People's Democratic Republic of Algeria
Ministry of Higher Education for Scientific Research
University 8 May 45 –Guelma-
Faculty of Mathematics, Computer Science and Sciences of Matter
Department of Computer Science

Master Thesis

Specialty: Computer science

Option:

Science and Technology of Information and Communication

Theme

An Arabic sign language recognition system for word-level generation and translation

Presented by: Rania Boufelfel

Jury Members

Chairman	Dr. Abderrahmane kefali
Supervisor	Dr. Hassina Bouressace
Examiner	Dr. Yamina Bourdjiba

June 2024

Acknowledgments

First of all, I am here by the grace of Allah. Alhamdulillah, who facilitated my path and gave me the strength and ability to finish my course, and I ask Him for continued success. I would like to thank my supervisor, Dr.Bouressace Hassina, for her support, dedication, and wisdom, which have been instrumental in shaping this thesis. I am truly grateful for her guidance.

Additionally, I would like to honor myself for my persistence despite the challenges; I have persevered, matured, and learned a lot. I wish her well going forward. I thank every member of my family, big and small, who has been supportive, even with a small smile.

I thank everyone in this department who worked hard for our comfort.

Abstract

Considering that communication is essential for human connection, the deaf community faces unique obstacles. Therefore, sign language is the best alternative for overcoming these communication barriers, as it is considered the most effective means of communication, involving many hand movements. However, sign language is often misunderstood by those not part of the deaf community, necessitating the use of interpreters. This has led the community to develop techniques to facilitate interpretation tasks. Despite progress in deep learning, there is still limited research on recognizing and translating Arabic sign language. This lack of research has prompted us to focus specifically on advancing studies in Arabic sign language. This thesis introduces improved methodologies to construct a comprehensive framework for processing, translating, and generating Arabic sign language from input videos. We begin by utilizing the Mediapipe library for identifying human body parts. Then, for sign language recognition, particularly in Arabic, we employed four distinct models: YOLOv8, Convolutional Neural Networks (CNN), Long Short-Term Memory (LSTM), and a hybrid CNN-LSTM approach. Using the ArabSign-A dataset [59], we adapted it to focus on individual words, achieving an accuracy of 87.37% for YOLOv8, 95.23% for the CNN model, 88.09% % for the LSTM model, and 96.66% for the hybrid model. A comparative analysis was conducted to evaluate our methodology, demonstrating superior discrimination between static signs compared to prior research.

Keywords: Arabic sing language, ArSL, CNN, LSTM, Hybrid CNN-LSTM, YOLOv8, Mediapipe.

Résumé

Tenant compte que la communication est essentielle pour la connexion humaine, la communauté sourde fait face à des obstacles uniques. Par conséquent, la langue des signes est la meilleure alternative pour surmonter ces barrières de communication, car elle est considérée comme le moyen de communication le plus efficace, impliquant de nombreux mouvements de la main. Cependant, la langue des signes est souvent mal comprise par ceux qui ne font pas partie de la communauté sourde, nécessitant l'utilisation d'interprètes. Cela a conduit la communauté à développer des techniques pour faciliter les tâches d'interprétation. Malgré les progrès réalisés dans l'apprentissage en profondeur, il existe encore peu de recherches sur la reconnaissance et la traduction de la langue des signes arabe. Ce manque de recherche nous a incités à nous concentrer spécifiquement sur l'avancement des études en langue des signes arabe. Cette thèse introduit des méthodologies améliorées pour construire un cadre complet de traitement, de traduction et de génération de la langue des signes arabe à partir de vidéos d'entrée. Nous commençons par utiliser la bibliothèque Mediapipe pour identifier les parties du corps humain. Ensuite, pour la reconnaissance de la langue des signes, notamment en arabe, nous avons employé quatre modèles distincts : YOLOv8, Réseaux de Neurones Convolutionnels (CNN), Mémoire à Long Terme à Court Terme (LSTM), et une approche hybride CNN-LSTM. En utilisant l'ensemble de données ArabSign-A [59], nous l'avons adapté pour nous concentrer sur des mots individuels, atteignant une précision de 87,37 % pour YOLOv8, 95.23% pour le modèle CNN, 88.09% pour le modèle LSTM, et 96.66% pour le modèle hybride. Une analyse comparative a été menée pour évaluer notre méthodologie, démontrant une discrimination supérieure entre les signes statiques par rapport aux recherches antérieures.

Mots-clés : Langue des singer arabe, ArSL, CNN, LSTM, Hybride CNN-LSTM, YOLOv8, Mediapipe.

List of content

List of Figures

List of Tables

List of abbreviations

ArSL	Arabic Sign Language
ArSLR	Arabic Sign Language recognition
SVM	Support Vector Machines
PCA	Principal Component Analysis
KNN	K-Nearest Neighbors
CNN	Convolutional Neural Networks
3D-CNN	3-Dimensional Convolutional Neural Networks
ML	Machine learning
LSTM	Long Short-Term Memory
GAN	Generative Adversarial Networks
HMM	Hidden Markov Model
ANN	Artificial Neural Network
NMT	Neural Machine Translation
ANFIS	Neuro-Fuzzy Inference System
SLID	Sign Language Identification
ReLU	Rectified Linear Unit
DBN	Dynamic Bayesian Networks
MLP	Multilayer Perceptron
SMO	Sequential Minimal Optimization
VFI	Voting Feature Intervals
CSOM	Convolutional Self-Organizing Map
BiLSTM	Bi-directional Long Short-Term Memory

General Introduction

Oral language is the most useful way for communication among individuals, encompassing a diverse range of dialects that differ from country to country and even from region to region. However, there is a segment of the population that struggles with spoken language due to issues related to pronunciation, hearing, or both, which leading to many challenges and obstacles that oral language cannot handle. Therefore, sign language emerges as a powerful communication tool and an efficient substitute for this state. Sign language operates as a communication system that utilizes hand and body movements, along with facial expressions, to convey meaningful messages instead of spoken words. This unique language provides a means to express their thoughts and engage with others. The sign language system varies across regions; therefore, we have dozens of sign languages, each specified by the local language, for example, there are many sign languages for the Arabic language, with each dialect characterized by unique signs. This diversity complicates the understanding process of the system, which is expressed through hand and body movements, coupled with facial expressions and other gestures that convey thoughts and emotions without relying on spoken words.

In this project, we aim to develop a system that can handle Arabic sign language by translating body movements including face and hands movements into comprehensible Arabic words. This task is challenging due to dialect variations and input quality. Therefore, we focused on using offline videos of individuals using sign language to convey messages, where our goal is to extract a complete and comprehensive Arabic phrase from each video, which may consist of more than one word.

Our work is organized into four chapters, where each one focuses on key aspects of our developed system.

- **The first chapter** presents the generalities of sign language recognition, which detailed the background of sign language, its recognition methods, and its application in the Arabic language.
- **The second chapter** discusses the state-of-the-art methods for sign language recognition, specifically in the context of the Arabic language.
- **The third chapter** outlines the conception of the developed system for Arabic sign language recognition.
- **The fourth chapter** describes the implementation of the developed system, along with its results and a comparative study with another research in the same field.

Chapitre 1. Sign language recognition

1.1. Introduction

Sign language recognition represents the best solution and for the individuals with hearing impairments, and with technological advancements and the incorporation of artificial intelligence, researchers are actively investigating novel methods to improve the precision and effectiveness of sign language recognition systems. This chapter explores the basic concepts of sign language, detailing into the detection, identification, and categorization of sign language recognition methods, as well as the utilization of hand-sensing technology, with a specific focus on Arabic sign language recognition (ArSLR).

1.2. Sign language detection identification and recognition

Sign language, in general, is a visual language based on hand gestures used by humans to communicate. It involves coordinated movements of different parts of the body, including the hands, face, and body [20]. Sign language is recognized as the most structured form of gesture-based communication. Similar to spoken languages, sign language naturally develops within communities of individuals with hearing impairments, which evolve independently from the spoken language of the region. Each sign language has its own grammar and rules, all of which share the common feature of being visually perceived. Just as there are numerous spoken languages worldwide, there are also various sign languages used globally [21].

Figure 1-1 *Example of identification of the word ' الله ' in Arabic sign gesture[59].*

Sign language detection involves determining whether a video from a diverse and unrestricted collection includes sign language content. This process has various applications, such as automatically tagging and categorizing videos, serving as an initial phase toward automatically captioning sign language videos, and facilitating the automatic initiation of Automatic Sign Language Recognition (ASLR) without relying on predetermined assumptions about the input videos [25].

Figure 1-2 *Example of detection of the word ' الله ' in Arabic sign gesture[59].*

Sign language recognition is the final phase in the system which includes identifying and understanding sign language gestures, signs, and expressions using machine learning algorithms, deep learning models, or computer vision techniques to interpret them into letters or words which can be either in spoken language or text, which involves the classification and interpretation of hand gestures, finger positions, and body movements to accurately translate sign language into meaningful text or speech, enabling effective communication between individuals using sign language and those who do not understand sign language.[16]

Figure 1-3 *Example of recognition of the word ' الله ' in Arabic sign gesture[59].*

1.3. Categorization of hand gesture recognition methods

Hand gesture recognition methods can be categorized into multiple methods based on different criteria. One of the main criteria is the type of input device used to capture the sign language gestures. Depending on the input device, sign language recognition methods can be classified as glove-based, utilizing specialized gloves equipped with sensors for hand sensing, or vision-based (see **Figure 1-4**), using cameras or other visual sensors to capture hand movements and gestures. These categorizations are essential for understanding the diverse approaches in hand sensing technology within sign language recognition.[24]

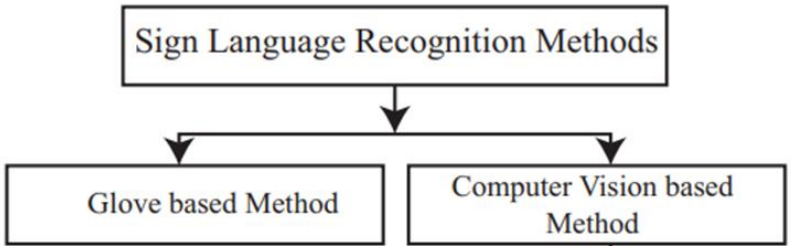


Figure 1-4 *Categorization of sign language recognition methods [1].*

1.3.1. Glove-based method

The glove-based method has become a vital component for gesture detection and recognition, playing an essential role in virtual reality applications. This method extracts hand postures or gestures by measuring joint angles, finger bending, and hand orientation using sensors. Over time, several versions of gloves have been developed specifically for virtual reality purposes where its ability is to capture 3D hand information without the need for intricate segmentation. However, drawbacks include its intrusive nature, high cost, and potential discomfort for the wearer. The following are the most important versions that have been developed for this purpose:

A. Sayre Glove: Developed at the University of Illinois at Chicago in 1976, it used flexible tubes with light sources and photocells to monitor hand movements for multidimensional control.

B. MIT LED Glove: Developed in the early 1980s at the MIT Media Lab, it used a camera-based LED system to track body and limb positions for computer graphics animation.

C. Digital Data Entry Glove: Developed in 1983 by Gary Grimes of Bell Telephone Laboratories, it had sensors to recognize hand signs for data entry but was not commercially developed.

D. DataGlove: Developed in 1987 by Thomas Zimmerman and others, it monitored finger joints and hand position in real-time using optical fibers and magnetic tracking, leading to widespread use.[22]

E. PowerGlove: Marketed by Mattel in 1989 for Nintendo game systems, offering hand-measuring capabilities at a lower cost but with reduced performance.[23]

F. Dexterous HandMaster: Originally developed as a master controller for a robotic hand, it was later adapted for use as an exoskeleton glove with Hall-effect sensors in 1989.

G. ***Space Glove:*** Developed in 1991 by W Industries for their virtually system.[22]
H. ***CyberGlove:*** Developed by Jim Kramer at Stanford University, using flexible strain gauges in a Wheatstone bridge configuration for accurate joint angle measurement.
I. **5th Glove:** Developed by Fifth Dimension Technologies and released in 1995, providing a cost-effective option with improved accuracy compared to the PowerGlove.[23]

1.3.2. Computer vision-based method

The second major approach for detecting and recognizing hands involves utilizing standard video camera equipment, such as cameras or webcams, to capture a visual image of the user. Computer vision techniques are then employed to extract data about the hands from this image, including acquiring image series depicting hand postures or gestures. Subsequently, computer vision and image processing approaches are implemented to delineate the hand from the environment, elicit pertinent attributes, and discern the gestures. Benefits of this strategy encompass its naturalness, absence of intrusiveness, and affordability. Nevertheless, limitations emerge owing to susceptibility toward illumination variations, hindrances, and cluttered surroundings. Moreover, computer vision's capabilities are highlighted within this context. Vision-based methods can be further divided into two types as shown in **Figure 1-5**.

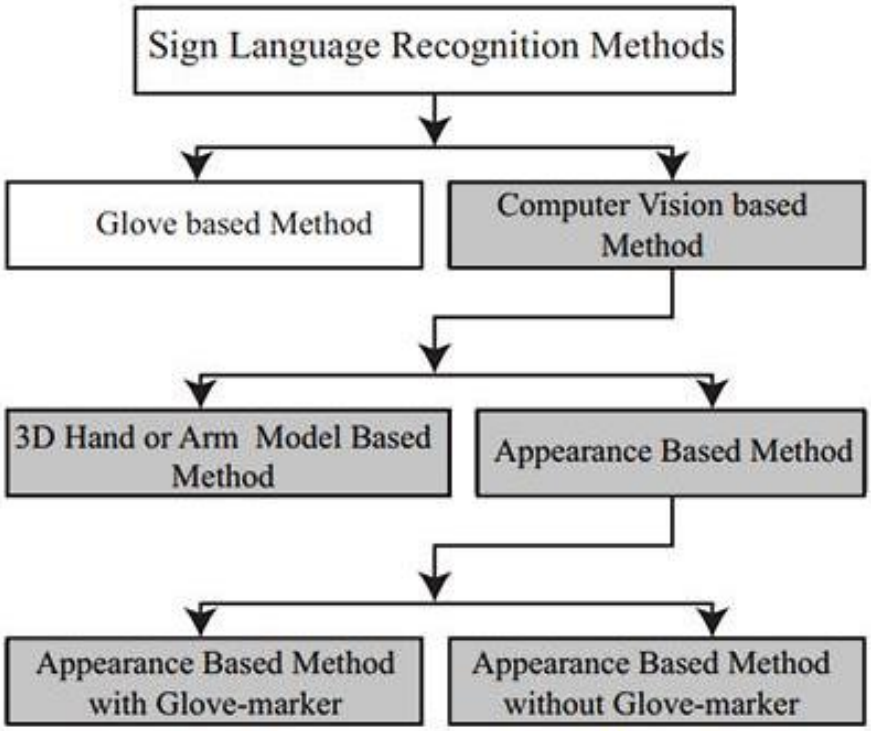


Figure 1-5 *Categorization of sign language recognition [1].*

A. **3D hand or arm model-based method:** Encompassing a subset of computer vision-driven strategies, this category utilizes a 3D representation of hands or arms to portray gestures. Model parameters are derived from pictures via tactics such as edge matching, form matching, or model adjustment. Strengths of this method comprise handling elaborate gestures and occlusions. Weaknesses incorporate elevated

computational expenses and an extensive volume of instructional data required [1]. **Figure** 1-6 illustrates an example of this method.

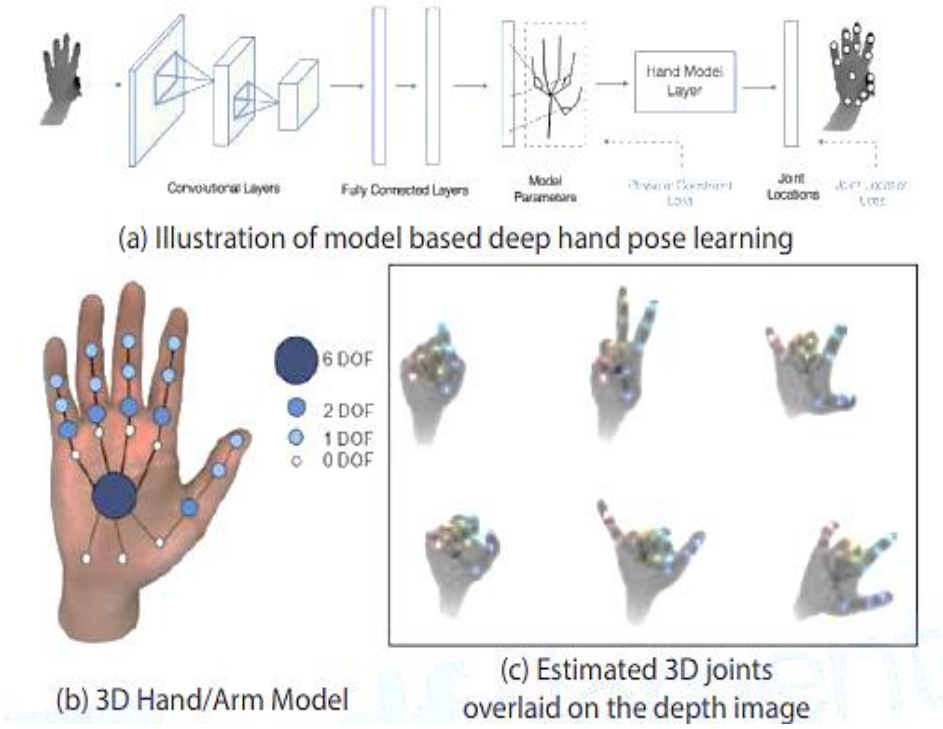


Figure 1-6 *An example of a 3D hand or arm model-based system[1].*

B. Appearance-based method: The appearance-based method utilizes the appearance of the hand or the gesture as the feature and can be either with or without glove-markers. Glove-markers (**Figure** 1-8) are colored or illuminated markers that are attached to the glove to facilitate the segmentation **Figure** 1-7and feature extraction process. The proposed system in this paper uses an appearance-based method without glove-markers, which relies on skin-color detection and edge detection to segment the hand from the background and uses geometrical features to recognize the gestures [1]**.**

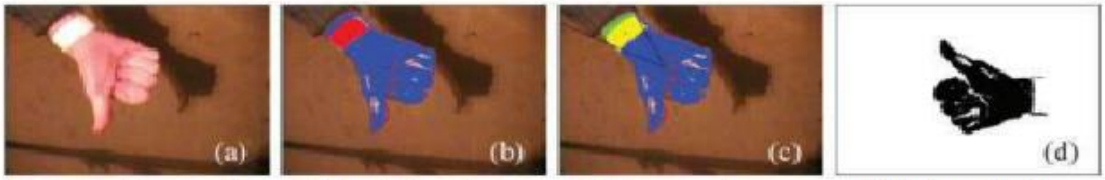


Figure 1-7 *An example of the hand-sign segmentation process using color wristband [1].*

Figure 1-8 *An example uses of glove-markers [1].*

1.4. Vision based gesture recognition existing techniques

The devices and technologies mentioned in the previous section represent various types of inputs used to capture the sign language gestures. In this part, we focus on the recognition process and its phases within vision-based aspect, which involve specializing hand position and orientation using position tracking techniques to achieve the recognition goal.

1.4.1. Measuring joint positioning

This phase includes feature extraction and hand gesture classification by determining and calculating how the joints in the signer's body, especially in the hands and other related parts, are positioned or aligned in space. Measuring joint positioning accurately is very important technique that is used for interpreting the exact gestures and signs that are being shown [30]. The following are the essential techniques used for feature extraction approaches:

A. **Joint Angles:** Accurate measurement of joint angles allows for more precise recognition and interpretation of sign language gestures. This information is crucial for developing effective sign language recognition systems that can understand and interpret the complex movements and configurations of the user's hands and body.[30]
B. **Finger Positions:** Additionally, the precise recognition of finger positions plays a vital role in sign language interpretation. By precisely assessing the placement of each finger and its alignment with the hand and other fingers, sign language recognition systems can differentiate between various signs that may share similar hand shapes but vary in finger positions.[31]
C. **Hand Orientation:** is a critical aspect of sign language as it contributes to the clarity and accuracy of communication, where the orientation of the hand, including its position, angle, and movement, conveys specific meanings and nuances, which can change the interpretation of signs and gestures, highlighting the importance of precise hand positioning in sign language communication.[32]

In the context of its approaches using the techniques above, they can be categorized as follows:

A. **Model based Approaches (Kinematic Model):** These approaches focus on inferring the pose of the palm and joint angles to enable realistic interactions in virtual environments. By searching for kinematic parameters that align a 2D projection of a 3D hand model with an edge-based image of a hand, these

approaches aim to accurately represent hand movements.

B. **View based Approaches:** View-based approaches, also known as appearance-based approaches, represent the hand as a collection of 2D intensity images. Gestures are then modeled as sequences of views, offering an alternative to kinematic model-based approaches.
C. **Low Level Features based Approaches:** Instead of reconstructing the entire hand, these approaches focus on extracting robust, low-level image measurements that are resistant to noise and can be quickly obtained. Examples of low-level features include the centroid of the hand region, principal axes defining an elliptical bounding region, and optical flow/affine flow of the hand region in a scene [34].

1.4.2. Position Tracking

Position tracking is a crucial component in gesture recognition. It involves monitoring the precise spatial location of the hand using cameras, motion detectors, or wearable devices. One of the most used technologies that are essential for accurately interpreting sign language gestures in computer vision aspect are presented in the following elements.

A. **Optical Systems:** These systems use ambient light or dedicated light emitters with sensors (e.g., using cameras on a VR headset) to track the subject's position. They can be 'inside-out' or 'outside-in' systems and may use pattern recognition for tracking [26].
B. **Depth Sensors:** There are famous techniques such as Intel RealSense cameras also used for gesture recognition applications which are based on measure the distance to objects in a scene. [48]
C. **Marker-based Tracking:** This technique is based on placing markers on specific points of interest, such as fingertips, which is often used in applications where high precision is required.[49]
D. **Feature-based Tracking:** There are famous techniques such as the Kanade-Lucas-Tomasi (KLT) feature tracker, which is commonly used for real-time gesture recognition, that track points (features) on the hands or objects, allowing for robust and efficient tracking in varying conditions.[50]

1.4.3. Hand gesture recognition

Hand gesture recognition is a fundamental aspect of systems designed for sign language interpretation. This process entails the examination and comprehension of hand movements to decode and grasp the associated concepts within sign language. The efficacy of hand gesture recognition within these systems is underpinned by an amalgamation of tactile sensing devices, responsive control mechanisms, and algorithms dedicated to pattern identification. The findings from the aforementioned research underscore the proficiency and promise that hand sensing technologies hold in the domain of sign language recognition [23]. Hand gestures can be classified using the following approaches.

A. Machine Learning based Approaches

In the machine learning aspect, there are many approaches that are constantly used for hand gesture recognition. In the following, we present the most commonly used ones:

a. Artificial Neural Network (ANN)

ANNs are computational models used for various machine learning tasks like pattern recognition, clustering, and classification. Inspired by biological systems, ANNs mimic the human brain's ability to learn and store knowledge through synaptic weights between neurons. This unique characteristic sets neural networks apart, which utilized Multilayer Perceptron, that consists of multiple hidden layers of neurons [39].

The use of artificial neural networks (ANNs) in sign language involves the application of machine learning techniques to recognize and interpret signals. ANNs are particularly useful in this context because they can learn to recognize complex patterns from visual and temporal data, which is essential for understanding the nuances of sign communication.

b. Hidden Markov Model (HMM)

HMMs are a statistical model used to model temporal series and are known for their high recognition rate in dynamic gesture recognition [35] [36]. These models consist of a finite number of states with associated random functions, where transitions between states generate observation symbols based on the current state's function. HMMs efficiently model temporal dependencies and employ algorithms like Baum-Welch and Viterbi for evaluation, learning, and decoding. **Figure 1-9** depicts an example of an HMM model used for hand gesture recognition.

