Republic of Iraq

Ministry of Higher Education and Scientific Research

University of Babylon

College of Information Technology

Software Department

Classifying Arabic Sign Language Based on Machine Learning Techniques

A Dissertation

Submitted to the Council of the College of Information Technology for Postgraduate Studies of University of Babylon in Partial Fulfillment of the Requirements for the Degree of Doctor of Philosophy in Information Technology-Software

By

Maha Sabri Zbala Altememe

Supervised by

Prof. Dr. Nidhal K. El Abbadi

2023 A.C. **1444 A.H.**

بِسْمِ الله الرحمن الرحيم

﴿ قُلْ هُوَ الَّذِي أَنشَأَكُمْ وَجَعَلَ لَكُمُ السَّمْعَ وَالْأَبْصَارَ وَالْأَفْئِدَةَ ۖ قَلِيلًا مَّا تَشْكُرُونَ ﴾

صدق الله العلي العظيم

سورة الملك\آية 23

2023 A.C. 1444 A.H.

Supervisor Certification

I certify that this dissertation is prepared under my supervision at the Department of Software / Collage of Information Technology / Babylon University, by **Maha Sabri Altememe** as a partial fulfillment of the requirements for the degree of **Ph.D. in Information Technology**.

Signature:

Supervisor Name: **Prof. Dr. Nidhal K. El Abbadi**

Date: / /2023

The Head of the Department Certification

In view of the available recommendation, we forward this dissertation for debate by the examining committee.

Signature:

Name: **Dr. Ahmed Saleem Abbas**

Title**: Professor**

Date: / / 2023

Certification of the Examination Committee

We hereby certify that we have studied the dissertation entitled (**Arabic Sign Language Recognition Based on Machine Learning**) presented by the student (**Maha Sabri Zbala Altememe**) and examined him/her in its content and what is related to it, and that, in our opinion, it is adequate with (**Excellent**) standing as a thesis for the degree of Doctor of Philosophy in Information Technology-Software.

Signature:
Name**: Dr. Tawfiq A. Alassadi**
Title: **Professor**
Date: / / 2023
(**Chairman**)

Signature:
Name:**Dr. Baheja Khudair Shukur**
Title: **Professor**
Date: / / 2023
(**Member**)

Signature:
Name: **Dr. Ahmed Habeeb Said AL-Azawei**
Title: **Assistant Professor**
Date: / / 2023
(**Member**)

Signature:
Name: **Dr. May A. Salih**
Title: **Assistant Professor**
Date: / / 2023
(**Member**)

Signature:
Name: **Dr. Sura Zaki Alrashid**
Title: **Assistant Professor**
Date: / / 2023
(**Member**)

Signature:
Name: **Dr. Nidhal K. El Abbadi**
Title: **Professor**
Date: / / 2023
(**Member and Supervisor**)

Approved by the Dean of the College of Information Technology, University of Babylon.

Signature:
Name: **Dr. Hussein Atiyah Lafta**
Title: **Professor**
Date: / / 2023
(**Dean of Collage of Information Technology**)

Declaration

I hereby declare that this dissertation, **Arabic Sign Language Recognition Based on Machine Learning** submitted to the University of Babylon in partial fulfillment of requirements for the degree of Doctorate of Philosophy in Information Technology-Software has not been submitted as an exercise for a similar degree at any other University. I also certify that this work described here is entirely my own except for reports and summaries whose sources are appropriately cited in the references.

Signature:

Name: Maha Sabri Altememe

Date: / / 2023

Acknowledgement

In the name of Allah, all praise to Allah Most Gracious, Most Merciful, Alhamdulillah I have finished this research. All praise be to ALLAH Almighty who helped me complete this task successfully and my utmost respect to his last Prophet Mohammad PBUH.

First, I would like to express my deep appreciation to my supervisor **Prof. Dr. Nidhal K. El Abbadi**, for his invaluable guidance, supervision, and untiring efforts during this work.

I also extend my thanks and gratitude to **the Imam Hussein Academy for the Deaf,** especially **Mr. Basem Al-Atwani** for providing assistance

I must express my very profound gratitude to my mother and my dear husband for providing me with unfailing support and continuous encouragement throughout my years of study and through the process of researching and writing this dissertation.

I would like to declare my deep thanks and appreciation to my lovely daughters, my darling son, and my sister for their patience, encouragement, and help during the work.

Thanks, and appreciation to the staff of the College of Information Technology, especially my friends for their patience and for the great assistance they provided me.

Last but not least, most importantly, I would like to thank all the kind, helpful, and lovely people who helped me directly or indirectly to complete this work and apologize to them for not being able to mention them by name here, though they are in my heart.

Maha Sabri Altememe

Dedication

"اللهم صلى على محمد وال محمد"

To present this effort as a gift to

ALLAH (God Almighty)

To the Prophet of Mercy,

Prophet Mohammad

(May God bless him and grant him peace).

Al-Imam Al-Mehdi

(Peace be upon him)

The survival of religion on earth.

Abstract

Sign language is a visual language that uses a combination of hand gestures, facial expressions, and body language to communicate meaning. According to the World Health Organization (WHO) report, for the year 2021, approximately 13% of the world's population lives with some forms of hearing loss. This emphasizes the importance of raising awareness about hearing loss, promoting prevention measures, and ensuring access to appropriate hearing healthcare services.

Sign language recognition is a complex area of research that poses many challenges, such as the rapid movement of hands and bodies that can change the intended meaning of signs, as well as the overlap of signs. Building an Arabic Sign Language (ArSL) interpreting system based on deep learning is an ambitious and valuable goal. The method of combining machine learning and deep learning shows promising results in various domains, including computer vision and natural language processing, which can be leveraged for sign language recognition.

The phases of work are organized in the present dissertation to create an integrated system for the classification and interpretation of Arabic sign language. The first phase is the data collection process. It should include many different locations and a wide range of signs and expressions. These collected images or videos must be classified and labeled for each word or letter.

The second phase is preprocessing. This includes frame splitting, noise reduction, contrast enhancement, converting color image into grayscale, and size standardization. The third phase is the feature extraction stage. This stage is to obtain important features from images or videos. It is done using the Linear Discriminant Analysis (LDA) algorithm and Viola and Jones algorithm. The LDA is used to extract important features that pertain to the hand, whereas the second algorithm is used to extract facial features and to determine the important area in the images.

In the fourth phase, the focus is on designing and building models capable of classifying signs or movements. Five main models can be used for this purpose: The

LDA-CNN of the Alphabet model combines the LDA method with one-dimensional CNN technology to produce excellent accuracy and efficiency at the cost of time. This model is a process for letter sign recognition. The **LDA-CNN of Word** model employs a combination of a feature extraction LDA algorithm and CNN to classify and distinguish gestures of sign language words based on dynamic images. The **LDA-ML of the Alphabet & Word** model is used to classify the alphabet and words in sign language. The **Face-CNN** model is implemented to extract and identify faces. Finally, **Hybrid CNN-ML Word** Model combines the facial detection model with the first model's high accuracy and the second model's fast speed to make a definitive call on how to recognize and classify the Arabic sign language's unique movements.

The fifth phase is training and evaluation. The training of deep learning models usually includes a large amount of classified dataset. The special dataset is divided into training, validation, and test sets. The use of appropriate loss functions and optimization techniques is to train the model. The **LDA-CNN Alphabet** model has an accuracy rate of 99.9% in classifying letters. Similarly, the **LDA-CNN Word** model achieves an accuracy of up to 98.9% in classifying words. On the other hand, the **Hybrid CNN-ML** model surpasses both with an even higher accuracy rate of up to 99.9%, all while maintaining efficient processing speed.

The significance of this work in Arabic Sign Language lies in its potential to improve accessibility, inclusion, and communication for deaf and hard-of-hearing individuals. By advancing sign language recognition technology, it contributes to breaking down communication barriers and promoting equal participation in various aspects of life.

Declaration Associated with this Dissertation

Some of the works presented in this dissertation have been published or accepted as listed below.

List of Dissertation-Related Publications

(First Paper)

Maha S. Altememe, Nidhal K. El Abbadi. "A Review for Sign Language Recognition Techniques", 2021 1st Babylon International Conference on Information Technology and Science (BICITS), 2021

DOI: 10.1109/BICITS51482.2021.9509905

(Second Paper)

Maha S. Altememe, Nidhal K. El Abbadi. "Gesture Interpreting of Alphabet Arabic Sign Language Based on Machine Learning Algorithms", 2022 Iraqi International Conference on Communication and Information Technologies (IICCIT), 2022

DOI: 10.1109/IICCIT55816.2022.10010478

(Third Paper accepted and under publication)

Name of Journal: AIP Publishing

Paper Title: A Hybrid Model Between a One-Dimensional Convolution Neural Network and Machine Learning Algorithms for Arabic Sign Language Word Recognition

Authors:

- Maha Sabri Altememe.
- Prof. Dr. Nidhal K. El Abbadi

(Fourth Paper accepted and under publication)

Name of Journal: Iraqi Journal of Science

Paper Title: Alphabets Arabic Sign Language Recognition Based on A Hybrid Model Combining Linear Discrimination Analysis and A One-Dimensional Convolutional Neural Network

Authors:

- Maha Sabri Altememe.
- Prof. Dr. Nidhal K. El Abbadi

IICCIT 2022
Iraqi International Conference on Communication & Information Technologies

Iraqi International Conference on Communication and Information Technologies IICCIT'22
7-8 September 2022

BEST PAPER AWARD
Awarded to

Maha Sabri Altimeme

University of karbala

Paper Title: Gesture Interpreting of Alphabet Arabic Sign Language Based on Machine Learning Algorithms

Asst. Prof. Dr. Salma A. Mahmoud

IEEE Iraq Section

University of AlKafeel

Acceptance Letter

Author(s) : Maha S. Altememe [1],Nidhal K. El Abbadi [2]

Affiliation : College of Information Technology, University of Babylon [1],Faculty of Education, University of Kufa [2]

Email of Corresponding Author : maha.sabri@uokerbala.edu.iq

Dear (author) We are pleased to inform you that your manuscript

A Hybrid Model Between A One-Dimensional Convolution Neural Network and Machine Learning Algorithms for Arabic Sign Language Word Recognition

after being peer- reviewed , has been accepted for participating in the

4th INTERNATIONAL SCIENTIFIC CONFERENCE OF ALKAFEEL UNIVERSITY , ISCKU 2022

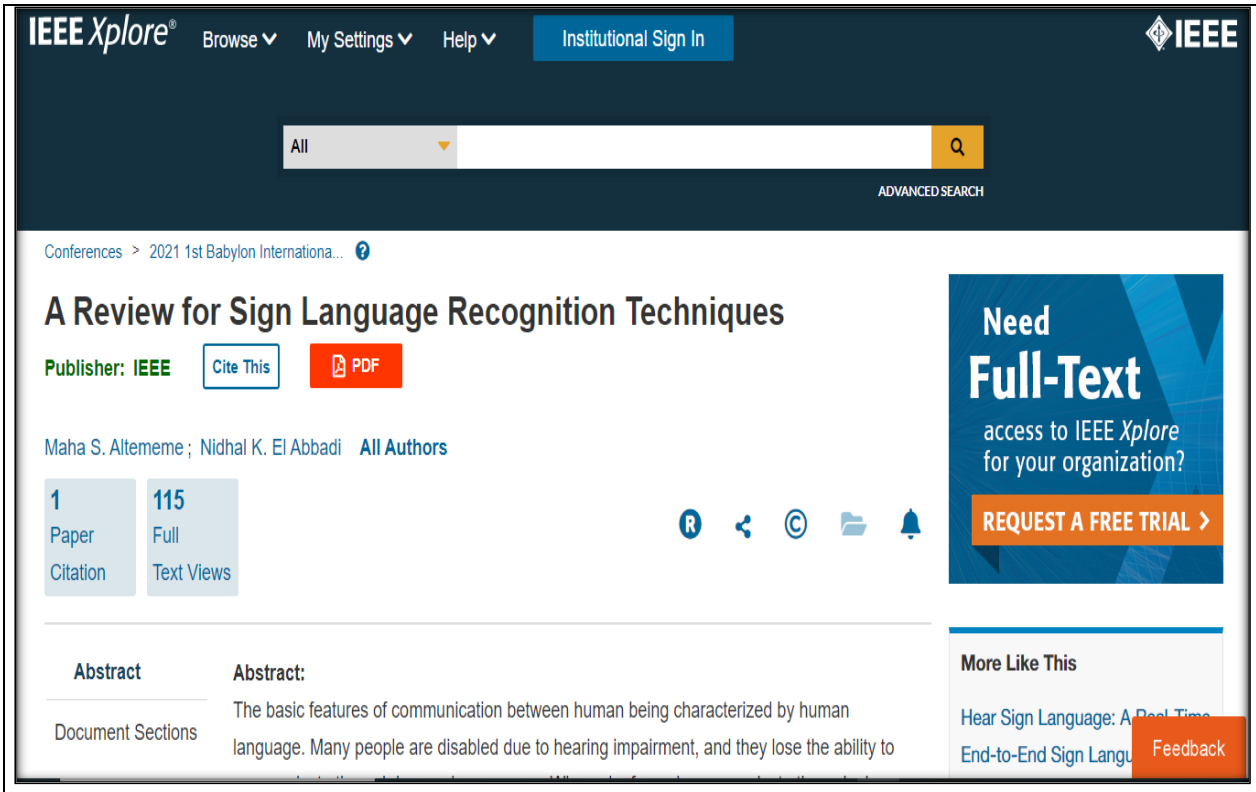
in Al-Najaf Al-Ashraf, Iraq ,December 20th -21st , 2022 . Given that your manuscript will be published after passing. the conference Scopus-indexed journals requirements

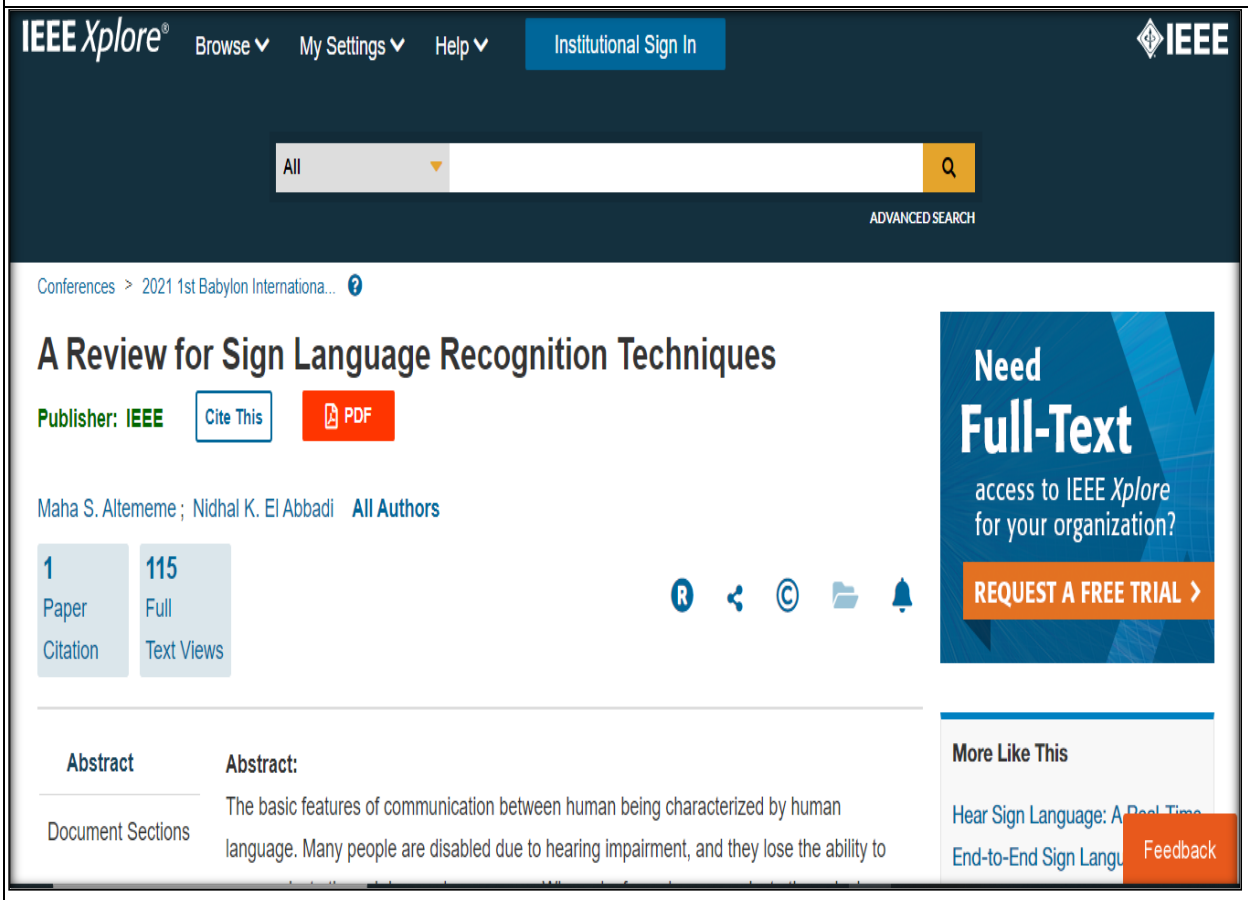
With Best Regards

N. AL-DAHAN

Prof. Dr. N. Al-Dahan
Rector of Alkafeel University

Scopus AIP Publishing


IEEE Xplore
Browse
My Settings
Help
Institutional Sign In
IEEE
All
ADVANCED SEARCH
Conferences > 2021 1st Babylon Internationa...
A Review for Sign Language Recognition Techniques
Publisher: IEEE
Cite This
PDF
Maha S. Altememe ; Nidhal K. El Abbadi
All Authors
1
Paper
Citation
115
Full
Text Views
Need
Full-Text
access to IEEE Xplore
for your organization?
REQUEST A FREE TRIAL
Abstract
Document Sections
Abstract:
The basic features of communication between human being characterized by human language. Many people are disabled due to hearing impairment, and they lose the ability to
More Like This
Hear Sign Language: A
End-to-End Sign Langu
Feedback


IEEE Xplore
Browse
My Settings
Help
Institutional Sign In
IEEE
All
ADVANCED SEARCH
Conferences > 2021 1st Babylon Internationa...
A Review for Sign Language Recognition Techniques
Publisher: IEEE
Cite This
PDF
Maha S. Altememe ; Nidhal K. El Abbadi
All Authors
1
Paper
Citation
115
Full
Text Views
Need
Full-Text
access to IEEE Xplore
for your organization?
REQUEST A FREE TRIAL
Abstract
Document Sections
Abstract:
The basic features of communication between human being characterized by human language. Many people are disabled due to hearing impairment, and they lose the ability to
More Like This
Hear Sign Language: A
End-to-End Sign Langu
Feedback

بسم الله الرحمن الرحيم

العنوان :جامعة بابل \كلية تكنولوجيا المعلومات

م/تعاون مشترك

نؤيد طلب طالبة الدكتوراه (مها صبري التميمي) في تصميم برنامج لتمييز لغة الاشارة الخاصة للصم ..ويبدي مركز الحسين التخصصي للصم الاستعداد للتعاون لإنجاز هذا المشروع والذي سيعود بالفائدة على شريحة من الصم وذويهم في العراق ونقدم المساعدة من خلال توفير المصادر والبيانات المطلوبة للطالبة .

