalization. These techniques aim to achieve a uniform data distribution, reduce noise, and improve the model’s learning ability [73].
- **Rescaling** Rescaling, also known as resizing, refers to the process of adjusting the size of images. It involves shrinking images to fit the detection and prediction of hand gestures. This decision is made to reduce training time [44]. The hand gesture image is preprocessed to ensure compatibility with subsequent operations such as feature extraction and classification algorithms. Additionally, rescaling facilitates picture normalization and lowers the computational difficulty of the following analysis [79].

3.2.2 Feature extraction Phase

During the feature extraction stage for image input, the goal is to convert the visual content of the image into a compressed and meaningful representation in the form of a vector. This is achieved by capturing distinctive features and patterns present in the image that are relevant to the given task, such as object recognition or classification. Various techniques can be used for this purpose, such as Convolutional Neural Networks (CNNs) which rely on deep learning and are capable of automatically extracting features, or manually crafted feature descriptors that depend on expert knowledge and use specialized algorithms for feature extraction (show Figure 31).

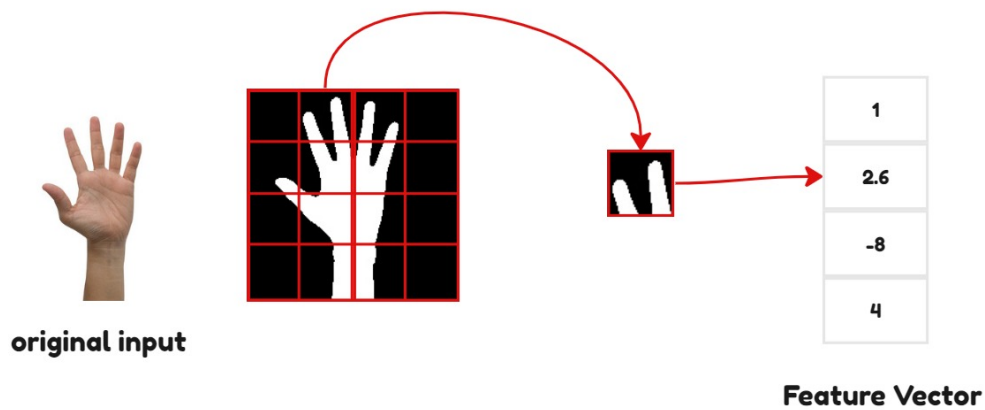


Figure 31: Feature extraction in local image image regions.

Convolutional Neural Networks (CNNs) have proven to be a powerful tool in feature extraction from images, as they excel in understanding the visual content of an image and converting it into a meaningful and compressed representation. These networks consist of multiple layers of interconnected neurons that learn from data to discover different levels of visual patterns, ranging from simple patterns like edges to complex patterns like large objects. The network processes the image through successive stages of analysis, including convolutional layers that identify important features in the image, pooling layers that reduce the size of processed data, and activation layers that activate discovered patterns. The final result of this process is a compressed vector representation commonly known as an embedding or feature vector, which effectively encodes important information about the image. In addition, feature extraction techniques can be used to recognize sign language [76] or hand gestures.

- **Hierarchical Classification-based Feature Extraction:** Hierarchical classification is a technique used in sign language recognition to break down images into smaller parts that represent different components of the hand gesture, such as fingers, the hand itself, and other geometric shapes. By extracting features from each of these subparts separately, this method helps create a comprehensive and effective representation of the hand gesture. These extracted features can include characteristics like angles, symmetry, and distances between fingers, all of which contribute to capturing the various geometric intricacies of the hand gesture.
- **Appearance Patterns-based Feature Extraction:** Appearance patterns play a crucial role in feature extraction for sign language recognition. This technique leverages information related to color, boundaries, and shapes within images to extract features that accurately depict the hand gesture. Features such as changes in color intensity, unique patterns within boundaries, and identifiable colored regions in images are extracted, ensuring a detailed and precise representation of the hand gesture's visual characteristics.
- **Neural Networks-based Feature Extraction:** Neural networks are employed in sign language recognition for their ability to extract intricate and distinctive features. This technique involves using complex layers within the neural network to extract features encompassing geometric patterns, color information, unique boundary patterns, and structural elements of the hand

gesture. By utilizing neural networks, this method comprehensively represents the hand gesture, capturing both its visual and structural aspects.

- **Geometric Transformations-based Feature Extraction:** Geometric transformations offer a powerful approach to feature extraction in sign language recognition. By applying transformations such as rotation and scaling, this technique extracts geometric features that are crucial for accurately representing the hand gesture. Features such as angles, symmetry, and variations in shapes and dimensions are extracted through these transformations, providing a detailed and precise depiction of the hand gesture’s geometric properties.

3.2.3 Proposed CNN Architectures

In this section, we will discuss the proposed geometries for classifying hand gestures or sign languages in the presented task. Various techniques are used for feature extraction and classification.

- **MIMO Techniques for Classifying Static Hand Gestures**

Advanced preprocessing techniques have been employed in the quest for more accurate static hand gesture recognition with mmWave radar technology. These techniques encompass converting from multistatic to monostatic signals, focusing on the intricate complexities of the radar’s complex-valued signals to refine classification accuracy. Range or range-angle analysis was applied to these complex data sets, preserving the signal’s real and imaginary components. This preservation enhanced the intrinsic relationships between peak level and phase, improving the classification rate significantly.

This comprehensive data preprocessing forms a critical component in the overarching process of recognizing static hand gestures and optimizing the performance of deep neural network models in classification tasks. The utilization of complex-valued MIMO signals underscores the radar’s robust capabilities in discerning subtle variations in hand reflectivity, leading to enhanced classification outcomes.

- **Feature Extraction and Classification for Static Hand Gestures:** The feature extraction process for static hand gestures using mmWave radar involves applying multistatic-to-monostatic conversion techniques to preprocess the radar data. This preprocessing step focuses on the complex-valued signal to enhance classification accuracy. The radar data undergo range

or range-angle analysis, retaining the signal’s real and imaginary values to strengthen the fundamental relationships between peak level and phase. This enhances the classification rate significantly. The chosen architecture includes convolutional layers with ReLU activation, fully connected layers and a softmax layer for classification. Specifically, for the Range-Angle FFT CNN, the input image size is $64 \times 16 \times 2$, with convolutional layers using kernels of size $13 \times 4 \times 2 \times 16$ and $13 \times 4 \times 16 \times 16$. On the other hand, the Range FFT CNN has an input image size of $64 \times 8 \times 2$, with convolutional layers using kernels of size $13 \times 2 \times 2 \times 16$ and $13 \times 2 \times 16 \times 16$. These architectures as show in Figure 32, coupled with the complex-valued signal processing, contribute significantly to the success of feature extraction and classification for static hand gestures using mmWave radar.

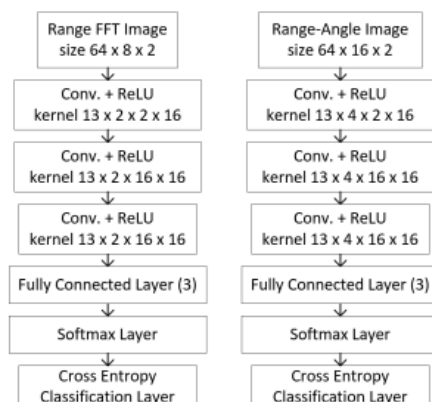


Figure 32: The network architecture of CNN [61].

3.2.4 proposed approach for hand gesture detection

The proposed approach aims to develop an efficient system for hand gesture detection using advanced computer vision and deep learning techniques. The proposed methodology encompasses a series of key steps, including preprocessing of input data, feature extraction using Convolutional Neural Network (CNN) models, and classification of hand gestures based on the extracted features. This introduction provides an overview of our approach and its significance in enhancing human-computer interaction through accurate and real-time hand gesture recognition.

- **Preprocessing Phase:** This phase involves several key steps to enhance the

quality of input data. It begins with motion-based background separation to isolate the hand from the background, followed by binary thresholding for precise segmentation of the hand region. Contour region selection is then utilized to outline the hand's shape accurately. After segmentation, the images undergo resizing and morphological operations to reduce noise and improve feature clarity, ensuring a clean input for the subsequent stages.

- **Feature Extraction using CNN Model:** In this stage, pre-trained CNN models such as VGG16, VGG19, and ResNet50 are employed for feature extraction. Thanks to their deep learning architecture, these models are capable of capturing intricate features from hand gesture images. Transfer learning techniques are applied to leverage the knowledge learned by these pre-trained models, allowing for efficient feature extraction with reduced training time and computational resources. The extracted features serve as the basis for the classification and recognition of hand gestures, enabling accurate and robust detection in real-time applications.

see in Figure 33

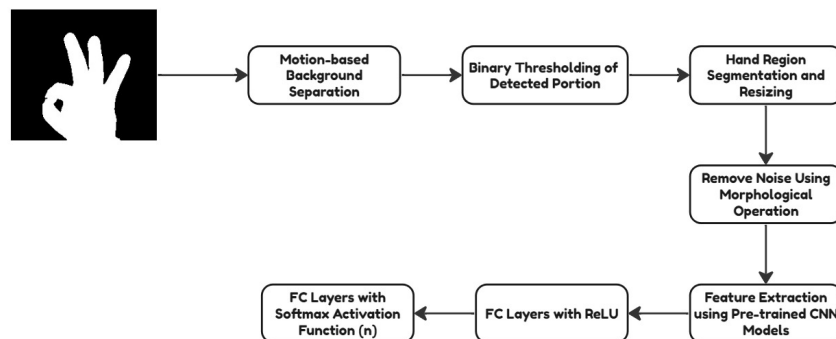


Figure 33: Feature Extraction with Pre-trained CNN Models Workflow.