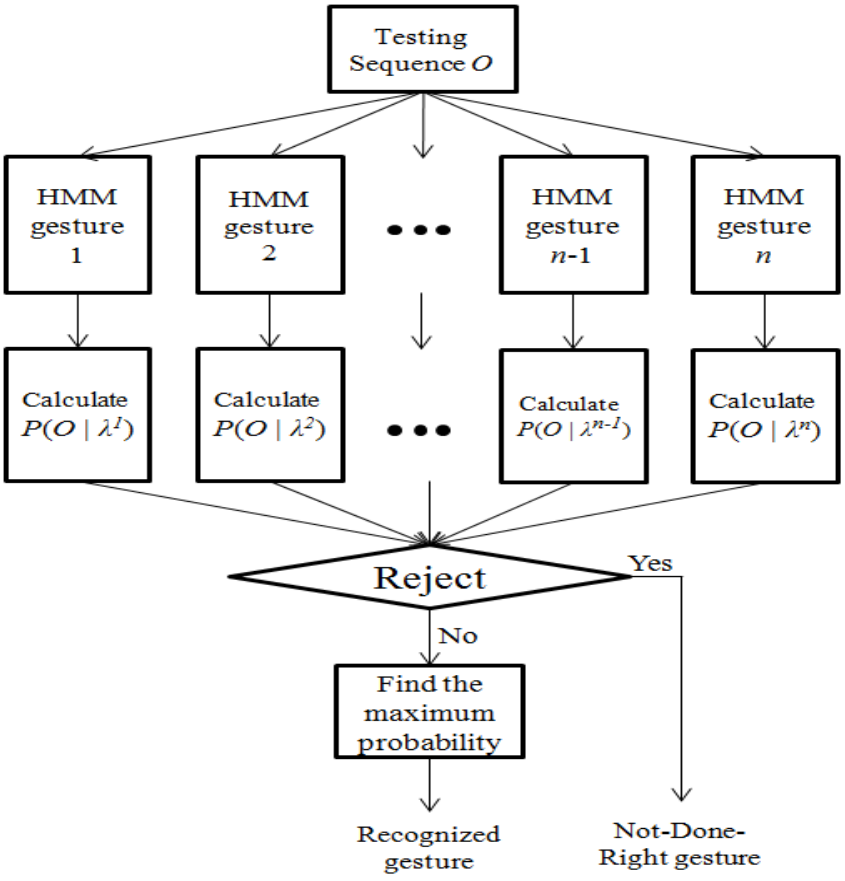


Figure 1-9 *The architecture of hand gesture recognition using an HMM model.[51]*

Successful applications of HMMs include hand and face recognition tasks, utilizing 2-D projections from 3-D models and experimental feature extraction. HMM classifiers effectively capture the temporal dynamics of dynamic gestures, trained on data and validated with test sets. Challenges in HMM usage encompass evaluation, training for probability optimization, and decoding to infer the underlying state sequence [37].

c. Support Vector Machines (SVM)

The objective of a SVM classifier is to find the most effective separation between classes. Unlike other classifiers, SVM relies on the principle of structural risk minimization (SRM), ensuring robust generalization in machine learning tasks. SVMs have been successfully utilized for hand gesture recognition tasks that provide efficient performance in distinguishing between different hand gestures. **Figure 1-10** depicts an example of a SVM model used for hand gesture recognition.[51]

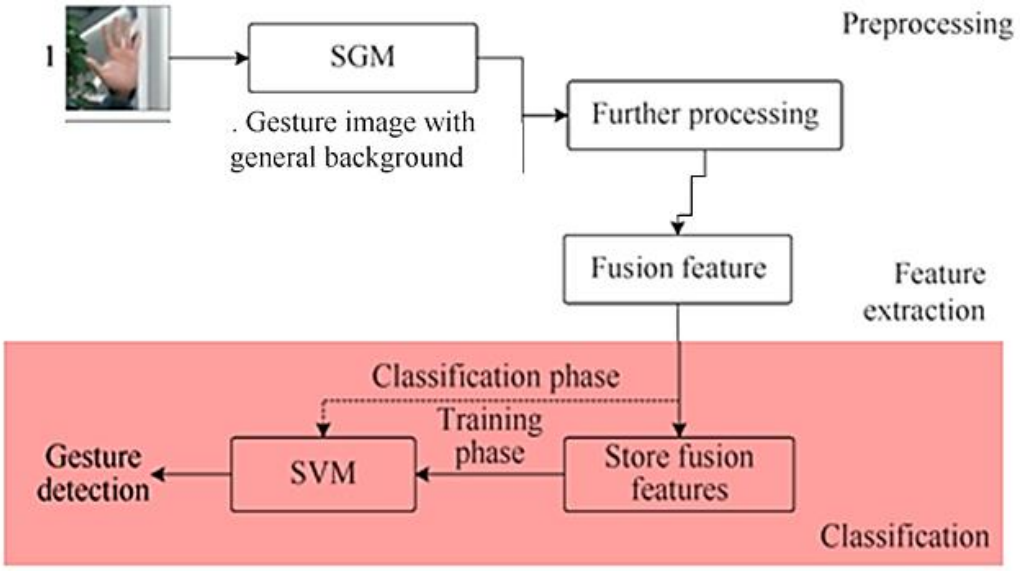


Figure 1-10 *The architecture of hand gesture recognition using a SVM model. [53[*

B. Deep learning approaches

In the deep learning aspect, there are many approaches that are constantly used for hand gesture recognition. In the following, we present the most commonly used ones:

a. Convolutional Neural Network (CNN)

CNNs are deep learning models well-suited for image processing and are crucial in gesture recognition. They excel at extracting features from gesture images by utilizing multi-layer convolution and pooling operations to automatically learn spatial structures and local features. **Figure 1-11** depicts an architecture example of a CNN model used for hand gesture recognition.

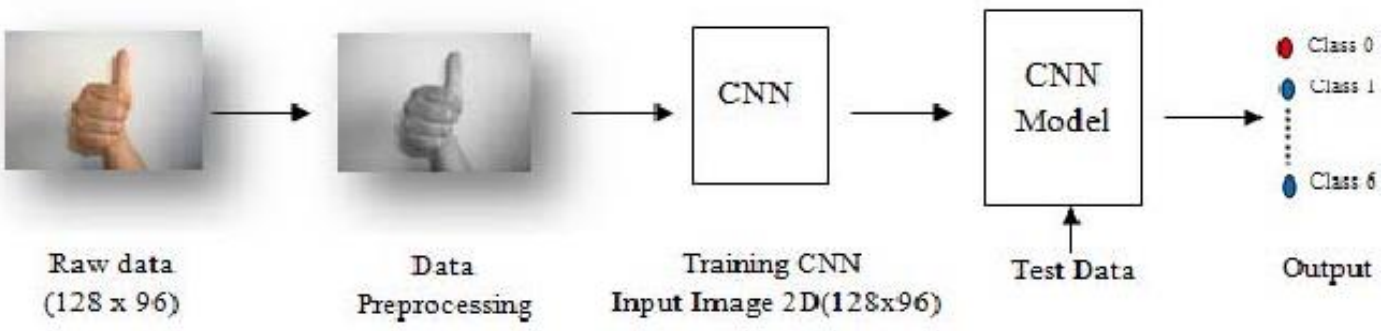


Figure 1-11 *The architecture of hand gesture recognition using a CNN model.[54]*

Where CNN phase is described in the following elements:

- **Feature Extraction:** CNNs automatically extract features from images, such as edges, textures, and shapes, through convolution and pooling layers. These features effectively represent essential information in gestures, enabling accurate gesture recognition.
- **Hierarchical Representation:** Through successive convolution and pooling operations, CNNs construct a hierarchical representation of images. This representation captures features at various levels, enhancing the understanding of gesture structures and content for improved recognition accuracy.
- **Weight Sharing:** CNNs employ weight sharing in the convolutional layer, where the same set of weights is utilized across the entire image. This mechanism reduces model parameters, enhancing training efficiency and aiding in local feature extraction for gesture recognition.
- **Translation Invariance:** CNNs exhibit translation invariance, enabling them to recognize gestures even when translated within an image. This property enhances robustness in gesture recognition tasks, accommodating changes in gesture positions [40].

b. Long Short-Term Memory (LSTM)

Recurrent Neural Networks (RNN) are pivotal for processing sequential data, making them ideal for handling gesture sequences in recognition tasks. Specifically, LSTM, a specialized RNN variant, excels in managing long sequence data effectively by addressing gradient-related challenges encountered in traditional RNN models. In gesture recognition, LSTM enhances the modeling of time sequence information, crucial for continuous gesture tracking and recognition. **Figure 1-12** depicts an architecture example of a LSTM model used for hand gesture recognition.

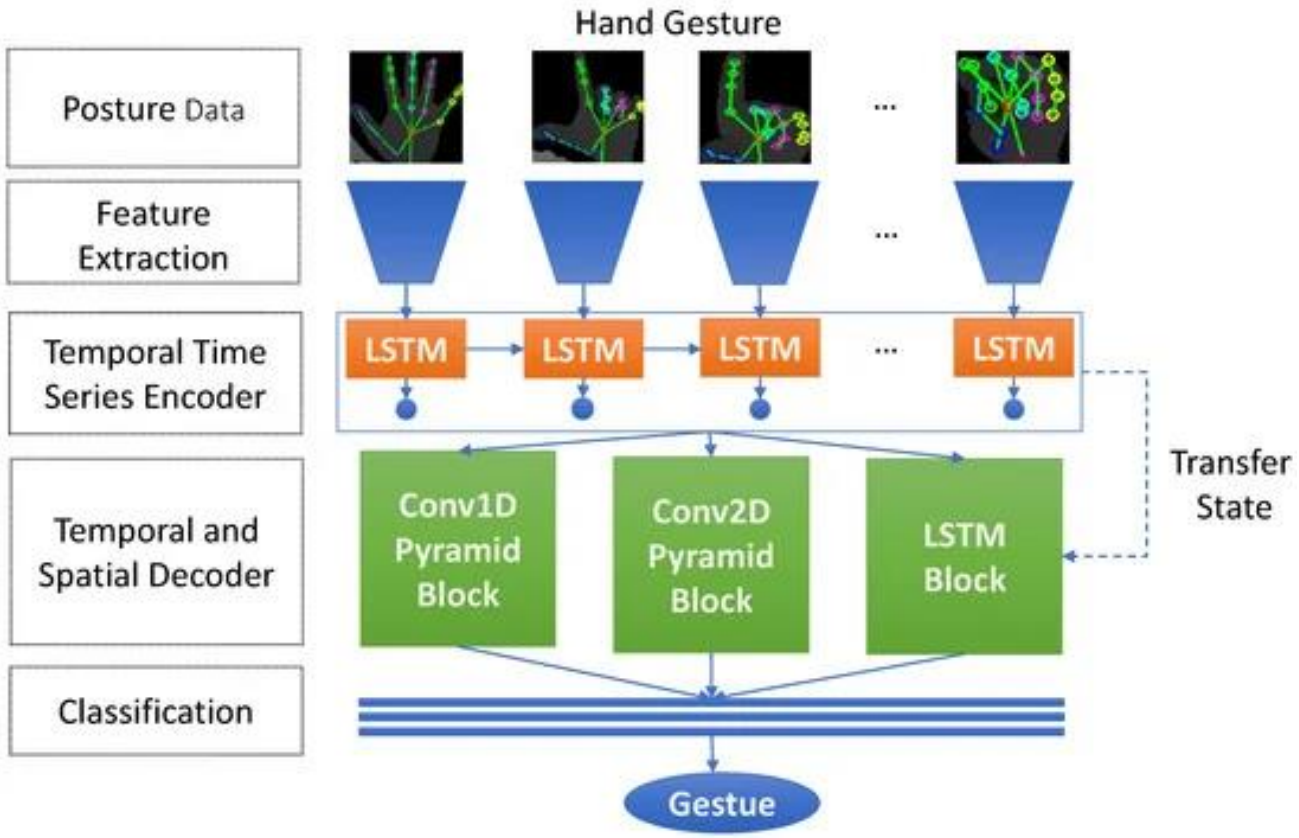


Figure 1-12 *The architecture of hand gesture recognition using a LSTM model.[55]*

Where the key points of LSTM in gesture recognition include:

- **Processing Time Series Data:** LSTM effectively processes time series data, capturing temporal changes in gestures to understand timing features crucial for accurate recognition.
- **Long-Term Dependency Modeling:** LSTM mitigates issues like gradient vanishing or exploding gradients when dealing with long-term dependencies in gesture features. Its gating mechanism, particularly the forgetting and input gates, enables robust modeling of long-term dependencies.
- **Modeling Context Information:** LSTM can model interrelated gestures by passing context information from previous gestures to subsequent ones. This capability enhances the understanding of gesture sequences, improving recognition accuracy[40].

C. Other Approaches

Besides the machine and deep learning models, there are few other approaches that are used for hand gesture recognition. In the following, we present the most commonly used ones:

a. Fuzzy Logic Based Approaches

Fuzzy logic model is described as a multivalued logic allowing for intermediate values between standard evaluations, that was presented as a versatile tool [42]. Fuzzy logic approaches in Sign Language recognition are used to improve the recognition, by dealing with

the uncertainty and imprecision inherent in the nature of sign language signs, which can vary considerably from person to person in terms of size, speed and style.

b. Genetic algorithm-based approaches

Genetic Algorithm is highly effective for finding optimal or near-optimal solutions with minimal constraints. GA operates using a generate-and-test mechanism over a set of possible solutions, known as a population, to arrive at an optimal acceptable solution. A significant advantage of GA is its parallel processing capability, allowing faster computations by simultaneously working on multiple solution points.[73]

1.4.4. Hand gesture interpretation

This phase is the final step in the whole process, which goes beyond understanding the meaning behind the body gestures and involves extracting the sign language and interpreting the generated text into meaningful communication, often using virtual models or other mediums. The following section outlines the approaches used for sign language interpretation.

A. **Avatar Approaches:** Utilizing 3D animated models to display signed conversations efficiently and accurately in the absence of human signers.
B. NMT Approaches: Implementing Neural Machine Translation (NMT) techniques to translate text from spoken language to human pose sequences.
C. Motion Graph (MG) Approaches: Incorporating Motion Graph methods to dynamically animate characters and generate new sequences to meet specific goals.
D. Conditional Image/Video Generation Approaches: Exploring techniques for generating images or videos based on specific conditions or inputs. [60]

1.5.Applications of hand gesture recognition

Hand gesture recognition technology has a wide array of applications, particularly in enhancing communication and learning. Here are two significant applications that is specified in sign language processing:

1.5.1. Sign language translation system

This application uses hand gesture recognition to translate sign language into spoken language or text in real-time (see **Figure 1-13**). It’s a vital tool for facilitating communication between deaf or hard-of-hearing individuals and those who do not understand sign language. Advanced systems may include wearable sensor arrays and wireless circuitry for sensitivity and quick response [2].

Figure 1-13 *Example for sign language translation system [2].*

1.5.2. Sign language training system

This is adept at monitoring and interpreting the hand movements of users. These systems provide instantaneous feedback, aiding learners in honing their sign language abilities (see **Figure 1-14**). Structured to be interactive, they offer extensive curricula that encompass a broad spectrum of vocabulary and grammar, as well as a deep dive into the cultural fabric of the Deaf community, thereby facilitating a thorough and engaging learning experience. [29]

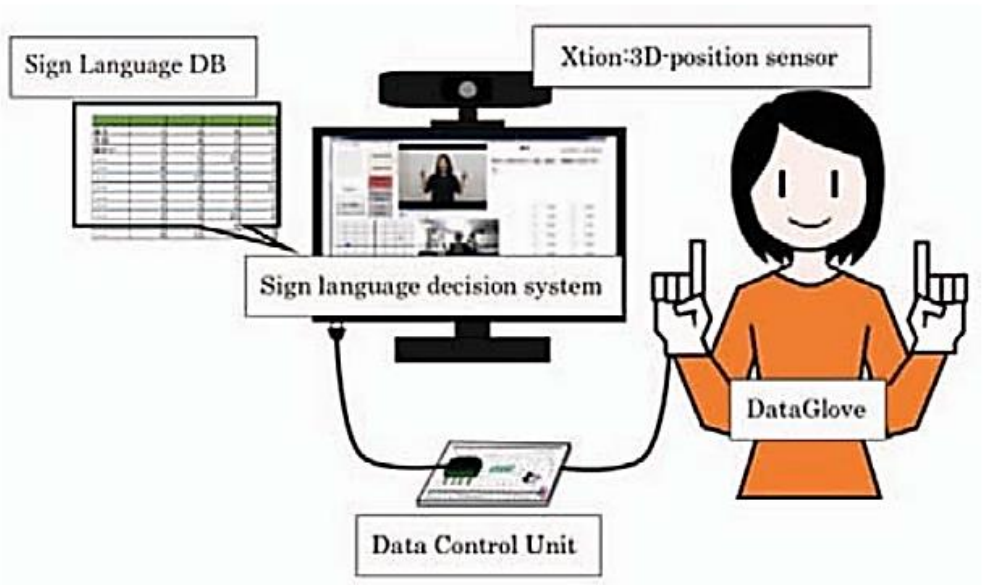


Figure 1-14 *Overview of sign language training system.[29]*

1.6. Sign language recognition in Arabic language (ArSLR)

Arabic Sign Language (ArSL) is a comprehensive and organic means of communication adopted by the deaf population across Arab countries. The widespread unfamiliarity with ArSL exacerbates the marginalization of deaf people, further alienating them from societal participation. ArSL stands apart from spoken Arabic, showcasing unique grammatical

structures, sentence arrangements, and a specialized lexicon. It embodies the cultural and linguistic identity of the deaf community, serving as a vital tool for inclusion and empowerment [3].

1.6.1. Structural components of signs

Arabic sign language gestures are characterized by two primary feature sets: manual and non-manual elements.

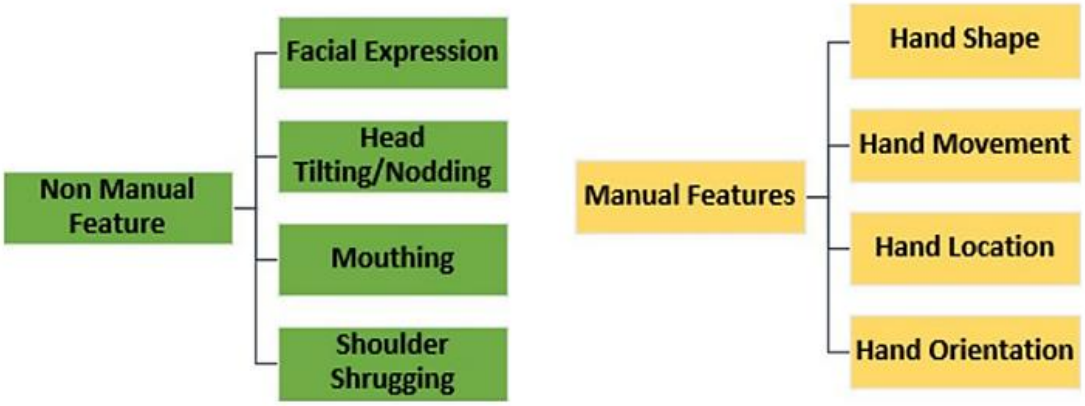


Figure 1-15 *Components of non-manual and manual features [4].*

A. **Manual elements:** Those are based on the hand's shape, motion, position, and orientation. Some gestures are executed with one hand, while others require both hands. An illustration can showcase these manual elements through various gestures [4].

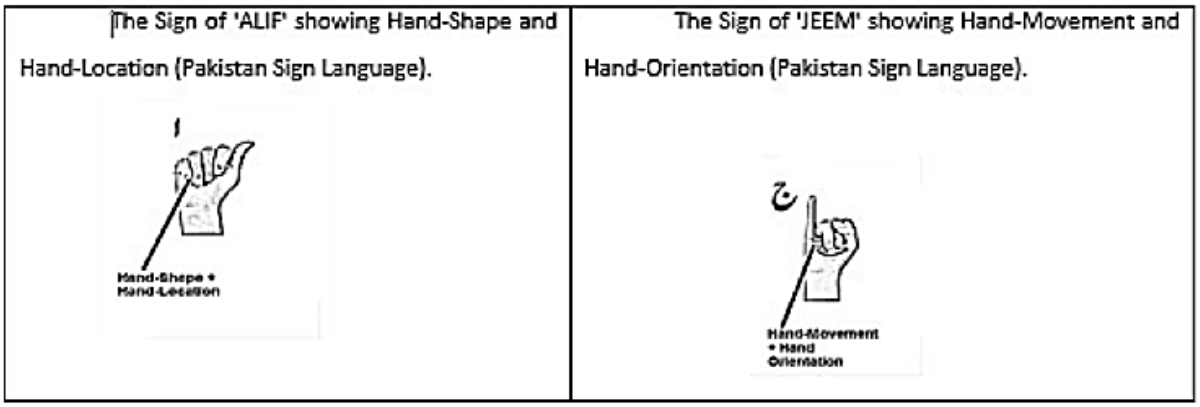


Figure 1-16 *Examples showing manual features [4].*

a. **Gestures:** Gestures involving hand movements are known as dynamic signs, whereas those without hand movements are static signs. Gestures can also be categorized based on the number of hands used: single-handed for one hand, and double-handed for both hands. **Figure** 1-17 displays examples of single and double-handed gestures, both static and dynamic, with dynamic gestures illustrated through a sequence of frames [4].

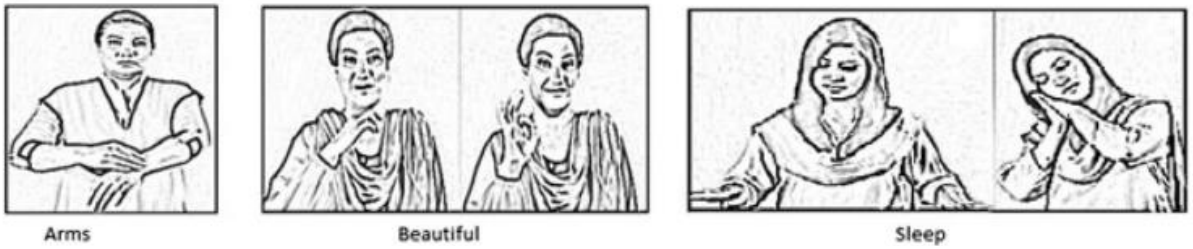


Figure 1-17 *An example of Single- and double handed gestures [4].*

B. **Non-manual elements:** Encompass a range of facial expressions, head movements such as tilts and nods, shoulder movements, mouthing, and other actions that enhance the meaning of a sign. These non-manual markers typically accompany manual signs to convey complete information. **Figure** 1-18 can depict these non-manual elements, and another can demonstrate the differences between manual and non-manual gestures [4].

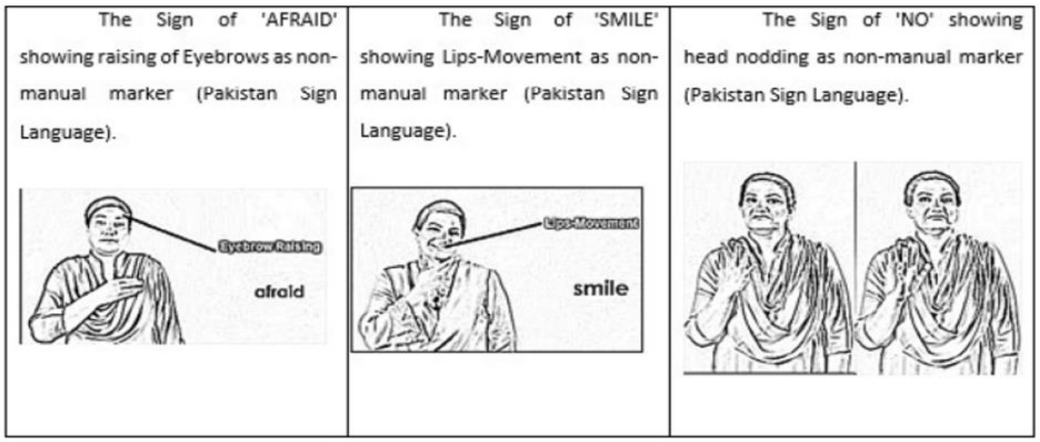


Figure 1-18 *An example showing non manual features [4].*

1.6.2. Components of Arabic signs

In Arabic sign language, basic sign gestures consist of several key components including handshape, hand orientation, hand location, and movement. Understanding these components and measuring joint positioning and finger locations is essential for interpreting gestures and signs accurately in the context of Arabic sign language communication.

A. Arabic hand shapes: Arabic hand shapes are indeed a crucial component of ArSL. They comprise specific hand configurations and motions that express various words, letters, or ideas within the language. These hand shapes are integral to the grammar and lexicon of ArSL, allowing for effective communication within the deaf community in Arab regions [33].

Figure 1-19 *Hand Shapes Used for Arabic Alphabets.[56]*

B. Arabic hand orientations: refers to the specific way that the hands are positioned while using ArSL. It plays a crucial role in conveying meaning and distinguishing between different signs, as the orientation of the hands can significantly alter the interpretation of a sign within the language.

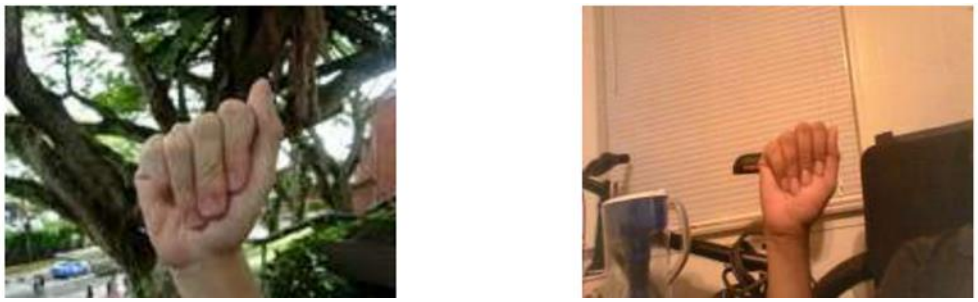
C. Arabic hand locations: Arabic hand locations refer to the specific areas of the body or space where the hands are placed during the formation of signs in ArSL. These locations are important in ArSL as they contribute to the clarity and understanding of the signs being conveyed. The placement can affect the meaning of a sign and is an integral part of the language's structure.

D. **Arabic hand motions:** Hand motions are indeed a vital component of ArSL. They refer to the various movements and gestures made by the hands while signing. These motions can include actions such as tapping, waving, pointing, or combining different movements to create meaning and convey complex concepts within the language. [33]

1.6.3. Sign language translation problems and challenges

A. Uncontrolled Environment

Translation systems often struggle in environments with variable lighting, background noise, and other unpredictable factors that can affect the visibility and clarity of sign language gestures. **Figure 1-20** presents samples of the challenges that can be encountered.