٢٧٥٩

١٢/١٢

سفير مترجمي لغة الاشارة

باسم العطواني

مدير مركز الامام الحسين (ع) التخصصي للصم

هيئة الصحة والتعليم الجامعي
AUTHORITY OF HEALTH & HIGHER EDUCATION

أكاديمية الإمام الحسين (ع) التخصصية للصم

العدد : ١٤
الموافق :١٧ / رجب/١٤٤٤هـ
التاريخ : ٩/ ٢ /٢٠٢٣م

٤٤٠
٢٠٢٣/٢/١٤

الى / كلية تكنلوجيا المعلومات – جامعة بابل

الموضوع / نظام ترجمة لغة الإشارة الالي

اطلعنا على نتائج العمل الذي أنجزته الباحثة مها صبري والخاص بإيجاد نظام آلي لترجمة وتمييز لغة الإشارة العربية.. وبهذا الصدد نوضح ما يلي:

١. ان السيدة مها صبري راجعت أكاديمية الامام الحسين علية السلام التخصصية للصم قبل ان تبدأ العمل، للحصول على استشارة فنية عن أهمية هذا العمل، وابدى المختصون في المركز اهتماما كبيرا بهذا المقترح لما له من فائدة كبيرة على شريحة الصم والبكم في الوطن العربي ونظرا لعدم وجود أنظمة متطورة تعنى بهذا الموضوع.
٢. تم تقديم النصح والإرشاد وكذلك تم توفير بعض المتطلبات المهمة لإنجاز هذا العمل من قبل المركز وعلى سبيل المثال توفير فديوات للغة الإشارة سجلها الخبير المختص عضو مجلس ادارة المنظمة العربية لمترجمي لغة الإشارة العربية الأستاذ باسم العطواني والذي خص بها هذا المشروع.
٣. تابع خبير لغة الإشارة المعروف السيد باسم العطواني الكثير من مراحل العمل وابدى الملاحظات والعون الفني حسب تخصصه.
٤. ان ما تم إنجازه يعتبر عملا مبتكرا من النواحي الفنية في معالجة لغة الإشارة العربية وخصوصا جانب الكلمات منها، وهو يلبي جانب كبير من احتياجات شريحة الصم والبكم فضلا عن كونه خطوة مهمة في طريق انجاز مشروع متكامل مستقبلا.
٥. ان مثل هكذا مشاريع سيكون لها مردود إيجابي على تواصل شريحة الصم والبكم مع المجتمع وإمكانية انخراطهم بسوق العمل بسهولة فضلا عن إمكانيات التحصيل الدراسي التي ستكون أسهل وأيسر.
٦. نؤكد بهذا الصدد على دعمنا وتشجيعنا لكلية تكنلوجيا المعلومات في تطوير هذا الإنجاز الرائع.. وكذلك نشجع السيدة أعلاه ومشرفها على الاستمرار بتطوير العمل ليكون شاملا لكل جوانب لغة الإشارة العربية.

تمنياتنا للباحثة بالتوفيق

سفير مترجمي لغة الإشارة
باسم شمران العطواني
م/اكاديمية الامام الحسين (ع) التخصصية للصم

أكاديمية الإمام الحسين (ع) التخصصية للصم

الدراسات العليا
اجراءات الطلبة

العراق | كربلاء المقدسة | الحي الحسيني
+964 743 500 5147 | +964 781 602 4873 | imam.hussein.academy.for.the.deaf@gmail.com

Table of Contents

List of Figures		
Figure No.	**Title**	**Page No.**
2.1	Most signs alphabet for ArSL	**15**
2.2	Feature Extraction Techniques	**20**
2.3	The Process of LDA between two classes	**22**
2.4	The Process LDA in class	**23**
2.5	The Process of LDA	**25**
2.6	The Haar-like Features	**27**
2.7	The concepts of Integral Image + Haar-like Features	**28**
2.8	The AdaBoost algorithm process	**29**
2.9	The Cascade Classifier	**30**
2.10	The Original shape and corrected landmarks	**31**
2.11	The algorithms in Machine Learning	**33**
2.12	The performance of Random Forest	**36**
2.13	One-Dimensional Convolution Neural Network Layers	**40**
2.14	1D-CNN visualization	**41**
2.15	Representation of a convolutional layer visually	**43**
2.16	Convolution Layer Operations	**44**
2.17	The pooling Layer Operations	**45**
2.18	Types of Activation Function	**47**
2.19	The Confusion matrix.	**49**
3.1	Three Basic Concepts of Hearing-Impaired Communication Frameworks**.**	**54**
3.2	The Flow Diagram of the General Proposal System.	**55**
3.3	Input Sample of the collected dataset ArSL	**56**
3.4	Input sample of the FaceDataset	**57**
3.5	Input Sample of the collated dataset.	**58**
3.6	A Block Diagram for All Models.	**65**

List of Tables

List of Abbreviations	
Abbreviation	**Meaning**
1DCNN	One-Dimensional Convolution Neural Networks
2DCNN	Two-Dimensional Convolution Neural Networks
ArSL	Arabic Sign Language
ASL	American Sign Language
BSL	British Sign Language
CNN	Convolution Neural Network
DT	Decision Trees
HSI	Hearing And Speech Impaired
KNN	K-Nearest Neighbors
LDA	Linear Discriminant Analysis
Leaky ReLU	Rectified Linear Unit
LMC	Leap Motion Controller
MAE	Mean Absolute Error
MLP	Multilayer Perceptron
NB	Naive Bayes
ReLU	Rectified Linear Unit
RF	Random Forest
RMSE	Root Mean Square Error
SGD	Stochastic Gradient Descent
SL	Sign Language
Tanh	Hyperbolic Tangent
VGG	Visual Geometry Group
WHO	World Health Organization

CHAPTER ONE

General Introduction

1.1 Introduction

Sign language is a type of communication used by hearing-impaired people all across the world, with significant variances between countries [1]. Currently, hand gestures are very important for the communication of Hearing and Speech Impaired (HSI) people. There are many features related to sign language. Sign language has five main components: shape, position, gesture, orientation, and non-hand gestures.

Deaf/dumb and hearing-impaired people face problems in communicating with others. Sign language helps a deaf person to communicate with others and deal with communication technology [2]. Hand gestures are a promising component of human-computer interaction, and they are used in the extremely practical application of sign language recognition [3].

This language is comprised of a set of gestures, hand movements, facial expressions, and head movements used to represent letters and words, with each letter of the alphabet having a corresponding sign [4]. More specifically, there are two types of hand gestures, static gestures, and dynamic gestures, static time-independent gestures are those in which the hand position does not change during the gesturing period, and dynamic time-dependent hand gestures, where the hand position changes continuously concerning time [5].

Machine learning and deep learning algorithms enable computers to learn from data and make predictions or decisions without being explicitly processing. In the context of sign language, machine learning techniques have been applied to develop models that can understand and interpret the meaning of different sign gestures [6].

Deep learning algorithms are employed at learning complex patterns and representations from large amounts of data. In sign language classification, deep learning models are trained on extensive dataset consisting of sign language videos or images. These models learn to extract meaningful features and capture the

temporal dynamics of sign language gestures, allowing them to accurately interpret and classify different signs [7].

Deep learning models for sign language often benefit from transfer learning, where pre-trained models trained on largescale visual recognition tasks are fine-tuned or used as feature extraction. By leveraging the knowledge and representations learned from extensive image datasets, these models can be adapted to sign language tasks, reducing the need for large sign language-specific datasets [4, 8].

While deep learning has shown promising results in sign language classification, there are still challenges to overcome. Obtaining diverse and representative sign language datasets, addressing variations in sign language dialects and individual signing styles, and ensuring real-time performance on resource-constrained devices are ongoing research areas [9].

1.2 Problem Statement

According to data from the World Health Organization (WHO), there are more than 1.5 billion individuals worldwide who are either deaf or have varying degrees of hearing loss [2]. Subsequently, extensive research has been conducted to identify the issues and challenges that people with disabilities face in society [3]. For this reason, the problems recognized in SL recognition systems are listed through two important aspects that are relevant to this dissertation and are listed below:

In the social aspect, the WHO reports that the percentage of deaf people has increased to 13% of the world's population, consisting of the hard of hearing and the deaf. Numerous issues faced by individuals with disabilities prevent them from communicating with others in society [2]. As a result, those who have hearing impairments have fewer opportunities to study and live a normal life than those who

do not. Hence, researchers who work with the deaf and hard of hearing are very interested in developing solutions for them.

In the scientific aspect, indeed, the automatic interpretation of ArSL gestures is a complex and challenging problem in the scientific domain. While there have been significant advancements in recent years, the performance quality of SL recognition methods still varies and can be relatively low. One particular unresolved problem in ArSL gesture recognition is the association of facial expressions with the movement of the hands. Facial expressions play a crucial role in conveying meaning and adding nuance to sign language communication. Understanding and accurately capturing these associations between facial expressions and hand gestures in ArSL poses a unique challenge. Addressing this problem requires developing robust techniques that can effectively integrate and analyze both hand movements and facial expressions in ArSL. This may involve using multimodal approaches that combine computer vision techniques with facial expression recognition, machine learning, and possibly other technologies. As research in this area progresses, the solution to these problems will contribute to the development of more accurate and comprehensive systems for interpreting and translating ArSL gestures.

1.3 The Challenges of the Dissertation

Although numerous researchers have made significant progress in detecting Sign Language recognition, there are still several challenges that need to be addressed to achieve more results that are precise. Some of these challenges are:

1. Signs overlapping: Sometimes signs overlap with each other, making it difficult to determine the meaning of a sentence.

2. Movement: It should be noted that sign language is a dynamic language, and rapid hand and body movements can change the intended meaning.
3. Lack of sufficient data sets: Training smart systems used in sign language recognition requires the existence of sufficient and diverse data sets, which may be available in a limited way in some sign languages.
4. One unresolved limitation in ArSL language recognition is the association of facial expressions with hand movements, which is critical to accurately interpreting the meaning of ArSL words.

1.4 Aims of the Dissertation

The dissertation aims to contribute towards creating innovative solutions that facilitate communication and social integration for both deaf and hearing individuals, ultimately fostering greater inclusivity and confidence in society.

The technical aims are:

1. Developing a system to identify the alphabet signs of ArSL from both images and videos of hand gestures.
2. Creating a recognition system that can interpret word gestures in ArSL by analyzing images and videos of hand gestures with facial movements and converting them into meaningful words.
3. Constructing a new dataset that contains a comprehensive set of ArSL words. This dataset can be used to create a dictionary that facilitates the identification and understanding of ArSL words.

1.5 The Contributions of the Dissertation

The contributions of this dissertation are illustrated as:

1. Developing an accurate and robust system for recognition of ArSL gestures: This contribution involves designing and implementing a computer vision-based system that can accurately and efficiently recognize ArSL gestures. The system should be capable of processing images or videos of hand gestures and extracting meaningful information from them.
2. Integrating facial expression analysis for enhanced ArSL interpretation: To improve the understanding and interpretation of ArSL, incorporating facial expression analysis into the recognition system can be beneficial. To propose a new technique to construct a CNN by hybrid with ML algorithms to identify important features for employment in the high-accuracy classification.
3. Creation of an ArSL small dictionary using a new database of Arabic words: The dissertation aims to build a new database comprising a set of Arabic words in ArSL. This database serves as the foundation for creating an ArSL small dictionary, facilitating word search and enhancing accessibility for both deaf and hearing individuals.

1.6 The Scope of the Dissertation

The scope of this work includes many types of video files to generate a video interpreted in Arabic Sign Language (ArSL). The establishment of datasets based on ArSL video sign data acquired from three sources: the standard ArSL lexicon, the compilation of ArSL videos obtained from Imam Al-Hussain Specialist Center Deaf "أكاديمية الامام الحسين التخصصي للصم", The Qatari Center for the Deaf "المركز القطري للصم" and video signs from different websites. The research on ArSL recognition can be

divided into three levels: the recognition of the Arabic signs alphabet and gestures words.

1.7 Related Works

A review of previous studies is applied to Sign Language recognition and in particular to ArSL is conducted. When relevant works in the context of sign language were selected, several criteria were considered to ensure the relevance and quality of the selected studies. These criteria are focusing on research that specifically deals with sign language classification, interpretation, or related tasks using machine learning or deep learning techniques. Recently, the selected works have been identified within the period from 2015-2022 to cover aspects of the subject.

This work also searched for studies that provide new algorithms, architectures, data sets, or approaches that work to develop the field of sign language recognition using machine learning or deep learning, based on the dataset for Arabic sign language only. Table (1.1) Overview of SL recognition in most research studies. According to the techniques in this dissertation, this section is divided into two parts:

1.7.1 Related Work with Machine Learning

In [10], an automatic system for translating ArSL was developed. Here, the procedure begins with frame-by-frame transcoding of the video, then uses the Euclidian distance function to determine which frame is the best. After locating the group of letters, at every good frame, the system crops the image to resize the hand region. Depending on the observed orientation of the hand and wrist, this method splits the alphabetic ArSL into three groups. Edge detection and feature vector construction are used for feature extraction. It

aims to identify and highlight edges or boundaries in the data, which can be useful for distinguishing different objects or patterns. Then, the Minimum Distance Classifier (MDC) and the Multilayer Perception neural network (MLP) are employed for recognition with an accuracy of 91.3% and 83.7%, respectively. The first classifier mentioned is the MDC. It is a simple classification algorithm that calculates the distances between feature vectors and class prototypes or centroids. The second classifier mentioned is the MLP. It is a type of artificial neural network with multiple layers of interconnected nodes or neurons.

Another study focuses on the development of a system that utilizes the Leap Motion Controller (LMC) for recognizing Arabic letter signs [11]. The LMC is used for data acquisition, capturing hand movements and gestures. The study proposes a method that applies a subset of twelve features extracted from the LMC data. The selected features are used as input for classification models, namely Naive Bayes (NB) and Multilayer Perceptron (MLP). These models are trained on the extracted features and are capable of classifying Arabic letter signs. The reported accuracy of the classification is up to 90%.

In [12], a method that comprises four stages is outlined. The hand segmentation is the first stage which is based on using the face color tone to segment the hand. The skin blob is used for recognition and tracking the hand in the second stage. The third stage involves the extraction of geometric features, including the orientation of the hand axis, hand speed, hand center, and hand axis, before using them in the final stage for classification based on the Euclidean distance. This system is evaluated using a dataset consisting of 30 isolated words. The performance of the proposed method was 97% of accuracy in recognizing these words.

In [13], a vision-based signs identification tool that can identify static and dynamic gestures made with one or both hands from a live video is suggested. The feature vector is extracted from the gesture sequence, and these features are acquired to be used for classification using SVM. The key frame is isolated to reduce computation speed to a large range using Zernike moments and improve the process for co-articulation removal of hand spelling alphabets. Subsequently, the accuracy is 89% for single-handed dynamic words and 91% for distinctive static gesture alphabets.

In [14], the Microsoft Kinect is performed to develop a dynamic ArSL identification system. The Ada-Boosting method, along with the use of the Decision Tree and Bayesian Network machine learning algorithms, is what allows the system to recognize objects better. The technology was put to the test with 42 Arabic gestures related to medicine. Data from the testing show that the suggested system's recognition rate for the Decision Tree classifier was 91.18%. Meanwhile, for the Bayesian classifier, it was 92.50%, and for the Ada-Boosting method, it was 93.7%.

1.7.2 Related Work with Deep Learning

In [15], this work describes the use of a 3Dimantional Convolutional Neural Network for identifying twenty-five signs from the dictionary of ArSL. The approach involves processing videos, which are divided into multiple frames, each with a specific rate determined by a scoring mechanism. To capture the essence of the intended sign, only frames with the highest priority are selected. These prioritized frames are then used as input data for the 3DCNN. Additionally, depth maps are utilized to provide information for the

recognition algorithm. According to the results presented in the reference, the proposed method achieved an average accuracy of 85% for new, unseen data. This indicates that the model performed well in correctly identifying signs that it had not encountered during training. For observed data, which likely refers to the signs used in the training set, the accuracy was reported as 98%, indicating a high level of performance in recognizing signs that the model was already familiar with.