3.3 Conclusion

In conclusion, Chapter 3 presents a comprehensive system design for hand gesture recognition using deep learning. The chapter addressed the problem of hand gesture classification, emphasizing the importance of data preprocessing, encompassing techniques such as filtering, normalization, and segmentation, to prepare the hand gesture image dataset for analysis. Additionally, the chapter highlighted the process of feature extraction, focusing on extracting relevant features from the preprocessed hand gesture images to enable effective classification.

choosing an appropriate deep learning model architecture was also discussed, emphasizing the key considerations involved in this selection process. Overall, this chapter provides a solid foundation for the subsequent chapters, which will delve into experimentation and results analysis in the field of hand gesture recognition.

4 Implementation and Results

4.1 Introduction

This research focuses on the field of hand gesture recognition using deep learning techniques. It presents a comprehensive model for recognizing various hand gestures, including signals and fine movements, through advanced deep neural networks. The emphasis is on different stages of the recognition process, including data presentation, feature extraction, and classification stages. A detailed evaluation of the proposed model's performance is provided using multiple metrics, contributing to a deep understanding of the depth and effectiveness of these techniques in accurately and professionally analyzing hand gestures.

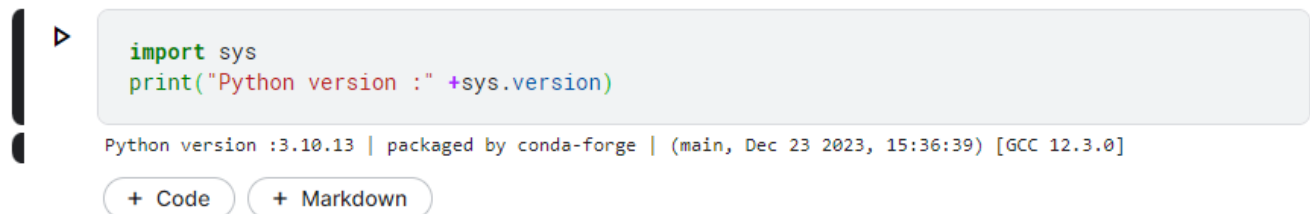
4.2 Implementation and Evaluation Tools

In this section on implementation and framework tools, we delve into the codes and tools utilized in this study and the different evaluation metrics employed. Our implementation relied on popular frameworks such as TensorFlow and PyTorch, which offered robust support for developing and training deep learning models. These frameworks enabled efficient implementation and facilitated the utilization of pre-trained models for effective feature extraction. Furthermore, we leveraged the capabilities of TensorFlow and PyTorch for effective model evaluation, including the use of established metrics such as accuracy, precision, and recall, providing us with an accurate assessment of the model's ability to recognize and classify hand gestures effectively.

4.2.1 Python language

The history of the Python language dates back to the late 1980s when it was developed by Guido Van Rossum in the Netherlands. Python was designed to be an easy-to-use and readable programming language, and it has evolved to become one of the most popular languages in programming due to its powerful features and active community support [72].

The Python language has experienced tremendous growth in popularity within the scientific computing community over the past decade and continues to do so in 2023, thanks to its high-level structure and easy readability, making it a favorite among users and beginners alike, the Python version is shown in Figure 34.



```
import sys
print("Python version :"+sys.version)
```

Python version :3.10.13 | packaged by conda-forge | (main, Dec 23 2023, 15:36:39) [GCC 12.3.0]

+ Code + Markdown

Figure 34: The Python version utilized on Kaggle.

4.2.2 Kaggle

Kaggle (Figure 35) is an online platform specifically designed for data science and machine learning. It provides a range of tools and resources for participants to learn about data, develop models, and participate in competitions to solve challenges and problems across various domains. This has made it one of the largest platforms for data science.

Kaggle also provides many services and features, such as competitions and challenges, where participants can analyze data and build predictive models to solve problems. Additionally, as a community and communication platform, Kaggle hosts an active community of data scientists and machine learning specialists who can interact with each other, share knowledge, and learn from others' experiences. Moreover, regarding educational resources, Kaggle offers a wide range of learning materials such as courses, articles, and academic notes that help users learn data science concepts and develop their skills. Lastly, as a large dataset repository, Kaggle provides access to a vast collection of datasets available for use in data analysis and model building.



Figure 35: Kaggle logo.

4.2.3 Keras

One of the most popular libraries used in developing neural networks is Keras (Figure 36), which is considered a high-level API that simplifies the construction of neural models straightforwardly and efficiently. Keras is known for its ease of use and learning, providing interfaces for several computational libraries such as TensorFlow, Theano, and CNTK. This makes it adaptable and able to leverage the latest technologies in artificial intelligence and deep learning.



Figure 36: keras logo.

4.2.4 OpenCv

OpenCV (Figure 37) (Open Source Computer Vision) is an open-source library for computer vision and machine learning. It started as a research project in 1999 and was developed by the University of Illinois and a team of developers. Later, Intel adopted it as an open-source project. OpenCV provides many tools and algorithms for image and video processing, including image analysis, object detection, image enhancement, video processing, and optical analysis. OpenCV is a powerful and well-known library in computer vision and machine learning, with extensive support for various platforms and languages.



Figure 37: OpenCV logo.

4.2.5 Tensorflow

TensorFlow (Figure 38) is an open-source and flexible machine learning platform used to build machine learning models. It was developed by researchers and engineers at Google Brain and offers a comprehensive environment of tools and libraries for building and deploying models. TensorFlow is known for its flexibility, scalability, and support for multiple programming languages such as Python and JavaScript. Thanks to its robust architecture and the large community supporting it, TensorFlow has become a leading platform in the field of machine learning and is used in a wide range of industries and applications.



Figure 38: TensorFlow logo.

4.2.6 Numpy

NumPy (Figure 39) is an open-source Python library used for scientific computing and data analysis. It provides support for arrays and multi-dimensional arrays, offering a wide range of high-level mathematical functions to work with these arrays. NumPy is a powerful and efficient tool in scientific research and data analysis, such as linear algebra, Fourier analysis, and statistics. It is widely used in a variety of applications that require numerical computations, scientific analysis, and applied mathematics, making it an essential tool for data scientists and researchers in various fields.



Figure 39: Numpy logo.

4.2.7 Matplotlib

Matplotlib (Figure 40), a library in Python, is designed for creating plots and visualizing data through graphs. Its object-oriented API facilitates seamless integration of graphics into applications, leveraging a range of graphical interface tools like Tkinter, wxPython, Qt, or GTK to enhance versatility and usability. Matplotlib can generate a diverse array of plots, from simple bar charts and line plots to intricate 3D visualizations and more.

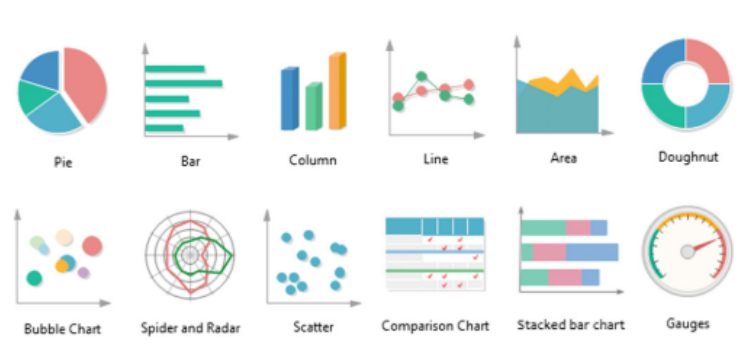


Figure 40: Visualization Data by Matplotlib [39].

4.3 Performance Evaluation Metrics in Classification Tasks

To ensure the reliability and dependability of our system, it is essential to evaluate it using a variety of metrics. By analyzing the system’s performance across different evaluation measures, we can comprehensively understand its strengths and weaknesses. We will explore various evaluation metrics, including accuracy, precision, recall, F1-score, and ROC. Each metric provides valuable insights into the system’s performance, enabling us to make informed decisions and progressively enhance efficiency.

- **TP:** True Positive refers to the cases where the model correctly predicts the positive class by identifying a positive instance as positive.
- **TN:** True Negative refers to the cases where the model correctly predicts the negative class by identifying a negative instance as negative.
- **FP:** False Positive refers to the cases where the model incorrectly predicts the positive class by identifying a negative instance as positive.
- **FN:** False Negative refers to the cases where the model incorrectly predicts the negative class by identifying a positive instance as negative.

4.3.1 Accuracy

The accuracy is defined as the ratio of correct answers to the total number of answers. In other words, Accuracy is used to measure the correctness or precision of a model or algorithm in predicting the correct results. Accuracy is typically expressed as a percentage.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \quad (5)$$

Equation 5 Accuracy equation.

4.3.2 Precision

The precision refers to the ratio of true positive results to the total positive results predicted by the model. In other words, Precision focuses on the model's accuracy in correctly identifying positives.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \quad (6)$$

Equation 6 Precision equation.

4.3.3 Recall

The recall expresses the ratio of true positive results correctly predicted among all actual positive results. In other words, Recall focuses on the model's ability to identify all positive cases correctly.

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

Equation 7 Recall equation.

4.3.4 F1-score

The F1 Score is a composite metric used to evaluate the performance of binary classification models. It combines Precision and Recall to determine how well the model correctly identifies positive classes. The F1 Score is an important metric because it considers both Precision and Recall.

$$\text{F1 Score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} = \frac{2 \times \text{TP}}{2 \times \text{TP} + \text{FP} + \text{FN}} \quad (8)$$

Equation 8 F1 Score equation.

4.3.5 Confusion Matrix

The confusion matrix is a table summarizing four categories of predictions made by the model: True Positives (TP), True Negatives (TN), False Positives (FP), and False Negatives (FN) as show in Figure 41. It is used to evaluate the model's performance in predicting binary outcomes.