(a) Hand gesture with fair skin-tone and variable back-ground

(b) Hand gesture with fair skin-tone and poor lighting conditions

Figure 1-20 *Example Images of Hand Gestures with Variable Background and Lighting Conditions [13].*

B. Occlusion

Parts of the sign language gestures may be obscured by other body parts or objects, making it difficult for recognition systems to interpret the signs correctly. **Figure 1-21** and **Figure 1-22** present some cases where occlusion can lead to inaccurate detection.

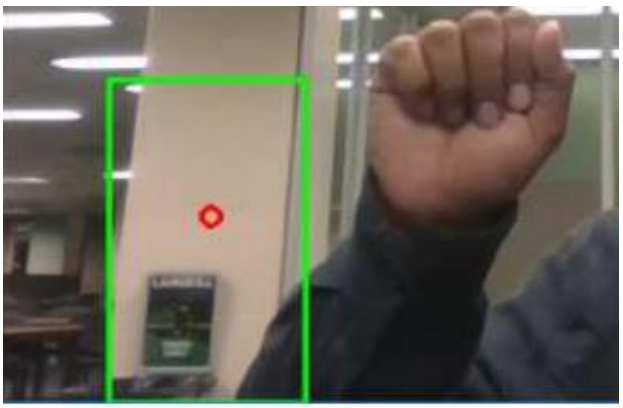


Figure 1-21 *An Example of False Detection of Hand Region in Skin Based Detection Technique [13].*

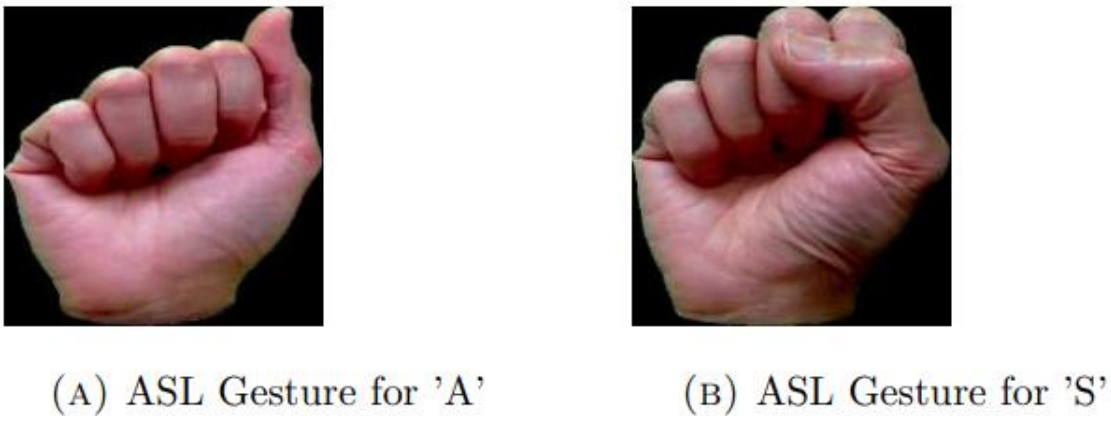
(A) ASL gesture for 'R' (B) ASL gesture for 'D'

Figure 1-22 *An example of Occlusion: R Gesture can Look like D in 2D Projection because of Occlusion [13].*

C. Low Inter-Class Variability

Signs that are similar to each other can be hard to distinguish, leading to errors in translation. This is especially challenging when signs differ only by slight variations in hand

shape or movement. **Figure 1-23** presents an example of low inter-class variability, which can lead to misclassification of hand gestures.



(A) ASL Gesture for 'A' (B) ASL Gesture for 'S'

Figure 1-23 *An example of Low Inter-Class Variability: A gesture can be Misclassified as S because of Low Inter-Class Variability [13].*

D. A. Robust Hand Detection

Accurate detection of hands is critical for sign language recognition, but it can be complicated by factors like hand speed, angle, and overlap with other body parts.

E. Trigger to Recognition Process

Determining the start and end of a sign can be challenging, especially in continuous signing without clear breaks. Systems must be able to identify these cues accurately to ensure proper translation [4].

1.7.Conclusion

In conclusion, the recognition of sign language is an important field that has to be developed and expanded more for the benefit of the individuals with disabilities, which consists of viable strategies for enhancing hearing-impaired or deaf people's communication and increasing inclusivity and accessibility for sign language users. The upcoming chapter will examine current studies aimed at Arabic sign language recognition.

Chapitre 2. State-of-the-Art

2.1 Introduction

The field of sign language recognition is a well-established area of research with a diverse range of implementation methods. In this chapter, we endeavored to find the largest possible number of studies in Arabic language, considering the significant challenges associated with studying this field, especially when focusing solely on the Arabic language. Hereafter, we have gathered the most recent and important works focused on Arabic sign language, irrespective of its dialects.

2.2 Basic methodology

All the works that are presented here are focused on Arabic language generation from variation type of signs which can be extracted from either video or image data. The figure below represents the basic steps of any approach developed under the vision-based method or with sensors (for more details, see Chapitre 1).

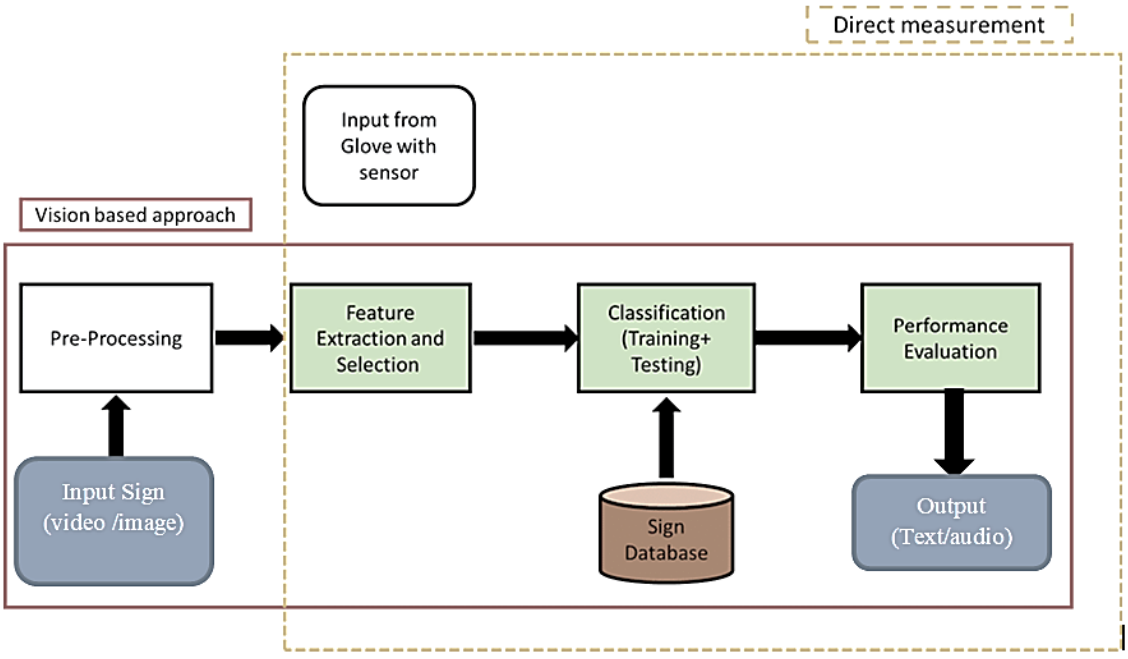


Figure 2-1 *System's general block design for recognizing sign language.[79]*

The presented works have been divided into three main groups: image processing/statistical modeling-based recognition, classic machine learning-based recognition, and deep learning-based recognition.

2.2 Image Processing/ Statistical Modelling based approaches

In these methodologies, traditional image processing algorithms such as rule-based approaches, fuzzy logic-based approaches, and classifiers were employed to analyze and interpret visual data for gesture recognition tasks within this domain.

Omar Al-Jarrah et al. [45] developed a system for recognizing gestures in Arabic sign language using neuro-fuzzy systems. The system aimed to automatically translate the gestures of the manual alphabets into Arabic sign language without the need for gloves or visual markings. The proposed architecture is functionally equivalent to the first-order Sugeno-type fuzzy inference system, where the models were trained to produce a value of 1 as output if the presented data corresponded to the associated gesture and 0 otherwise. The system employed a fuzzy inference system to identify the gestures, and the training was accomplished using a hybrid learning algorithm. The subtractive clustering algorithm and the least-squares estimator were used to identify the fuzzy inference system, making the system more flexible and accurate in recognizing the 30 Arabic manual alphabets. The result of this system is a recognition rate of 93.55% for the 30 Arabic manual alphabets. By enhancing some of the models with 13 rules, they were able to improve the recognition rates for specific gestures such as "sad," "thah," and "he." The recognition rates for these gestures increased to 90.00%, 86.66%, and 95.00%, respectively. Overall, the enhancements made to the models resulted in a slight increase in the overall recognition rate from 92.94% to 93.55%. The limitations of the system include the similarity between gestures and the performance impact of enhancements.

Safaa M. Elatawy et al. [46] proposed a system that utilizes the neutrosophic technique and fuzzy c-means for the detection and recognition of alphabet Arabic sign language. The dataset used in the experiment for the proposed system was provided by the "Al-Amal Institute Damietta for deaf students," containing 300 images of Arabic Sign Language (ArSL) organized into 28 classes. They employed a fuzzy approach in their system through the clustering method based on fuzzy c-means. The fuzzy c-means clustering algorithm helps in accurately categorizing the input images for the recognition of Arabic sign language. The results demonstrated a total classification accuracy of 91%. The system for recognizing Arabic sign language faces several limitations, including the absence of linguistic studies on Arabic sign language, challenges in representing output sign sentences, and a lack of methods to evaluate sign language translation systems. Moreover, the system encounters difficulties due to the absence of grammatical structure and rules in Arabic sign language, making it challenging to construct a sign language corpus. Additionally, there is currently no method to describe the properties of Arabic sign language using text or symbols, such as movement, hand shape location, direction, and non-manual gestures like facial expressions.

Another study developed by A. M. Riad et al. [47] which they utilized color-based for hand localization to account for the significant variations in hand size and shape. The system

incorporated a hand region description algorithm to identify border points in images and extract geometric features. A rule-based classifier was employed in their hand gesture recognition system to classify the specific geometric features extracted. By integrating a vision-based geometric model with the rule-based classifier, the system could recognize static ArSL gestures. It is worth noting that the proposed model may lead to false positives or incorrect identifications of hand gestures. Despite this limitation, the approach enabled accurate classification of hand gestures, resulting in a high recognition rate of approximately 95.3% on the dataset comprising seven ArSL words.

Abdelmoty M. Ahmed et al. [75] engineered a sophisticated system capable of intelligently interpreting isolated dynamic gestures in Arabic Sign Language (ArSL) and converting them into Arabic text. To measure similarity within their system, they employed the weighted Euclidean distance. This metric calculates the distance between feature vectors stored in the database and the feature vector of the current image being matched. The use of the weighted Euclidean distance was crucial for comparing the dataset's stored image classes with the matched image, thereby facilitating the accurate recognition and translation of dynamic isolated ArSL gestures. The research group utilized a dataset that included 100 distinct one- and two-handed dynamic ArSL signs. To create this dataset, they recorded 1500 video files, which were then categorized into five distinct classes. The system demonstrated a high degree of accuracy, achieving a rate of 95.8%. However, a notable challenge was the lack of a standardized dataset for ArSL, prompting the researchers to develop their own dataset for the project.

2.3 Classic Machine Learning based approaches

In these methodologies, traditional machine learning algorithms such as Support Vector Machines (SVM) [8] classifiers and k-Nearest Neighbors (KNN) [9] classifiers were employed. Feature engineering was identified as a crucial aspect of system design in these conventional machine learning-based strategies [11][14]. The discussion briefly outlines a range of feature extraction, classification, and clustering techniques, along with their effectiveness in gesture recognition within this domain.

A. M. Zakariya and colleagues [15] developed an Arabic Sign Language Recognition System on the smartphone platform. The system utilizes a client-server architecture where the smartphone captures sign gestures, sends them to a server for processing and classification, and then displays the predicted sign gesture back to the user. Image processing techniques are employed to detect sign image backgrounds and extract features from frames using binary pixels. For training the Support Vector Machine (SVM) for classification, a dataset containing 200 images for each of the 10 Arabic Sign Language gestures was utilized. While the system demonstrates the capability to recognize 10 Arabic Sign Language gestures with an experimental accuracy result of 92.5%, it has some limitations. These include the system being restricted to recognizing only 10 Arabic Sign Language gestures, and the constraint of smartphones having limited computational power compared to computers, potentially impacting the system's performance. Furthermore, although the system achieved an

experimental accuracy result of 92.5%, there is still room for improvement in terms of both accuracy and speed.

A.M.Ahmed et al. [74] developed the ATASAT System, which stands for Automatic Translation of Arabic Sign Language to Arabic Text. This innovative system is engineered to convert gestures from Arabic Sign Language into written Arabic text by utilizing advanced image and pattern recognition technologies. For the purpose of classification, a variety of algorithms including C4.5, Naïve-Bayesian, K-Nearest Neighbors (K-NN), Multilayer Perceptron (MLP), Sequential Minimal Optimization (SMO), and Voting Feature Intervals (VFI) are implemented to accurately categorize new hand gestures based on their distinctive features. The system processes inputs in the form of videos and captured images, and the output is rendered as Arabic text. During the development phase, the dataset compiled for the system encompassed approximately 700 images for each symbol of the Arabic alphabet, culminating in a comprehensive collection for all 28 characters. This dataset was instrumental in both training and evaluating the performance of the classification algorithms. The deployment of the system yielded successful outcomes, with the precise classification of hand gestures into their respective Arabic letters being a notable achievement. The system demonstrated exceptional proficiency and efficiency in the recognition and translation of sign language gestures into textual form.

R. Alzohairi et al[76]crafted a recognition system for Arabic Sign Language (ArSL) that is image-based and harnesses visual descriptors to identify sign language gestures. These descriptors are extracted directly from the images to effectively represent the gestures. The classification process was conducted using a kernel SVM approach. The experimental dataset comprised 210 gray-level images of ArSL, each depicting one of the thirty characters in the Arabic alphabet. The images were standardized by centering and cropping them to a uniform size of 200×200 pixels. The system's performance was marked by a recognition accuracy of approximately 63.5%. However, the system faced challenges in differentiating between similar gestures, such as "Ra" and "Zay," "Tah" and "Thah," as well as "Dal" and "Thal," which points to potential areas for enhancement in the system's ability to discriminate between gestures.

2.4 Deep learning-based approaches

In the deep learning area, the authors employed different approaches including CNN [10] and RNN, where the feature engineering was highlighted as a critical component in system design within these deep learning-based methodologies. The discussion below briefly covers a range of feature extraction, classification, and clustering techniques, along with their impact on gesture recognition performance in this category.

Al-Hammadi et al. [7] developed a hand gesture recognition system for sign language that utilizes a 3DCNN architecture for spatiotemporal feature learning. This system was designed using two approaches: the first approach involved a 3DCNN extracting features from the entire video sample for classification through a SoftMax layer, while the second approach aimed to enhance the temporal dependency of video frames by training the 3DCNN to extract

features from various regions within the video sample. Additionally, transfer learning was implemented to address the scarcity of large labeled hand gesture datasets. The system's performance was evaluated using three color video gesture datasets, each containing 40, 23, and 10 gesture classes, respectively. The recognition rates achieved were 98.12%, 100%, and 76.67% for the signer-dependent mode, and 84.38%, 34.9%, and 70% for the signer-independent mode. The research paper did not explore or incorporate certain aspects such as limited exploration of traditional computer vision techniques, absence of data augmentation techniques, evaluation on diverse environmental conditions, comparison with state-of-the-art traditional methods, and exploration of transfer learning.

Ali A. Alani et al. [16] developed the ArSL-CNN, a deep learning model based on a CNN specifically tailored for translating Arabic Sign Language (ArSL) gestures. The ArSL-CNN model underwent training and testing using the extensive ArSL2018 dataset [80], comprising 54,049 images of 32 sign language gestures. The system demonstrated impressive training and testing accuracies of 98.80% and 96.59%, respectively. Furthermore, the researchers investigated the impact of imbalanced data on model accuracy and applied resampling techniques, such as the synthetic minority oversampling technique (SMOTE), to enhance the overall performance of the ArSL-CNN model. This led to a notable improvement in the system's test accuracy to 97.29%, showcasing its ability to effectively manage imbalanced data and enhance gesture recognition performance. The limitations of the ArSL-CNN system stem from its dependence on the quality and quantity of training data, potentially leading to challenges in accurately recognizing gestures for classes with fewer samples despite efforts to address imbalanced data using techniques like SMOTE [81]. Furthermore, the system's performance may vary when encountering sign language gestures outside its training dataset. The computational demands of CNNs, especially with large datasets, can restrict scalability on resource-constrained devices. Additionally, the nature of deep learning models like CNNs presents obstacles in understanding decision-making processes, particularly in applications where interpretability is crucial. Environmental factors such as lighting conditions, background noise, and variations in hand gestures can also influence the system's accuracy in gesture recognition.

Rawf et al.[17] employed a novel approach utilizing 2D CNN and transfer learning models to detect and classify Arabic-script-based sign language gestures. The system was trained and evaluated using the ASSL2022 dataset was generated by combining and upgrading two publicly available datasets, the ASL alphabet dataset and ArSL2018[80], which contains labeled Arabic-script pictures for 40 classes of sign language gestures. The results of the study demonstrated close to 100% accuracy in recognizing and translating hand gestures into legible Arabic characters, showcasing the effectiveness of the proposed models. However, limitations of the system include potential challenges related to dataset size and diversity, generalization to new gestures, hardware dependency, and interpretation of complex gestures, language specificity, and usability considerations.

AL Moustafa et al. [18] developed an innovative approach that integrates Mediapipe for hand landmark detection with a CNN model for ArSL. The system achieved a high precision level in recognizing ArSL alphabets, with a validation accuracy of 97.1%. The dataset [82]

used for training consisted of over 7,000 images categorized into 28 different classes of ArSL motions. However, the study noted that certain ArSL characters, such as Beh, Teh, and Teh, exhibited strong similarities in gestures, which could potentially complicate the classification process. Additionally, the system's performance may be influenced by variations in hand gestures and orientations, highlighting a limitation in the system's robustness to diverse signing styles and conditions.

S. Aiouez et al.[19] developed a real-time Arabic sign language hand posture recognition system based on YOLOv5. They constructed a dataset [80] of 28 Arabic signs containing around 15,000 images with variations in hand sizes, lighting conditions, backgrounds, and accessories. The authors then trained and tested different variants of YOLOv5 on this dataset. The adapted YOLOv5 model showed effectiveness in recognizing Arabic sign language, outperforming the Faster R-CNN detector. The system achieved a recognition rate of 93.41% on a dataset of 2323 samples.

Saad Al Ahmadi et al.[72] developed a methodology that leverages CNNs and Transfer Learning to enhance Arabic Sign Language. This methodology involves phases such as data collection, data preprocessing, feature extraction using CNN, and transfer learning-based classification. By utilizing the Arabic Alphabets Sign Language Dataset (ArASL2018) [80], which consists of images representing each sign of the Arabic alphabet, they were able to train a model that achieved a recognition accuracy of 94.46%. Their research contributes to bridging communication gaps for the deaf and hard of hearing community by improving the accuracy and accessibility of sign language interpretation.

Saleh Aly et al.[78] developed a proficient sign language recognition system tailored to identify dynamic isolated gestures. This system tackled three pivotal challenges: hand segmentation, hand shape feature representation, and gesture sequence recognition. For hand segmentation, they utilized Deeplabv3+, an advanced semantic segmentation model. To describe the hand shapes, they employed Convolutional Self-Organizing Map (CSOM), and for recognizing the sequence of gestures, they implemented a p Bi-directional Long Short-Term Memory (BiLSTM) network. The dataset they created included 23 distinct Arabic word signs, each demonstrated by three different individuals. The system’s efficacy was reflected in its high recognition accuracy rate of 89.59%.

Salma Hayani et al. [77] Developed a system that automatically recognizes Arabic Sign Language using Convolutional Neural Networks (CNN). Their goal was to identify the 28 letters of the Arabic alphabet and digits from 0 to 10 through RGB images processed by the CNN. They trained and validated their model using a genuine dataset tailored for Arabic Sign Language. The system they proposed attained a recognition accuracy of 90.02%, showcasing its effectiveness in sign language interpretation.

2.5 Comparative Analysis of Previous Studies

In this part, we detailed the most recent and significant works that based on sign language recognition specifically in Arabic language. **Table 2-1** presents a brief comparison among

these studies where we relied on the dataset used, techniques employed, characteristics of the input and output, and the highest recognition rate achieved. Additionally, it includes information about the authors and the publication year.

Author	Year	Language	Dataset	Approach	Input	Output	Accuracy
Omar Al-Jarrah et al. [45]	2001	Arabic	Manually	Neuro-Fuzzy	Image	Alphabets (text)	93.55%
Safaa M. Elatawy et al. [46]	2020	Arabic	Manually	fuzzy c-means	Image	Alphabets (text)	91%
A. M. Riad et al. [47]	2014	Arabic	Manually	Rule-based	Frame capturing	Words (text)	95.3%
Abdelmoty M. Ahmed et al. [75]	2020	Arabic	Manually	Euclidean distance	Video	Words (text)	95.8%
A. M. Zakariya and colleagues [15]	2019	Arabic	Manually	SVM	Image	Alphabets (text)	92.5
A.M.Ahmed et al .[74]	2016	Arabic	Build two datasets: ASL dictionary and gestures from different human gestures	KNN, MLP, C4.5, VFI and SMO	Image	Alphabets (text)	80.67, 88.66, 90.7%, and 84.4%
R.Alzohairi [76]	2018	Arabic	Manually	SVM	Image	Alphabets (text)	63.5%
Al-Hammadi et al. [7]	2020	Arabic	(KSU_SSL), ArSL, RVL_SLLL	3DCNN	Video	Words (text)	96.69%, 100%, 76.67%
Ali A. Alani et al. [16]	2021	Arabic	ArSL2018	CNN	Images	Alphabets (text)	96.59%
Rawf et al.[17]	2022	Arabic	ASSL2022	2D CNN	Images	Alphabets	99.32%
AL Moustafa et al. [18]	2023	Arabic	ArSL	CNN	Image	Alphabets	97.1%
S. Aiouez et al.[19]	2022	Arabic	ArSL2018	YOLOv5	Image	Alphabets	93.41%
Saad Al Ahmadi et al.[72]	2024	Arabic	ArSL2018	CNN	Image	Alphabets	94.46%
Saleh Aly et al.[78]	2020	Arabic	Manually	BiLSTM	Videos	Words	89.59%
Salma Hayani et al[77].	2019	Arabic	Manually	CNN	Images	Alphabets	90.02%

Table 2-1 Summary of existing work focusing on Arabic sign language.

As seen in the table, the majority of the dataset is manually generated, with each work having its own dataset. This practice leads to inaccurate assessments and less objective evaluations. Additionally, the input type mainly consists of images rather than videos, with

videos being rare in only a few works. Moreover, the outputs mostly represent letters, while only four works focus on word types. Notably, the best-performing work in this area was developed by Al-Hammadi et al. [7], who achieved a 96.69% accuracy using 3DCNN for word generation, and by Rawf et al. [17], who utilized a 2D CNN-based model with a 99.32% accuracy for letter generation. Based on this, we can conclude that the field of Arabic sign language is currently limited to generating alphabets from image inputs, indicating a need for further improvement and development, particularly in the area of word generation.

2.5 Conclusion

In conclusion, the sign language recognition in Arabic language still lacks extensive research, with few works dedicated to this area, where each of these works adopts a different approach, each with its own set of strengths and weaknesses. Deep learning-based approaches such as CNNs, have demonstrated promising results in recent years, achieving high accuracy rates in both detection and recognition tasks. Additionally, techniques such as SVM and KNN have been also giving good results. The evaluation of these works can be also challenging due to the variation of the dataset used which leads to vague comparison.

Chapitre 3. Conception

3.1 Introduction

In this chapter, we propose an approach for Arabic sign language recognition using deep learning techniques. We suggested different solutions utilizing various techniques by generating different scenarios, which can be divided into four propositions: the CNN individually, the LSTM individually, and the combination of CNN and LSTM, which has a different structure than the individual ones. Additionally, we used the YOLOv8 technique for performance comparison, where the goal is to efficiently recognize phrases in Arabic from the provided input. Through meticulous preprocessing steps and the utilization of the ArabSign-A dataset [59], we leveraged the hybrid technique due to its efficiency and accuracy. The chapter outlines a structured approach to capturing and interpreting Arabic sign language gestures. The proposed methodology not only aims to facilitate seamless communication but also underscores the importance of inclusivity and understanding in linguistic interactions.

3.2 Problems and system goals

The essential problems in this study that make it difficult and challenging for achieving high accuracy in recognizing signs in Arabic which can be presented as follows:

1. The limitation of Arabic dataset existence;
2. The limitation of video type input for the sign language system;
3. The limitation of studies that focused on Arabic language which leads to narrow observation and comparison;
4. The complexity of preprocessing and feature extraction from Arabic sign language gestures;
5. The high computational cost associated with training deep learning models on large datasets;
6. The difficulty in capturing subtle variations and nuances in sign language gestures;

Based on these problems, we proposed solutions to address the presented challenges. The primary goal of this project is to create a deep learning system capable of detecting and classifying signs from a given input, as well as accurately recognizing and extracting the content of the detected signs. By leveraging advanced deep learning techniques, the system aims to identify and extract the necessary information, specifically the Arabic words, from the video segments. **Figure 3-1** presents the general architecture of our system.