In [16], a CNN-based automatic method for identifying ArSL letters as well as numbers is introduced. Seven layers make up the proposed CNN. The first four levels concentrated on feature extraction and the final three layers on classification. Around 7869 RGB color images of 256 x 256 pixels were utilized, along with 5839 pictures of the 28 letters and 2030 images of digits (from 0 to 10) of the ArSL. The results were compared with the results of the k-nearest neighbors (KNN) algorithm and support vector machines (SVM). The recognition rate is about 90%.

In [17], an automatic ArSL letter recognition and voice conversion for each letter is proposed. A dataset of 100 images was used for the training phase, and 25 images were used for the testing phase for each of the 31 ArSL hand signs. The suggested CNN architecture comprises two layers, two max pools, and 100 epochs. This system has up to 90% accuracy.

In [18], a signal classification in real series is presented. This work proposed a brand-new model combines deep convolutional layers (ConvNets) with Long Short-Term Memory (LSTM) layers which was so-called DeepConvLSTM. LeapMotion sensors are then utilized to supply the input data to several typical machine-learning techniques that are intended to

classify American SL. The authors propose a kinematic model of the left and right forearm/hand/fingers/thumb and use basic data augmentation to avoid overfitting to enhance the generalization of the Neural Network (NN). The substantial variety in executing movements amongst participants is one of the study's shortcomings that could lower categorization performance. This technique's accuracy rate is 91.1%.

In [19], a model using a combination of ontology and deep learning techniques to convert ArSL into different meanings is created. Ontology is used to address several challenges in sign language processing. The convolution neural network (CNN) was trained and tested using a variety of Arabic words as well as static Arabic alphabetic signs. This approach had a classification accuracy of up to 88.87%. This system's flaw is a significant rate of inaccurate recognition for some Arabic letters, which may exceed 30%.

In [20], a system for identifying ArSL that makes use of the CNN model using RGB images as input and automatically recognizes 28 letters is proposed. In this paper, the authors suggest a new convolutional neural network-based architecture that can recognize ArSL letters automatically when fed with a real dataset. An evaluation of the proposed method's effectiveness and robustness in comparison to other methods is necessary to validate the scheme. In this work, the CNN model was put to the test on 10810 images. Both the model's error rate and accuracy were measured; during the training and testing phases, the accuracy rate increased while the error rate decreased, and to obtain 92.9% recognition accuracy.

Table 1.1: A summary of the reviewed literature

No.	Ref.	Years	Types of Signs	Input Device	Modality	Features	Technique	Accuracy
1.	[10]	2011	Alphabet Arabic	Camera	Videos	Euclidean distance	MLP & MDC	91.3% 83.7%
2.	[11]	2015	Alphabet Arabic	LMC	LMC	Naive Bayes (NB)	MLP	90%
3.	[12]	2018	30 isolated words	Camera	Videos	Geometric Features	SVM & ANN	97%
4.	[13]	2019	28-handed static &10 words dynamic hand	Camera	Videos	extraction feature vector from the gesture sequence and Zernike moments	SVM	91%
5.	[14]	2019	42 Arabic gestures	Micros oft Kinect	RGB	Ada-Boosting strategy	Decision Tree & Bayesian Network	93.7 %
6.	[15]	2017	25 letters of Arabic sign	Camera	Videos	CNN	3DCNN	85%
7.	[16]	2019	11 numbers & 28 letters	Camera	RGB	LeNet-5	CNN	90.2%
8.	[17]	2020	31 alphabets	Leap Motion or Xbox	Images &videos	CNN	CNN	90%

9.	[18]	2020	28 letters	Leap Motion sensors	RGB	DeepConvLSTM	NN	91.1%
10.	[19]	2020	10 Arabic words	Camera	RGB	CNN	Deep learning	88.87%
11.	[21]	2020	28 letters	Camera	Grey	CNN	CNN	88.87%
12.	[20]	2020	28 letters	Camera	RGB	CNN	CNN	92.9%

1.8 Dissertation Outline

The dissertation's remaining chapters are structured as below:

Chapter Two: provides a comprehensive description of the basic principles that are used in this dissertation.

Chapter Three: Explains the basic processes of developing the suggested approach for exploring ArSL utilizing a deep learning neural network.

Chapter Four: Explains the use of the suggested system on a set of actual video data and the experimental results of this implementation, in addition to each step.

Chapter Five: summarizes the findings derived from this dissertation and makes suggestions for future work.

CHAPTER TWO

Theoretical Background

2.1 Overview

Human culture and language are fundamental to human communication and are passed down from generation to generation, varying greatly within and between cultures. People with hearing disabilities may be labeled as deaf due to their inability to communicate using spoken language, leading to communication challenges with others [22]. Sign Language (SL) is a significant way for the deaf community to communicate, and one application of human-computer interaction is the SL recognition system, which converts sign language into text or spoken language [23].

In Arabic communities, the deaf community most commonly uses Arabic Sign Language (ArSL) as their native tongue, which was formally launched by the Arab Federation of the Deaf in 2001 [24]. However, ArSL is still in the developmental stage and relies on the shapes of letters and words [25]. It is important to note that sign languages, including ArSL, are not universal and vary between different regions and cultures. For example, American Sign Language (ASL) differs from British Sign Language (BSL), and both are distinct from ArSL. This variation in sign languages, similar to the variation in spoken languages, reflects the diversity of human culture and communication [26].

Thus, it is important to develop accurate and reliable sign language recognition systems to improve the communication and integration of people with hearing disabilities into society [27].

However, developing such systems requires a deep understanding of the unique features and structures of sign languages, as well as the ability to accurately capture and recognize subtle variations in hand gestures, facial expressions, and body movements, achieving these goals can greatly benefit the deaf community and facilitate their full participation in society [8].

2.2 Sign Language Overview

Sign language is a visual language that uses hand gestures, facial expressions, and body movements to communicate meaning. It is used primarily by people who are deaf or hard of hearing but can also be used by hearing individuals who wish to communicate with deaf individuals. Each sign language has its unique grammar, vocabulary, and syntax [25].

Sign language can be conveyed in two main modalities: visual-manual and tactile. In visual-manual sign language, communication is conveyed through hand gestures and facial expressions, while in tactile sign language, communication is conveyed through touch, usually on the hands or back [28].

Sign language recognition systems are being developed to translate sign language into spoken language or text. These systems use computer vision and machine learning techniques to capture and recognize the hand gestures, facial expressions, and body movements used in sign language [29].

Overall, sign language is a rich and diverse language that provides a means of communication for millions of people around the world who are deaf or hard of hearing. The development of sign language recognition systems and other assistive technologies can greatly improve the communication and integration of people with hearing disabilities into society [30]. An ArSL alphabet is illustrated in Figure 2.1.

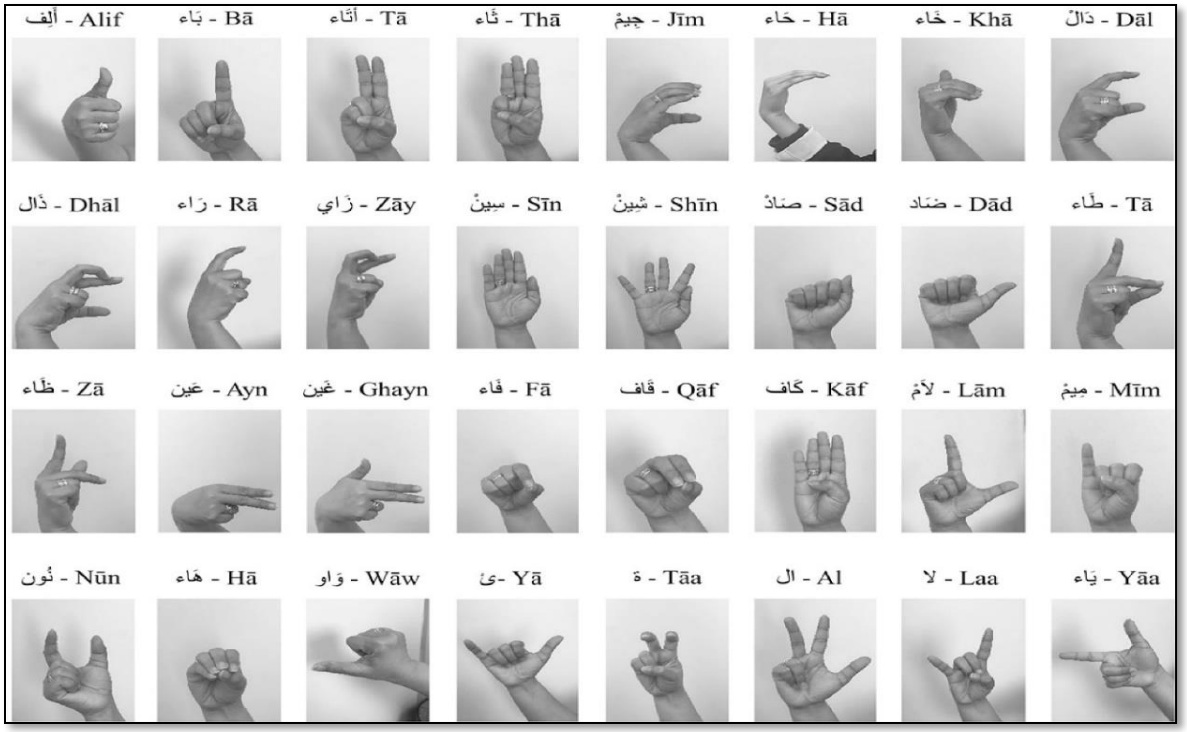


Figure 2.1: Most signs alphabet for ArSL [6].

2.2.1 Gestures

The act of identifying a movement expression for conveying various numbers, phrases, or letters using (hands, head, face, as well as body) with various meanings is known as gesture recognition. Gestures must be widely understood to develop applications that employ gestures or signs virtually and to enhance SL-dependent systems. There are two main categories of hand gestures [8], such as:

- **Static Gestures**: Since gestures are time-independent and do not change hand position while being made, they are less computationally complex. In SL, static gestures are frequently employed to identify between spelling fingers since they are thought of as a single group of images that represent each image in a separate frame [31].

- **Dynamic Gestures**: Dynamic hand gestures are more sophisticated yet appropriate for real-time environments since the hand position continuously varies with time. Then, the hand movement is recorded over time and is not based on only one frame but on multiple frames. Sequences of image frames are required as input for the data-collecting process for a dynamic sign to be extracted from the video [32].

Given the hand gestures' significance in communication, SL translations into natural language require the assistance of an interpreter to accurately and appropriately translate signs into text or meaningful speech, a mechanism for signal differentiation must be provided. As a result, there are ways needed to accomplish this because it is difficult to recognize motions [33].

2.2.2 Finger-spelling

The finger's alphabet is a helpful technique for SL that helps deaf people learn. This method equals writing the alphabet with one hand or utilizing names that are known to the deaf population but do not have a sign. The meaning of the alphabet written with fingers is clearer since each of the 28 letters uses a certain movement with the hand's fingers.

The finger-spelling technique is one of the visual physical communication approaches. It is centered on writing the alphabet's shapes in the air rather than on paper, expressing a manual manner, and simulating written language [29].

The alphabetical technique is considered a crucial way to communicate with the deaf since it is a vital component of communication generally, especially when discussing words, persons' names, or addresses without any established signs. The quickest way for deaf people to see words is by finger-spelling, which is suited for reading [31].

As a result, the rational for utilizing the fingerspelling technique involves the fact that training in this technique will not need the hearing-impaired student to apply much effort since it will be an implementation of what they learned using the alphabet's writing skill. This means that the student must first practice lips reading and pronunciation activities, write words and letters, and then move on to the training phase before mastering this approach [31, 29].

2.3 Digital Image Enhancement

Digital image enhancement refers to the process of improving the quality of an image by applying various algorithms and techniques to remove noise, increase contrast, and sharpen details. Image enhancement is a crucial step in many computer vision applications, including sign language recognition [34]. In sign language recognition systems, image enhancement techniques are used to preprocess the input image or video frames and enhance the quality of the images before extracting features and recognizing the sign. Some common techniques used in image enhancement include histogram equalization, noise reduction filters, and edge enhancement filters. These techniques can help to improve the accuracy and robustness of sign language recognition systems by reducing the impact of noise and other image distortions [35]. This dissertation uses the histogram equalization technique.

Histogram equalization is a technique used in digital image processing to improve the contrast and brightness of an image. It works by redistributing the pixel intensities in an image so that they are spread out more evenly across the entire intensity range. The result is an image with higher contrast and a brighter appearance. The basic steps of histogram equalization are as follows [36]:

1. Calculate the histogram of the input image, which is a plot of the frequency of occurrence of each pixel intensity value.
2. Normalize the histogram so that the sum of all histogram values is equal to 1.
3. Calculate the Cumulative Distribution Function (CDF) of the normalized histogram. The CDF represents the probability that a pixel in the image has an intensity less than or equal to a certain value.

$$CDF(r) = \text{sum of } H(i) \quad \text{for } i=0 \text{ to } r / (M \times N) \ldots. (2.1)$$

In Equation (2.1), 'H(i)' represents the histogram value at bin 'i' of the histogram distribution. The histogram represents the frequency or count of occurrences of the random variable 'r' across the matrix. This is a two-dimensional matrix with dimensions 'M' for the number of columns and 'N' for the number of rows.

4. Calculate the new intensity values for each pixel in the image using the CDF. This is done by mapping the old intensity values to the new intensity values based on their position on the CDF.
5. Create a new image using the new intensity values.

The formula for histogram equalization is given by:

$$s = T(r) = \text{round}((L-1) * CDF(r)) \ldots (2.2)$$

In Equation (2.2), s is the new intensity value, r is the old intensity value, L is the maximum intensity level (e.g., 256 for 8-bit images), CDF is the cumulative distribution function, and T is the transformation function.

2.4 Feature Extraction Approaches

Feature extraction is a technique used in machine learning and data analysis to reduce the dimensionality of a dataset. It involves selecting and combining relevant variables from a larger set of raw data to create a smaller set of features that can be used for analysis or modeling [34]. The primary goal of feature extraction is to extract the most informative and useful features from the raw data while reducing the dimensionality of the dataset. By doing so, it helps to improve the accuracy and efficiency of machine learning algorithms and other data processing techniques [37].

Typically, feature extraction involves a combination of mathematical and statistical methods to identify the most important variables or features in a dataset. These features are chosen based on their ability to accurately and distinctly define the underlying patterns or relationships in the data. The resulting feature set is typically simpler and more manageable than the original raw data, making it easier to analyze and process. The characteristics of a good feature are [38]:

- Robust: the feature has similar results under different conditions, such as ambient lighting, lenses, and contrast.
- Salient: the feature must be informative such that it discriminates between the categories of the object of interest.
- Reliable: It corroborates flexible criteria for similar classes (objects).
- Independent: It is unique for the specific class and non-redundant, it is not related to other features.

Here, the two groups of features are:

- **Low-level features:** These are features that are directly extracted from the raw data, such as pixels in an image or raw signal data. They are typically simple, low-level characteristics of the data and can include color, texture, edge orientation, and frequency content. Low-level features are often used as building

blocks for more complex features and are important in many machine-learning applications.

- **High-level features:** These are features that are derived from low-level features and are more complex and abstract representations of the data. High-level features capture more complex characteristics of the data, such as shape, texture, and spatial relationships, and are often specific to the problem domain. They are typically defined using mathematical or statistical techniques. High-level features are often used in more advanced machine learning applications, such as object recognition, natural language processing, and speech recognition.

Figure 2.2 shows most of the algorithms used to extract features in general.

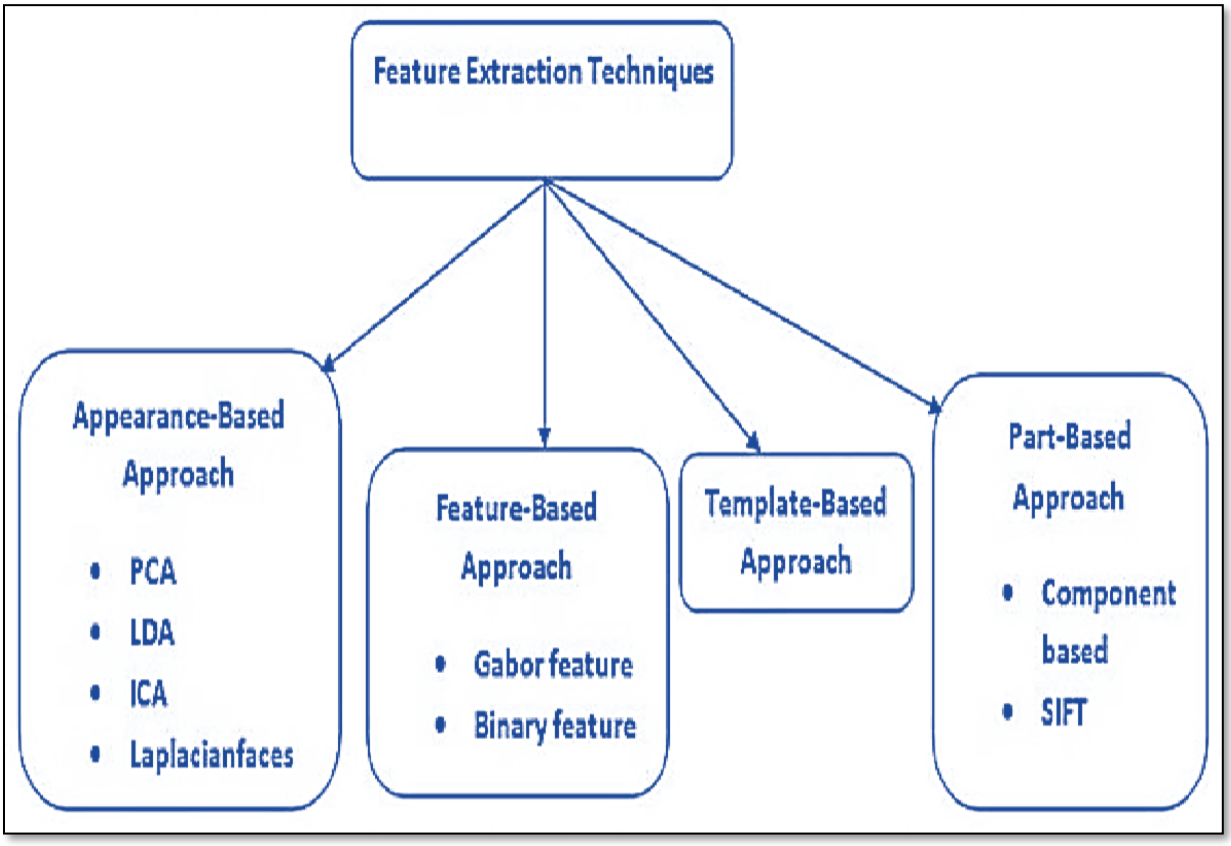


Figure 2.2: Feature Extraction Techniques [39].