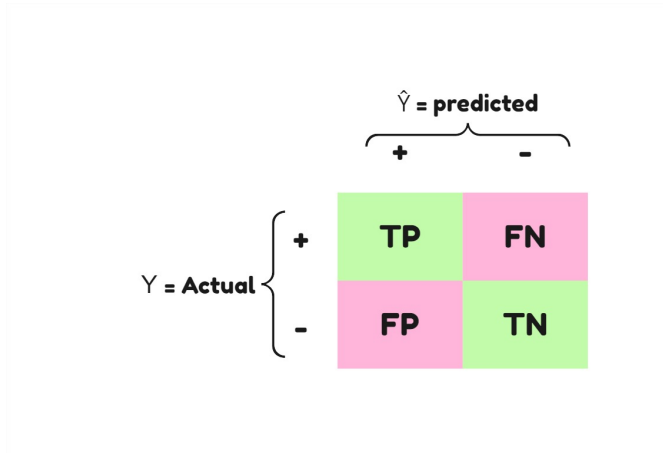


Figure 41: Confusion Matrix.

4.3.6 ROC

The ROC curve, which stands for Receiver Operating Characteristic Curve, is a tool for evaluating the performance of binary classification models. It illustrates the relationship between the True Positive Rate (Sensitivity) and the False Positive Rate.

4.4 Hand Gesture detection model implementation

4.4.1 Dataset description

The hand gesture recognition dataset was used as the data source in the study, containing six different categories obtained from the Kaggle platform. These categories were carefully chosen to represent a variety of hand movements, enabling the deep model to recognize a large and diverse set of motor actions. The data was meticulously processed and categorized to train the deep model, resulting in excellent performance in hand gesture recognition and a comprehensive understanding of motor signals with precision.

4.4.2 Data splitting

In our study, we used two distinct datasets obtained from Kaggle, each containing six categories (see Table 5).

The dataset was divided into two parts: 80% of the data was allocated for training purposes, while the remaining 20% was reserved for testing as shown

Number of Samples:	10321
Number of Categories:	6
Data Source:	The hand-gesture-recog-dataset from Kaggle database.
Classes:	['five','thumbsdown', 'ok', 'blank', 'fist', 'thumbsup']

Table 5: Hand Gesture Recognition Dataset.

in table 6. This division allowed for comprehensive model training on a significant portion of the data while also providing an independent dataset for robust evaluation.

Class	Train	Test	Total Number
five	1615	404	2019
thumbsdown	1312	329	1641
ok	1340	335	1675
blank	1293	323	1616
fist	1402	350	1752
thumbsup	1295	323	1618

Table 6: Number of samples per class in both training and testing sets.

4.4.3 Preprocessing phase

In this context, a complex-valued MIMO beat signal was utilized in radar data processing to extract essential information such as temporal, frequency, and spatial patterns. This approach enhanced data quality, improving classification accuracy within the system. The preprocessing stage encompasses several fundamental steps, starting with converting complex, multi-point radar signals into single-point signals for ease of subsequent analysis. Following this, analysis is conducted in the range or angle domain to extract vital information from the data, which is then utilized in analysis and classification stages. show in Figure 42

```

# Load and preprocess SAR dataset (complex-valued MIMO beat signals)
X_complex = [] # Complex-valued MIMO beat signals
y = [] # Class labels
gestures = ['blank', 'ok', 'thumbsup', 'thumbsdown', 'fist', 'five']
for i, gesture in enumerate(gestures):
    gesture_path = f"/kaggle/input/hand-gesture-recog-dataset/data/{gesture}/*.jpg"
    for img_path in glob.glob(gesture_path):
        img = cv2.imread(img_path, cv2.IMREAD_GRAYSCALE)
        img_complex = cv2.merge([img, np.zeros_like(img)]) # Convert to complex-valued representation
        img_complex = cv2.resize(img_complex, (100, 120))
        X_complex.append(img_complex)
        y.append(i)

```

Figure 42: complex-valued, MIMO beat signal for preprocessing.

So the main contribution of this study focuses on the discovery and recognition of hand gestures, where a preprocessed and partitioned dataset was used, specifically prepared for this particular purpose.



Figure 43: Example of an input preprocessed image.

These gestures were preprocessed and segmented, with each segment corresponding to a specific hand gesture as illustrated in Figure 43.

The first step in the process involved loading images from specific folders (identified in Figure 44), indicating that the set of images was ready for subsequent analysis.

```

for i, gesture in enumerate(gestures):
    gesture_path = f"/kaggle/input/hand-gesture-recog-dataset/data/{gesture}/*.jpg"
    for img_path in glob.glob(gesture_path):
        img = cv2.imread(img_path, cv2.IMREAD_GRAYSCALE)
        img_complex = cv2.merge([img, np.zeros_like(img)]) # Convert to complex-valued representation
        img_complex = cv2.resize(img_complex, (100, 120))
        X_complex.append(img_complex)
        y.append(i)

```

Figure 44: Loading Data.

4.4.4 Feature extraction phase and Model Training

To extract the most relevant features, Synthetic Aperture Radar (SAR) technology was employed as a technique, enabling the extraction of crucial information. These extracted features were then utilized as inputs for a pre-trained CNN (Convolutional Neural Network) model. The process involved several steps. After loading, splitting, and converting the data, a SAR technology function was defined to extract important features from the images. Then, the data size was reduced to facilitate learning. Synthetic Aperture Radar (SAR) technology extracts crucial features from images, creating high-resolution images that reflect the physical characteristics. Subsequently, image analysis and processing operations were applied to reduce data size and enhance learning effectiveness (see Figure 45).

```
# Define CNN model for SAR image classification
model = Sequential([
    Conv2D(32, (3, 3), activation='relu', input_shape=(120, 100, 2)),
    MaxPooling2D((2, 2)),
    Conv2D(64, (3, 3), activation='relu'),
    MaxPooling2D((2, 2)),
    Flatten(),
    Dense(128, activation='relu'),
    Dropout(0.5),
    Dense(len(gestures), activation='softmax')
])
```

Figure 45: Define CNN Model for SAR Image Classification.

In addition, we created CNN architecture (see Figure 46) to illustrate the aforementioned.

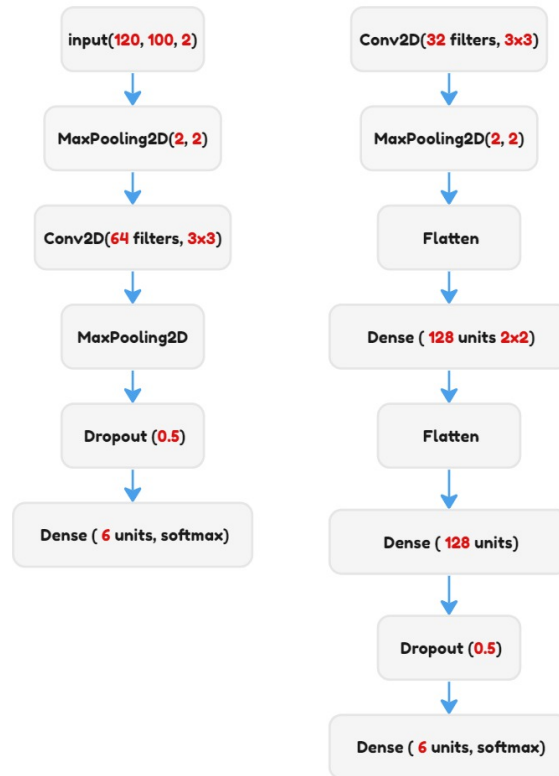


Figure 46: Defining the CNN model architecture.

The EarlyStopping callback (Figure 47) was implemented to prevent overfitting and improve the model’s generalization. This callback monitored the model’s performance during training and halted the training process if no improvement was observed. Additionally, the number of training epochs was increased to 100, allowing the model more opportunities to learn and converge towards optimal solutions.

```

# Define early stopping callback
early_stop = EarlyStopping(monitor='val_loss', patience=3, verbose=1, restore_best_weights=True)

```

Figure 47: Early stopping callback.

The study investigated the effectiveness of SAR technology in extracting important features from images, proposed neural network models, learning rates, and early

stopping techniques to achieve better results.

4.4.5 Experiments and results

In this section, we present the experiments and results of our study. For our experiment, we utilized the type of CNN used in your code is a standard feedforward CNN, also known as a convolutional neural network (CNN) and re-trained it for 100 epoch. We evaluated the model's performance using metrics such as training accuracy, precision, recall, and F1 Score.

The model information was analyzed through a training process conducted in batches, where the model is updated based on the data provided in each batch. The output of the training process shows details such as model accuracy, representing its ability to predict correct results, and loss, measuring the deviation between the model's predictions and the actual values.

Accuracy and loss results are typically displayed for each step of the training process, along with clarification of the time required for each step's execution. Relevant warnings related to the model's performance during training may also be shown and reviewed to ensure they don't impact the final model's quality.