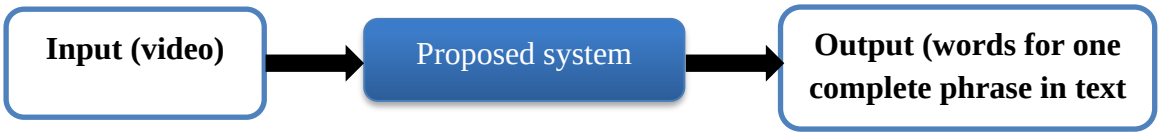


Figure 3-1 *The general form of the proposed architecture of our system.*

3.3 Characteristics of Arabic dataset used

For developing our system using deep learning techniques, we attempted to find the most suitable dataset that meets our needs. However, there was a significant absence of datasets where the input is video and in Arabic language. We found only two datasets where video input was available, and currently, only one of them is accessible. This limited our options to develop our models based on this dataset.

3.3.1. ArabSign-A dataset

In this research, we utilized the ArabSign-A dataset [59], which originally consisted of videos containing 50 sentences in Arabic sign language. The dataset was recorded by 6 signers, all male, with varying skin colors and ages ranging from 21 to 30 years old. Each sentence was performed by each signer at least 30 times during different sessions. Notably, all signers were right-handed, and one of them wore eyeglasses. The data captured by the Kinect V2 camera included three types of information: color, depth, and skeleton joint points. Additionally, the dataset was annotated according to both Arabic sign language and Arabic language structures, facilitating the study of linguistic characteristics in ArSL.

Figure 3-2 *A sample of the dataset[59].*

Each video in this dataset represents a complete sentence without any word segmentation. In the figure below, one sample of the complete sentence is shown, covering the total video duration without taking into account segment variations.

Figure 3-3 *A sample of the complete phases (الله اكبر) that exist in this dataset*

3.3.2. Our used dataset

Basing on the provided dataset (ArabSign-A dataset), we created a new alternative for our system, where we extracted the segments of each video, with each segment representing one word. We manually divided 30 videos of a sentence into segments, resulting in 5 distinct words, each represented by 30 videos. Therefore, rather than working on a full sentence or full video, we focused on individual words in Arabic sign language. **Figure 3-4** presents the new generation of our dataset used.

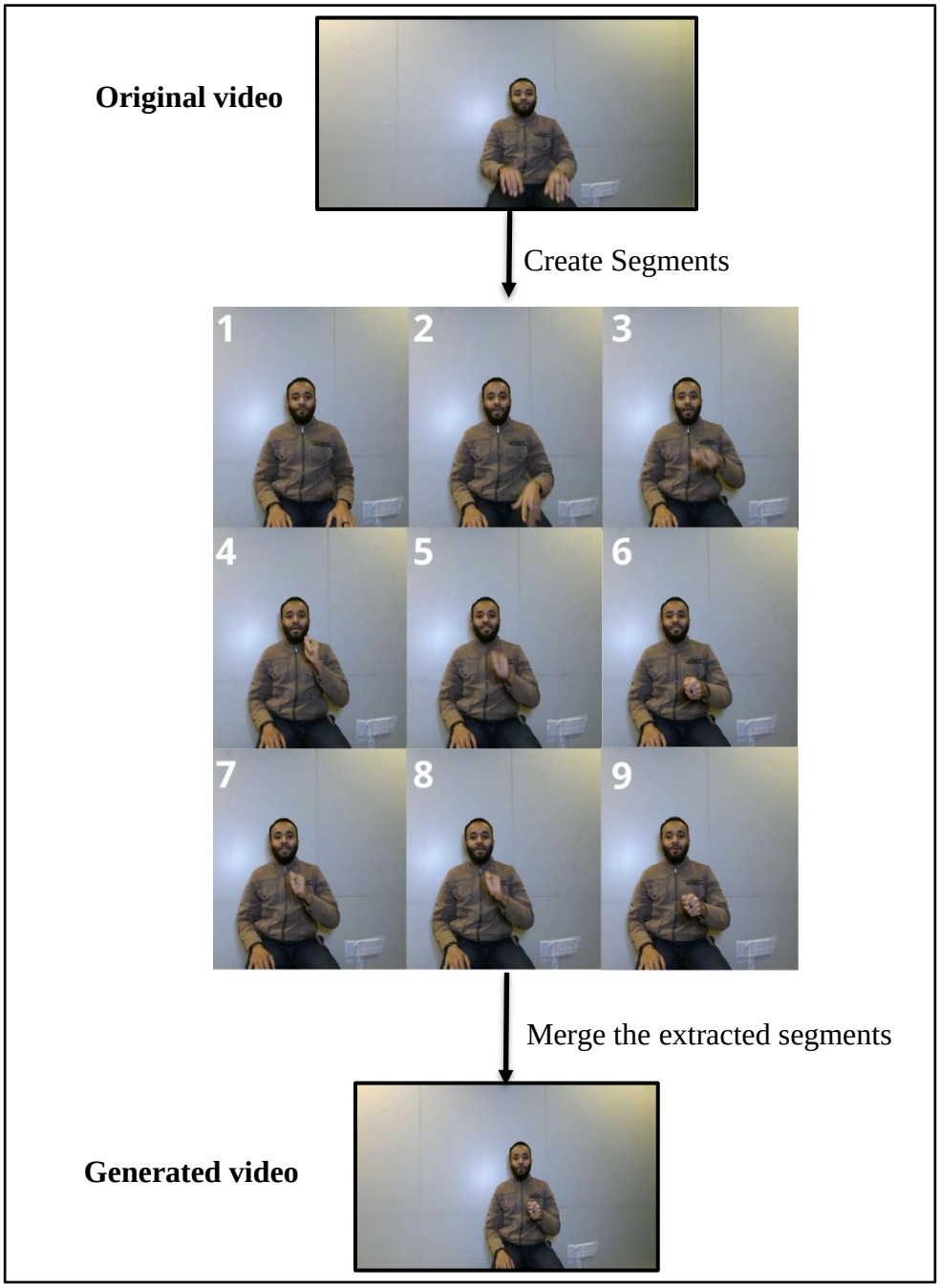


Figure 3-4 *A sample of the new generation dataset based on ArabSign-A dataset.*

3.4 Proposed system architecture

The proposed approach of our system is presents in **Figure 3-5** which illustrates the basic phases of the sing language recognition for translating the provided sign language into Arabic words. The proposed system consists of the following phases:

• **Preprocessing phase:** In this stage, we prepare the video, extract a set of frames from it, normalize it, and optimize its lighting after resizing it

• **Object Localization and Extraction:** This stage complements the previous stage by locating the primary objects for badge recognition, including hands and face.

• **Holistic detection:** After locating the objects in this stage, we detail the extracted objects for more information about the face, hands, and body features.

• **Segmentation phase:** At this stage, the video is segmented into 1-second clips in order to deal with each tag separately

• **Recognition phase:** In this phase, we use deep learning techniques to recognize Arabic sign language using four separate models that we have developed: CNN, LSTM, YOLOv8, and a hybrid CNN-LSTM model.

This systematic approach ensures the accurate identification and translation of sign language into

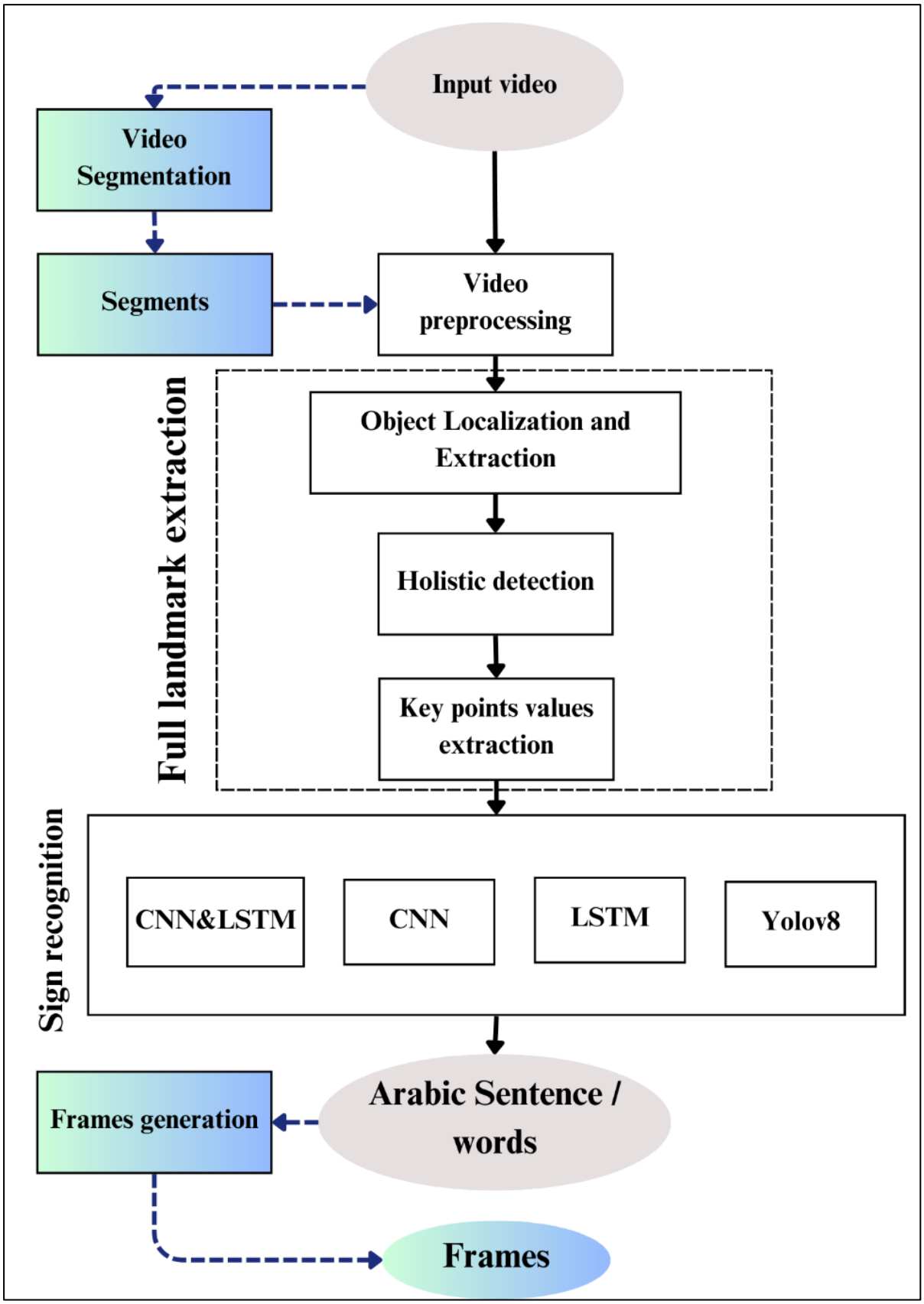


Figure 3-5 *The architecture of the proposed Arabic sign language recognition system.*

3.4.1 Video preprocessing

The first phase in our system is preprocessing the provided video to ensure accurate results in subsequent phases of our proposed system. The following pseudocode represents a generalized preprocessing pipeline of this phase:

```
function preprocessVideo(video):
    frames = extractFrames(video)
    for frame in frames:
        normalized_frame = normalizeImage(frame)
        resized_frame = resizeImage(normalized_frame, width=256, height=256)
        corrected_frame = correctLighting(transformed_frame)
    return concatenated_data
```

A. Frame extraction: Considering that the videos used in our system have duration of 1 second, we conducted numerous tests to determine the appropriate number of frames that can be extracted from each video; we have concluded that 10 frames are the most appropriate for our system, with each frame representing one movement. In **Figure** 3-6, we demonstrate dividing one video into 10 frames and converting each frame to RGB format. Subsequently, the detected human body is processed before pose detection and hand tracking. Following this, various techniques are applied to each generated frame, treating them as images.

Figure 3-6 *Ex example of dividing video into frames.*

B. Normalization: At this level, the images are normalized so that the pixel values are between 0 and 1. This means that the initial pixel values, which range from 0 to 255 (for an 8-bit per channel image), are converted to a range from 0 to 1. We use this process in image processing and machine learning to improve performance and model convergence. **Figure** 3-7presents a sample of the normalization technique.

Figure 3-7 *An example of normalization technique.*

C. **Resizing:** The normalized image is then resized to a specific size of 256 pixels wide by 256 pixels high, which ensures that subsequent algorithms and models can operate on images with consistent dimensions, facilitating reliable and efficient processing.
D. **Lighting Correction:** We also applied the histogram normalization that can correct the lighting variations and improve the visibility of crucial features.

3.4.2 Full landmark extraction

After applying the necessary preprocessing steps, the next phase includes landmark extraction, which is the core of good and accurate recognition. This phase extracts the essential objects for sign recognition, including the hands and the face, which are crucial steps for preparing the images for accurate tracking models. In the following, we detail this phase in detail:

A. Object Localization and Extraction

Here, we apply specific techniques to localize the objects correctly. Even if there are incorrect rotations, these techniques will correct them.

a. **Rotation and Mirroring:** The transformations technique also used in our system including the rotation and the mirroring the objects that can be not in correct orientation for enhancing the model.
b. **Concatenation:** after that change result of pose and hand and face to table is created with the x, y, z coordinates and visibility of each fixture. If no pose marker is detected, an array of zeros the resized frames are further processed to a size of (640, 480) and converted to RGB format.

B. Holistic detection

MediaPipe Holistic is a machine learning model developed by Google that enables simultaneous, integrated detection of facial landmarks, hands and body pose. This model combines several sub-models to provide a complete solution for human motion analysis

This phase followed the previous step which based on detailing the extracted objects for extracting more details which combines pose, face, and hand tracking components to create a comprehensive set of landmarks for the human body. In this phase, we utilized a machine

learning holistic model on a continuous stream of images, for producing a total of 543 landmarks (33 pose landmarks, 468 face landmarks, and 21 hand landmarks per hand. This model was trained on around 30,000 real images, as well as on several synthetic hand models displayed on different backgrounds, which there are two types of landmarks.

a) Pose landmark detection

First, the pose of the attached image is estimated and then the overall pose of the object is determined using the CNN model to detect the pose parameters of the object and its parts, which is estimated to be 33 parameters. **Figure 3-8** shows the model used in our case for the whole body.

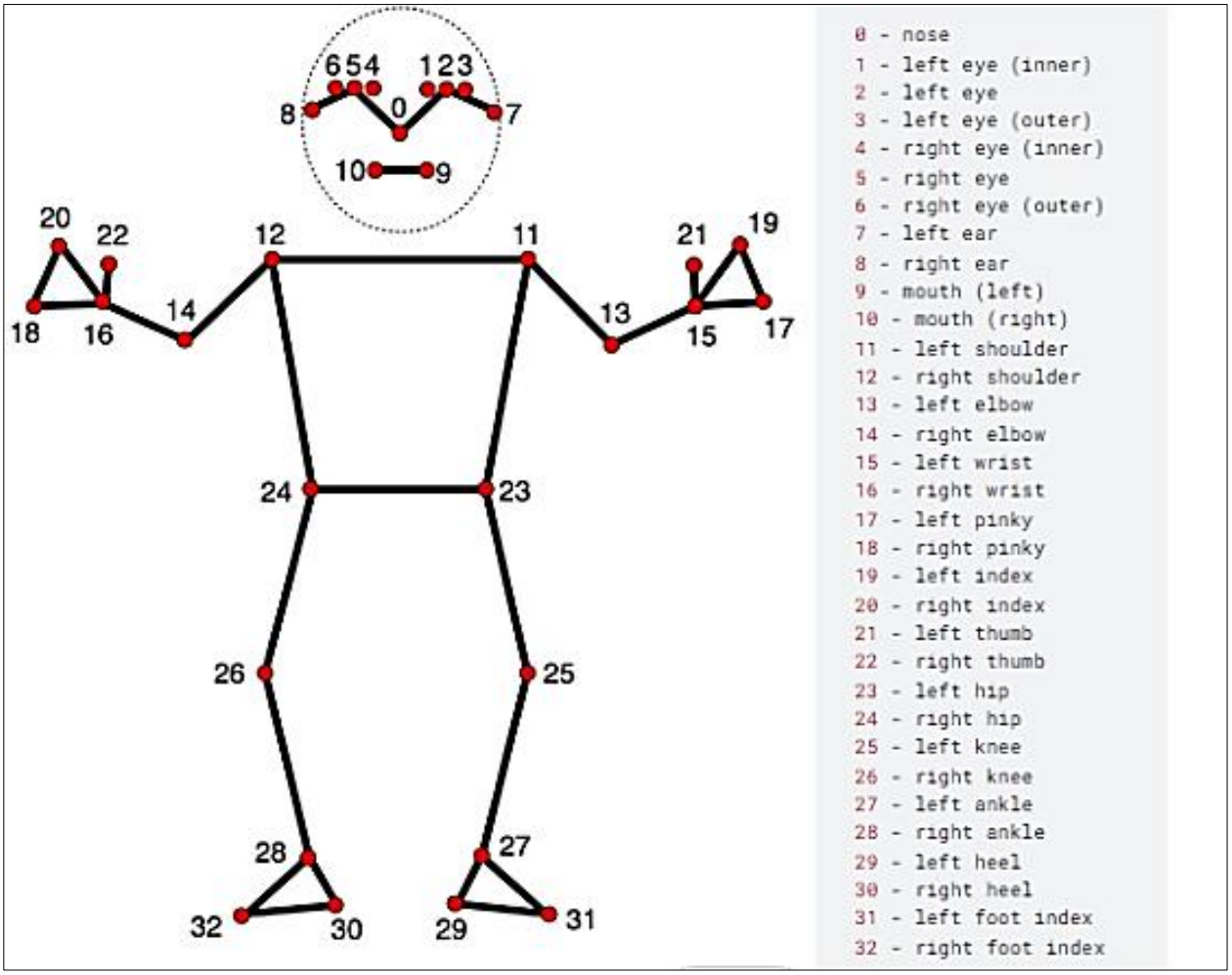


Figure 3-8 *The existing landmarks that can exist in the whole body.*

b) Hand landmark detection

After determining the overall pose of the object, we use its landmarks to determine the palm regions. There are 20 landmarks as shown in **Figure 3-9**

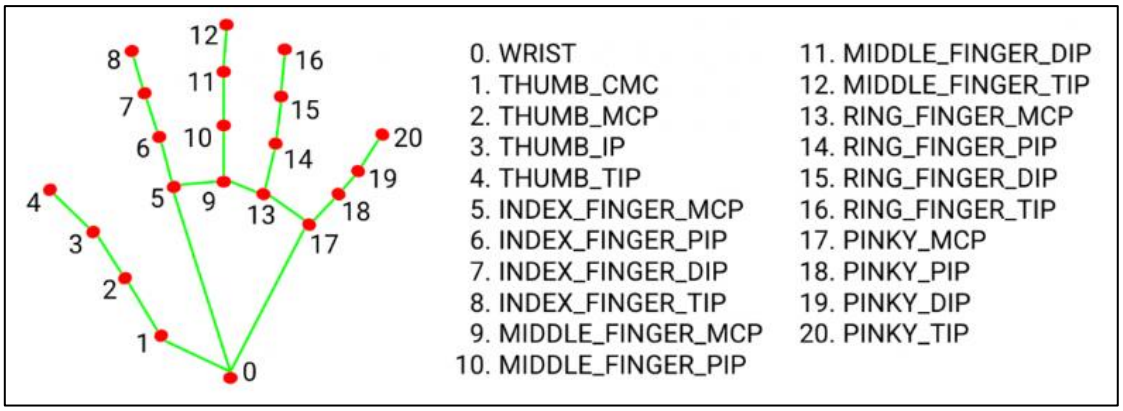


Figure 3-9 *The existing landmarks that can exist in one hand.*

c) Face landmark detection

The last step after hand and pose detection is facial contouring and refinement. 468 points are used to draw the detailed facial features.

C. Key points values extraction

After extracting the landmarks for the entire body, including the face and hands, we extract and organize the key points from the pose, face, and hand detection results. The x, y, and z coordinates of each landmark are extracted and represented in a 1D vector (pose, right-hand, left-hand, face), where the length of the vector is:

- Pose: NumPy array and flattened into a 1D vector of 33*4 elements;
- Hand: where each hand has NumPy array and flattened into a 1D vector of 21*3 elements;
- Face: NumPy array and flattened into a 1D vector of 468*3 elements;

The output is a large vector with a size of 1662 elements. **Figure 3-10** presents the final result of the object localization and extraction phase.

Figure 3-10 *An example of landmark body building extraction.*

3.4.3 Sign recognition

After extracting the vector of key points, we use deep learning techniques to interpret the significance of these points. We developed four separate models for this purpose: CNN, LSTM, YOLOv8, and a hybrid CNN-LSTM model to determine the best technique for ASL. In the following sections, we provide a detailed explanation of each component of these models.

A. CNN model

The proposed CNN architecture that we used in our study is shown in the following figure.

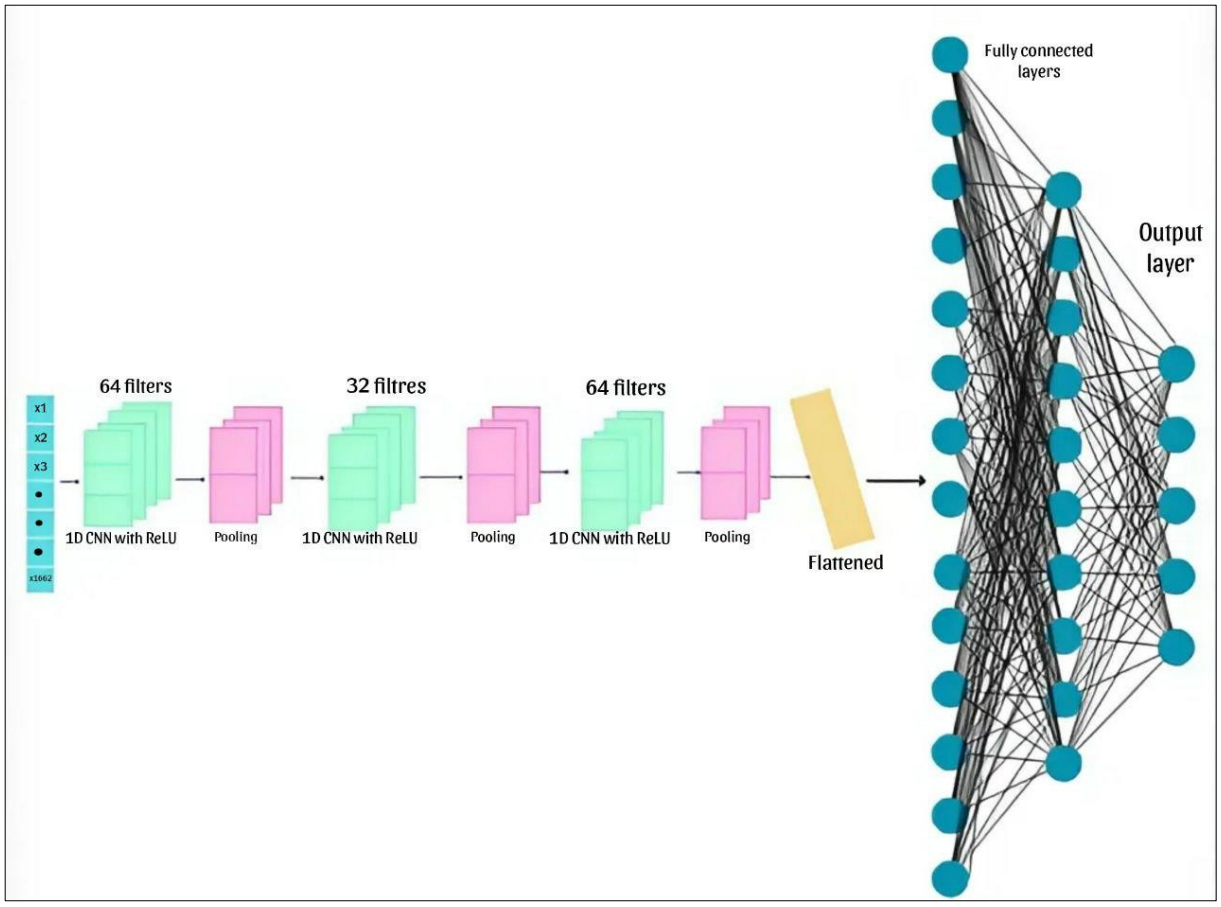


Figure 3-11 *The proposed architecture of the CNN model.*

In the following, the details of the provided architecture:

a. **Convolutional layers:** The model consists of three 1D convolutional layers. The first layer has 64 filters, the second layer has 32 filters, and the third layer has 64 filters, each of size 3. These filters are applied to the input vector, and a modified linear unit activation function (ReLU) is used to introduce nonlinearity to learn complex patterns. After each convolutional layer, a maximum clustering layer with a clustering size of 2 is applied. This layer reduces the size of the sequence by selecting the maximum value from every two consecutive values, which aids in feature capture and reduces computational complexity.
b. **Flatten Layer:** The Flatten layer converts the multidimensional data into a one-

dimensional vector, preparing the data for connection to Dense layers.

c. **Dense Layers:** The model includes two Dense layers for classification. The first Dense layer has 64 units with ReLU activation, and the second Dense layer has 32 units with ReLU activation. The final Dense layer consists of 5 units with a softmax activation function, suitable for multiclass classification tasks such as recognizing different actions.
d. **Model Compilation:** The model is compiled using the Adam optimizer, a popular choice for weight adjustment in neural networks.