2.4.1 Linear Discriminant Analysis Algorithm

Linear Discriminant Analysis (LDA) is a popular technique for dimensionality reduction, particularly in the fields of computer vision and pattern recognition [39]. It aims to reduce the number of features while preserving the most relevant information for classification tasks. LDA works by finding a lower-dimensional representation of the input data that maximizes the separation between classes. This is achieved by projecting the data onto a set of new features that are a linear combination of the original features [34]. The resulting LDA features are ordered according to their discriminative power, with the most significant features appearing first. LDA has proven to be an effective technique for reducing the dimensionality of high-dimensional data and improving the performance of classification algorithms. It is a strong yet straightforward technique for diverse implementations in computer vision as well as pattern recognition communities. This algorithm is one of the types of techniques of the appearance-based approach [40].

The process of applying LDA to images for feature extraction requires careful consideration of various factors, including image size, the complexity of the image features, available computational resources, and classification accuracy requirements. However, when it is applied appropriately, LDA can be a powerful tool for extracting discriminant features and improving classification performance [41]. In some cases, using a smaller image size may improve computational efficiency. LDA can be used to reduce the dimensions of image features, while preserving the most important information for classification tasks [42].

The LDA algorithm is used to project the initial data matrix into a lower-dimensional environment. To do this, three actions had to be taken.

- Determining the reparability between different classes, sometimes referred to as the between-class matrix or between-class variance (i.e., the distance between different classes' means), is the first stage.
- Finding the difference between each class's mean and samples, also known as the within-class variance or within-class matrix, is the second phase.
- Establishing a lower-dimensional environment that minimizes variance inside classes and maximizes variance across classes is required for the third phase.

Overall, LDA is a powerful algorithm for dimensionality reduction and classification, and its computational efficiency makes it a good choice for handling large and complex datasets. The LDA technique's steps are illustrated in Figures 2.3 and 2.4, and after that, a list of the equations that were utilized to create the LDA process is provided [43].

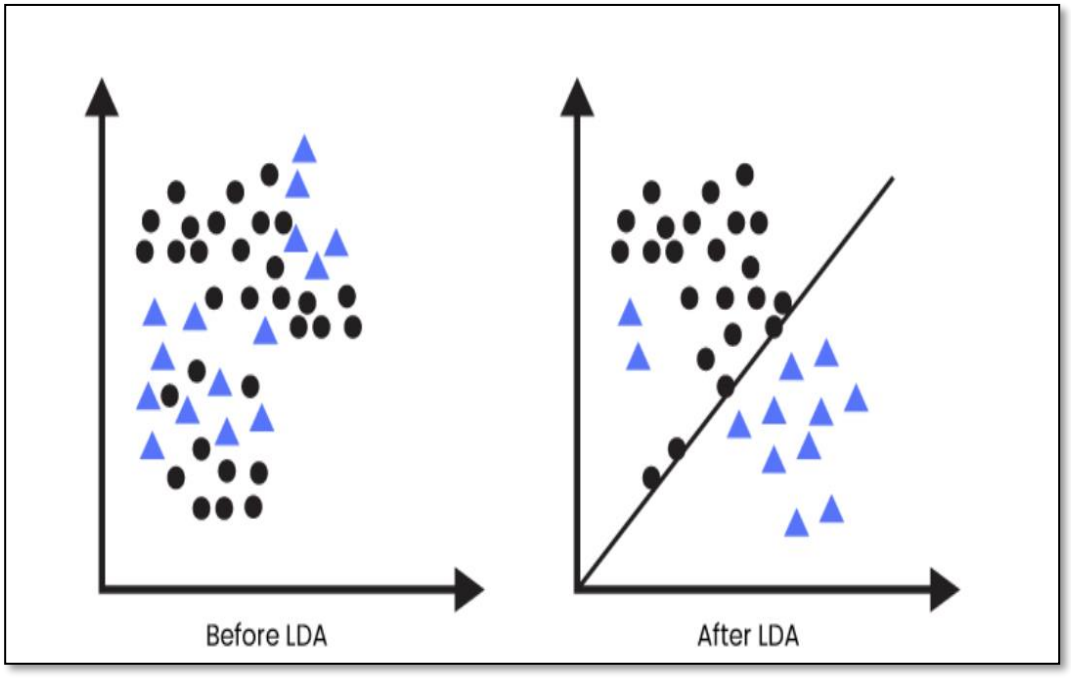


Figure 2.3: The Process of the LDA between one class [40]

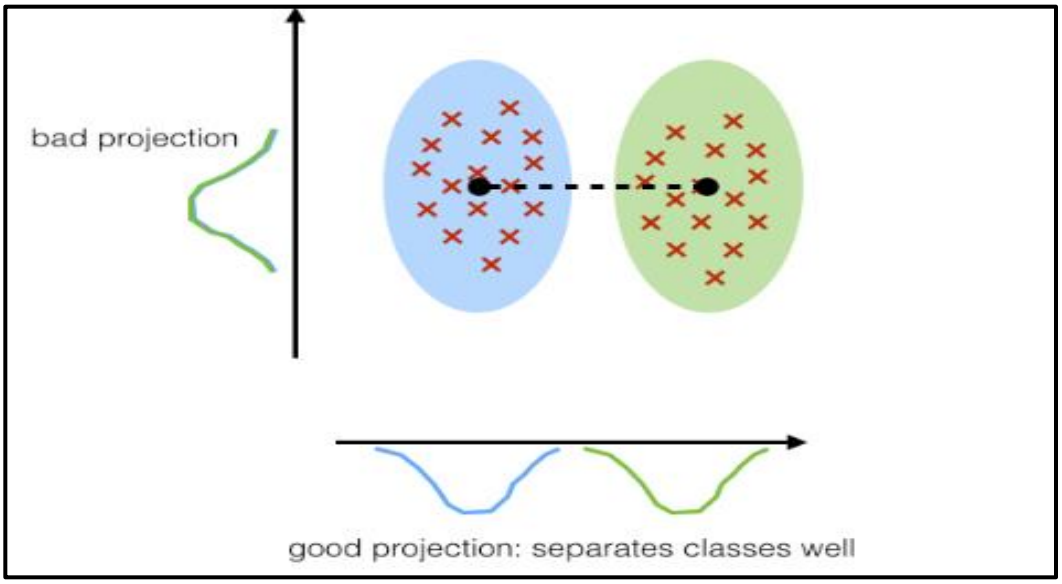


Figure 2.4: The Process of the LDA between two class [43]

Equations (2.3), (2.4), (2.5), (2.6), (2.7), (2.8) illustrate the process of the LDA algorithm:

$$\mu_j = \frac{1}{n_j}\sum_{x_i \in \omega_j} x_i \ldots\ldots\ldots\ldots (2.3)$$

Equation 2.3 calculates the mean vector μj for each class ω_j. μj represents the average values of the input feature vectors x_i belonging to class ωj.

$$\mu = \frac{1}{N}\sum_{i=1}^{N} x_i = \sum_{i=1}^{c} \frac{n_i}{N}\mu_i \ldots\ldots\ldots\ldots (2.4)$$

Equation 2.4 calculates the overall mean vector μ by taking the average of all the input feature vectors xi. N represents the total number of data points, ni represents the number of data points in class ωi, and μi represents the mean vector for class ωi.

$$S_B = \sum_{i=1}^{c} n_i(\mu_i - \mu)(\mu_i - \mu)^T \ldots\ldots\ldots\ldots (2.5)$$

Equation 2.5 computes the between-class scatter matrix S_B. It measures the spread between the mean vectors of different classes weighted by the number of data points in each class.

$$S_W = \sum_{j=1}^{c} \sum_{i=1}^{n_j} (x_{ij} - \mu_j)(x_{ij} - \mu_j)^T \ldots\ldots\ldots (2.6)$$

Equation 2.6 calculates the within-class scatter matrix S_W. It represents the spread within each class by measuring the covariance of the feature vectors around their respective mean vectors.

$$W = S_W^{-1} S_B \ldots\ldots\ldots . (2.7)$$

Equation 2.7 solves the generalized eigenvalue problem by multiplying the inverse of the within-class scatter matrix S_W with the between-class scatter matrix S_B. The resulting matrix W represents the projection direction for discriminant analysis.

$$Y = XV_K \ldots\ldots\ldots . (2.8)$$

Equation 2.8 projects the input feature vectors X onto the lower-dimensional space defined by the eigenvectors V_K. Y represents the transformed data in the lower-dimensional space.

These equations are used iteratively in the LDA algorithm to find the optimal projection direction that maximizes the separation between classes while minimizing the within-class scatter. The transformed data in the lower-dimensional space can then be used for classification tasks. Figure 2.5 and algorithm 2.1 show the implementation of the LDA process.

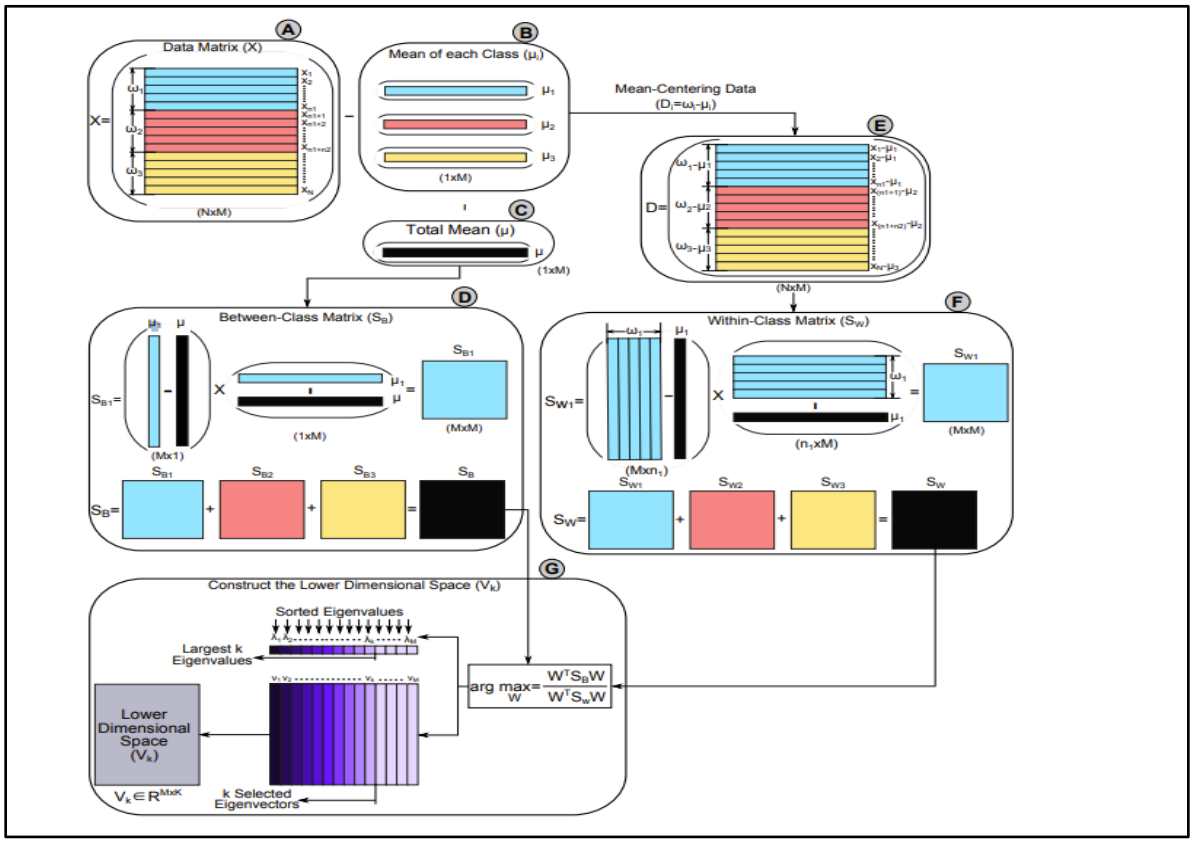


Figure 2.5: the process of the LDA [43]

Algorithm 2.1: Linear Discriminant Analysis
Input: Image
Output: Features of Vector
BEGIN **Step 1:** Read images of people's hands. **Step 2:** Assign a weighted average d-vector to each category in the dataset, using Eq (1): $\mu_j = \frac{1}{n_j}\sum_{x_i \in \omega_j} x_i$ …….. (1) Where: N_J… represents the number of samples of the *i*tch class. **Step 3:** Compute in between-class matrix S_B *(M×M)* using Eq (2)*:* $S_B = \sum_{i=1}^{c} n_i(\mu_i - \mu)(\mu_i - \mu)^T$ …… (2) Where: S_B scatter matrix (in between-class), μ_I … represents the projection of the mean of the *itch class.* μ … Projecting the overall distribution mean across categories **Step 4:** Compute within-class matrix S_W *(M ×M)* using Eq (3): $W = \sum_{j=1}^{c}\sum_{i=1}^{n_j}(x_{ij} - \mu_j)(x_{ij} - \mu_j)^T$ …… (3)

Where: S_W scatter matrix (within-class)

Step 5: Find the scatter matrices' eigenvectors (e1, e2,) and corresponding eigenvalues (λ1, λ2, ad).

Step 6: Eigenvalues determine the descending order of eigenvectors, and the k eigenvectors with the largest eigenvalues are used to represent the column of a d x k dimensional matrix W.

Step 7: Transforming the samples onto the new subspace makes use of the d*k eigenvector matrix. In this case, it can be used with the help of Eq (4).

$$Y=X\times W \qquad \ldots\ldots\ldots\ldots\ldots\ldots (4)$$

where X is a nxd matrix representing the original n samples and Y is an updated n*k matrix reflecting the modified samples.

END

2.4.2 Viola and Jones Algorithm

The Viola-Jones algorithm, created in 2001 by Paul Viola and Michael Jones, is used to identify faces in images [44]. It has been demonstrated to be effective in bringing to the front images of faces and is capable of handling a 45-degree rotation of the face around both the horizontal and vertical axes. The algorithm has demonstrated excellent real-time performance [45].

The integral image, AdaBoost, and cascade structure are the three key ideas that enable it to operate in real-time. The Viola and Jones algorithm's concepts are [46]:

1. **Integral image:** this is a preprocessing step that is used to compute the sum of pixel intensities in a designated rectangle in an image quickly. It works by calculating the cumulative sum of pixel intensities along the x and y axes of the

image. Once the integral image has been computed, the sum of pixel intensities in any rectangular region of the image can be calculated using just four values from the integral image, regardless of the size of the rectangle. This makes the computation of Haar-like features, which are used by the Viola-Jones algorithm to detect faces, much faster [35].

Three types of face detection features, namely two-rectangle, three-rectangle, and four-rectangle features, are used to compute Haar features relatively quickly. The difference in the sums of the pixels within two rectangular sections is the two-rectangle feature. These rectangular areas lie next to one another either horizontally or vertically and are the same size and form [45].

In a three-rectangle feature, the sum of the pixels in the two outer rectangles is calculated and deducted from the sum of the pixels in the central rectangle.

The last feature computes the difference between diagonal pairs of rectangles using four rectangles. These features are displayed in Figure 2.6 and Figure 2.7 describes the principles of the Integral Image and the Haar Features diagram.