From the presented results (see Figure 48), it can be inferred that the model was trained for several epochs, with accuracy and loss evolving with each epoch. An early stopping strategy was also employed when the results improved marginally, aiming to avoid issues like performance loss or overfitting that can occur with continued training.

```

Epoch 7/100
26/26 ----- 2s 66ms/step - accuracy: 0.9995 - loss: 0.0018 - val_accuracy: 1.000
0 - val_loss: 2.6517e-04
Epoch 8/100
26/26 ----- 2s 66ms/step - accuracy: 0.9996 - loss: 0.0014 - val_accuracy: 0.999
4 - val_loss: 0.0011
Epoch 9/100
26/26 ----- 2s 66ms/step - accuracy: 1.0000 - loss: 0.0016 - val_accuracy: 1.000
0 - val_loss: 1.0460e-04
Epoch 10/100
26/26 ----- 2s 65ms/step - accuracy: 0.9999 - loss: 7.0207e-04 - val_accuracy:
1.0000 - val_loss: 2.7756e-04
Epoch 11/100
26/26 ----- 2s 66ms/step - accuracy: 0.9997 - loss: 0.0010 - val_accuracy: 1.000
0 - val_loss: 2.7124e-04
Epoch 12/100
26/26 ----- 2s 67ms/step - accuracy: 0.9999 - loss: 8.0240e-04 - val_accuracy:
1.0000 - val_loss: 4.1647e-05
Epoch 13/100
26/26 ----- 2s 67ms/step - accuracy: 0.9999 - loss: 9.0128e-04 - val_accuracy:
1.0000 - val_loss: 2.5115e-04
Epoch 14/100
26/26 ----- 2s 66ms/step - accuracy: 0.9999 - loss: 6.2848e-04 - val_accuracy:
1.0000 - val_loss: 1.6727e-04
Epoch 15/100
26/26 ----- 2s 66ms/step - accuracy: 0.9998 - loss: 8.2561e-04 - val_accuracy:
1.0000 - val_loss: 1.6575e-04
Epoch 15: early stopping
Restoring model weights from the end of the best epoch: 12.

```

Figure 48: Analyzing Model Performance and Optimization Strategies.

The best model was saved at epoch 12, while the earliest stopping occurred at epoch 12, as shown in Figure 48.

Figure 49 the results of the confusion matrix of our proposed CNN architecture: According to this confusion matrix, our model has achieved perfect predic-

tions for all five classes, with zero false predictions. This is an exceptional result, indicating that our model is performing flawlessly in classifying the given data. It demonstrates high accuracy and reliability, providing accurate predictions for each class without any errors. This outstanding achievement suggests that our model is well-trained and highly capable of handling the given task.

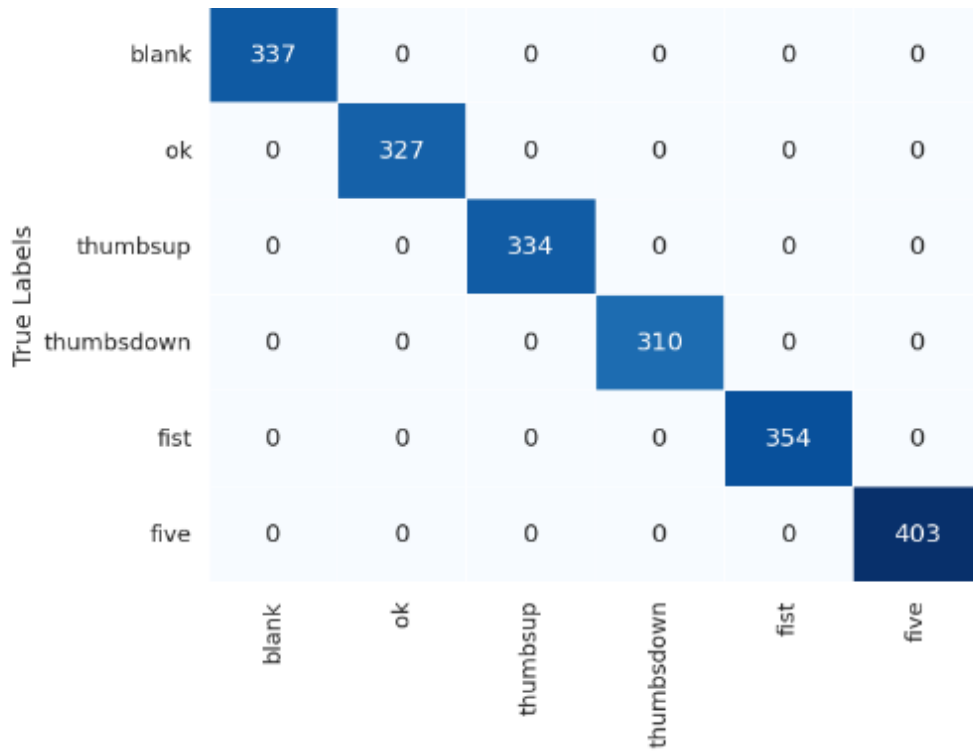


Figure 49: Confusion Matrix for the model.

The Receiver Operating Characteristic (ROC) curve and Area Under the Curve (AUC) are crucial evaluation tools in assessing the performance of classification models. The ROC curve illustrates the trade-off between the model's ability to correctly identify true positives (TPR) and its tolerance for false positives (FPR) across various decision thresholds. The ROC curve expands towards the upper-right corner when the model can detect all true positives without any false positives, resulting in an AUC of 1.00, indicating optimal classification performance. This optimal area validates the model's effectiveness in accurately distinguishing between classes.

Figure 50 depicts the ROC curve of our CNN model.

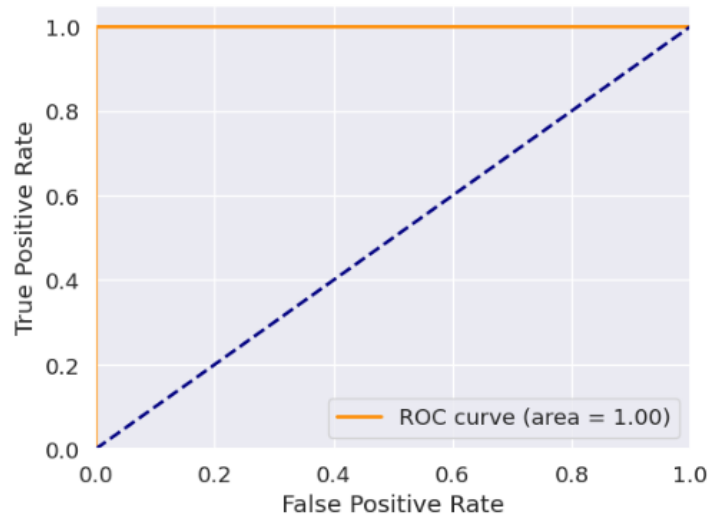


Figure 50: Receiver Operating Characteristic.

4.5 Creating a Model for Detecting Hand Gestures

4.5.1 Preprocessing phase

In this part, complex-valued images representing SAR (Synthetic Aperture Radar) signals are converted into RGB representation. This is achieved by creating a new array with specific dimensions to represent the images in RGB format, where the data extracted from the grayscale channel and the complex part of the images are allocated to the red and green color channels, while zero values are added to the blue color channel. This conversion is necessary because neural networks expect images in RGB format, so complex-valued images need to be transformed into this required representation before being inputted into the model for training and testing (see Figure 51).

```
# Convert complex-valued images to RGB
X_train_rgb = np.zeros((len(X_train), 120, 100, 3), dtype=np.float32)
X_train_rgb[:, :, :, 0] = X_train[:, :, :, 0] # Red channel
X_train_rgb[:, :, :, 1] = X_train[:, :, :, 1] # Green channel
X_train_rgb[:, :, :, 2] = np.zeros_like(X_train[:, :, :, 0]) # Blue channel

X_test_rgb = np.zeros((len(X_test), 120, 100, 3), dtype=np.float32)
X_test_rgb[:, :, :, 0] = X_test[:, :, :, 0] # Red channel
X_test_rgb[:, :, :, 1] = X_test[:, :, :, 1] # Green channel
X_test_rgb[:, :, :, 2] = np.zeros_like(X_test[:, :, :, 0]) # Blue channel
```

Figure 51: Complex Image Conversion to RGB Format.

Data augmentation techniques are vital, as our dataset is small. It is necessary to apply data augmentation techniques to effectively increase the size of our hand gesture dataset artificially. These techniques can include flipping, rotation, zooming, cropping, and adding noise to the gesture. Flipping and rotation can be used to simulate different orientations of hand gestures while zooming and cropping can simulate different levels. This process is crucial for enhancing the robustness and generalization abilities of our hand gesture recognition models (see Figure 52).

```

# Data Augmentation
datagen = ImageDataGenerator(
    rotation_range=20,
    width_shift_range=0.1,
    height_shift_range=0.1,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True,
    vertical_flip=True,
    fill_mode='nearest'
)
datagen.fit(X_train_rgb)

```

Figure 52: Data Initialization Using Data Augmentation Techniques.

4.5.2 Feature extraction phase

For feature extraction, we used the deep learning model VGG16 as shown in Figure 53, which is pre-trained on the ImageNet dataset. This model can highly extract features from images due to its deep architecture composed of numerous convolutional layers. In Figure, we instantiate VGG16 model where the layers are frozen to prevent weight modification during the training process. A new sequential model is then created, containing a flattening layer to convert the outputs into a flat matrix, followed by a dense layer with ReLU activation and a dropout layer to prevent overfitting. Finally, another dense layer with softmax activation is added to classify hand gestures into the defined categories.

```

# Pre-trained Model (VGG16) for Feature Extraction
base_model = VGG16(weights='imagenet', include_top=False, input_shape=(120, 100, 3))

# Freeze the base model layers
for layer in base_model.layers:
    layer.trainable = False

model = Sequential([
    base_model,
    Flatten(),
    Dense(128, activation='relu'),
    Dropout(0.5),
    Dense(len(gestures), activation='softmax')
])

```

Figure 53: Proposed features extractor function using VGG16.

4.5.3 Model Learning

During the model training process using early stopping and data augmentation techniques, as shown in Figure, a significant improvement in the model's performance on the test data was observed. Initially, there were no specific values for the model's accuracy or loss during the training epochs, while the model's accuracy on the test data was relatively high with low loss. As the epochs progressed, the model's accuracy on the training data improved significantly, and appropriate accuracy and loss values appeared on the test data, indicating the model's ability to learn and adapt to the data. In epoch number 18, early stopping was utilized due to the model's lack of performance improvement on the test data for a certain period, and the best model weights that achieved better performance earlier were restored, confirming the effectiveness of using early stopping to avoid overfitting and enhance the model's performance on new data.