B. LSTM model

The proposed LSTM architecture that we used in our study is shown in the following figure.

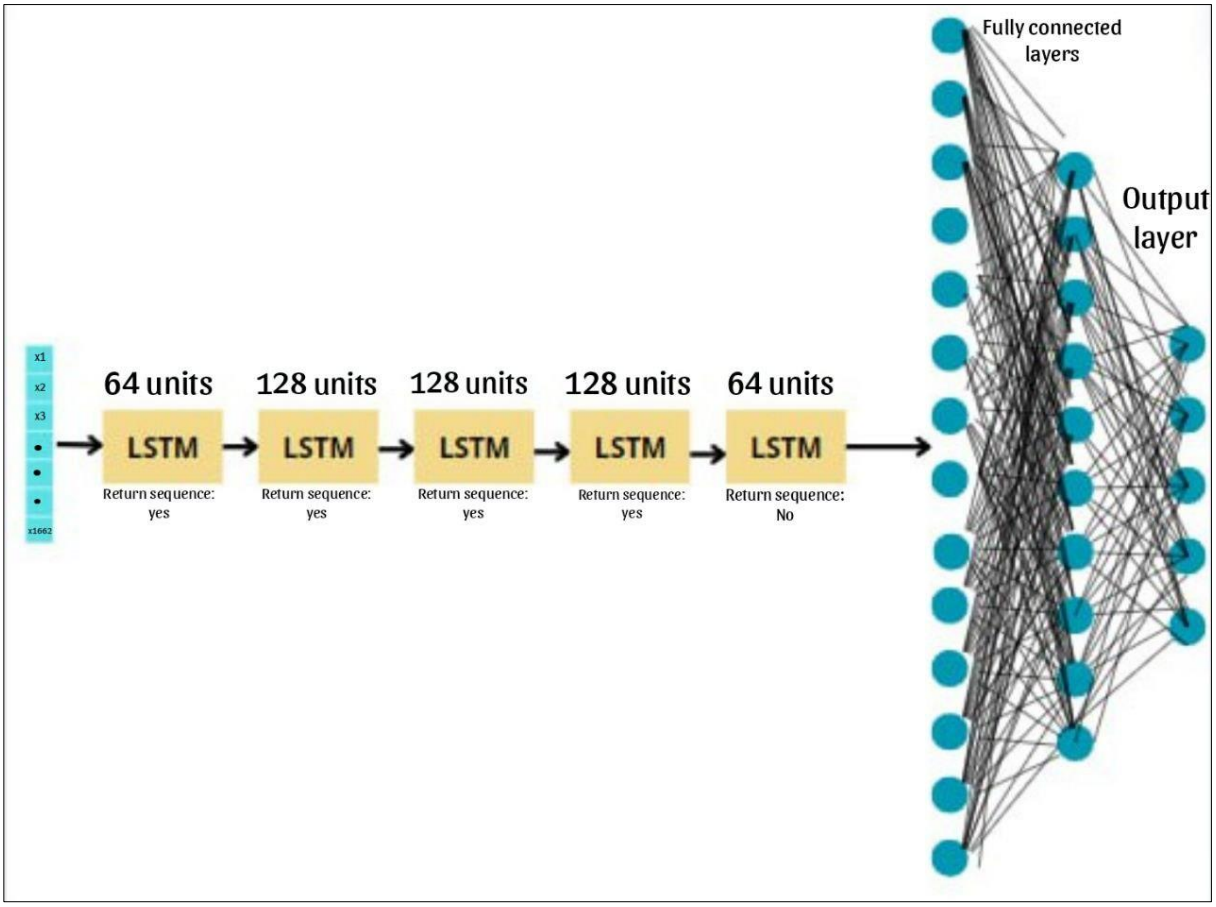


Figure 3-12 *The proposed architecture of the LSTM model.*

In the following, the details of the provided architecture:

a. **LSTM Layers:** The model consists of 5 LSTM layers. The first layer contains 64 units with a return sequence, the next three layers (second, third, and fourth) contain 128 units with a return sequence, and the fifth layer contains 64 units without a return sequence.
b. **Dense Layers:** The model includes two Dense layers for classification. The first Dense layer has 64 units with ReLU activation, and the second Dense layer has 32 units with ReLU activation. The final Dense layer consists of 5 units with a softmax activation function, suitable for multiclass classification tasks such as recognizing different actions.

c. Model Compilation: The model is compiled using the Adam optimizer, a popular choice for weight adjustment in neural networks.

C. YOLOv8 model

We utilize the YOLOv8 model for comparison, whose architecture is depicted in **Figure 3-13**

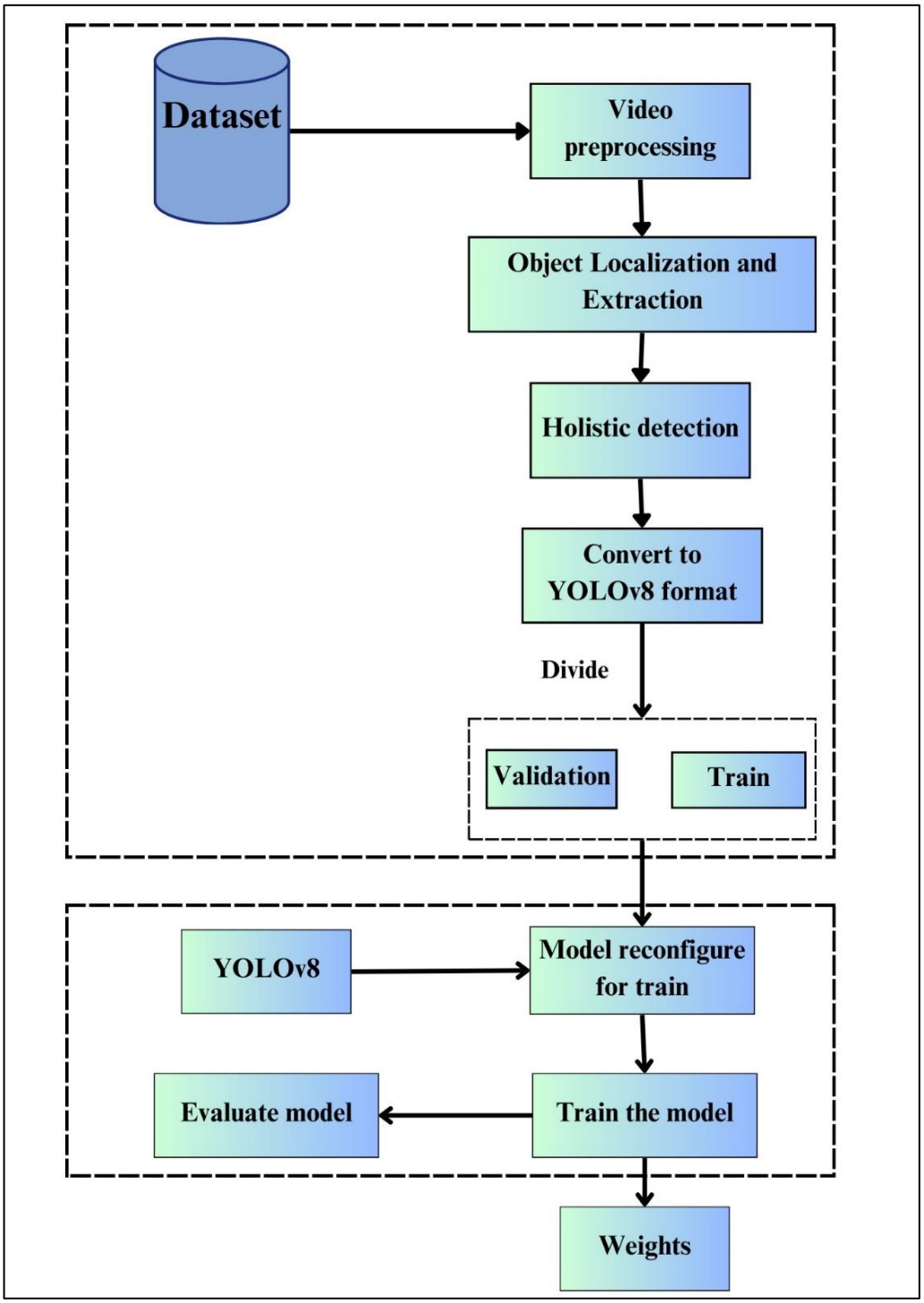


Figure 3-13 *Phases of YOLOv8 Model.*

We utilized the same dataset (ArabSign-A dataset), dividing it into training and validation sets. Next, the video dataset was converted into sequential images, each accompanied by a text file in a label folder for streamlined training. However, the model's performance gave poor results. Therefore, we used the holistic detection technique before dataset transformation. In the following the basic steps of YOLO model:

a. **Creating data file:** based on YOLOv8, we create data.yaml file which configures the training of an object detection model with YOLOv8. It specifies paths to the directories containing training and validation images. It defines 5 classes to detect, specifically ArSL, providing essential details for setting up the training process, including class names and counts.
a. **Training the model:** Images are resized to 416 x 416 pixels, and a batch size of 100 is used for efficient processing which processes 100 images together. Training occurs over 100 epochs to optimize performance on the ArSL task, utilizing the configuration from the data.yaml file.
b. Model Evaluation: The YOLOv8 architecture is utilized for evaluation using the model.val() script. This script assesses the model's ability to recognize ArSLR effectively. Evaluation employs the best-performing model weights specified by the –weights parameter, referencing the best.pt file. Configuration details, such as dataset and class information, are defined in the data.yaml file. Images are resized to 416x416 pixels during evaluation using the –img parameter to maintain consistency. The –task test argument ensures evaluation on the test dataset, providing reliable insights into the model's generalization capability to unseen sign images. This process evaluates the model's accuracy and potential practical applicability.

D. The hybrid model (CNN & LSTM)

The hybrid model was the most efficient one that gave the most accurate results; **Figure 3-14** presents the architecture of our hybrid model, which leverages the strengths of CNNs for feature extraction and LSTMs for capturing temporal dependencies, making it well-suited for tasks requiring both spatial and sequential information processing.

In the following, the details of the proposed architecture:

a. Convolutional Layers: The model consists of two 1D convolutional layers. The first layer has 64 filters, and the second layer has 128 filters, both of size 3. These filters are applied to the input vector, and the Rectified Linear Unit (ReLU) activation function is used to introduce non-linearity for learning complex patterns. Following each convolutional layer, a max-pooling layer with a pool size of 2 is applied. This layer reduces the sequence size by selecting the maximum value from every 2 consecutive values, aiding in feature capture and computational complexity reduction.
b. LSTM Layers: After the convolutional and max-pooling layers, three LSTM layers are included to capture temporal dependencies within the sequences. The first LSTM layer has 64 units and returns full sequences for connection to subsequent layers. The second LSTM layer also returns sequences, while the third LSTM layer outputs only the final output (not sequences).

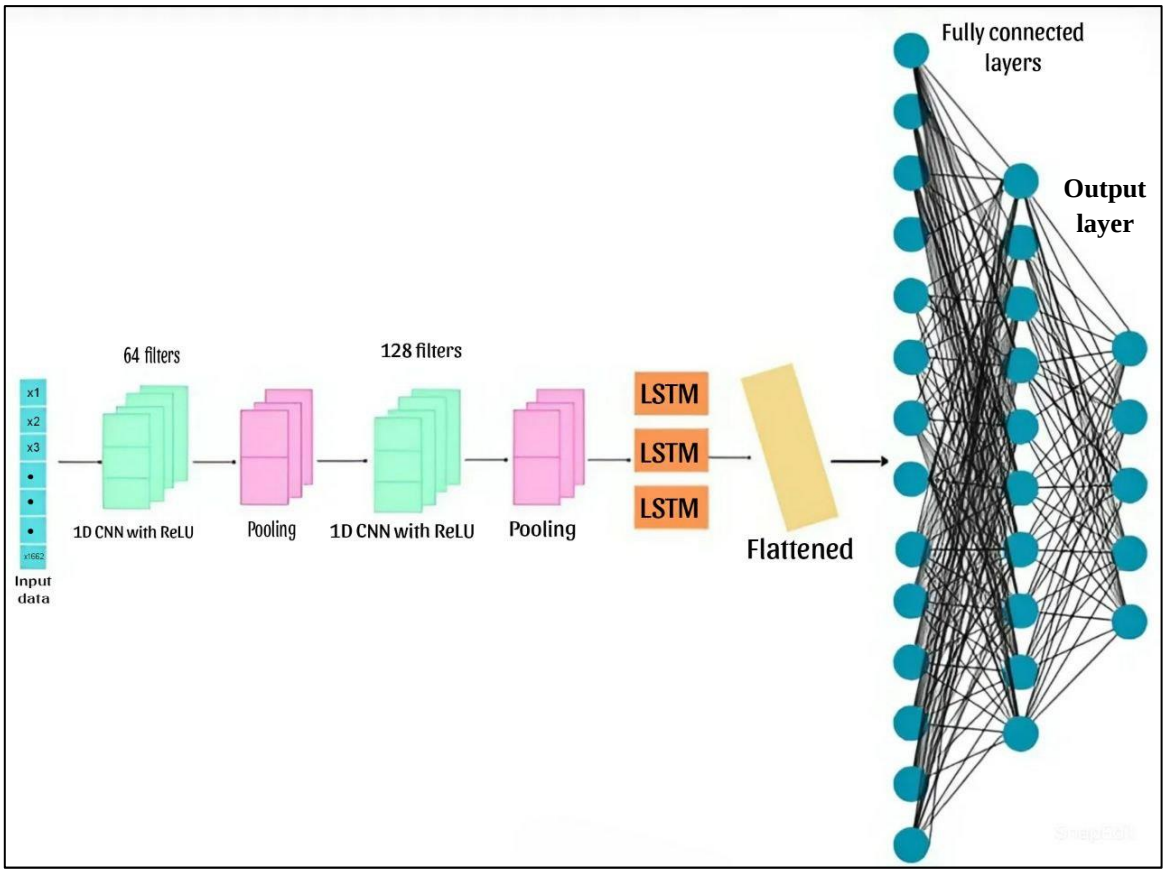


Figure 3-14 *The proposed architecture of the hybrid model.*

c. **Flatten Layer:** The Flatten layer converts the multidimensional data into a one-dimensional vector, preparing the data for connection to Dense layers.
d. **Dense Layers:** The model includes two Dense layers for classification. The first Dense layer has 64 units with ReLU activation, and the second Dense layer has 32 units with ReLU activation. The final Dense layer consists of 5 units with a softmax activation function, suitable for multiclass classification tasks such as recognizing different actions.
e. **Model Compilation:** The model is compiled using the Adam optimizer, a popular choice for weight adjustment in neural networks.

3.4.4 Word generation

The result of the model will consist of each gesture in a video, with the word in the Arabic language. When the gesture changes, the word will change accordingly based on the gesture.

Figure 3-15 *An example of Arabic word generating based on the recognition phase.*

3.4.5 Video Segmentation

This stage involves segmenting the video based on the detected tags. For example, if the video contains four tags (equivalent to four words), it will be divided into four segments of 1 second each. Each segment is treated as a single video, with the distinction that each segment represents only one word, whereas the original video may contain multiple words. **Figure 3-16** shows that a single video can contain more than one word (one segment).

A. Segment: Each segment is considered as an input video where we can perform the same steps of detection, prediction, and output of Arabic text. By dividing the entire video into segments, with each segment being 1 second long, we ensure that each segment contains only one word.

B. Frames: Each segment is divided into frames, and each gesture frame will undergo the same steps of detection, prediction, and output of Arabic text on the video.

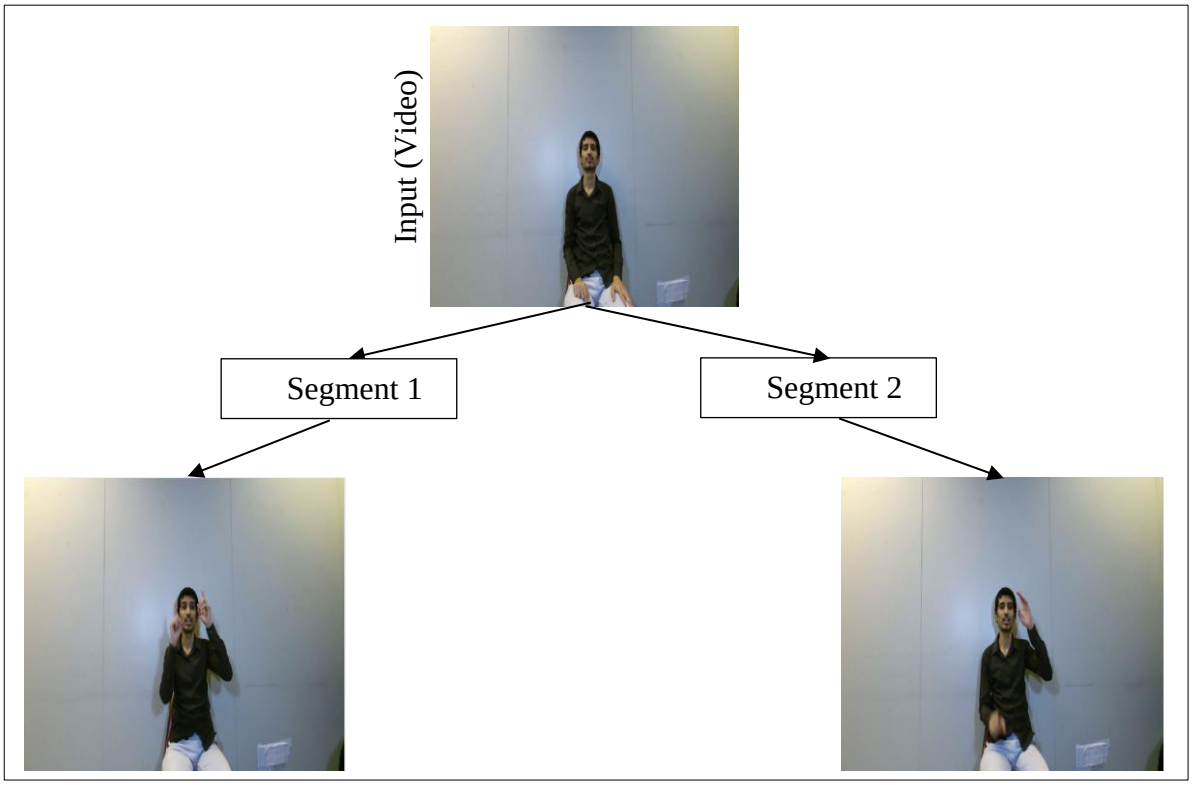


Figure 3-16 *An example 'شكرا لكم' of segment generation.*

3.5 Conclusion

In conclusion, after presenting the chapter on Conception, we present the global architecture that constitutes the building blocks of our project. In this chapter we have provided a brief overview of the most important steps involved in our work, providing a clear overview of the process. In the next chapter Implementation Details we will present the architectural framework into practical solutions.

Chapitre 4. Implementation

4.1 Introduction

In the previous chapter, we focused into the detailed conception of our system. In this chapter, our attention shifts towards the development environment and the libraries utilized in our system. Furthermore, we will offer insights into the essential components of our code and showcase the results we have achieved.

4.2 Development environment

To implement our application, we utilized a personal computer with the following specifications:

Model Part	Used Laptop
Processor	AMD QC-4000 CPU @ 1.30 GHz
RAM	8,00 Go
System type	64-bit operating system, x64 processor
Edition	Windows 10 Famille

Table 4-1 Characteristics of the material used.

4.2.1 Programming language

In this study, we used the Python language, which is detailed as follows:

A. Python

Python is renowned as a high-level programming language that prioritizes code readability and simplicity. Its hallmark lies in its clear and succinct syntax, enabling developers to articulate ideas using fewer lines of code compared to alternative programming languages. Python accommodates diverse programming paradigms, encompassing procedural, object-oriented, and functional programming styles. Boasting an extensive standard library and a robust ecosystem of third-party packages, Python proves versatile for a broad spectrum of applications.[57]

B. Jupyter notebook

Jupyter Notebook stands as a freely available web application enabling the creation and dissemination of documents featuring live code, equations, visualizations, and explanatory text. Compatible with multiple programming languages like Python, R, and Julia, Jupyter Notebook furnishes an interactive computational platform facilitating the structured and systematic writing and execution of code.[58]

4.2.2 Libraries

In this study, we used several libraries, which are detailed as follows:

A. Tensorflow

TensorFlow is an open-source library that provides tools for machine learning and deep learning. It allows developers to easily create complex models, with a focus on training and inference for deep neural networks, including acquiring data, training models, making predictions, and refining future results. It is widely used in applications such as image recognition, natural language processing, and more.[62]

B. Keras

Keras is a high-level neural networks API, written in Python. It is designed for fast experimentation with deep neural networks, and it focuses on being user-friendly, modular, and extensible. It's particularly useful for researchers and developers in the field of deep learning.[61]

C. Mediapipe

MediaPipe is a cross-platform framework designed for building applied multimedia machine learning pipelines. It allows developers to create complex systems for processing time-series data such as video and audio with a focus on real-time applications, and is widely used for tasks such as hand and face detection, object tracking, and more.[63]

D. OpenCv

OpenCV, which stands for Open-Source Computer Vision Library, is a comprehensive open-source library that provides a multitude of computer vision and image processing algorithms. OpenCV is well-documented and supports various programming languages, including Python, which makes it accessible for a wide range of projects.[64]

E. NumPy

NumPy, short for Numerical Python, is an open-source Python library that's essential for scientific computing. It provides support for large, multi-dimensional arrays and matrices, along with a collection of mathematical functions to operate on these arrays efficiently. NumPy is the foundational package for many other scientific libraries makes it a popular choice for tasks such as data analysis, machine learning, and simulation.[65]

F. Tkinter

Tkinter is the standard Python interface to the Tcl/Tk GUI toolkit. It's a thin object-oriented layer that allows Python developers to create simple and effective graphical user interfaces (GUIs). With Tkinter, you can create windows, dialogs, buttons, labels, and other widgets in a platform-independent manner.[66]

G. PIL (Python Imaging Library)

PIL or the Python Imaging Library enhances your Python interpreter with the ability to process images. The fundamental part of this library is optimized for quick retrieval of data in several essential pixel formats, serving as a reliable base for any general image processing utility.[67]

H. Moviepy

MoviePy is a Python library for video editing, enabling tasks like cutting, concatenations, title insertions, video compositing (also known as non-linear editing), video processing, and the creation of custom effects. It supports reading and writing most common video formats, including GIFs, and is designed for both basic operations and creating advanced effects.[70]

I. Bidi

The bidi library is a Python package that implements the bidirectional (BiDi) layout algorithm. This is essential for handling text in languages written from right to left, like Arabic and Hebrew, ensuring that the characters are displayed in the correct sequence and direction.[69]

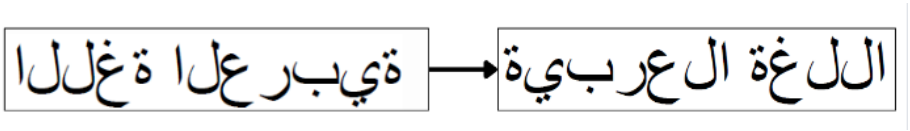


Figure 4-1 *Example of used of python-bidi for the correct order from right to left.[68]*

J. Arabic reshaper

The Arabic Reshaper library is a Python tool that adjusts Arabic text to display correctly in applications that don't natively support Arabic script. It ensures that characters are shown in their proper forms and that text flows from right to left as required by the Arabic language.[68]

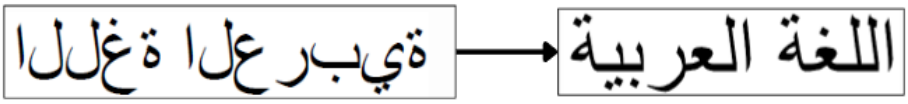


Figure 4-2 *Example of Arabic reshaper works.[68]*

K. Os

The os library in Python is a versatile tool that provides a wide range of functions for interacting with the operating system. It allows you to manage files, directories, and paths, and to access environment variables and system information, making it a fundamental part of Python for system-related operations.[71]

4.3 System overview

We dedicate this section to show the interfaces of our system and the function of each of them, as our system has three different interfaces, including the frontend and the main interface, with a special focus on the main modules. The figure below shows the front end of our application.