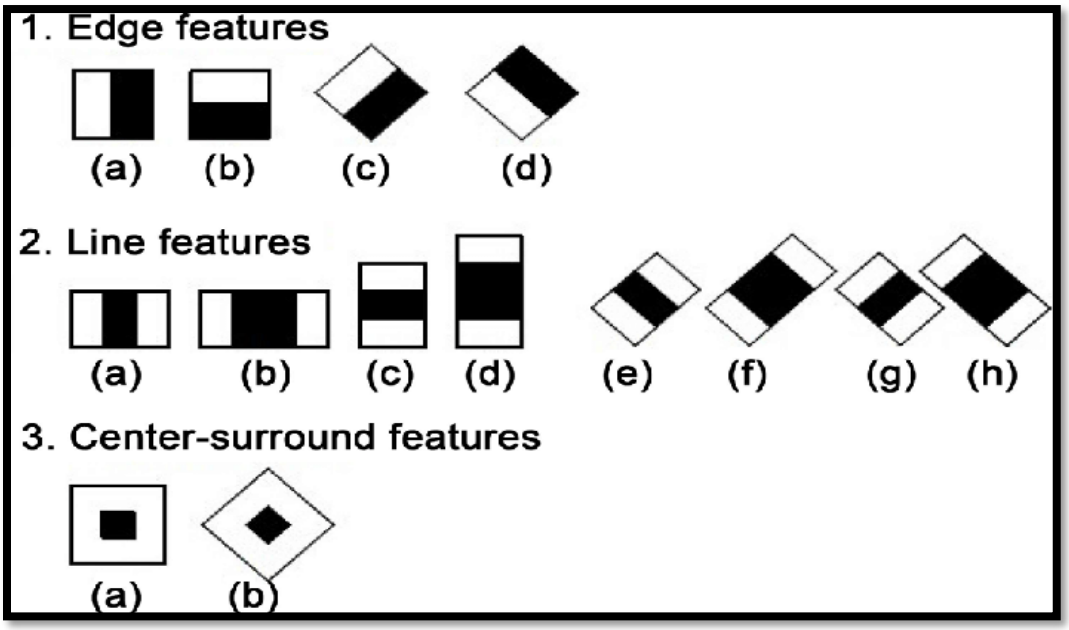


Figure 2.6: The Haar-like Features [35]

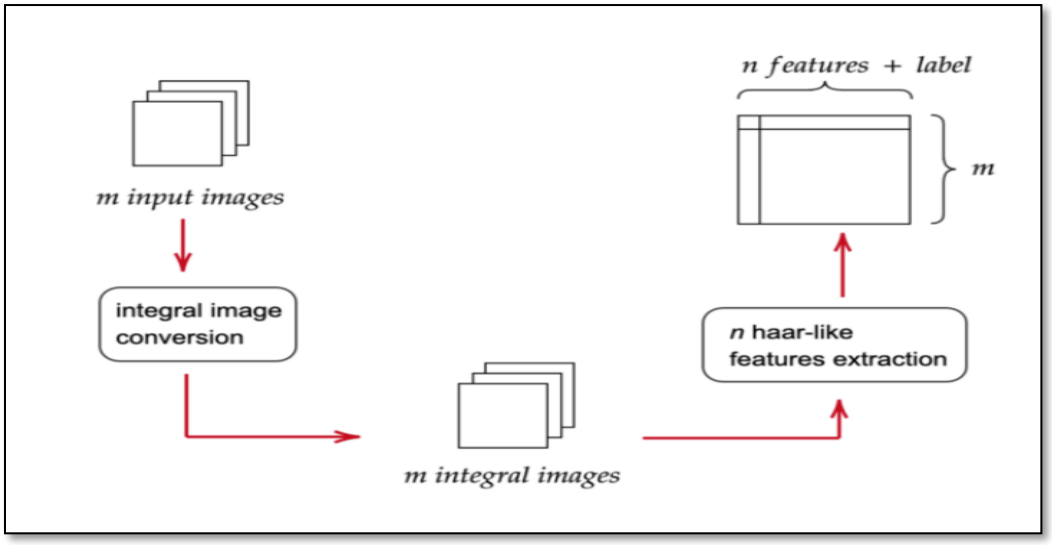


Figure 2.7: The concepts of Integral Image + Haar-like Features [45].

2. **The AdaBoost Learning algorithm:** A large number of Haar features (approximately 180,000) are computed for each window. Most of these features are superfluous, which then leads to using AdaBoost as a redundancy-reduction method. The AdaBoost algorithm can be referred to as a learning classification function that minimizes a large list of features by removing those that are superfluous. This classifier is made up of a weighted combination of weak classifiers. Each window feature functions as a weak classifier [47].

This approach can also be viewed as a feature selection algorithm since it selects the best features from all the available possibilities. The chosen features are the ones that describe a face most accurately. A few hundred features are left after this technique condenses thousands of features [47, 48]. The AdaBoost algorithm is utilized to retrieve the best features from a set of features, as shown in Figure 2.8.

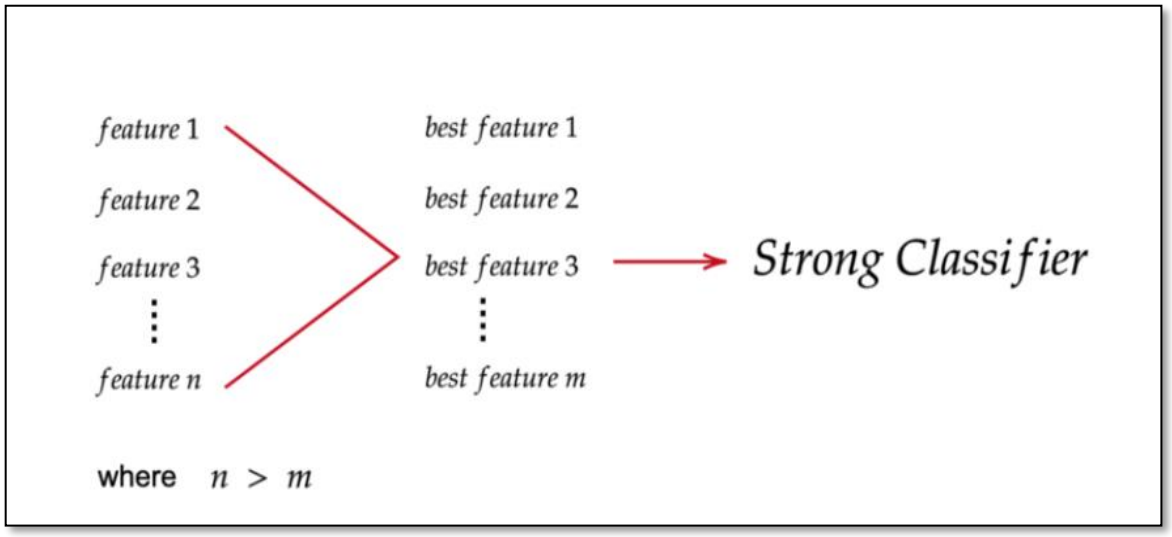


Figure 2.8: The AdaBoost algorithm process [47]

3. **The cascade classifier**: it is a multi-stage classifier that has a speedy and precise detection capability. A strong classifier created by the AdaBoost algorithm makes up each stage. A strong classifier has an increasing number of weak classifiers as it advances through the stages. A cascade of strong classifiers is utilized after choosing the best features in each window to reduce detection time through effective computation. The goal is to build classifiers that are smaller and more effective based on the input image's sub-windows. The best to worst classifiers in the strong classifier is ranked. Every stage lowers the number of false positives or areas that are mistakenly identified as faces. A classifier is created for each stage utilizing a few features. Every level after that adds more and more features, complicating the classifier. The discovered region can either be rejected or moved on to the following level in each stage. As a result, only the area that completes all

stages are categorized as a face. Here, Figure 2.9 illustrates the Cascade process [48].

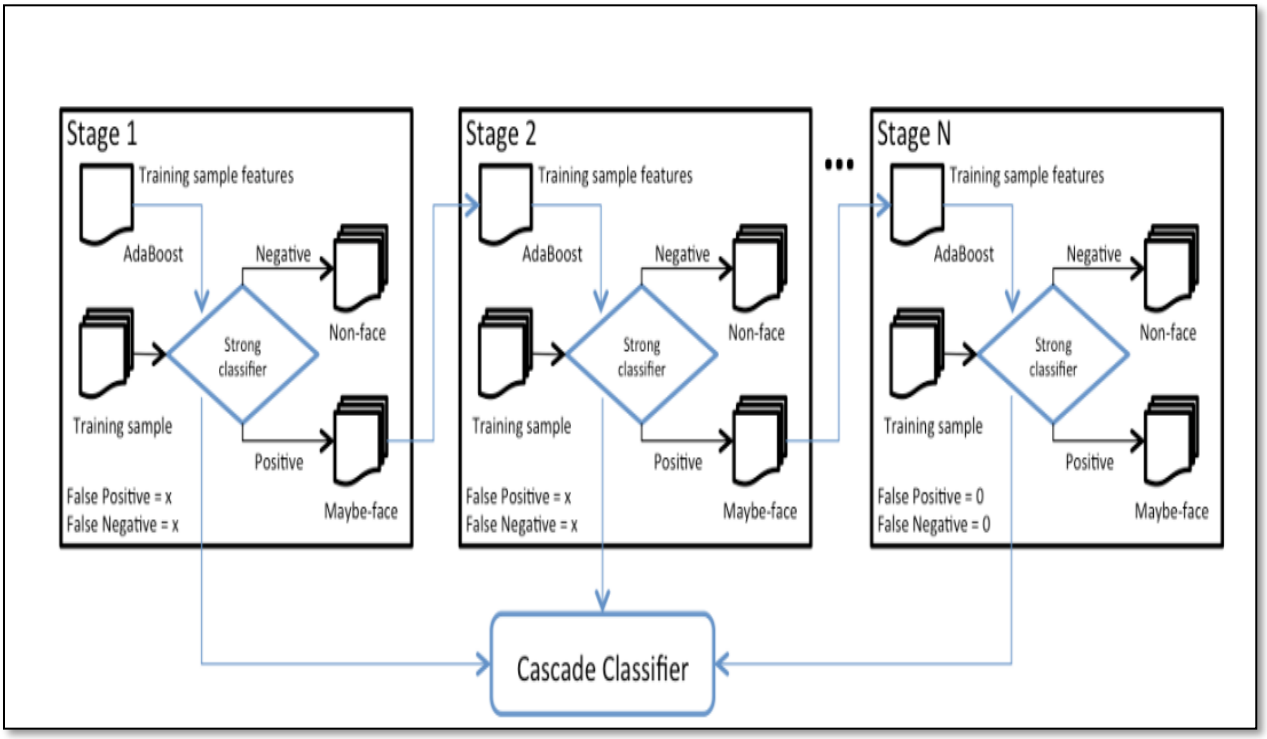


Figure 2.9: The cascade Classifier [49]

It uses the Viola-Jones facial feature algorithm, which uses a combination of Haar-like features to represent texture information in an image. A detailed description of this method and training method can be found in AdaBoost. A landmark is a recognizable natural or man-made feature used for navigation that stands out from its close environment and is often visible from long distances [50].

Facial landmarks are defined as the detection and localization of certain key points on the face that have an impact on the subsequent face-focused task, such as animation, face recognition, gaze detection, face tracking, expression recognition, gesture understanding, etc. An outstanding feature that can play a differentiating role or serve as anchor points on a file face chart. The facial features are the tip of the

nose, corners of the eyes, chin, corners of the mouth, corners of the nose, eyebrow arches, earlobes, etc. [51]. Figure 2.10 shows the output of facial features.

Figure 2.10: The original shape and corrected landmarks [51].

2.5 Sign Language Recognition

Sign language recognition is an important application of computer vision and machine learning, particularly for individuals who are deaf or hard of hearing. There are several approaches to sign language recognition, but one common method is to use computer vision techniques to extract features from the video of sign language gestures and then use machine learning algorithms to classify the gestures [52].

Another approach is to use a combination of computer vision and sensor-based techniques, such as using gloves equipped with sensors to capture the movements of the hands during sign language gestures. The sensor data can be combined with visual data to improve the accuracy of the recognition system [8].

Overall, sign language recognition is a challenging task due to the complexity and variability of sign language gestures. However, recent advances in deep learning and computer vision have shown promising results, and sign language recognition systems have the potential to greatly improve communication and accessibility for individuals who use sign language as their primary means of communication [53]. The recent techniques used to distinguish SL are divided into two main groups:

- Machine Learning Techniques
- Deep Learning Techniques

2.5.1 Machine Learning Techniques

An area of Artificial Intelligence (AI) known as Machine Learning (ML) is a well-liked technique for automating system performance in several complicated issue domains. Supervised, Unsupervised, and Semi-supervised learning issues can be broadly categorized as ML [54]. Research and development in practically every discipline, including the natural sciences, engineering, social sciences, medicine, and the arts and humanities, have all benefited from the use of machine learning [55]. which is commonly understood to include automatic computing procedures predicated on binary or logical operations that learn a task from a sequence of instances, and investigate automatic approaches for learning to make precise predictions based on past observations [56]. The most popular machine-learning algorithms are depicted in Figure 2.11.

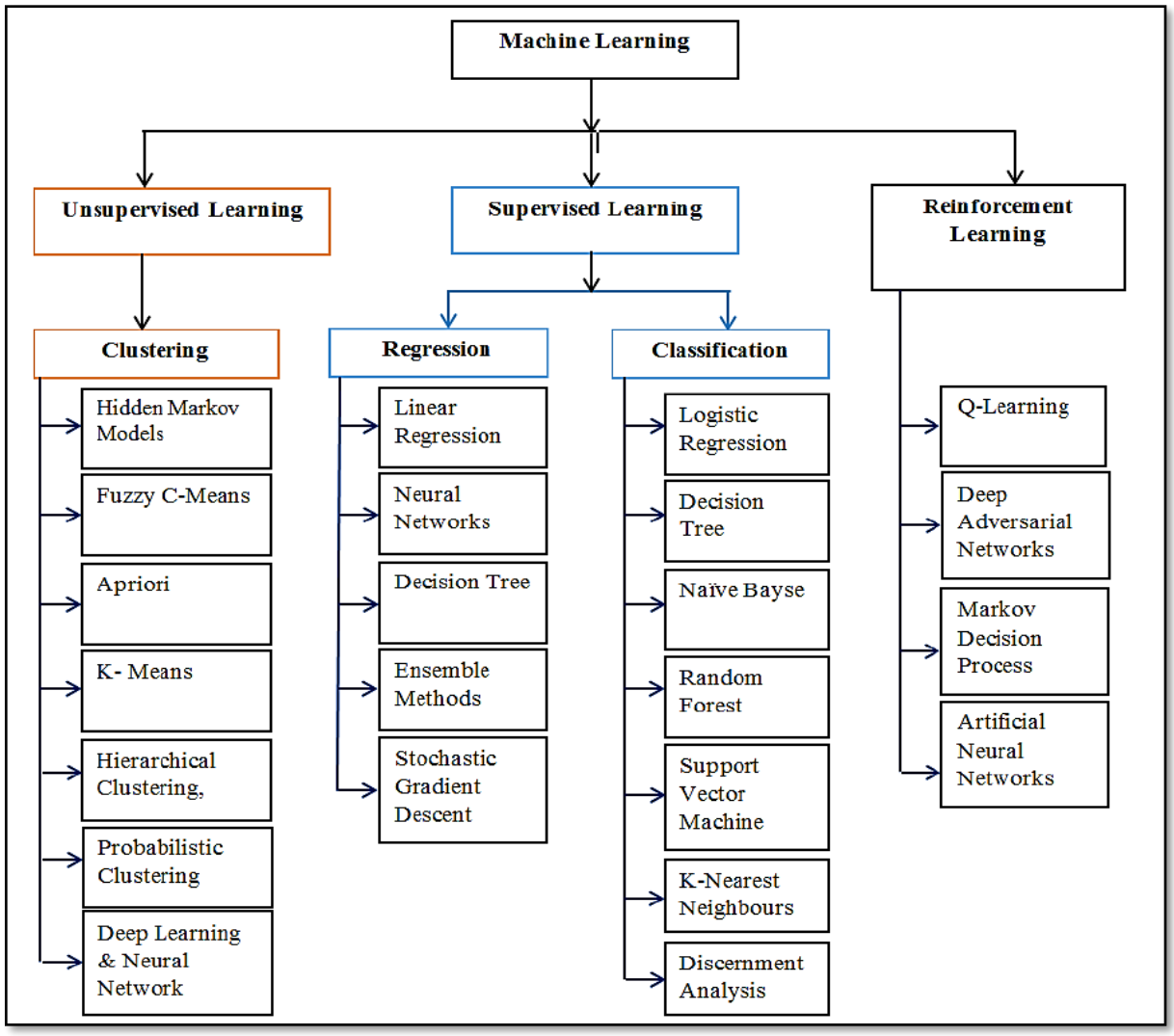


Figure 2.11: The algorithms in Machine Learning [57]

The goal of ML is to produce classification expressions that are simple enough for a human to understand. They must closely resemble human reasoning to offer an understanding of the decision-making process [58, 59]. In this section, machine learning algorithms that are widely implemented in the field of classification, especially the classification and recognition of images, are described. Among these techniques that have been effective in the classification field are Decision Trees (DT), Naive Bayes (NB), K-Nearest Neighbors (KNN), Random Forest (RF), and Stochastic Gradient Descent (SGD) [60, 61].

2.5.1.1 Naïve Bayes (NB)

The Naive Bayes classifier assumes that the features (or attributes) that are used to describe an object are conditionally independent given the class label. In other words, the model assumes that the presence or absence of a particular feature does not depend on the presence or absence of any other feature, given the class label [55]. It can be referred to as a probabilistic model that utilizes the joint probability of terms and categories to evaluate the probabilities of categories provided in a test text. The classifier's naive elements result from the straightforward presumption that every term in a given category is conditionally independent of every other term in the category. Owing to this concept of independence, each term's parameters can be learned independently, simplifying and speeding up calculation procedures. The categorization used to determine this probability is shown in Equation (2.9) [62].

$$P(C \mid X) = (P(X \mid C) * P(C)) / P(X) \ldots\ldots \quad (2.9)$$

Where:

P (C | X) is the posterior probability of class C given the feature value X.

P(X | C) is the likelihood probability of observing the feature values X given class C.

P(C) is the prior probability of class C.

P(X) is the probability of observing the feature value X.

2.5.1.2 K-Nearest Neighbor (KNN)

The algorithm is a well-known image classification method because it is straightforward to use. The KNN approach is an instance-based ML technique that

is thought to be very straightforward when compared to other ML classification techniques [63]. When it comes to supervised statistical pattern recognition, the KNN technique consistently outperforms other methods. When drawing the training sample, no previous distribution presumption is required. Then, every new test sample is categorized in a very simple and straightforward way by comparing it to every other sample in the training set. To assess the test sample's category, the neighbors' category labels are employed.