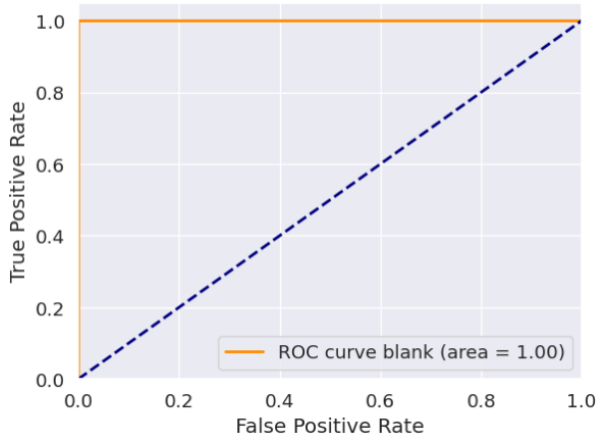
```
# Train the model with early stopping and data augmentation
history = model.fit(datagen.flow(X_train_rgb, y_train, batch_size=32), epochs=100,
                    steps_per_epoch=len(X_train) // 32, verbose=1,
                    validation_data=(X_test_rgb, y_test), callbacks=[early_stop])
```

Figure 54: Improving Model Performance with Early Stopping and Data Augmentation.

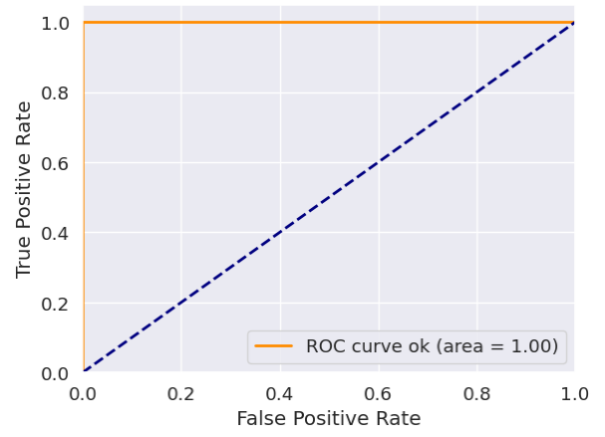
4.5.4 Experiments and results

We utilized a pre-trained VGG16 model for feature extraction in our project, setting it up to process our data with an input shape of (120, 100, 3). This approach allowed us to benefit from the feature extraction capabilities of VGG16 while customizing the classification layers for our specific task.

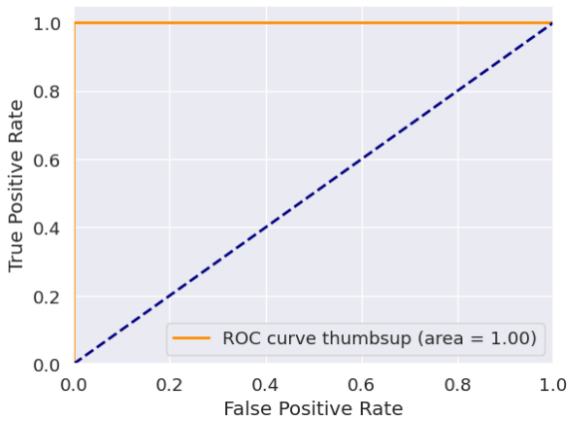
Figure 55 depicts the ROC curve of our VGG16 Model for each gesture category in a classification task.



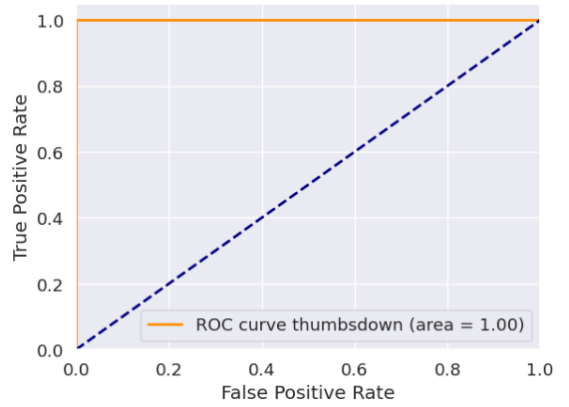
(a) ROC Curve for Blank class.



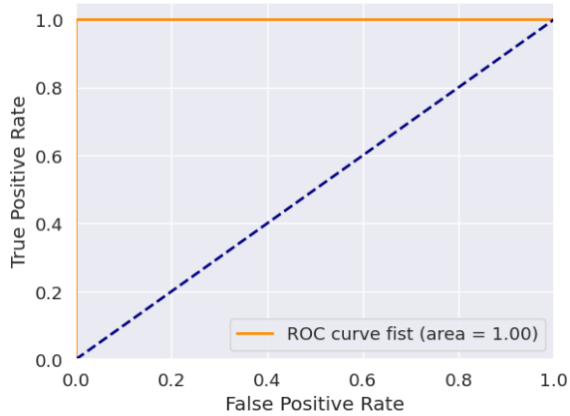
(b) ROC Curve for Ok class.



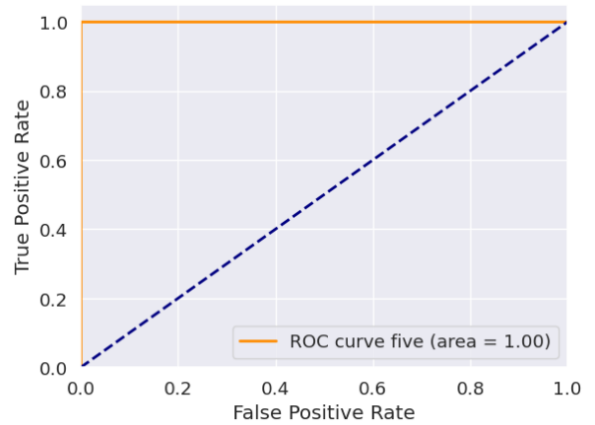
(c) ROC Curve for Thumbs Up class.



(d) ROC Curve for Thumbs Down class.



(e) ROC Curve for Fist class.



(f) ROC Curve for Five class.

Figure 55: ROC Curves for all classes.

According to our confusion matrix (Figure 56), our model achieved flawless predictions for all six categories. This very good result indicates the perfect performance of our model inefficiently categorizing the provided data. It reflects a significant level of accuracy and reliability, delivering accurate predictions for every category except one. This notable achievement highlights the efficiency and competence of our well-trained model in successfully handling the assigned task.

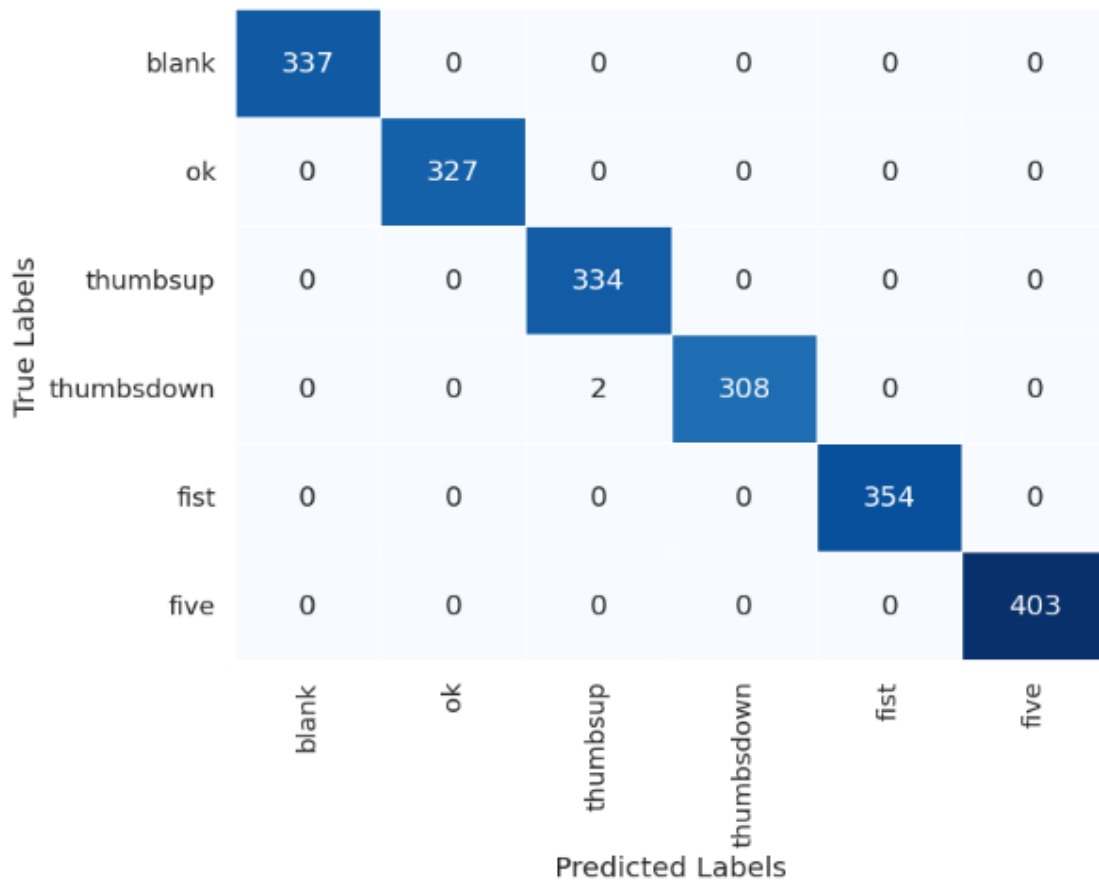


Figure 56: Confusion Matrix.