Figure 4-3 *Home page of our system*

After clicking on the Start button, the basic interface shown in the image below appears **Figure 4-4**:

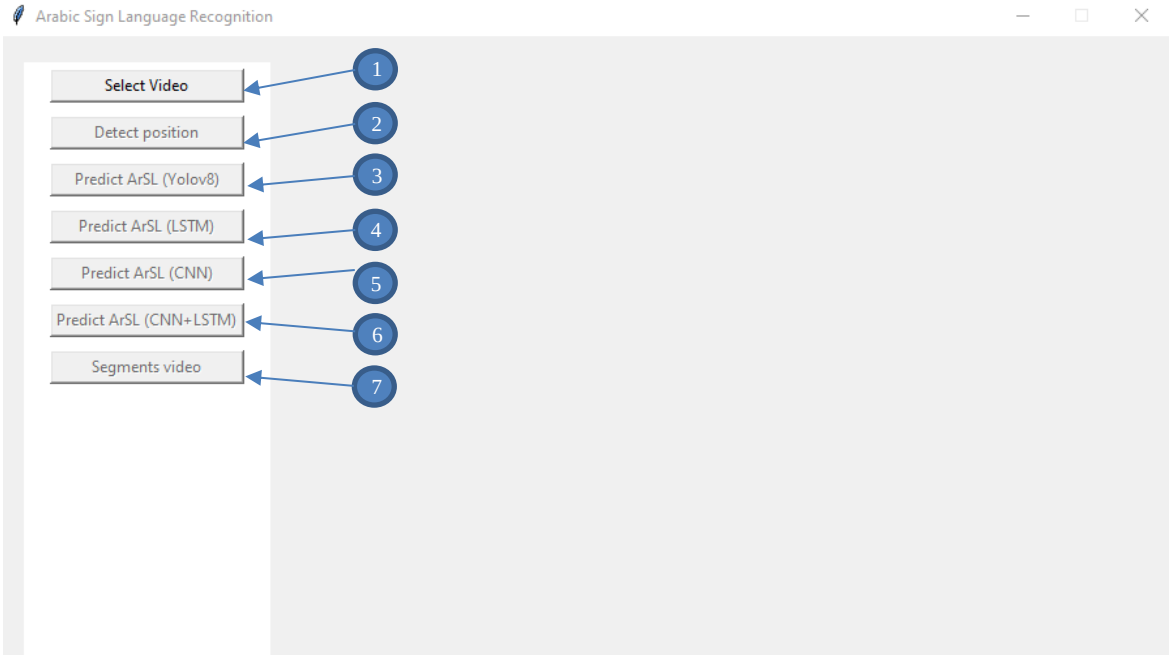


Figure 4-4 *Basic interface of our system*

The main modules of our application are summarized by the following numbers:

1. The initial button enables the user to select and open video.
2. The second button is designed to perform the task of human position detection and subsequently draw styled landmarks of face hand and pose.
3. The third button is responsible for predict of Arabic sign language (ArSL) using YOLOv8 model.
4. The fourth button is responsible for predict of Arabic sign language (ArSL) using LSTM model.
5. The fifth button is utilized for predict of Arabic sign language (ArSL) using CNN model.
6. The sixth button is responsible for predict of Arabic sign language (ArSL) using CNN&LSTM model.
7. The seventh button is for dividing the video for segments.

4.4 Usage scenario

In this section, we will provide an overview of the working principles of our system. We will describe the different stages involved in Arabic sign language recognition:

To commence the Arabic sign language recognition process, we start by opening a video file using the command "Select Video" This allows us to access the desired video and utilize it for further analysis and detection procedures. As shown in **Figure 4-5**.

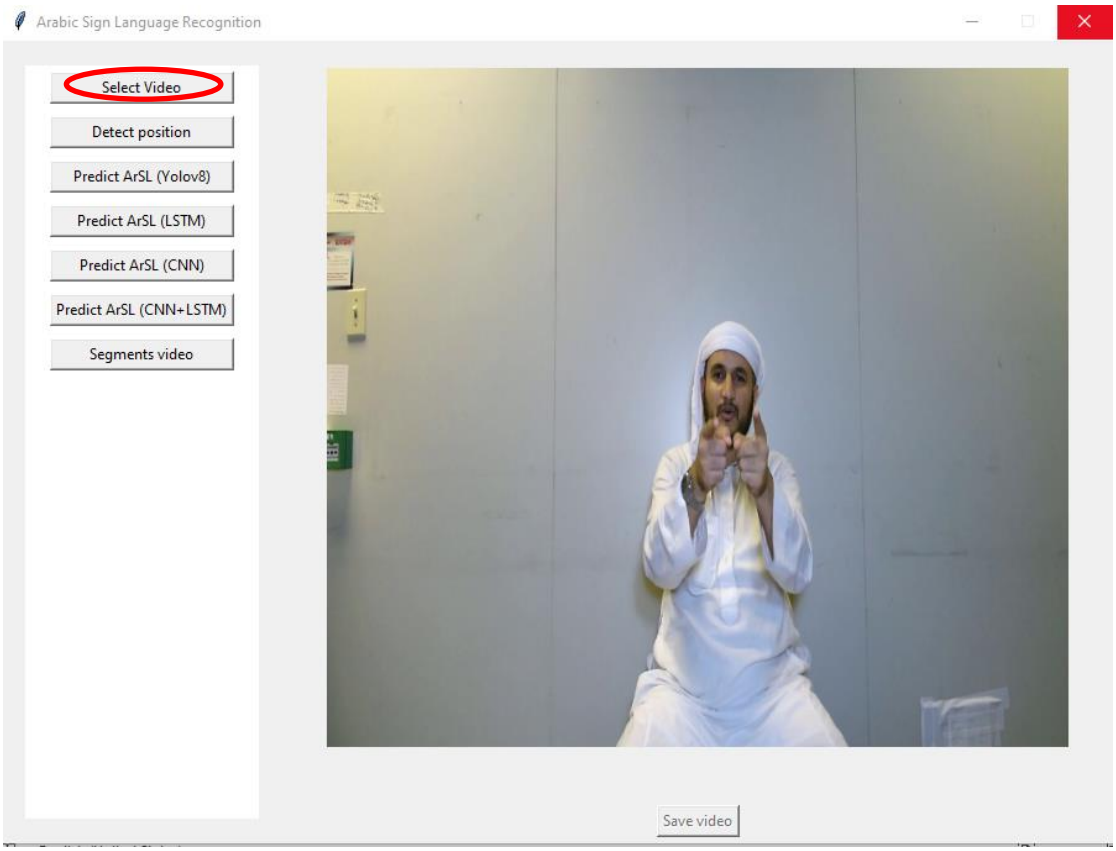


Figure 4-5 *Select video.*

Once the video is successfully opened. By selecting the "detect position" button, we initiate the process of detecting the face, hand, pose of human within the opened video. The of “Mediapipe Library”, which has been previously trained on a large dataset, enables accurate and efficient of human position detection by leveraging advanced object detection techniques and if the user want to saved video with detection, they clicked in button save the video saved in folder ‘Detection’. The result is shown in **Figure 4-6**.

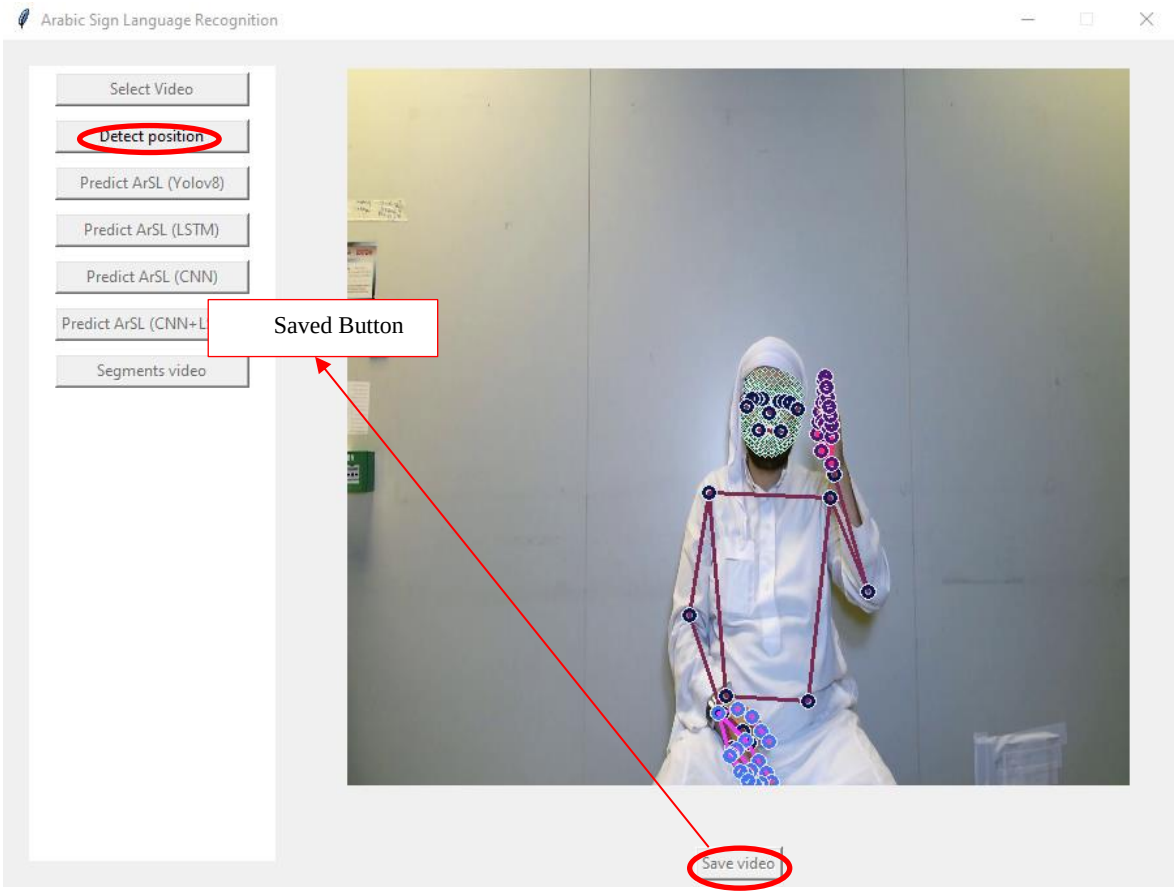


Figure 4-6 *Detect position.*

The "Predict ArSL (YOLOv8)" button predicts the labeling of Arabic words using the trained YOLOv8 model. It splits the video into sequence of images and then the model predicts the class probabilities for each image, and then the result appears on the video in the form of words as shown in **Figure 4-7**.

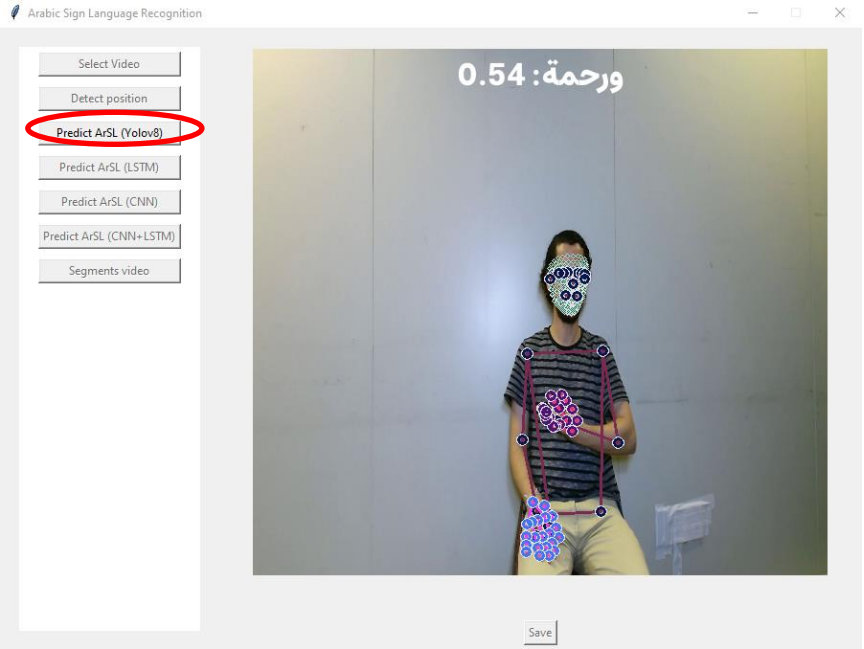


Figure 4-7 *Predict ArSL (YOLOv8)*

The "Predict ArSL (LSTM)" button predicts the labeling of Arabic words using the trained Long Short-Term Memory (LSTM) model. It splits the video into a sequence of images after that extract key points of hand and face and pose and then the model predicts the class probabilities for each key points, and then the result appears on the video in the form of sequential words as shown in **Figure 4-8**.

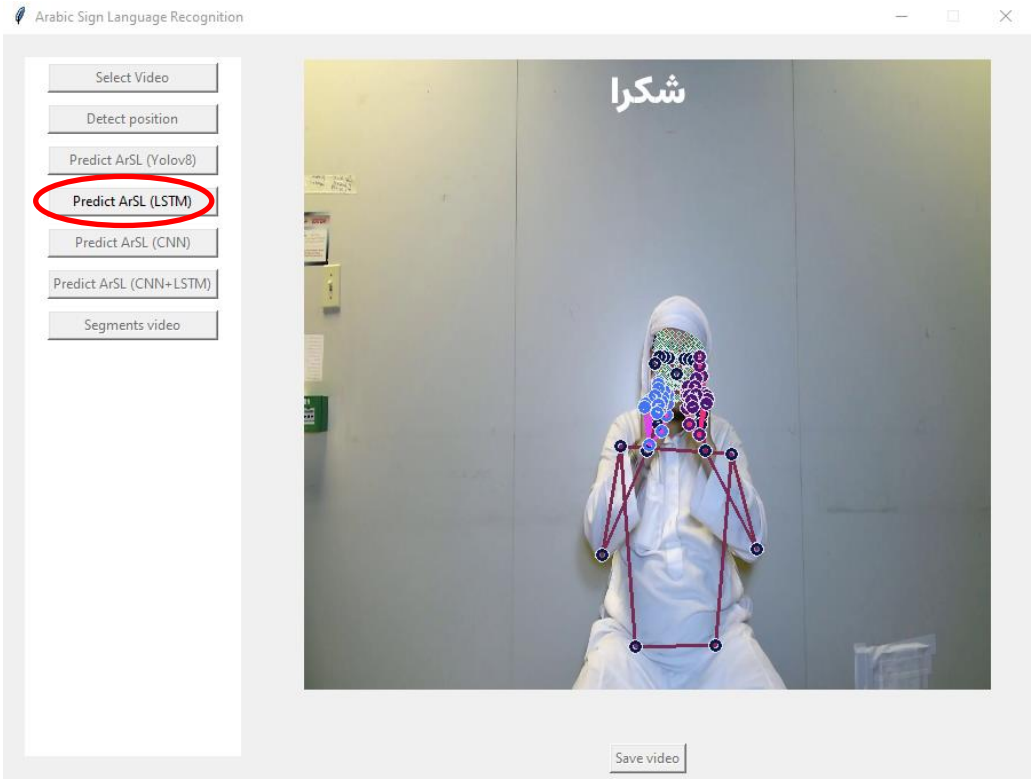


Figure 4-8 *Predict ArSL (LSTM)*

The "Predict ArSL (CNN)" button predicts the labeling of Arabic words using the trained Convolutional Neural Networks (CNN) model. It splits the video into a sequence of images after that extract key points of hand and face and pose and then the model predicts the class probabilities for each key points, and then the result appears on the video in the form of sequential words as shown in **Figure 4-9**.

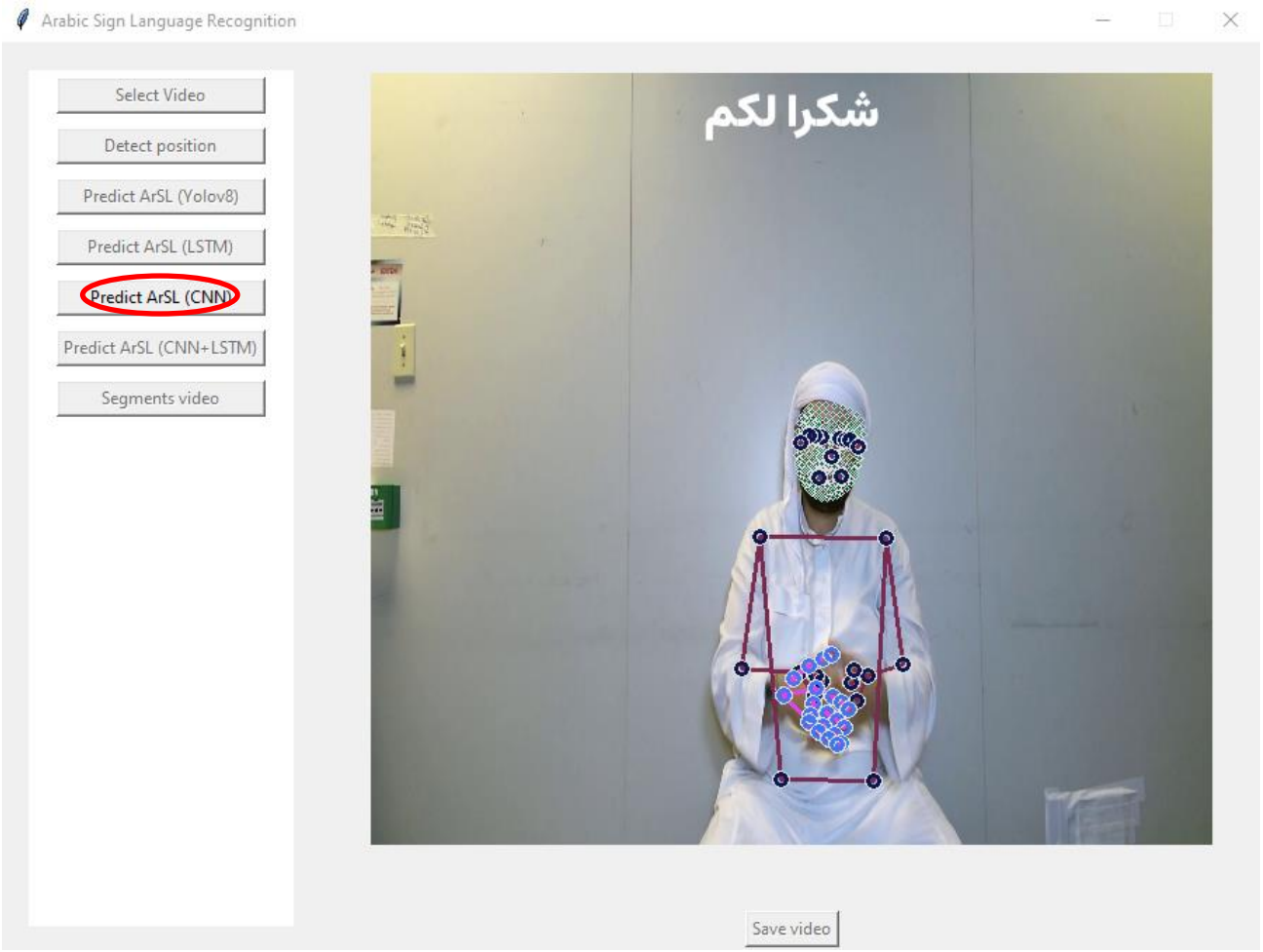


Figure 4-9 *Predict ArSL (CNN)*

The "Predict ArSL (CNN+LSTM)" button predicts the labeling of Arabic words using the trained hybrid between Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM) model. It splits the video into a sequence of images after that extract key points of hand and face and pose and then the model predicts the class probabilities for each key points, and then the result appears on the video in the form of sequential words as shown in **Figure 4-10**

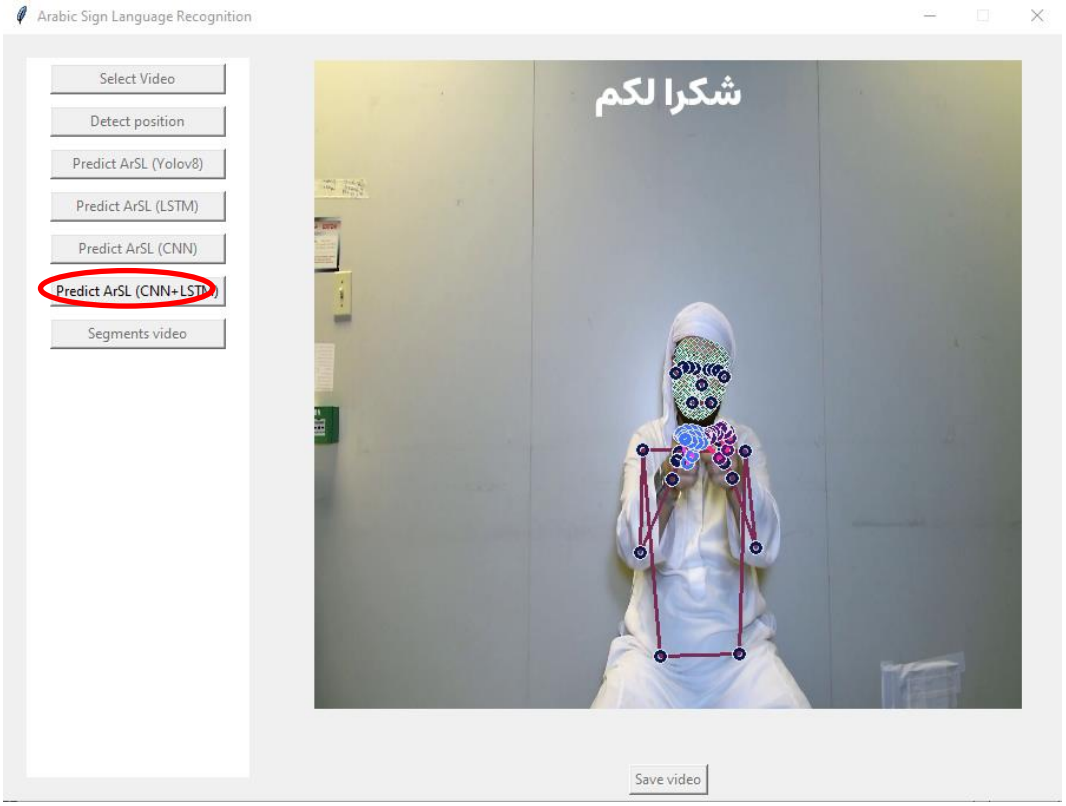


Figure 4-10 *Predict ArSL (CNN+LSTM).*

If the user wants to know the tag for each word individually, we created a video clip button that cuts the video into a series of 1-second videos, so that each part of the video contains only one tag. **Figure 4-11** and **Figure 4-12** show the basic steps for this process.

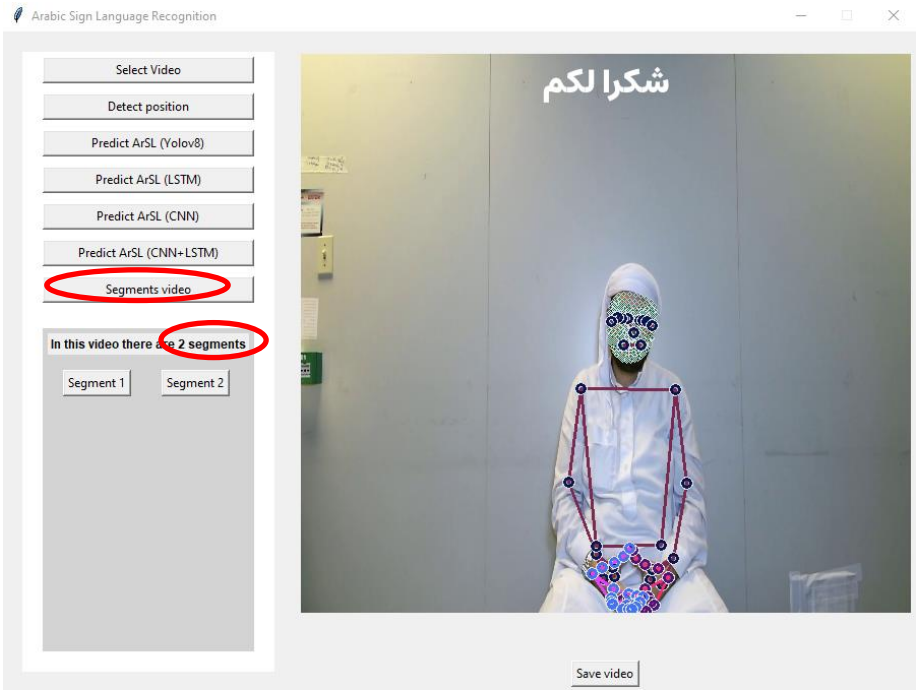


Figure 4-11 *Segments video.*

Clicking on a clip brings up a new window:

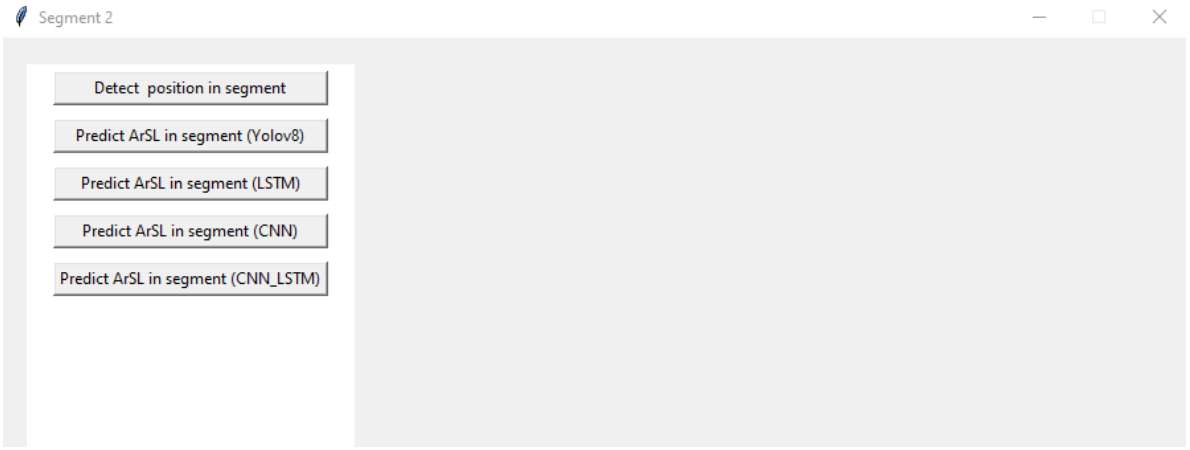


Figure 4-12 *Segment's window.*

In the generated window (see **Figure 4-12**), it works the same as the previous window, but focuses on a segment of the video, which there are images (frames) that combine all the hand and body gestures that form the specific word, and the word is written in the last image in the case of prediction (see **Figure 4-13** and **Figure 4-14**).