Measures like Euclidean Distance are shown in Equation (2.10) and Hamming Distance is displayed in Equation (2.11). These are used to determine how similar the two instances are [62].

$$d(x,y) = \sqrt{\sum_{i=1}^{n}(yi - xi)^2} \ldots\ldots\ldots (2.10)$$

$$Hd = \left(\sum_{i=1}^{k}|xi - yi|\right) \ldots\ldots\ldots\ldots\ldots (2.11)$$

Where: xi and yi represent the corresponding elements of the vectors x and y.
n represents the total number of elements in the vectors. But x=y represents d=0, x≠y when d≠1.

2.5.1.3 Decision Tree (DT)

The algorithm is categorized as a Supervised classification technique and it is one of the inductive learning algorithms. The decision tree is a hierarchical structure with numerous layers, beginning with the base's root node and ending with several internal sub-nodes and leaves (end-nodes). The direction of the categorizing process is indicated by the branches (Arches) that connect the nodes. Each step of the test is administered to one or more features, with one of two probable results. That is, within the decision tree, each node has a root node (father) [60].

2.5.1.4 Random Forest (RF)

The random forest can be referred to as an ensemble classifier that increases prediction accuracy using many models from different DTs. Here, it creates a lot of classification trees [64]. Each of them is trained utilizing a set of training data, and a technique called bootstrapping. This method just searches for a random subset of variables to obtain a split at each node. Each tree in the RF is given the input vector for classification, and every tree votes for a class. Now, the class that obtains the most votes is ultimately chosen by the RF. It is more flexible than previous methods and can handle bigger input datasets. Figure 2.12 illustrates how the algorithm is performed [57].

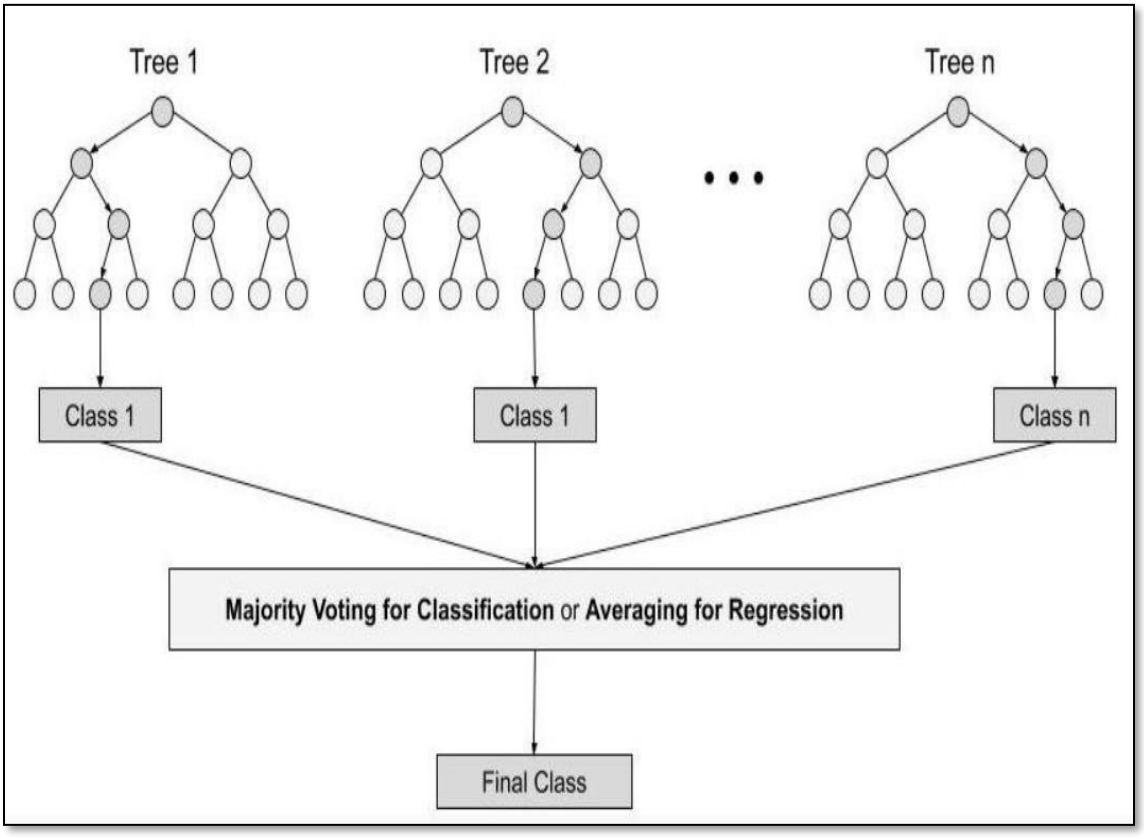


Figure 2.12: the performance of Random Forest [57].

2.5.1.5 Stochastic Gradient Descent (SGD)

This algorithm is among the essential algorithms for deep neural network training. The gradient descent algorithm has a few disadvantages. A differentiable or sub-differentiable goal function can be found for the optimum smoothness qualities using this iterative method. Since this method replaces an estimate for the actual gradient (obtained from the entire data set), it can be thought of as a stochastic approximation of gradient descent optimization (derived from a randomly chosen subset of the data). This lowers the computing cost, particularly for high-dimensional optimization problems, in exchange for a slower convergence rate and quick iterations [60].

2.5.2 Deep Learning Techniques

Deep learning is a subfield of machine learning that involves building and training artificial neural networks with multiple layers. These deep neural networks have revolutionized many fields, including computer vision, natural language processing, and speech recognition [65].

Deep learning uses multiple layers of artificial neural networks to extract features and learn representations of data at different levels of abstraction. The neural networks learn from large amounts of data, often requiring no human intervention to design specific features or rules. The use of multiple layers allows for the modeling of increasingly complex patterns and relationships in the data. This approach is effective in a wide range of applications, including computer vision, natural language processing, and speech recognition [66].

Here, a type of deep learning technique commonly used in various applications is Convolutional Neural Networks (CNNs) is a type of deep neural network that is particularly suited for image and video analysis tasks. They use convolutional layers to extract features from the input images, and pooling layers to reduce the spatial dimensions of the features. CNNs have achieved state-of-the-art results in many image recognition tasks, such as object detection and segmentation [67]. CNN networks are classified into several types, including:

1. One-Dimensional Convolution Neural Networks (1DCNN)
2. Two-Dimensional Convolution Neural Networks (2DCNN)

All networks for CNN types are similar in the way they work but differ in the way the input structure is organized, the way the filter is transmitted, the convolution kernel, and the feature map. Thus, deep CNNs are widely used and have a wide range of applications due to the following benefits [68]:

1. CNNs combine the operations of feature extraction and feature categorization into one learning entity. During the training phase, they can directly learn how to maximize the features from the raw input.
2. In comparison to typical fully-connected Multi-Layer Perceptions (MLP) networks, massive inputs can be processed by CNNs with excellent computational efficiency. This results from the sparse coupling of CNNs neurons with associated weights.
3. CNNs can withstand slight data alterations involving translation, skewing, scaling, and distortion.
4. CNNs can handle a range of input sizes. This allows CNNs to process images of different sizes while maintaining a fixed output size.

2.5.2.1 One-Dimensional Convolution Neural Networks (IDCNN)

The One-Dimensional CNN consists of the same architecture as the Two-Dimensional CNN, but the input data differs, as data is entered into the network in a one-dimensional format, such as a vector or a one-dimensional matrix. The convolution, pooling, and fully linked layers are the three primary components of the 1DCNN [20]. To create the input feature maps, convolution operations are carried out between the input signal and matching convolution kernels. The activation function is then applied to the input feature maps to produce the output feature maps of the convolution [69]. Recently, this network has been widely applied to images or videos for several reasons [70]:

The first reason is the significant differences between the computational complexity of 1D and 2D convolutions exist. This can be illustrated in a way that an image with N×N dimensions convolved with a K×K kernel will possess a computational cost of O (N^2 K^2). Meanwhile, this is O for the comparable 1D convolution (with the same dimensions, N and K) O(NK) [59].

The second reason is that when this method is compared to 2DCNNs it often requires specialized hardware such as GPUs or cloud computing resources for efficient training, 1DCNNs can be trained using standard computing resources available on typical computers. This makes 1DCNNs more feasible and convenient for researchers or developers who may not have access to high-end hardware [71].

Figure 2.13 shows that the one-dimensional network consists of three main layers, which are the layer (L+1), layer (L), layer (L-1), as well as the middle layer (L), consisting of the number of neurons denoted by kth, and the last layer (L+1) has

an activation function added to it. The sub-sampling factor is 2, while the size of the 1D filter kernels is 3.

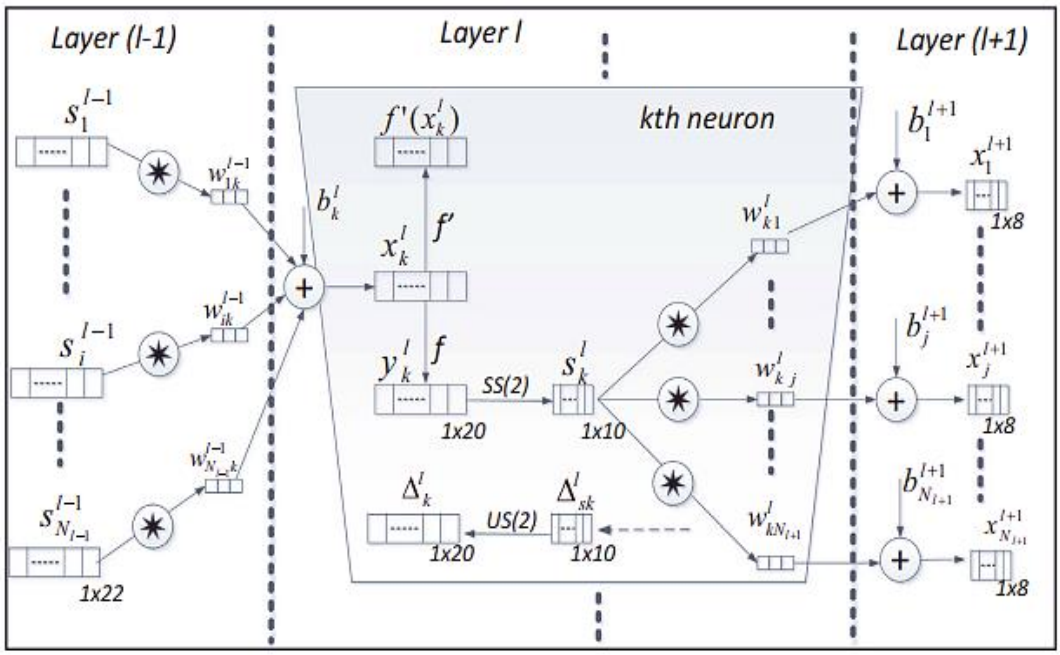


Figure 2.13: One-Dimensional Convolution Neural Network Layers [72].

The sub-sampling procedure is followed by function, f. The primary distinction between 1D and 2DCNNs is that instead of using 2D arrays to represent the feature maps and kernels, 1D arrays are employed. The CNN layers then go through the unprocessed 1D input and "learn to extract" the features that will be utilized by the fully connected layers for classification. Subsequently, the processes of feature extraction and classification are merged into a single process that may be modified to enhance the performance of the classification [69]. Figure 2.14 shows representation visualization for 1DCNN.

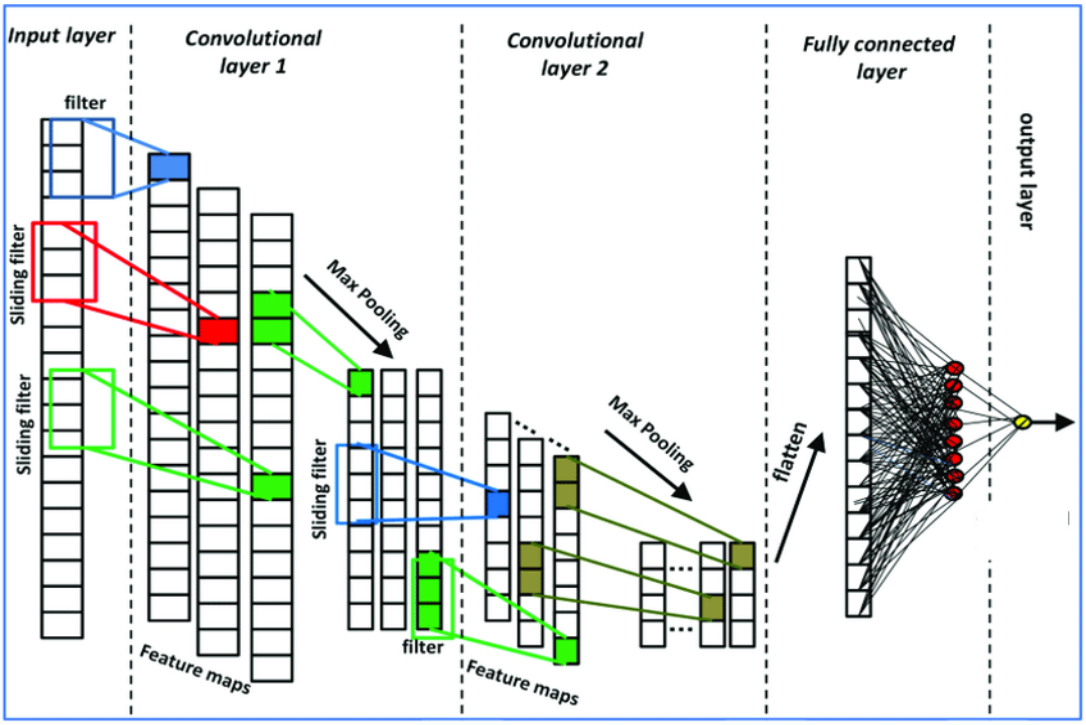


Figure 2.14: 1DCNN visualization [18]

2.5.2.2 Two-Dimensional Convolution Neural Networks (2DCNN)

A subclass of discriminative deep architecture, known as CNN, has demonstrated adequate performance for processing two-dimensional data with grid-like topologies, for instance, images and videos [37]. The design of CNNs is inspired by the structure and function of the visual cortex in the human brain. In the visual cortex, there are many cells called receptive fields that are responsible for detecting light in different areas of the visual field. Similarly, in CNNs, the filters are applied locally in the input space, and larger receptive fields are constructed by combining multiple layers of smaller receptive fields. This allows CNNs to detect patterns and features in images hierarchically and efficiently [42].

The Deep 2D-CNNs are very powerful in the image and video processing tasks when such techniques are trained on large datasets. They can automatically

learn complex features and patterns from the data, which can be used for various applications such as object detection, image segmentation, and classification. With proper training and tuning of hyperparameters, 2D-CNNs can achieve state-of-the-art performance on many benchmark datasets. [73].

2.6 Building Blocks of CNN Architecture

All current CNN designs generally include convolutional layers and pooling layers as standard components. More, along with some differences, the common training method is the fundamental method for training a network. This part presents the basic structure for building a network architecture [55].

The feature extraction and the classification functions of these characteristics are carried out by CNN accordingly. Each function makes use of a few layers, each of which serves a particular role. These layers are convolution, nonlinear (activation), pooling, and fully connected (dense) layers, in addition to some generalization layers. These layers are stacked for each function as follows:

2.6.1 The Convolution Layer

The convolution layer is a fundamental component of a CNN that performs the convolution operation on the input data. The convolution operation involves a set of filters or kernels that slide over the input data to produce a feature map.

During the convolution operation, the filters scan the input data by moving horizontally and vertically, performing a dot product between the filter and the input data at each location. The resulting values are summed up and form a single value in the output feature map. This process is repeated for all the filters, generating multiple feature maps.

The filters or kernels are learnable parameters that get updated during the training process to extracting relevant features from the input data. Typically, the first few convolution layers in a CNN extract low-level feature, such as edges and corners, while the subsequent layers extract more complex features, such as shapes and objects.

The size of the output feature map depends on several factors, such as the size of the input, the size of the filters, the stride, and the padding. The stride determines the step size between each filter application, while the padding adds extra pixels around the input data to maintain the spatial dimensions of the output feature map.

In summary, the convolution layer performs the feature extraction operation by applying filters to the input data, generating multiple feature maps that are used as input to the next layer in the CNN. A visual illustration of a convolutional layer is shown in Figure 2.15 [71] [72].

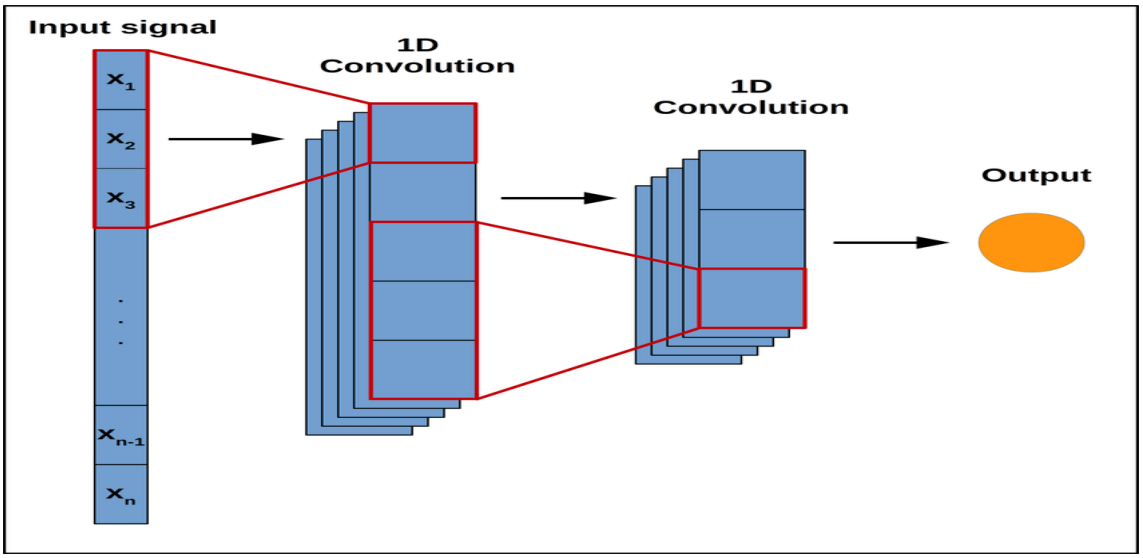


Figure2.15: Representation of a convolutional layer visually [72].