4.6 System deployment

4.6.1 Predicting Image Classes from a File Path

The predict image function (see Figure 57) loads an image from a specified file path and processes it by converting it to a grayscale image and then to a complex-valued representation. The image is resized and converted to an RGB format, with values normalized. The processed image is then passed through the trained

model to predict the class. Finally, the function returns the predicted class from a predefined list of gestures. This method is useful for predicting classes of images stored in local files.

```
def predict_image(model, img_path):
    # Load and preprocess the image
    img = cv2.imread(img_path, cv2.IMREAD_GRAYSCALE)
    img_complex = cv2.merge([img, np.zeros_like(img)]) # Convert to complex-valued representation
    img_complex = cv2.resize(img_complex, (100, 120))
    img_rgb = np.zeros((120, 100, 3), dtype=np.float32)
    img_rgb[:, :, 0] = img_complex[:, :, 0] # Red channel
    img_rgb[:, :, 1] = img_complex[:, :, 1] # Green channel
    img_rgb[:, :, 2] = np.zeros_like(img_complex[:, :, 0]) # Blue channel
    img_rgb = img_rgb.astype('float32') / 255.0
    img_rgb = np.expand_dims(img_rgb, axis=0) # Add batch dimension

    # Predict the class
    y_pred = model.predict(img_rgb)
    y_pred_class = np.argmax(y_pred, axis=1)[0]
    gestures = ['blank', 'ok', 'thumbsup', 'thumbsdown', 'fist', 'five']

    # Return the predicted class
    return gestures[y_pred_class]
```

Figure 57: Predicting image class from any link.

4.6.2 Displaying and Predicting Image Classes from a Test Set

The predict and show image function (see Figure 58) selects an image from a test set using a specified index, expands its dimensions to match the input shape required by the model, and predicts its class using the trained model. The function then displays the image along with the predicted class as a title and returns the predicted class from a predefined list of gestures. This method is ideal for visualizing and predicting classes of images already present in a test dataset as show in Figure 59.

```

def predict_and_show_image(model, X_test_rgb, index):
    # Select the image from the test set
    img_rgb = X_test_rgb[index]

    # Expand dimensions to match the input shape of the model
    img_rgb_expanded = np.expand_dims(img_rgb, axis=0)

    # Predict the class
    y_pred = model.predict(img_rgb_expanded)
    y_pred_class = np.argmax(y_pred, axis=1)[0]
    gestures = ['blank', 'ok', 'thumbsup', 'thumbsdown', 'fist', 'five']

    # Show the image
    plt.imshow(img_rgb, interpolation='nearest')
    plt.title(f"Predicted class: {gestures[y_pred_class]}")
    plt.axis('off') # Hide axes
    plt.show()

    # Return the predicted class
    return gestures[y_pred_class]

```

Figure 58: Predicting image class using test set index and displaying full image.

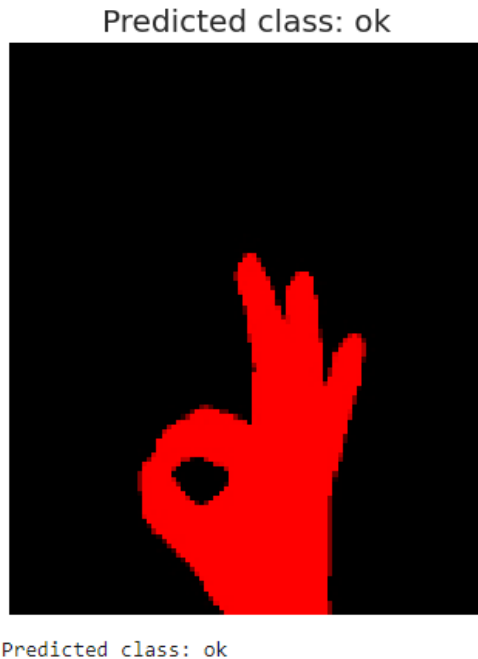


Figure 59: Predict of gesture ok.

4.6.3 Visualizing Image Predictions from Test Set Using Indexing

The function illustrates a called predict and show image (see Figure 60) that predicts the class of an image from the given test set and displays the image along with its predicted class as show in Figure 61. This function utilizes Matplotlib's capabilities for image display and employs Python functions to perform the prediction task. The process begins by selecting the desired image from the test set using an index, expanding its dimensions to match the input format expected by the model, and predicting its class using the trained model. The predicted class is then displayed alongside the image, visually representing the classification result. This function is crucial for examining and verifying the accuracy of image classification models in practical applications. A value was used to specify the image index within the test set.

```
def predict_and_show_image(model, X_test, y_test, index):
    # Select the image from the test set
    img_rgb = X_test[index]

    # Expand dimensions to match the input shape of the model
    img_rgb_expanded = np.expand_dims(img_rgb, axis=0)

    # Predict the class
    y_pred = model.predict(img_rgb_expanded)
    y_pred_class = np.argmax(y_pred, axis=1)[0]
    gestures = ['blank', 'ok', 'thumbsup', 'thumbsdown', 'fist', 'five']

    # Show the image
    plt.imshow(img_rgb[:, :, 0], cmap='gray', interpolation='nearest')
    plt.title(f"Predicted class: {gestures[y_pred_class]}")
    plt.axis('off') # Hide axes
    plt.show()

    # Return the predicted class
    return gestures[y_pred_class]
```

Figure 60: Index-based Image Prediction and Display in Test Set.



Figure 61: predict of gesture fist.

4.7 Comparison and discussion

Our analysis focuses on several results obtained from our experiments and reported results in related studies. This thorough comparison and evaluation aim to shed light on the performance of different approaches, offering insights into their strengths and limitations.

4.7.1 Model Comparison: Performance and Effectiveness

Through the results between the different trained models, the values of Precision, Recall, F1 score, and Accuracy were found to be 100%, 100% , 100% , and 100% respectively for the CNN model, and for the VGG16 model, they were found to be 99.9%, 100%, 100%, and 100%.

These results show that both our VGG16 and CNN classifiers achieved high accuracy of 100%.

Given the overall accuracy of the model, which reached 100%, it can be said that the model can accurately predict all samples in the test set correctly. The precision, recall, and F1 scores of nearly 100% for each category further validate this

result, indicating that the model cannot accurately distinguish between different gestures. The macro average and weighted average, both reaching 100%, enhance the model’s strength in balanced classification across all categories, regardless of the number of samples in each category.

4.7.2 Proposed Model vs. Related Work

We will now compare our proposed CNN architecture with the related work. The results of this comparison are illustrated in the table 7.

	Model/paper	Precision (%)	Recall (%)	Accuracy (%)	F1 Score (%)	ROC (%)
Our proposed	CNN	1.0000	1.0000	1.0000	1.0000	1.0000
Transfer Learning	VGG16	1.0000	1.0000	0.9990	1.0000	1.0000
Related Work	S. Hussain et al. (2017)[21]	-	-	0.9309	-	-
	N. Jomhari et al. (2021)[42]	-	-	0.9800	-	-
	Y. Fang et al. (2007)[14]	-	-	0.938	-	-
	A. Gunduz et al. (2019)[33]	-	-	0.9404	-	-
	Z. Ren et al. (2013)[49]	-	-	0.9320	-	-
	Y. Lu et al. (2020)[38]	-	-	0.9840	-	-

Table 7: Comparing our proposed model with related work.

According to the table 7, our proposed CNN architecture has achieved outstanding performance with perfect precision, recall, accuracy, and an F1 score value of 1.0000. In comparison, the transfer learning model VGG16 achieved an accuracy of 0.9990. While some of these models performed well, none surpassed the accuracy our proposed CNN architecture achieved.

4.8 Conclusion

In this chapter, we covered the frameworks, tools, and evaluation metrics employed during the implementation phase of our project. We focused on two primary contributions, each analyzed in detail, encompassing various stages such as data description, preprocessing, classification, and model evaluation.

During this chapter, we examined various frameworks that provided the necessary infrastructure for building and training our models. We used powerful tools such as TensorFlow and its ecosystem, OpenCV, and others. These frameworks and tools provided a solid foundation for our implementation, enabling us to develop advanced machine learning models.

Given the development of advanced learning models, data played a prominent role in our project, as we dedicated a significant part of this chapter to describing the dataset and its features. We discussed the initial preprocessing steps necessary, normalizing, and transforming data into a suitable format for training and evaluation. Techniques such as resizing images, normalizing data, and augmenting data were used to improve the quality and diversity of our dataset.

Classification was a key task in our project, and we delved into various algorithms and models suitable for our problem domain. Instead of using traditional convolutional neural networks (CNNs), we utilized the VGG16 model for feature extraction from images. We discussed the architecture of this model, the training process, and hyperparameter tuning to ensure optimal performance and high accuracy.

Model evaluation was a fundamental aspect of our project implementation, where we utilized a variety of evaluation metrics to measure our models' performance. These metrics included accuracy, precision, recall, F1-score, and ROC-AUC, allowing us to assess different aspects of performance. By relying on these multiple metrics, we gained a comprehensive understanding of the models' effectiveness and accurately compared their performance.

In conclusion, this chapter provided a comprehensive exploration of the frameworks, tools, and evaluation metrics utilized during our project's implementation phase. By covering data description, preprocessing, classification, model evaluation, and system deployment, we ensured a thorough understanding of the entire implementation process. This knowledge forms a robust foundation for subsequent chapters, where we analyze results, draw conclusions, and make recommendations based on our findings.

General conclusion

This study presents a comprehensive system for detecting and classifying hand gestures using artificial intelligence and deep learning techniques. The system can recognize various hand gestures accurately, providing opportunities for effectively analyzing and interpreting hand movements.

This research leverages artificial intelligence and deep learning techniques to introduce a comprehensive system for accurately recognizing and classifying hand gestures. This system reduces human errors and enhances efficiency in analysis and interpretation processes, opening the door to innovative and effective use of hand gestures, thus significantly improving outcomes and interventions.