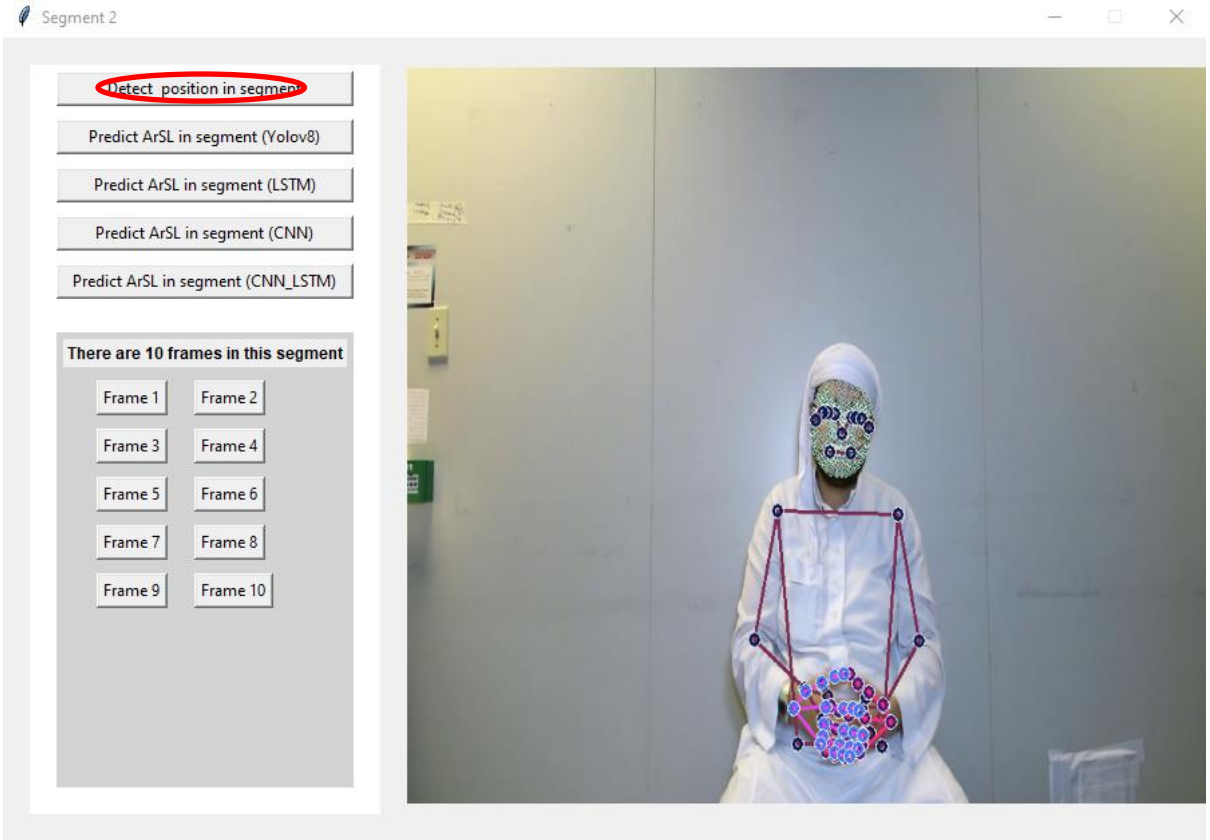


Figure 4-13 *Detect position in segment.*

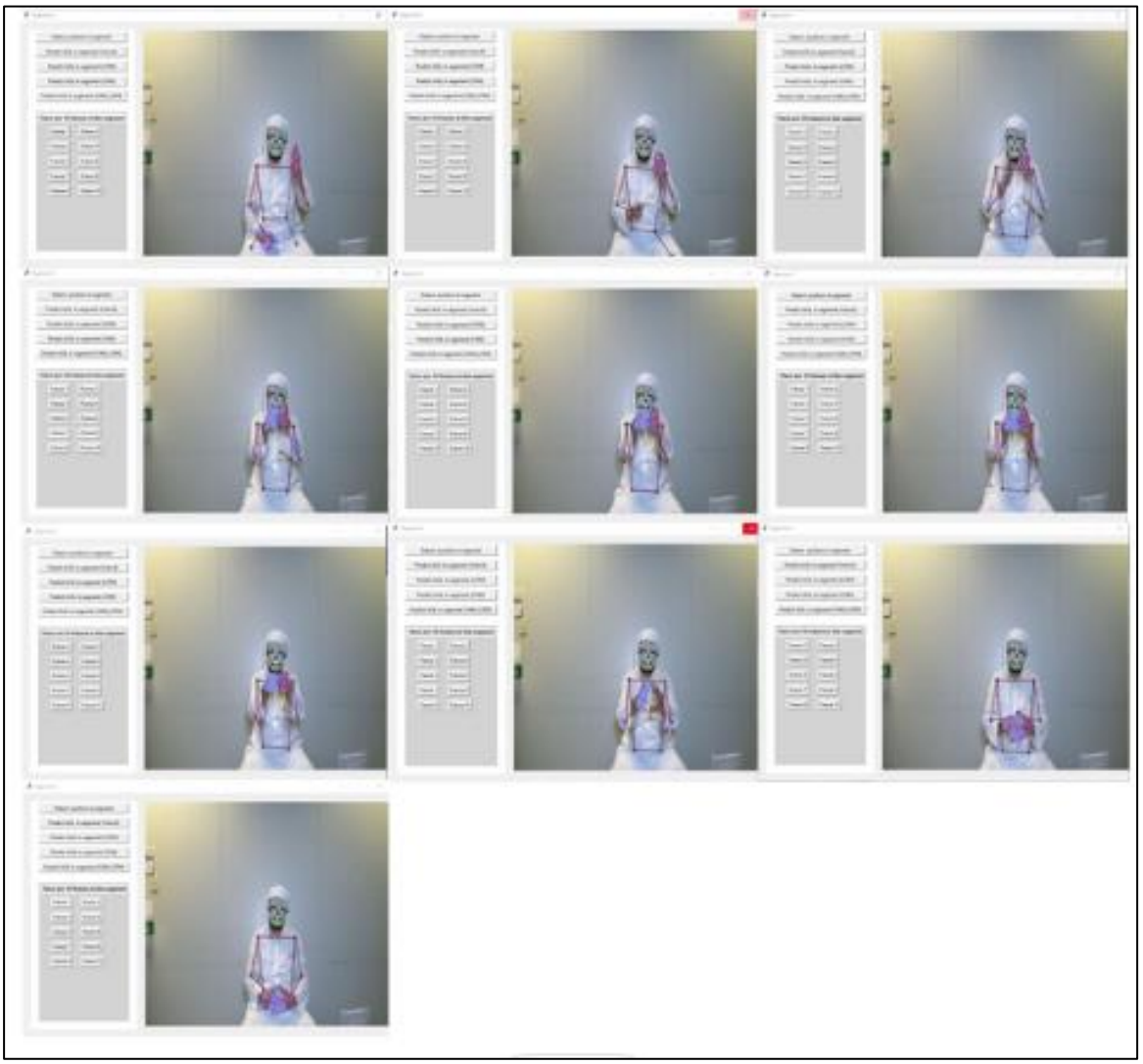


Figure 4-14 *Frames after detection position in segment.*

4.5 Model Performance and Analysis

In this section, we detail the results, which consist of three parts: presentation of results, analysis, and comparison of the models among themselves and with other related works that share similar features. However, due to the limited data available in the dataset we used and the computational constraints, we did not cover all words. We focused instead on the most frequent phrases that are commonly used, reflecting the nature of the dataset and our machine's processing capabilities.

4.5.1 Model Results

Here, we present the results of each Arabic sign language recognition model. To evaluate these models, we use metrics derived from the confusion matrix, which consists of four components:

- **True Positives (TP):** The model predicted the label and matched it correctly according to the ground truth.
- **True negatives (TN):** The model does not predict the label and is not part of the ground truth.
- **False positives (FP):** The model predicted a label, but it is not part of the ground truth.
- **False negatives (FN):** The model does not predict a label, but it is part of the ground truth.

These metrics can be presented as the following:

Accuracy: Accuracy measures the correctness of the results produced by a system or model. It is calculated by dividing the number of correct predictions by the total number of predictions and multiplying it by 100 to get a percentage. High accuracy indicates that a system or model is performing well in its predictions:

$$Accuracy = \frac{TP+TN}{TP+TV+FP+FN} \quad (1)$$

Precision: Precision measures the ratio of correctly predicted bounding boxes for objects compared to the total predicted bounding boxes. It indicates the algorithm's ability to minimize false positives. With the following mathematical relationship:

$$Precition = \frac{TP}{TP+FP} \quad (2)$$

Recall: Recall measures the number of positive class predictions made out of all the positive examples in the dataset. It answers the question "Out of all the actual positive examples, how many positive examples did the model correctly predict as positive?" We calculate it using the formula:

$$Recall = \frac{TN}{TN+FP} \quad (3)$$

F1 score: is a metric used to evaluate the performance of a classification model, especially in cases where the data is unbalanced. It is the harmonic mean of precision and recall, providing a balance between the two. It is measured using the following formula:

$$F1\ score = 2 * \frac{Precition*Recall}{Precition+Recall} \quad (4)$$

Mean Average Precision (mAP): is used to measure the performance of computer vision models. mAP is equal to the average of the Average Precision metric across all classes in a model. You can use mAP to compare both different models on the same task and different versions of the same model. mAP is measured between 0 and 1.

Loss: is a measure of how well or poorly the model's predictions match the actual results. It measures the difference between predicted and actual values. The goal in many machine learning algorithms is to minimize this loss during training, which means making the model's predictions as accurate as possible.

In our case, we have seven distinct classes as shown in the following table. As a result, we evaluate the performance of our model using the confusion matrix, where it provides information of how well our model distinguishes between each class, showing the counts of true positives, true negatives, false positives, and false negatives for each class.

	السلام	عليكم	ورحمة	الله	وبركاته	شكرا	لكم
السلام	TP	FN	FN	FN	FN	FN	FN
عليكم	FP	TP	FN	FN	FN	FN	FN
ورحمة	FP	FP	TP	FN	FN	FN	FN
الله	FP	FP	FP	TP	FN	FN	FN
وبركاته	FP	FP	FP	FP	TP	FN	FN
شكرا	FP	FP	FP	FP	FP	TP	FN
لكم	FP	FP	FP	FP	FP	FP	TP

Table 4-2 Confusion Matrix of seven classes.

A. YOLOv8 model

Using the YOLOv8 model for comparison purposes, we achieved the following results, where the matrix of the phrase (السلام عليكم و رحمة الله و بركاته) is presented in Figure 4-3.

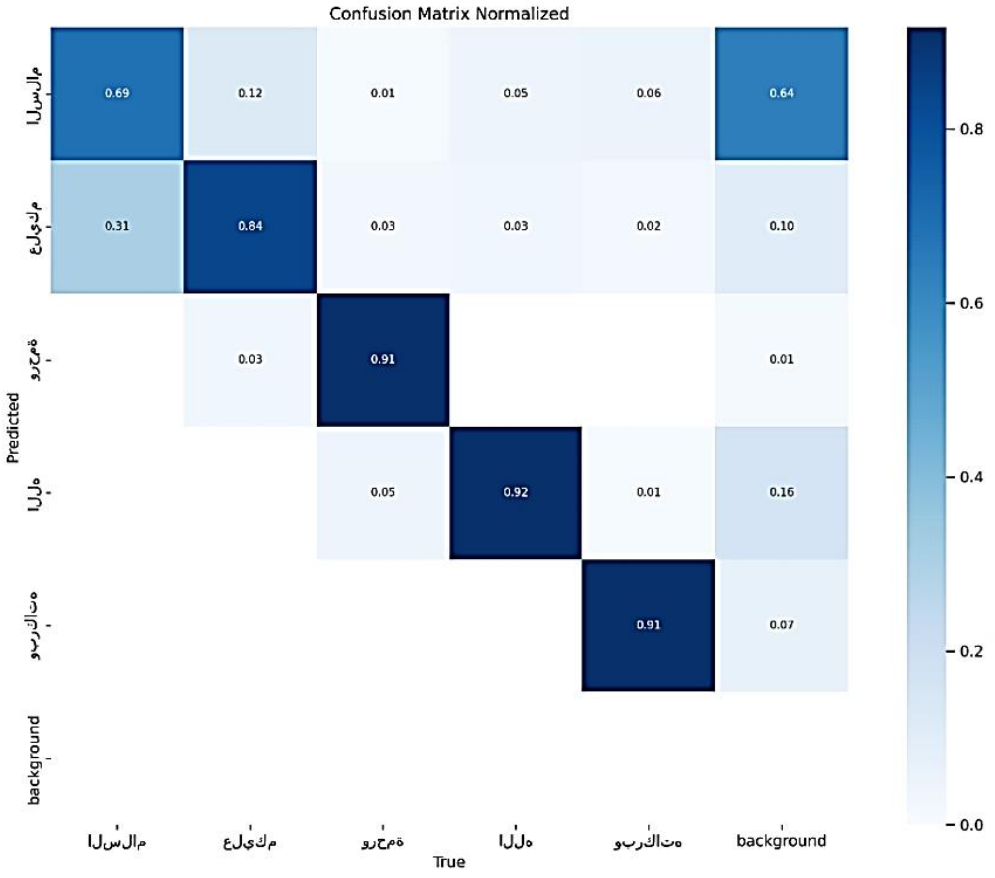


Table 4-3 *Result of Confusion Matrix of our algorithm*

The training result of this model is shown in **Figure 4-15** and **Table 4-4**.

Precision	Recall	mAP50
0.8641	0.8910	0.9267

Table 4-4 The training result statistics.

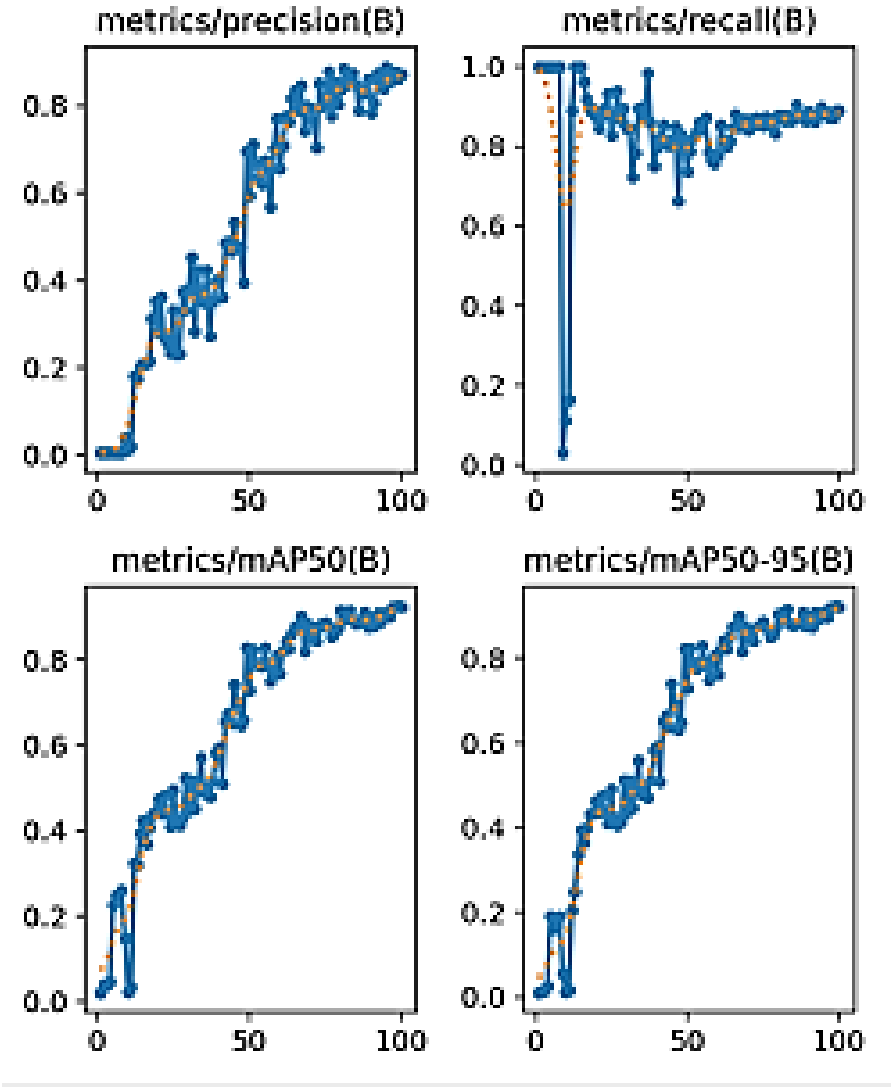
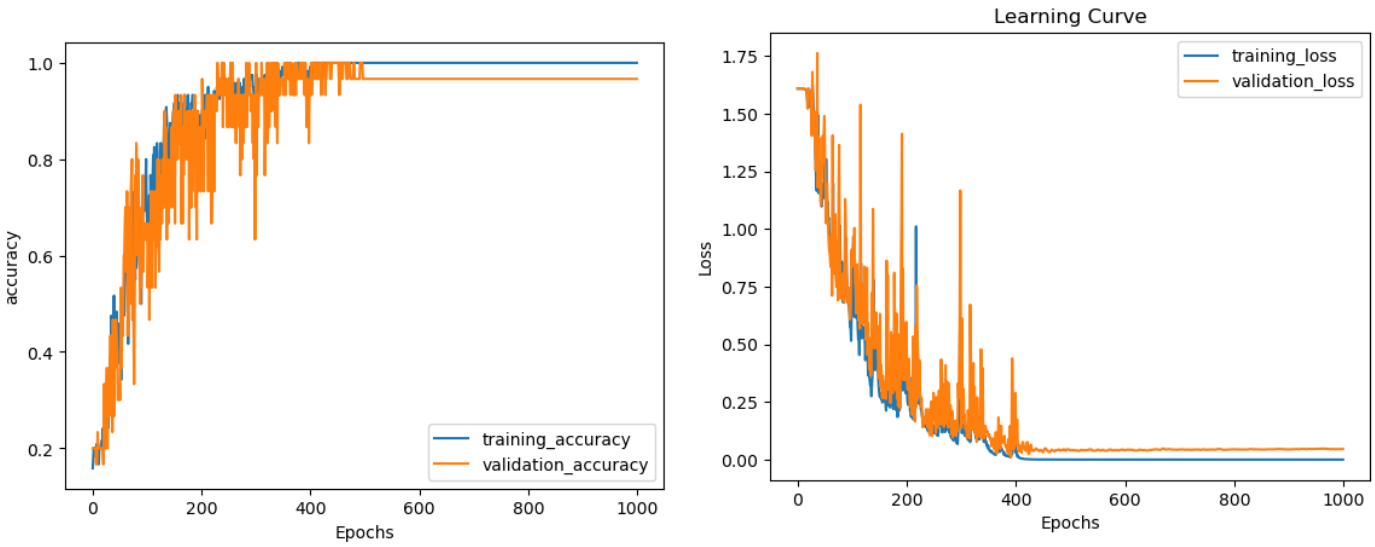


Figure 4-15 *The training results of YOLOv8*

B. CNN model

The model achieved a validation loss of 0.2137, indicating that it effectively minimized the discrepancy between the predicted and actual values. This suggests that the model's predictions are generally close to the ground truth. Furthermore, the validation accuracy of 0.*9523* reveals that the model correctly classified a significant proportion of the validation samples. These results indicate that the model exhibits a high level of accuracy in distinguishing between different classes or categories.

Overall, the evaluation results emphasize the efficacy and potential of the model in solving the problem at hand, validating its capability to make accurate predictions and contributing to the overall success of your research. The accuracy and loss curve are shown in ***Figure 4-16*****.**



(a) Accuracy curve (b) Training loss curve

Figure 4-16 *Accuracy and training loss Curve of CNN model.*

Below, we show the confusion matrix for our model that displays the true label compared to the predicted label shown in the figure below:

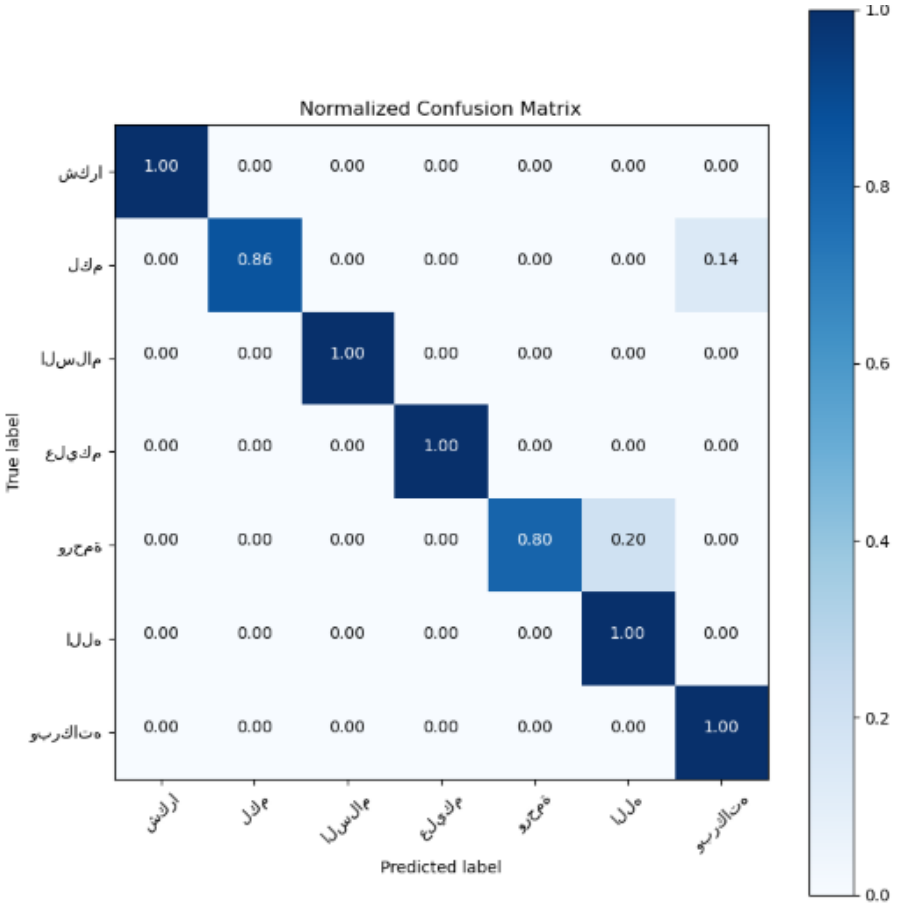


Figure 4-17 *Confusion Matrix of CNN model.*

C. LSTM model

The results of the LSTM model presented and detailed in the previous chapter are shown in the **Table 4-5**

Loss	Accuracy	Precision	Recall	F1_Score
0.5862	0.8809	0.9024	0.8809	0.8915

Table 4-5 Result of LSTM model

And confusion matrix of this model shown in **Figure 4-18**.