If the available conditions include an input of size W × W × D, an uncertain number of kernels having a spatial size of F, a stride of S, and an amount of padding

P, the output volume may be identified using equation (2.13). Figure 2.16 The illustrated representation of convolution layer operations.

$$Wout = \frac{Win-F+2P}{S} + 1 \ldots\ldots\ldots (2.13)$$

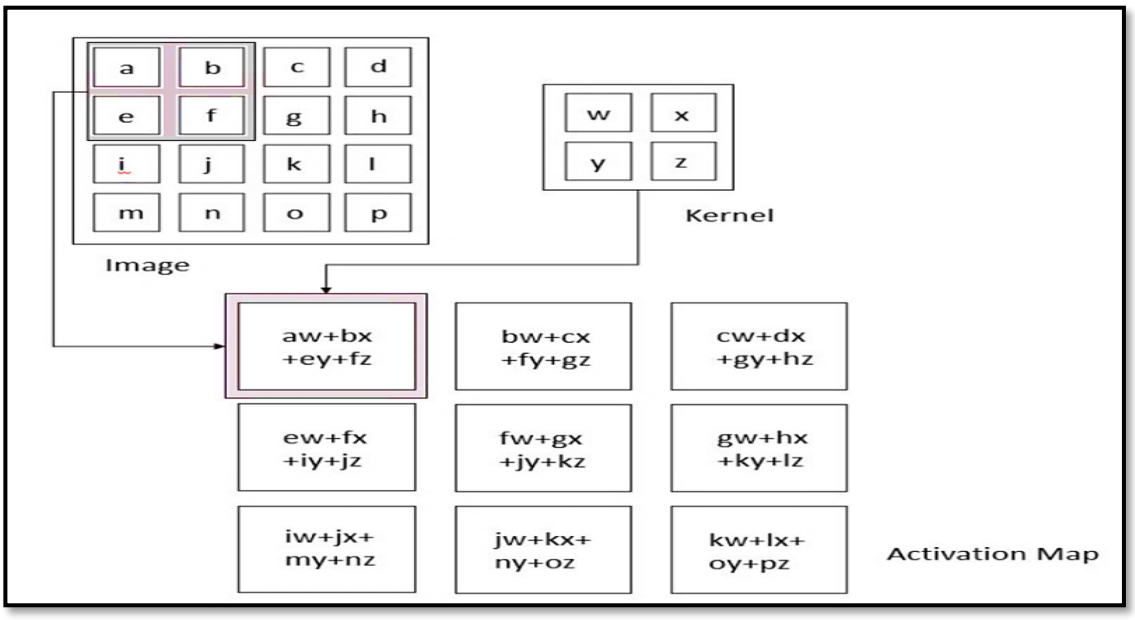


Figure 2.16: Convolution Layer Operations [71].

2.6.2 Pooling Layer

The pooling layer is usually placed after the convolutional layer in a CNN architecture. Its primary function is to down-sample the feature maps generated by the convolutional layer by extracting the most important features and discarding the unimportant ones. This not only reduces the computational load but also helps in preventing overfitting.

The most common type of pooling is max pooling, which extracts the maximum value from each window of the feature map. The window size and stride are also hyperparameters that can be tuned during the design process. The output of the pooling layer is a smaller feature map with reduced dimensions compared to the

input feature map. Figure 2.17 the illustrated representation of the pooling layer. The formula for calculating the output size of the max pooling layer is as follows [71]:

$$Wout = \frac{Win-F}{S} + 1 \ldots\ldots\ldots (2.14)$$

where Wout is the output size, Win is the input size, F is the size of the pooling filter, and S is the stride

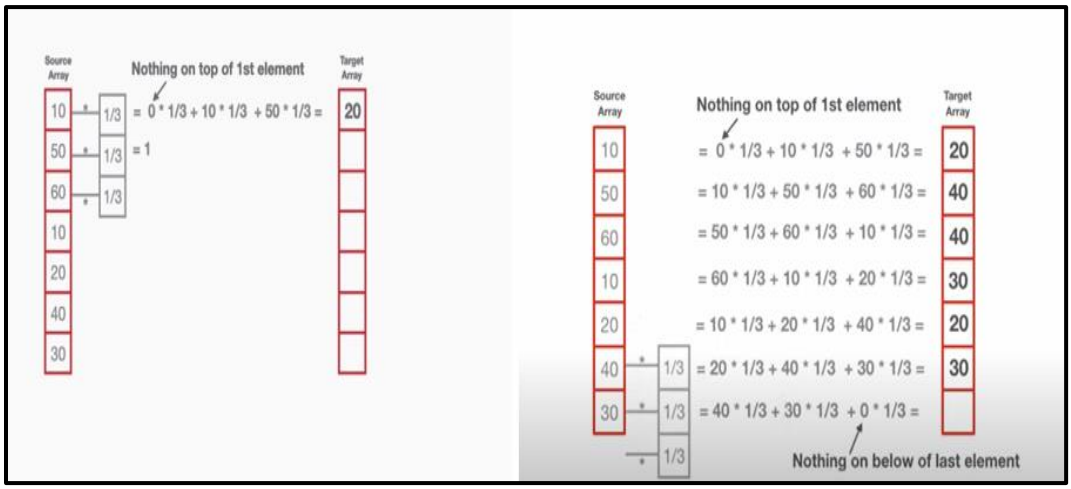


Figure 2.17: The pooling Layer Operations [71].

2.6.3 The Fully Connected Layer

The Fully Connected (FC) layer is a type of CNN layer that connects every neuron in one layer to every neuron in the next layer. It is also called a Dense layer or a Linear layer. In a CNN, the FC layer is typically used at the end of the network to perform the classification task [71].

It is responsible for performing the classification task in a CNN. The output of the convolutional and pooling layers is flattened into a 1D vector, which is then fed into the FC layer. The FC layer consists of a set of neurons that are fully connected to the previous layer. Each neuron in the FC layer computes a weighted sum of the input and applies an activation function to produce the output. The output of the FC

layer is then passed through a SoftMax function to produce the final output probabilities for each class [69].

2.6.4 The Activation Function

An Activation Function is a non-linear function applied to the output of a neural network layer to introduce non-linearity into the network. Without an activation function, the neural network would be a series of linear functions, and the overall network would still be a linear function.

The activation function is applied to the weighted sum of the inputs and biases of a neural network layer. The output of the activation function determines whether the neuron is activated or not. If the output is above a certain threshold, the neuron is activated, and its output is propagated to the next layer of the network. There are several commonly used activation functions in neural networks [70], including:

a. Sigmoid: The sigmoid function maps the input to a value between 0 and 1. It is commonly used in the output layer of binary classification problems.
b. ReLU (Rectified Linear Unit): The ReLU function is simple and computationally efficient, making it popular in deep learning models. It introduces non-linearity to the network, allowing it to learn complex relationships and patterns in the data. When the input to the ReLU function is positive, the output is the same as the input value. This means that positive values are passed through without any alteration. When the input is negative, the output is set to 0, effectively removing the negative values [69].
c. Tanh (Hyperbolic Tangent): The tanh function maps the input to a value between -1 and 1. It is commonly used in the hidden layers of a neural network.

d. SoftMax: it is a function that maps the input to a probability distribution over the output classes. It is commonly used in the output layer of multi-class classification problems [69].

e. Leaky ReLU (Rectified Linear Unit): this is a modification of the ReLU activation function, which addresses the "dying ReLU" problem. The "dying ReLU" problem occurs when a large number of neurons in the network have a ReLU activation function and become inactive, resulting in dead neurons that do not contribute to the network's output. Leaky ReLU works by introducing a small slope to the negative region of the function, allowing some degree of signal to pass through even when the input is negative [70].

   By introducing this small slope to the negative region of the function, Leaky ReLU can prevent neurons from dying and improve the performance of the network. However, choosing the value of alpha is a hyper-parameter that needs to be tuned to obtain the best performance on a specific problem.

The choice of activation function depends on the specific task and the characteristics of the dataset. In general, the ReLU function is a good default choice for most problems, but others can be used depending on the specific requirements of the problem. Figure 2.18 shows the schematic diagram of the two activation functions (ReLU, LeakyReLU) [18].

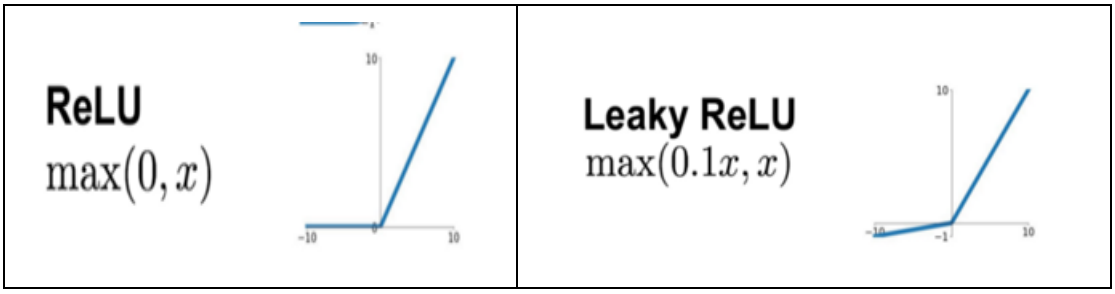


Figure 2.18: Types of Activition Function [70].

2.6.5 Loss Function

In convolutional neural networks (CNNs), the loss function plays a crucial role in training the network to make accurate predictions on an image or video data. CNNs are commonly used for tasks such as image classification, object detection, and semantic segmentation.

The loss function calculates the difference between the predicted output and the actual output, and this difference is used to update the model parameters during training. The goal of training the model is to minimize the loss function, which means that the model can make accurate predictions for a given input [71].

2.7 Video Splitting

Video splitting is the process of dividing a video into segments or regions based on certain criteria such as motion, color, or texture. This process is crucial in many video analysis applications such as object tracking, video compression, and video editing.

Regular video splitting refers to dividing the video into equal and non-overlapping segments, such as dividing a video into fixed-length segments of 1 minute each.

Non-regular video splitting, on the other hand, refers to dividing the video into segments based on specific criteria, such as detecting scene changes, object motion, or camera movement. In non-regular video splitting, the length of each segment may vary depending on the criteria used for segmentation [48].

Both regular and non-regular video splitting have their advantages and disadvantages. Regular video splitting is simple and easy to implement, but it may not be suitable for all types of videos as the content of each segment may vary widely. Non-regular video splitting, on the other hand, can provide more accurate

segmentation but may require more computational resources and more sophisticated algorithms. Ultimately, the choice between regular and non-regular video splitting depends on the specific application and the requirements of the user [50].

A shot, which can be considered the basic elements of an edited video, can be defined as a sequence of contiguous frames that have some degree of visual uniformity for a single camera take in an edited video. A group of contiguous shots composed logically is known as a "scene".

2.8 Evaluation Measures

The following measures are employed during the evaluation stage to guarantee the effectiveness and quality of the system [74]:

2.8.1 Confusion Matrix

A table termed a confusion matrix is then employed to assess how effectively a deep learning model performed on a classification task. However, many deep learning classification metrics are computed in addition to the Confusion matrix. This is because it is a very good performance indicator for the model. In more depth, the Accuracy, Precision, Recall and F1-score measures are predicated on the Confusion matrix. Figure 2.19 presents an example of a confusion table [75].

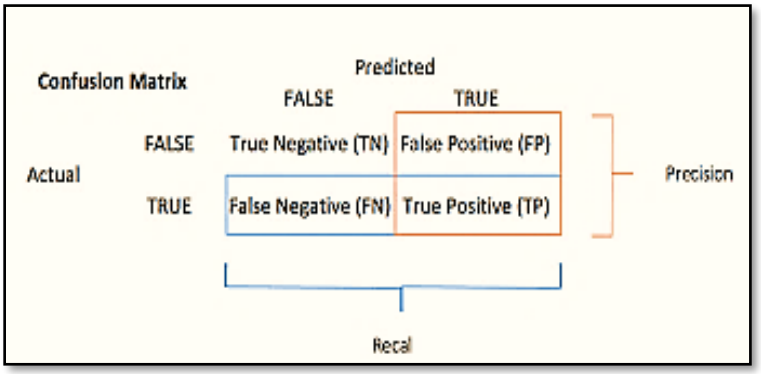


Figure 2.19: The Confusion Matrix [75].

2.8.2 Accuracy Measure

Based on Equation (2.15) [76], accuracy is described as the ratio of accurate predictions to all input samples. Consequently, accuracy is an indicator of the model's performance throughout all classes. When each class is given the same amount of weight, it is beneficial. Additionally, it is determined by dividing the total number of predictions by the number of accurate ones.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} * 100 \ldots\ldots\ldots (2.15)$$

2.8.3 Precision Measure

Precision is an evaluation measure used in classification tasks to assess the model's ability to correctly identify positive instances out of the total instances predicted as positive. It focuses on minimizing false positives. Precision is calculated in Equation (2.16) [77]:

$$Precision = \frac{TP}{TP+FP} *100 \ldots\ldots\ldots (2.16)$$

2.8.4 Recall Measure

The recall is determined as the proportion of Positive samples that were correctly identified as Positive relative to all Positive samples. On top of that, recall measures how effectively the model can differentiate Positive samples. As more positive samples are discovered, the recall value rises in Equation [77]:

$$Recall = \frac{TP}{TP+FN} *100 \ldots\ldots\ldots (2.17)$$

2.8.5 F1 Score Measure

F1 score is an evaluation measure that combines precision and recall into a single metric. It provides a balanced measure of a model's performance in classification tasks, taking into account both the ability to correctly identify positive instances (precision) and the ability to capture all positive instances (recall). The F1 score is calculated using Equation (2.18) [77]:

$$f1 - score = 2 * \frac{precision \times recall}{precision + recall} *100 \ldots\ldots\ldots (2.18)$$

2.8.6 Mean Absolute Error (MAE)

MAE measures the dissimilarity between two continual variables. Assuming that *X* and *Y* are paired observation variables manifest the same original sin. Examples of *Y* vs. *X* include comparing prophesy vs. observed, posterior time vs. initial time, and a gauge approach vs. an ersatz measurement approach. Considering a squander plot of *n* points, in which point *i* has coordinates (*xi*, *yi*). MAE is the average vertical distance between every point and the Y=X line, also referred to as the One-to-One line. MAE is the mean horizontal distance between each one of the points and the Y=X line. The MAE is calculated using the following Equation (2.19) [78].

$$MAE = \frac{1}{n}\sum_{i=1}^{n} |e_i| \qquad (2.19)$$

Assuming to already have n samples of model errors which are computed as (ei 5, i = 1, 2 . . . n) [79].

2.8.7 Root Mean Square Error (RMSE)

RMSE is utilized in a form of a standard statistical measure for measuring model efficiency in meteorology, air quality, and studies of climate research. RMSE is calculated using the following Equation (2.20) [78].

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n} e_i^2} \qquad (2.20)$$

Assuming that there are already n samples of model errors calculated as (ei. 5, i = 1, 2 . . . n).

2.8.8 Mean Square Error (MSE)

MSE quantifies the average squared difference between the predicted values and the actual values in a regression model. The MSR is calculated in Equation (2.21) as follows:

$$MSR = (1/n) * \sum (y - \hat{y})\ ^2 \ldots\ldots. (2.21)$$

In Equation (2.21), n represents the total number of instances or data points, y represents the actual values, and ŷ represents the predicted values.

MSR provides a measure of how well the regression model fits the data, with lower values indicating a better fit. It penalizes larger differences between the predicted and actual values more than smaller differences due to the squared term.

CHAPTER THREE

The Proposed System

3.1 Overview

This chapter focuses on creating and performing the suggested techniques for recognizing and classifying Arabic Sign Language (ArSL) to achieve the goal of helping deaf people communicate in society. It mainly includes two major techniques, each of which uses a different algorithm to extract features from sign language and then classify and translate these features into Arabic.

The name of the proposed system is an Interpretation system of Arabic Sign Language Recognition (ISArSLR). The proposed system uses a suite of methods from the fields of image processing, and video analysis employing deep learning, and machine learning to examine and categorize video data and generate predictions about sign language interpretation.

3.2 Hearing-Impaired Communication Framework

The work in this dissertation is divided into three main concepts of the ArSL structure that are dedicated to being utilized in this system. Figure 3.1 shows the three basic concepts that are handled by the system.

A. Arabic Sign Language Alphabet processing: These Arabic letters are created by finger-spelling in one hand. Static images were used to represent the sign of the fingerspelling and that each of the 32 letters of the alphabet has a specific sign using the hand.
B. Arabic Sign Language word processing: Arabic words are represented visually by gestures made with one or two hands. Gestures are recorded using either static or dynamic images utilizing a vision-based method.

C. Face Recognition Processing: This processing provides a method for facial recognition that helps increase the efficiency of recognizing indistinct gestures through the movement of the hands.