Continuing ahead, future research can concentrate on enhancing and broadening the system's capabilities regarding hand gesture recognition. This could entail delving into real-time monitoring and integrating data acquisition for ongoing assessment of hand gestures. By pursuing these developments, this study contributes to the progression of hand gesture recognition technology, empowering users to make informed decisions and provide tailored assistance. Ultimately, these endeavors aim to enhance user experiences and interaction efficiency in various applications, particularly in fields like human-computer interaction, virtual reality, and sign language interpretation.

One of the key advantages of this system is its ability to manage large volumes of hand gesture data efficiently. It can process substantial information quickly, leading to rapid analysis and decision-making. Additionally, the system's accuracy in identifying various types of hand gestures can assist in tailoring personalized interactions and responses.

The advancement of advanced algorithms in hand gesture recognition enhances their ability to interact with the environment more intelligently and accurately. These improvements contribute to improving user experience and providing more comprehensive and effective interactions. Thanks to progress in this field, hand gesture recognition can be used in various applications, including smart device control, virtual reality applications, and interactive robots. This opens up new avenues for smarter and easier human-machine communication, thus improving quality of life and user experience in various domains.

In summary, This topic delves into how integrating advanced machine learning techniques, such as deep learning algorithms, in hand gesture recognition systems can revolutionize how humans interact with machines. Similar to the paragraph you provided, it emphasizes the potential of these systems to detect and categorize

hand gestures accurately, enabling personalized interactions and driving advancements in various fields, including human-computer interaction, virtual reality, and assistive technologies. The continuous improvement and expansion of these systems through ongoing research efforts are expected to enhance user experiences and applications in hand gesture recognition, ultimately leading to more intuitive and efficient human-machine interactions.

References

- [1] Osama Abdeljaber et al. “1-D CNNs for structural damage detection: Verification on a structural health monitoring benchmark data”. In: *Neurocomputing* 275 (2018), pp. 1308–1317.
- [2] S Ahmed et al. *Hand Gestures Recognition Using Radar Sensors for Human-Computer-Interaction: A Review. Remote Sens.* 2021, 13, 527. 2021.
- [3] Anatomy.app. *Metacarpal bones*. Accessed: 2024-05-29. 2024. URL: <https://anatomy.app/encyclopedia/metacarpal-bones>.
- [4] SH Shabbeer Basha et al. “Impact of fully connected layers on performance of convolutional neural networks for image classification”. In: *Neurocomputing* 378 (2020), pp. 112–119.
- [5] Pia Breuer, Christian Eckes, and Stefan Müller. “Hand gesture recognition with a novel IR time-of-flight range camera—a pilot study”. In: *Computer Vision/Computer Graphics Collaboration Techniques: Third International Conference, MIRAGE 2007, Rocquencourt, France, March 28-30, 2007. Proceedings 3*. Springer. 2007, pp. 247–260.
- [6] Anthony Lewis Brooks and Eva Brooks. “Game-based (re) habilitation via movement tracking”. In: *Recent Advances in Technologies for Inclusive Well-Being: Virtual Patients, Gamification and Simulation* (2021), pp. 253–274.
- [7] Bossard Bruno. “Problèmes posés par la reconnaissance de gestes en Langue des Signes”. In: *Rencontre des étudiants chercheurs en informatique pour le traitement automatique des langues* (2002), p. 67.
- [8] Douglas Chai and Abdesselam Bouzerdoum. “A Bayesian approach to skin color classification in YCbCr color space”. In: *2000 TENCON proceedings. Intelligent systems and technologies for the new millennium (Cat. No. 00CH37119)*. Vol. 2. IEEE. 2000, pp. 421–424.
- [9] Ahmet Çınar, Muhammed Yıldırım, and Yeşim Eroğlu. “Classification of pneumonia cell images using improved ResNet50 model”. In: *Traitement du Signal* 38.1 (2021), pp. 165–173.
- [10] Wikipedia contributors. *Langue des signes française*. Accessed: 2024-06-05. 2024. URL: https://fr.wikipedia.org/wiki/Langue_des_signes_fran%C3%A7aise.

- [11] Djamila Dahmani. “Elaboration d’un système de reconnaissance de l’epellation digitale de la langue des signes”. PhD thesis. Faculté d’Electronique et d’Informatique, 2014.
- [12] Mathieu Domalain, Laurent Vigouroux, and Éric Berton. “Modélisation biomécanique de la main: influence des caractéristiques de l’objet sur la distribution des tensions des tendons lors d’une tâche de préhension: (Prix Jean-Claude Lyleire 2007)”. In: *Staps* 3 (2008), pp. 7–22.
- [13] Larisa Dunai, Martin Novak, and Carmen García Espert. “Human hand anatomy-based prosthetic hand”. In: *Sensors* 21.1 (2020), p. 137.
- [14] Yikai Fang et al. “A real-time hand gesture recognition method”. In: *2007 IEEE International Conference on Multimedia and Expo*. IEEE. 2007, pp. 995–998.
- [15] RAKOTOARISOA Jean Ferdinand. “RECONNAISSANCE DE GESTE DE LA MAIN BASEE SUR LA VISION PAR ORDINATEUR”. In: ().
- [16] William T Freeman and Craig D Weissman. “Television control by hand gestures”. In: *Proc. of Intl. Workshop on Automatic Face and Gesture Recognition*. 1995, pp. 179–183.
- [17] William T Freeman et al. “Computer vision for computer games”. In: *Proceedings of the second international conference on automatic face and gesture recognition*. IEEE. 1996, pp. 100–105.
- [18] Hossein Gholamalinezhad and Hossein Khosravi. “Pooling methods in deep neural networks, a review”. In: *arXiv preprint arXiv:2009.07485* (2020).
- [19] S Michael Goza et al. “Telepresence control of the NASA/DARPA robonaut on a mobility platform”. In: *Proceedings of the SIGCHI conference on Human factors in computing systems*. 2004, pp. 623–629.
- [20] N Hemavathy and P Indumathi. “Deep learning-based hybrid dynamic bi-ased track (DL-HDBT) routing for under water acoustic sensor networks”. In: *Journal of Ambient Intelligence and Humanized Computing* 12.1 (2021), pp. 1211–1225.
- [21] Soeb Hussain et al. “Hand gesture recognition using deep learning”. In: *2017 International SoC design conference (ISOCC)*. IEEE. 2017, pp. 48–49.
- [22] A Huysseune. “Skeletal system”. In: *The laboratory fish*. Elsevier, 2000, pp. 307–317.

- [23] Vladimir I Iglovikov et al. “Paediatric bone age assessment using deep convolutional neural networks”. In: *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support: 4th International Workshop, DLMIA 2018, and 8th International Workshop, ML-CDS 2018, Held in Conjunction with MICCAI 2018, Granada, Spain, September 20, 2018, Proceedings 4*. Springer. 2018, pp. 300–308.
- [24] Shuo Jiang et al. “Emerging wearable interfaces and algorithms for hand gesture recognition: A survey”. In: *IEEE Reviews in Biomedical Engineering* 15 (2021), pp. 85–102.
- [25] Tae Joon Jun et al. “ECG arrhythmia classification using a 2-D convolutional neural network”. In: *arXiv preprint arXiv:1804.06812* (2018).
- [26] Maria Karam et al. “A taxonomy of gestures in human computer interactions”. In: (2005).
- [27] Maria Karam and MC Schraefel. “A study on the use of semaphoric gestures to support secondary task interactions”. In: *CHI’05 Extended Abstracts on Human Factors in Computing Systems*. 2005, pp. 1961–1964.
- [28] JOHN MG KAUER. “Functional anatomy of the wrist.” In: *Clinical Orthopaedics and Related Research (1976-2007)* 149 (1980), pp. 9–20.
- [29] Nikhil Ketkar et al. “Convolutional neural networks”. In: *Deep Learning with Python: Learn Best Practices of Deep Learning Models with PyTorch* (2021), pp. 197–242.
- [30] Serkan Kiranyaz et al. “1-D convolutional neural networks for signal processing applications”. In: *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2019, pp. 8360–8364.
- [31] Eva Kollorz et al. “Gesture recognition with a time-of-flight camera”. In: *International Journal of Intelligent Systems Technologies and Applications* 5.3-4 (2008), pp. 334–343.
- [32] Tollmar Konrad, David Demirdjian, and Trevor Darrell. “Gesture+ play: full-body interaction for virtual environments”. In: *CHI’03 Extended Abstracts on Human Factors in Computing Systems*. 2003, pp. 620–621.

- [33] Okan Köpüklü et al. “Real-time hand gesture detection and classification using convolutional neural networks”. In: *2019 14th IEEE international conference on automatic face & gesture recognition (FG 2019)*. IEEE. 2019, pp. 1–8.
- [34] Moez Krichen. “Convolutional neural networks: A survey”. In: *Computers* 12.8 (2023), p. 151.
- [35] Abdelaziz Lakhfif and Mohamed Tayeb Laskri. *Un signeur virtuel 3D pour la traduction automatique de textes arabes vers la langue des signes algérienne*. 2010.
- [36] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. “Deep learning”. In: *nature* 521.7553 (2015), pp. 436–444.
- [37] Olivier Losson. “Modélisation du geste communicatif et réalisation d’un signeur virtuel de phrases en langue des signes française”. PhD thesis. Université des Sciences et Technologie de Lille-Lille I, 2000.
- [38] Yiqin Lu et al. “Designing and evaluating hand-to-hand gestures with dual commodity wrist-worn devices”. In: *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 4.1 (2020), pp. 1–27.
- [39] Rosy Madaan and Komal Kumar Bhatia. “Prevalence of visualization techniques in data mining”. In: *Data Visualization and Knowledge Engineering: Spotting Data Points with Artificial Intelligence* (2020), pp. 273–298.
- [40] Ana I Maqueda Nieto. “Development of a hand-gesture recognition system for human-computer interaction”. In: (2014).
- [41] Aleix M Martínez et al. “Purdue RVL-SLLL ASL database for automatic recognition of American Sign Language”. In: *Proceedings. Fourth IEEE International Conference on Multimodal Interfaces*. IEEE. 2002, pp. 167–172.
- [42] Noraini Mohamed, Mumtaz Begum Mustafa, and Nazean Jomhari. “A review of the hand gesture recognition system: Current progress and future directions”. In: *ieee access* 9 (2021), pp. 157422–157436.
- [43] A Mohanarathinam et al. “Study on Hand Gesture Recognition by using Machine Learning”. In: *2020 4th International Conference on Electronics, Communication and Aerospace Technology (ICECA)*. IEEE. 2020, pp. 1498–1501.