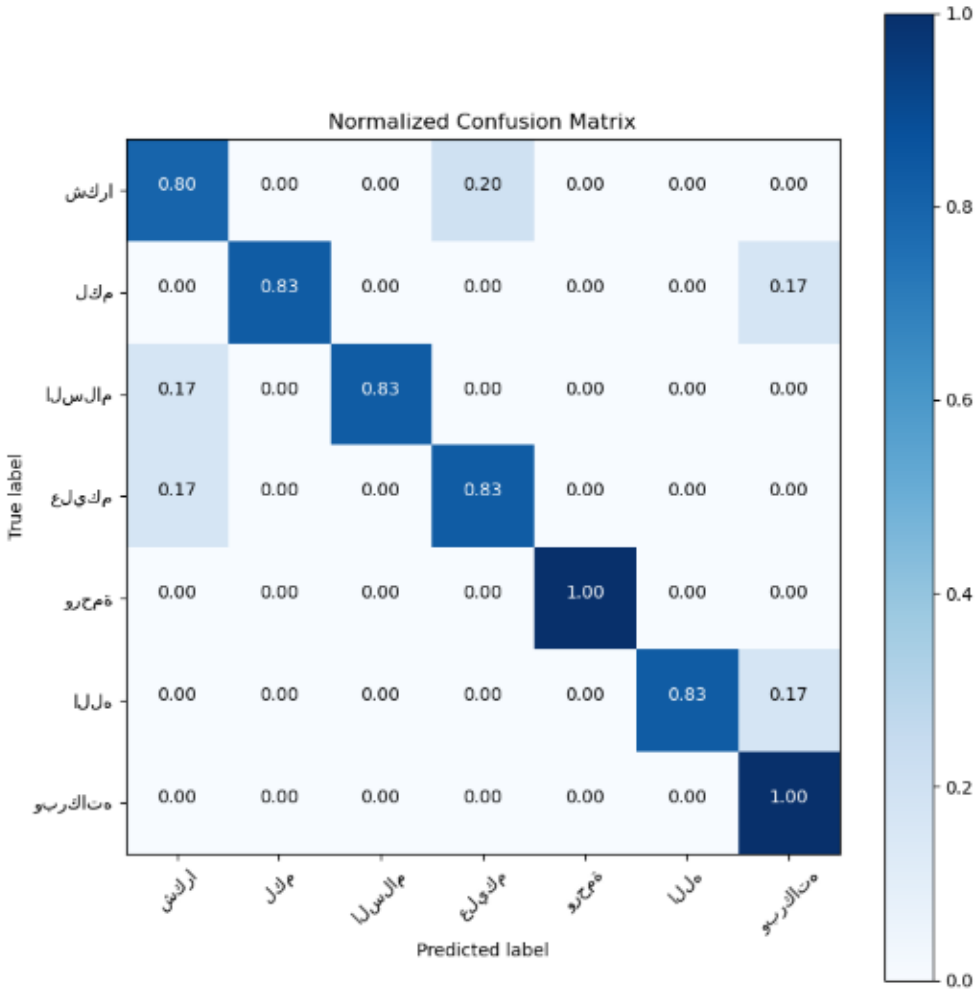
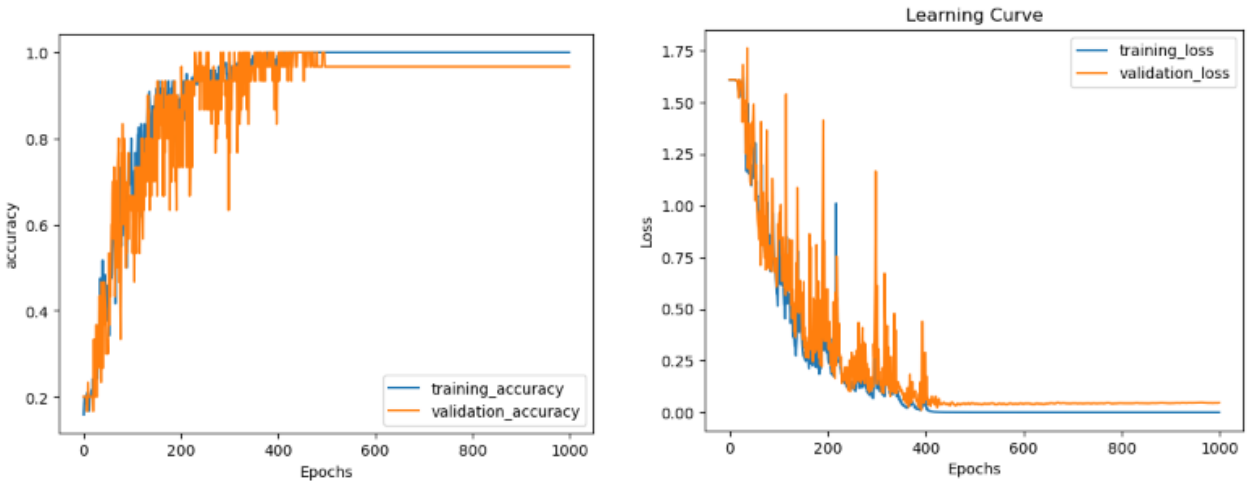


Figure 4-18 *Confusion Matrix of LSTM model.*

D. Hybrid model

As we said earlier, we wanted to optimize the results, so we combined two models CNN and LSTM into one and obtained a low validation loss of 0.0465 and an accuracy rate of 0.9666. The accuracy and loss curve are shown in



(a) Accurracy curve *(b) Training loss curve*

Figure 4-19 *Accuracy and training loss Curve of the hybrid model.*

4.5.2 Results Analysis

In this section, we provide a true comparison and analysis based on the models' results that were developed in this study which basically divided into four models. The table below represents a comparison between these four models:

	validation loss	Precision	Recall	F1_score
YOLOv8	0.2139	0.8641	0.8910	0.8773
CNN	0.2137	0.9523	0.9523	0.9523
LSTM	0.5862	0.9024	0.8809	0.8915
CNN&LSTM	0.04653	0.9666	0.9666	0.9666

Table 4-6 Comparison table of our system models.

From the results, we can see that our system works well in the detection and recognition phase, especially in normal situations. The high precision and recall values indicate that the system successfully recognizes Arabic sign language. This means that the system is effective in detecting and recognizing human body parts, demonstrating its robustness and reliability.

Regarding the YOLOv8 model developed for comparison purposes, it is not considered the best solution for recognizing Arabic sign language, as its accuracy is low compared to previously developed models. This may be due to the small dataset, which does not include all cases and is the same dataset used to train all models.

Based on these statistics, we can say that the hybrid model is the most effective for handling and covering the recognition task well, which was able to solve many issues and deal with many obstacles that can be characterized as follows:

1. Developed a new model based on the combination of two distinct models: CNN and LSTM.
2. Provided a real implementation of four different models, which gives more information about the models' performance in Arabic language.

3. Our system excelled in detecting and recognizing human location even under difficult conditions of light contrast and distortion.
4. Our system shows high accuracy in recognizing Arabic sign language, reliably recognizing the words that each sign represents.
5. We covered a new aspect of Arabic sign recognition where the input is a video, not an image, addressing a type of input that is very limited and covered in only a few papers.

4.5.3 Result discussion

In this section, we provide a comparison based on our models' results, specifically focusing on the combination of CNN and LSTM model, with other related works to demonstrate the effectiveness of the proposed model. **Table 4-7** shows the results of our model and several other models in the literature trained on the same dataset.

Author	Year	Language	Dataset	Approach	Input	Output	Error rate
Saleh Aly et al.[78]	2020	Arabic	Manually	BiLSTM	Video	Words	10.41%
Hamzah Luqman et al [59].	2023	Arabic	ArabSign-A	Encoder-decoder model.	Video	Sentence	50%
Our models	2024	Arabic	Manually created based on ArabSign-A	CNN&LSTM CNN LSTM YOLOv8	Video	Words/ Sentence	3.34% 4.77% 11.91% 8%

Table 4-7 Evaluation of the proposed model in relation to existing models.

We encountered difficulties in comparing our work with existing studies due to the absence of similar works using the same dataset. Each study is isolated and employs manually developed datasets specific to its own research. However, we selected the closest studies that appeared similar to ours. In contrast to the results of previous studies, our proposed architecture (hybrid model) obtained the best results and was proven to be effective in recognizing sign language, achieving an Error rate of 3.34%.

However, as with any system, we have encountered several challenges that still remain, which we present in the following points:

1. The first and biggest issue we faced was not finding a dataset for Arabic sign language words in video type, solely only one that we could find with complete sentences, and limited to whole inseparable sentence without words or letters options. Furthermore, the only available dataset does not include all cases which there are no variation in the videos; where the videos are close to each other which lead to narrow evaluation.

2. The second major issue that we encountered, is related to the high computational cost associated with training deep learning models, due to the high size of the video which each one of them cross 1.31 MO, therefore we chose a small number of words because the training process requires a lot of RAM capacity, which is not available in our machine. Also, when training in COLAB and using the trainer model, we face issues with the library versions.

Finally, we aspire to turn this system into a system for Algerian sign language and create a dataset that includes all cases and is specific to the Algerian language only, and extend it to cover the entire Maghreb region.

4.6 Conclusion

In this chapter, we provided details about our system for human position detection and Arabic sign language recognition. For the detection task, we used the Mediapipe library, which shows excellent performance in detecting the human body and its movements. In addition, we use the CNN, LSTM, YOLOv8, LSTM&CNN models for the ArSL task. After experiments, we obtained impressive results in terms of accuracy, which shows the effectiveness of our system.

General Conclusion

Research in the field of sign language recognition for global languages has reached its peak, but for Arabic sign language, most research is limited to recognizing the Arabic alphabet. Recognizing ASL in words and sentences poses a significant challenge in video analysis and linguistics, where a comprehensive dataset for this language is lacking. We could not find suitable data for recognition purposes; while existing datasets were images, we found only one dataset that partially met our needs, limited to complete phrases, which constrained our development and evaluation. Despite this issue, we were able to leverage the benefits of this dataset to develop three different models in addition to YOLOv8 for real evaluation and comparison analysis.

In this thesis, we propose different models using a deep learning-based computer system to easily and robustly recognize ASL. To achieve this goal, we developed four AI models. The hybrid model (CNN and LSTM) excelled in recognizing Arabic sign language and achieved encouraging results in all cases. Furthermore, we developed other models for comparison and attempted to achieve better results: one based on a CNN model and another using an LSTM model, by adding layers to each model to improve accuracy and processing time. For comparative purposes, we also trained the same dataset using the YOLOv8 algorithm, which did not yield satisfactory results, highlighting the superior performance of the hybrid model in Arabic sign language tasks that the existing YOLO model did not meet.

In conclusion, despite the lack of resources such as datasets and computational capacity, we achieved high levels of accuracy, efficiency, and robustness. This makes our approach suitable for application in the lives of hearing-impaired individuals, enabling them to effectively communicate their thoughts and opinions.

In our future work, we have identified several perspectives and goals to improve our Arabic sign language recognition system. These include:

- Creating a dataset that encompasses all cases. While our current system operates on a small and limited dataset, we recognize the importance of continuously expanding it.
- Aspiring to adapt the system for Algerian sign language and developing a dedicated dataset for the Algerian language.
- Expanding the dataset to encompass the entire Maghreb region.
- Exploring real-time applications and integration possibilities to make the system more accessible and practical for users.
- Collaborating with linguists and sign language experts to ensure cultural and linguistic inclusivity in dataset expansion efforts.
- Conducting user studies and feedback sessions to refine the system based on practical user experiences and needs.

Bibliography

[1] Muhammad Aminur Rahaman (2018). Computer vision-based Bangla sign language recognition. *Ph. D. dissertation*.

[2] Zhou, Z., Chen, K., Li, X. et al (2020). Sign-to-speech translation using machine-learning-assisted stretchable sensor arrays. *Nat Electron*, 3(9), 571–578.

[3] Luqman, H., & Mahmoud, S. A. (2019). Automatic translation of Arabic text-to-Arabic sign language. *Universal Access in the Information Society*, 18(4), 939-951.

[4] Farooq, U., Rahim, M. S. M., Sabir, N., Hussain, A., & Abid, A. (2021). Advances in machine translation for sign language: approaches, limitations, and challenges. *Neural Computing and Applications*, 33(21), 14357-14399.

[5] Elpeltagy, M., Abdelwahab, M., Hussein, M. E., Shoukry, A., Shoala, A., & Galal, M. (2018). Multi-modality-based Arabic sign language recognition. IET Computer Vision, 12(7), 1031-1039.

[6] Viswavarapu, L. K. (2018). Real-Time Finger Spelling American Sign Language Recognition Using Deep Convolutional Neural Networks, Master's thesis, *University of North Texas*. Retrieved from UNT Digital Library.

[7] Al-Hammadi, M., Muhammad, G., Abdul, W., Alsulaiman, M., Bencherif, M. A., & Mekhtiche, M. A. (2020). Hand gesture recognition for sign language using 3DCNN. *IEEE access*, 8, 79491-79509.

[8] Hearst, M. A., Dumais, S. T., Osuna, E., Platt, J., & Scholkopf, B. (1998). Support vector machines. *IEEE Intelligent Systems and their applications*, 13(4), 18-28.

[9] Zhang, M. L., & Zhou, Z. H. (2007). ML-KNN: A lazy learning approach to multi-label learning. *Pattern recognition*, *40*(7), 2038-2048.

[10] Albawi, S., Mohammed, T. A., & Al-Zawi, S. (2017, August). Understanding of a convolutional neural network. *In 2017 international conference on engineering and technology (ICET)* (pp. 1-6). Ieee.

[11] Sultan, A., Makram, W., Kayed, M., & Ali, A. A. (2022). Sign language identification and recognition: A comparative study. *Open Computer Science*, 12(1), 191-210.

[12] Abdi, H., & Williams, L. J. (2010). Principal component analysis. *Wiley interdisciplinary reviews: computational statistics*, 2(4), 433-459.

[13] Viswavarapu, L. K. (2018). Real-Time Finger Spelling American Sign Language Recognition Using Deep Convolutional Neural Networks *(Doctoral dissertation, University of North Texas).*

[14] Mohandes, M., Deriche, M., & Liu, J. (2014). Image-based and sensor-based approaches to Arabic sign language recognition. *IEEE transactions on human-machine systems*, 44(4), 551-557.

[15] Zakariya, A. M., & Jindal, R. (2019, July). Arabic sign language recognition system on smartphone. *In 2019 10th international conference on computing, communication and networking technologies (ICCCNT)* (pp. 1-5). IEEE.

[16] Alani, A. A., & Cosma, G. (2021). ArSL-CNN: a convolutional neural network for Arabic sign language gesture recognition. *Indonesian journal of electrical engineering and computer science*, 22.

[17] Rawf, K. M. H., Mohammed, A. A., Abdulrahman, A. O., Abdalla, P. A., & Ghafoor, K. J. (2023). A comparative technique using 2D CNN and transfer learning to detect and classify Arabic-script-based sign language. *Acta Inform Malays*, 7(1), 66.

[18] Moustafa, A. M. A., Mohd Rahim, M. S., Bouallegue, B., Khattab, M. M., Soliman, A. M., Tharwat, G., & Ahmed, A. M. (2023). Integrated Mediapipe with a CNN Model for Arabic Sign Language Recognition. *Journal of Electrical and Computer Engineering*.

[19] Aiouez, S., Hamitouche, A., Belmadoui, M. S., Belattar, K., & Souami, F. (2022). Real-time Arabic Sign Language Recognition based on YOLOv5. *In IMPROVE* (pp. 17-25).

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

[21] Kelly, D. (2010). Computational Models for the Automatic Learning and Recognition of Irish Sign Language (*Doctoral dissertation, National University of Ireland Maynooth*).

[22] Sturman, D. J., & Zeltzer, D. (1994). A survey of glove-based input. *IEEE Computer graphics and Applications*, 14(1), 30-39.

[23] Vamplew, Peter. (1999). Recognition of Sign Language Gestures Using Neural Networks. *Neuropsychological Trends*. 1.

[24] Mohamed, N., Mustafa, M. B., & Jomhari, N. (2021). A review of the hand gesture recognition system: Current progress and future directions. *ieee access*, 9, 157422-157436.

[25] Borg, M., & Camilleri, K. P. (2019, May). Sign language detection "in the wild" with recurrent neural networks. *In ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1637-1641. IEEE.

[26] Vamplew, P. (1996). Recognition of sign language using neural networks. *Doctoral dissertation, University of Tasmania*.

[27] Nerlekar, A. A. (2021). Sign Language Recognition Using Smartphones. *Doctoral dissertation, California State University, Northridge.*

[28] Viswavarapu, L. K. (2018). Real-Time Finger Spelling American Sign Language Recognition Using Deep Convolutional Neural Networks. *Doctoral dissertation, University of North Texas*.

[29] Mori, Y., & Toyonaga, M. (2018, December). Data-glove for japanese sign language training system with gyro-Sensor. *In 2018 Joint 10th International Conference on Soft Computing and Intelligent Systems (SCIS) and 19th International Symposium on Advanced Intelligent Systems (ISIS)* (pp. 1354-1357). IEEE.

[30] Eccarius, P., Bour, R., & Scheidt, R. A. (2012). Dataglove measurement of joint angles in sign language handshapes. *Sign Language & Linguistics*, 15(1), 39-72.

[31] Khan, R. (2022). Sign Language Recognition from a webcam video stream.

[32] Ong, S. C., & Ranganath, S. (2005). Automatic sign language analysis: A survey and the future beyond lexical meaning. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 27(06), 873-891.

[33] Aliwy, A. H., & Ahmed, A. A. (2021). Development of arabic sign language dictionary using 3D avatar technologies. *Indonesian Journal of Electrical Engineering and Computer Science*, 21(1), 609-616.

[34] Murthy, G. R. S., & Jadon, R. S. (2009). A review of vision-based hand gestures recognition. International Journal of Information Technology and Knowledge Management, 2(2), 405-410.

[35] Suarez, J., & Murphy, R. R. (2012, September). Hand gesture recognition with depth images: A review. In 2012 IEEE RO-MAN: the 21st IEEE international symposium on robot and human interactive communication (pp. 411-417). IEEE.

[36] Shrivastava, R. (2013, February). A hidden Markov model based dynamic hand gesture recognition system using OpenCV. In 2013 3rd IEEE International Advance Computing Conference (IACC) (pp. 947-950). IEEE.

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

[38] Lee, H. K., & Kim, J. H. (1999). An HMM-based threshold model approach for gesture recognition. IEEE Transactions on pattern analysis and machine intelligence, 21(10), 961-973.

[39] Bamwenda, J., & Özerdem, M. S. (2019). Recognition of static hand gesture with using ANN and SVM. *Dicle Univ. J. Eng*, *10*, 561-568.

[40] Wu, M. (2024). Gesture Recognition Based on Deep Learning: A Review. EAI Endorsed Transactions on e-Learning, 10.

[41] Chaudhary, A., Raheja, J. L., Das, K., & Raheja, S. (2013). Intelligent approaches to interact with machines using hand gesture recognition in natural way: a survey. arXiv preprint arXiv:1303.2292.

[42] Sivanandam, S. N., Sumathi, S., & Deepa, S. N. (2007). Introduction to fuzzy logic using MATLAB.

[43] Verma, R., & Dev, A. (2009, October). Vision based hand gesture recognition using finite state machines and fuzzy logic. In 2009 International Conference on Ultra-Modern Telecommunications & Workshops (pp. 1-6). IEEE.

[44] Trivino, G., & Bailador, G. (2007, June). Linguistic description of human body posture using fuzzy logic and several levels of abstraction. In 2007 IEEE International Conference on Computational Intelligence for Measurement Systems and Applications (pp. 105-109). IEEE.

[45] Al-Jarrah, O., & Halawani, A. (2001). Recognition of gestures in Arabic sign language using neuro-fuzzy systems. *Artificial Intelligence*, *133*(1-2), 117-138.

[46] Elatawy, S. M., Hawa, D. M., Ewees, A. A., & Saad, A. M. (2020). Recognition system for alphabet Arabic sign language using neutrosophic and fuzzy c-means. *Education and Information Technologies*, *25*(6), 5601-5616.

[47] Riad, A. M., Elminir, H. K., & Shohieb, S. M. (2014). Hand gesture recognition system based on a geometric model and rule-based classifier. *British Journal of Applied Science & Technology*, *4*(9), 1432-1444.

[48] Siena, F. L., Byrom, B., Watts, P., & Breedon, P. (2018). Utilising the intel realsense camera for measuring health outcomes in clinical research. *Journal of medical systems*, 42, 1-10.

[49] Siam, S. M., Sakel, J. A., & Kabir, M. H. (2016). Human computer interaction using marker-based hand gesture recognition. arXiv preprint arXiv:1606.07247.

[50] Herath, H. M. S. P. B., Ekanayake, M. P. B., Godaliyadda, G. M. R. I., & Wijayakulasooriya, J. V. (2015, August). Multi-feature-based hand-gesture recognition. *In 2015 Fifteenth International Conference on Advances in ICT for Emerging Regions (ICTer)* (pp. 63-68). IEEE.

[51] Liu, K., Chen, C., Jafari, R., & Kehtarnavaz, N. (2014, October). Multi-HMM classification for hand gesture recognition using two differing modality sensors. In 2014 IEEE Dallas Circuits and Systems Conference (DCAS) (pp. 1-4). IEEE.

[52] Yang, Y.T.; Fishbain, B.; Hochbaum, D.S.; Norman, E.B.; Swanberg, E. The Supervised Normalized Cut Method for Detecting, Classifying, and Identifying Special Nuclear Materials. *Informs J. Comput.* 2013, 26, 45–58.

[53] Li, J.; Li, C.; Han, J.; Shi, Y.; Bian, G.; Zhou, S. Robust Hand Gesture Recognition Using HOG-9ULBP Features and SVM Model. Electronics 2022, 11, 988. https://doi.org/10.3390/electronics11070988

[54] Alani, A.A., Cosma, G., Taherkhani, A., & McGinnity, T.M. (2018). Hand gesture recognition using an adapted convolutional neural network with data augmentation. *2018 4th International Conference on Information Management (ICIM)*, 5-12.

[55] Do, N.-T.; Kim, S.-H.; Yang, H.-J.; Lee, G.-S. Robust Hand Shape Features for Dynamic Hand Gesture Recognition Using Multi-Level Feature LSTM. Appl. Sci. 2020, 10, 6293. https://doi.org/10.3390/app10186293

[56] Ismail, M. H., Dawwd, S. A., & Ali, F. H. (2021). Static hand gesture recognition of Arabic sign language by using deep CNNs. *Indonesian Journal of Electrical Engineering and Computer Science*, 24(1), 178-188.

[57] Python Software Foundation. Python. https://www.python.org/. Accessed on 10th June 2023.

[58] Jupyter Project. Jupyter Documentation. https://jupyter.org/. Accessed: june, 2023.

[59] Luqman, H. (2023, January). ArabSign: A Multi-modality Dataset and Benchmark for Continuous Arabic Sign Language Recognition. In *2023 IEEE 17th International Conference on Automatic Face and Gesture Recognition (FG)* (pp. 1-8). IEEE.

[60] Rastgoo, R., Kiani, K., Escalera, S., & Sabokrou, M. (2021). Sign language production: A review. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 3451-3461).

[61] Keras Documentation. https://keras.io. Accessed :29/05/2024.

[62] TensorFlow Documentation. https://www.tensorflow.org/learn?hl=fr. Accessed :29/05/2024.

[63] MediaPipe – The Ultimate Guide to Video Processing. https://learnopencv.com/introduction-to-mediapipe/#What-is-MediaPipe?. Accessed :29/05/2024.

[64] What is OpenCV? – An Introduction Guide. https://pythongeeks.org/what-is-opencv/. Accessed :29/05/2024.

[65] Numpy Documentation. https://courspython.com/apprendre-numpy.html. Accessed :29/05/2024.

[66] Python Documentation. https://docs.python.org/3/library/tkinter.html. Accessed :29/05/2024.

[67] PIL Contributors. Python Imaging Library (PIL) Documentation. https://pillow.readthedocs.io/en/stable/. Accessed :29/05/2024.

[68] Arabic Reshaper. https://pypi.org/project/arabic-reshaper/. Accessed :29/05/2024.

[69] python-bidi. https://pypi.org/project/python-bidi/. Accessed :29/05/2024.

[70] moviepy documentation. https://pypi.org/project/moviepy/. Accessed :29/05/2024.

[71] Python Documentation. https://docs.python.org/3/library/os.html. Accessed :29/05/2024.

[72] Al Ahmadi, S., Muhammad, F., & Al Dawsari, H. (2024). Enhancing Arabic Sign Language Interpretation: Leveraging Convolutional Neural Networks and Transfer Learning. *Mathematics*, *12*(6), 823.

[73] Chaudhary, A., Raheja, J. L., Das, K., & Raheja, S. (2013). Intelligent approaches to interact with machines using hand gesture recognition in natural way: a survey. *arXiv preprint arXiv:1303.2292*.

[74] Ahmed, A. M., Alez, R. A., Taha, M., & Tharwat, G. (2016). Automatic translation of Arabic sign to Arabic text (ATASAT) system. *Journal of Computer Science and Information Technology*, 6, 109-122.

[75] Ahmed, A. M., Abo Alez, R., Tharwat, G., Taha, M., Belgacem, B., & Al Moustafa, A. M. (2020). Arabic sign language intelligent translator. *The Imaging Science Journal*, 68(1), 11-23.

[76] Alzohairi, R., Alghonaim, R., Alshehri, W., & Aloqeely, S. (2018). Image based Arabic sign language recognition system. *International Journal of Advanced Computer Science and Applications*, 9(3).

[77] Hayani, S., Benaddy, M., El Meslouhi, O., & Kardouchi, M. (2019, July). Arab sign language recognition with convolutional neural networks. *In 2019 International conference of computer science and renewable energies (ICCSRE)* (pp. 1-4). IEEE.

[78] Aly, S., & Aly, W. (2020). DeepArSLR: A novel signer-independent deep learning framework for isolated arabic sign language gestures recognition. *IEEE Access*, 8, 83199-83212.

[79] Sharma, S., & Singh, S. (2022). Recognition of Indian sign language (ISL) using deep learning model. *Wireless personal communications*, *123*(1), 671-692.

[80] G. Latif, N. Mohammad, J. Alghazo, R. AlKhalaf, and R. AlKhalaf, “Arasl: Arabic alphabets sign language dataset,” *Data in brief*, vol. 23, p. 103777, 2019

[81] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, “Smote: synthetic minority over-sampling technique,” *Journal of artificial intelligence research*, vol. 16, pp. 321–357, 2002.

[82] M. Al-Barham, “RGB Arabic alphabets sign language dataset,” 2023.