- [44] Cristina Nuzzi et al. “Deep learning based machine vision: first steps towards a hand gesture recognition set up for collaborative robots”. In: *2018 Workshop on Metrology for Industry 4.0 and IoT*. IEEE. 2018, pp. 28–33.
- [45] Nobuyuki Otsu et al. “A threshold selection method from gray-level histograms”. In: *Automatica* 11.285-296 (1975), pp. 23–27.
- [46] Munir Oudah, Ali Al-Naji, and Javaan Chahl. “Hand gesture recognition based on computer vision: a review of techniques”. In: *journal of Imaging* 6.8 (2020), p. 73.
- [47] Fabrizio Pedersoli et al. “XKin: an open source framework for hand pose and gesture recognition using kinect”. In: *The Visual Computer* 30 (2014), pp. 1107–1122.
- [48] Realite-Virtuelle. *Cybergloves: à quoi ça sert?* 2022. URL: <https://www.realite-virtuelle.com/cybergloves-3012/> (visited on 05/29/2024).
- [49] Zhou Ren et al. “Robust part-based hand gesture recognition using kinect sensor”. In: *IEEE transactions on multimedia* 15.5 (2013), pp. 1110–1120.
- [50] Md Repon Islam, Md Saiful Islam, and Muhammad Sheikh Sadi. “Towards developing a real-time hand gesture controlled wheelchair”. In: *Proceedings of International Conference on Trends in Computational and Cognitive Engineering: Proceedings of TCCE 2020*. Springer. 2020, pp. 355–366.
- [51] Ricardo Ribani and Mauricio Marengoni. “A survey of transfer learning for convolutional neural networks”. In: *2019 32nd SIBGRAPI conference on graphics, patterns and images tutorials (SIBGRAPI-T)*. IEEE. 2019, pp. 47–57.
- [52] Stefan Waldherr Sebastian Thrun Roseli Romero and Dimitris Margaritis. “Template-Based Recognition of Pose and Motion Gestures On a Mobile Robot”. In: (1998).
- [53] Jaya Prakash Sahoo et al. “Real-time hand gesture recognition using fine-tuned convolutional neural network”. In: *Sensors* 22.3 (2022), p. 706.
- [54] Shadman Sakib et al. “An overview of convolutional neural network: Its architecture and applications”. In: (2019).

- [55] Abir Sen, Tapas Kumar Mishra, and Ratnakar Dash. “A novel hand gesture detection and recognition system based on ensemble-based convolutional neural network”. In: *Multimedia Tools and Applications* 81.28 (2022), pp. 40043–40066.
- [56] Abir Sen, Tapas Kumar Mishra, and Ratnakar Dash. “Deep Learning-Based Hand Gesture Recognition System and Design of a Human–Machine Interface”. In: *Neural Processing Letters* 55.9 (2023), pp. 12569–12596.
- [57] Rajeev Sharma et al. “Speech/gesture interface to a visual-computing environment”. In: *IEEE Computer Graphics and Applications* 20.2 (2000), pp. 29–37.
- [58] Sagar Sharma, Simone Sharma, and Anidhya Athaiya. “Activation functions in neural networks”. In: *Towards Data Sci* 6.12 (2017), pp. 310–316.
- [59] Karen Simonyan and Andrew Zisserman. “Very deep convolutional networks for large-scale image recognition”. In: *arXiv preprint arXiv:1409.1556* (2014).
- [60] Gary M Smith and MC Schraefel. “The radial scroll tool: scrolling support for stylus-or touch-based document navigation”. In: *Proceedings of the 17th annual ACM symposium on User interface software and technology*. 2004, pp. 53–56.
- [61] Josiah W Smith et al. “Improved static hand gesture classification on deep convolutional neural networks using novel sterile training technique”. In: *Ieee Access* 9 (2021), pp. 10893–10902.
- [62] Peng Song et al. “A handle bar metaphor for virtual object manipulation with mid-air interaction”. In: *Proceedings of the SIGCHI conference on human factors in computing systems*. 2012, pp. 1297–1306.
- [63] Stefan Soutschek et al. “3-d gesture-based scene navigation in medical imaging applications using time-of-flight cameras”. In: *2008 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*. IEEE. 2008, pp. 1–6.
- [64] Thad Starner, Joshua Weaver, and Alex Pentland. “Real-time american sign language recognition using desk and wearable computer based video”. In: *IEEE Transactions on pattern analysis and machine intelligence* 20.12 (1998), pp. 1371–1375.

- [65] Thad Starner et al. “The gesture pendant: A self-illuminating, wearable, infrared computer vision system for home automation control and medical monitoring”. In: *Digest of Papers. Fourth International Symposium on Wearable Computers*. IEEE. 2000, pp. 87–94.
- [66] David Stotts, Jason McC Smith, and Karl Gyllstrom. “Facespace: endo-and exo-spatial hypermedia in the transparent video facetop”. In: *Proceedings of the fifteenth ACM conference on Hypertext and hypermedia*. 2004, pp. 48–57.
- [67] Mohammad Mustafa Taye. “Understanding of machine learning with deep learning: architectures, workflow, applications and future directions”. In: *Computers* 12.5 (2023), p. 91.
- [68] Parthivi Thakore and Divyansh Johari. “An Interface for Communication for the Deaf Using Hand Gesture Recognition through Computer Vision and Natural Language Processing”. In: *Journal of Image Processing & Pattern Recognition Progress* 9.3 (2022), 25–40p.
- [69] Clayton Valli and Ceil Lucas. *Linguistics of American sign language: An introduction*. Gallaudet University Press, 2000.
- [70] Michael Van den Bergh and Luc Van Gool. “Combining RGB and ToF cameras for real-time 3D hand gesture interaction”. In: *2011 IEEE workshop on applications of computer vision (WACV)*. IEEE. 2011, pp. 66–72.
- [71] Michael Van den Bergh et al. “Real-time 3D hand gesture interaction with a robot for understanding directions from humans”. In: *2011 Ro-Man*. IEEE. 2011, pp. 357–362.
- [72] Guido Van Rossum et al. “Python Programming Language.” In: *USENIX annual technical conference*. Vol. 41. 1. Santa Clara, CA. 2007, pp. 1–36.
- [73] Monu Verma, Ayushi Gupta, and Santosh K Vipparthi. “One for all: An end-to-end compact solution for hand gesture recognition”. In: *2021 International Joint Conference on Neural Networks (IJCNN)*. IEEE. 2021, pp. 1–8.
- [74] Christian Vogler and Dimitris Metaxas. “A framework for recognizing the simultaneous aspects of american sign language”. In: *Computer Vision and Image Understanding* 81.3 (2001), pp. 358–384.
- [75] Juan Pablo Wachs et al. “Vision-based hand-gesture applications”. In: *Communications of the ACM* 54.2 (2011), pp. 60–71.

- [76] Ankita Wadhawan and Parteek Kumar. “Deep learning-based sign language recognition system for static signs”. In: *Neural computing and applications* 32.12 (2020), pp. 7957–7968.
- [77] CH Wagner. “The pianist’s hand: anthropometry and biomechanics”. In: *Ergonomics* 31.1 (1988), pp. 97–131.
- [78] Manjula B Waldron and Soowon Kim. “Isolated ASL sign recognition system for deaf persons”. In: *IEEE Transactions on rehabilitation engineering* 3.3 (1995), pp. 261–271.
- [79] Mingqing Xiao et al. “Invertible image rescaling”. In: *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part I 16*. Springer. 2020, pp. 126–144.
- [80] Kai Xing et al. “Hand gesture recognition based on deep learning method”. In: *2018 IEEE Third International Conference on Data Science in Cyberspace (DSC)*. IEEE. 2018, pp. 542–546.
- [81] Dingjun Yu et al. “Mixed pooling for convolutional neural networks”. In: *Rough Sets and Knowledge Technology: 9th International Conference, RSKT 2014, Shanghai, China, October 24–26, 2014, Proceedings 9*. Springer. 2014, pp. 364–375.
- [82] Jinhua Zeng, Yaoru Sun, and Fang Wang. “A natural hand gesture system for intelligent human-computer interaction and medical assistance”. In: *2012 Third Global Congress on Intelligent Systems*. IEEE. 2012, pp. 382–385.
- [83] Xuesong Zhai et al. “A Review of Artificial Intelligence (AI) in Education from 2010 to 2020”. In: *Complexity* 2021 (2021), pp. 1–18.
- [84] Xu Zhang et al. “Hand gesture recognition and virtual game control based on 3D accelerometer and EMG sensors”. In: *Proceedings of the 14th international conference on Intelligent user interfaces*. 2009, pp. 401–406.
- [85] Jianfeng Zhao, Xia Mao, and Lijiang Chen. “Speech emotion recognition using deep 1D & 2D CNN LSTM networks”. In: *Biomedical signal processing and control* 47 (2019), pp. 312–323.