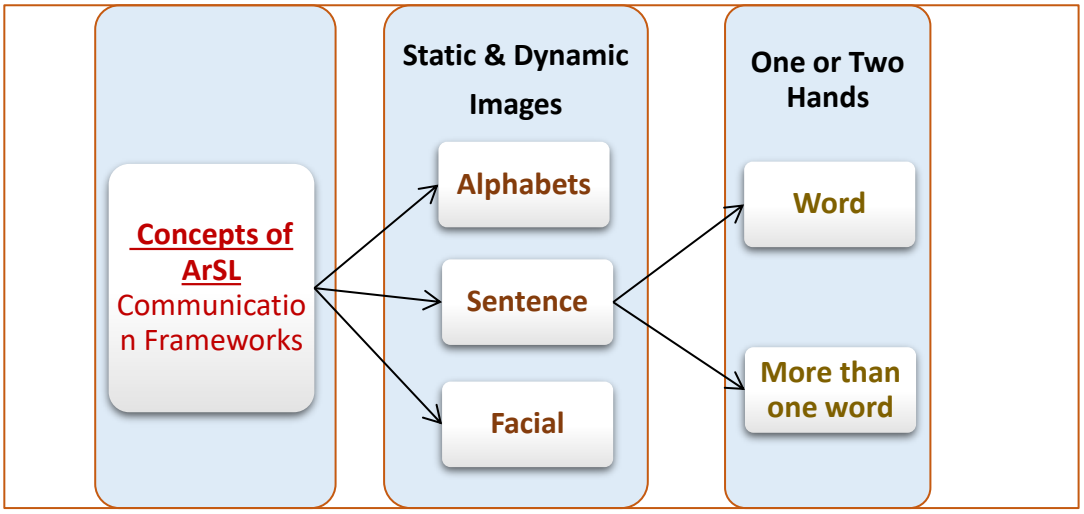


Figure 3.1: Three Basic Concepts of Hearing-Impaired Communication Frameworks.

3.3 The General Proposed Methodology

The suggested methodology architecture shown in Figure 3.2 depicts the design and development of the classification for Arabic Sign Language based on robust multi-feature extraction. There are four distinct stages to this process:

1. **Preparing Dataset Phase**
2. **Preprocessing Phase**
3. **Feature Extraction Phase**
4. **Recognition Phase**

The suggested system is based on the significant and well-known aspects of hand and facial images. It recognizes and categorizes cues primarily from the stored vocabulary. The following phases summarize the proposed approach for static and dynamic images. The phases are described in detail as follows:

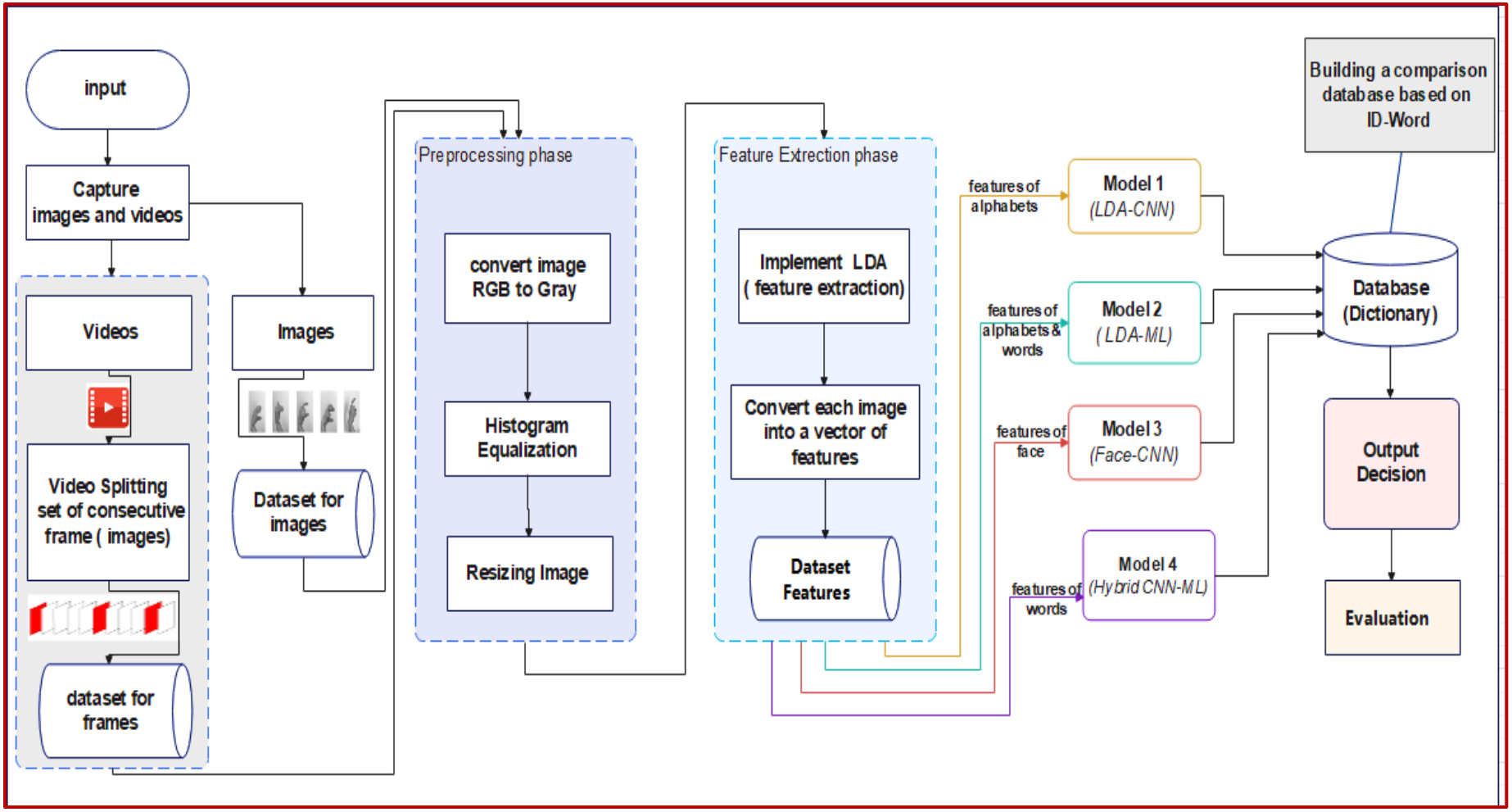


Figure 3.2: The Flow Diagram of the General Proposal System.

3.3.1 Preparing Dataset Phase

Many datasets are used in this dissertation according to the training requirements implemented in some stages. The captured images or videos provide the database for the system and indicate a sign or gesture for each letter or word. Two types of data are used: benchmark datasets and collected datasets. When gesture capture is done using a vision-based approach, the gestures are performed by many signers 6through the camera in different lighting environments with different background lights, and the dataset consists of static images and videos, as follows:

- **Benchmark ArSL Dataset (Dataset1):** ArSL2018 dataset [6], this data was obtained from King Abdul-Aziz University and named **Dataset1** for ArSL signals that use hand gestures on a static background. There are 54,049 grayscale 64x64 pixel images in the ArSL2018 dataset. There are 32 letters from the Arabic alphabet included here. Many different hand types and viewing angles of people signing the signs were used to introduce the visuals. Figure 3.3 shows a sample of this dataset. These datasets are labeled and distributed according to classifiers from the source.

Figure 3.3: Input sample of ArSL alphabets [6].

- **Benchmark Face Dataset (FaceDataset):** This is called referred to as the MUCT [49] database contains 3755 unique human faces and 76 unique manual landmarks. This dataset includes various more diverse lighting, ages, and

ethnicities in which these were included in the database rather than in previous landmark 2D face databases. Figure 3.4 shows a sample of this dataset.

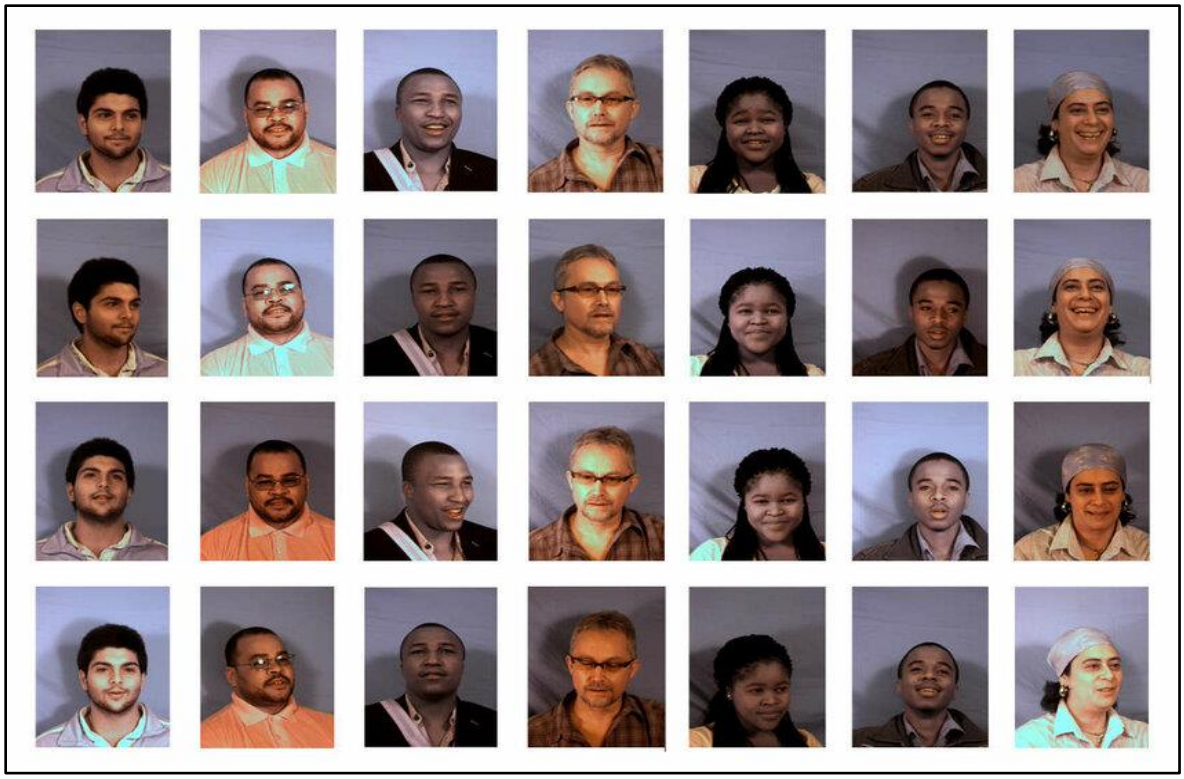


Figure 3.4: Input sample of FaceDataset [49].

- **The collected Dataset**: This data was collected from two sources, which are from Imam Hussein Academy for the Deaf and Al-Qatar Center for the Deaf. This data has been manually labeled and distributed to the workbooks. These datasets represent letters and words for ArSL and were gathered by utilizing the camera to capture static images or dynamic images through video recording. Their total number is approximately 2580 images of different sizes that have been saved in a separate database. There are 32 Arabic sign language letters and 80 commonly used terms included in this dataset.

These datasets are called Database2 and Database3. These are explained in detail in Chapter 4. A sample of these datasets is shown in Figure 3.5.

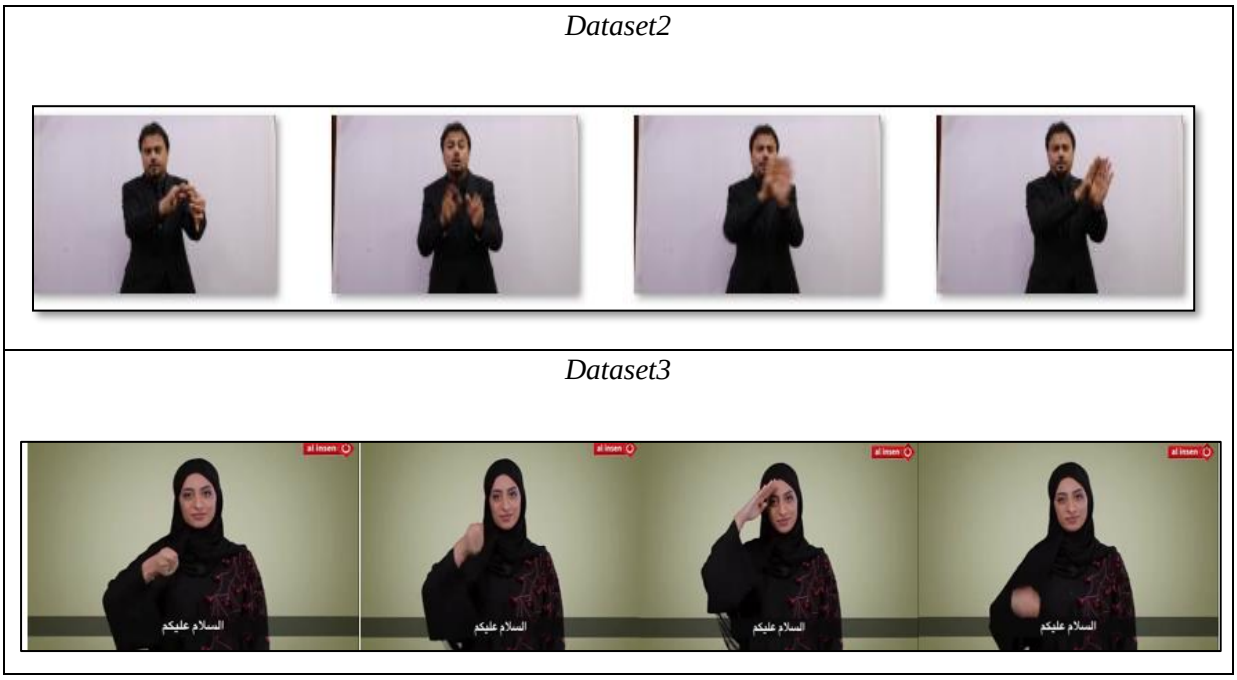


Figure 3.5: Input Sample of the collated dataset.

3.3.2 The preprocessing Phase

The second phase of this system is preprocessing operations. This phase aims to generate improved vocational and technical datasets that can be employed to create a more efficient and effective system. The same preprocessing steps are implemented for the test phase. It includes several operations for splitting videos, color conversion, image enhancement, and resizing. The preprocessing image operations are discussed in two steps:

A. The First Step: Splitting the videos

The raw video is divided into a series of frames based on a specific time sequence. Algorithm (3.1) outlines the process of processing the video and breaking it down into frames, where a word may be represented in one or more frames based on the time window.

Algorithm (3.1): Video Splitting

Input: Captured video

Output: set of frames for each gesture

BEGIN

Step 1: Initialize variables:

video_file_path: path to the video file

frame_directory_path: path to the directory where frames will be saved

frame_rate: number of frames to capture per second

Step 2: Load the video file from video_file_path.

Step 3: Initialize a frame counter to 0.

Step 4: While there are frames in the video:

Begin:

- Read the current frame from the video.
- Increment the frame counter.
- If the current frame's timestamp falls within the time window for a given word:

Begin

- Calculate the absolute difference between the current frame and the reference image.
- Save the current frame to the frame_directory_path.
- Label the frame with the corresponding word.

End if

Release the video file.

End while

Step 5: Return the list of frames with their corresponding labels.

END

In the context of video processing, a time window refers to a specific duration of time within a video. It is often used to represent a specific event or action that occurs in the video. The time window can be defined as a start and end time or as a duration from a specific time. For example, if we want to extract frames from a video where a person is speaking, we can define a time window that corresponds to the duration of the person's speech. The frames that fall within this time window will be selected and used for further processing or analysis.

Algorithm (3.1) takes in a video file and a time window for each word. It reads the video file frame by frame and checks whether each frame falls within the time window for any given word. If the frame does fall within the time window, it is saved to a directory as a separate image file with the corresponding label. The process continues until all frames in the video have been processed. Finally, the algorithm returns a list of frames with their corresponding labels.

B. The Second Step: Preprocessing

This step is performed on both all static images collected from multiple sources and the output of the first step, which is the set of frames for each gesture, called dynamic images. This step consists of three operations, namely:

a. Converting the input RGB images into a grayscale image**:** It is necessary to change all static and dynamic images in the databases to grayscale images because the converted images to a grey scale are suitable for feature extraction. The grayscale weighted average, Y, is represented by the following:

$$Y = 0.299\ R + 0.587\ G + 0.114\ B \ldots\ldots \quad (3.1)\ [79]$$

Equation (3.1) where R, G and B are integers representing red (R), green (G) and blue (B) with values in the range 0–255.

b. Enhancement image: The image's contrast is improved through the use of histogram equalization. This is a computer image processing technique that increases image contrast by making slight adjustments to the pixel intensities. It works by redistributing the most frequent intensity values across the image, thereby expanding the overall intensity range.

c. Resizing images: The size of the images from the dataset must be resized to a uniform size. This is because the images in the dataset may have different sizes. Determining the optimal size can be achieved by experimenting with several techniques and sizes, and testing the model's performance when using different image sizes, based on a trial-and-error method to obtain the best possible size. Although there is no one-size-fits-all solution, experimenting with different techniques and sizes can help determine the best way to perform the model. This is explained in detail in chapter four.

3.3.3 The Feature Extraction Phase

In this step, the goal is to identify and extract the most relevant features of sign letters, words, and facial expressions. This process involves analyzing the gestures and understanding their unique properties that contribute to their meaning. It may include aspects such as hand movements, hand shapes, facial expressions, and other visual cues. By isolating and focusing on these key aspects, the subsequent steps can be utilized effectively by processing two steps.

The two distinct steps mentioned in this phase using LDA and 1DCNN algorithms. Once the relevant aspects are isolated, the proposed system employs two algorithms, LDA and 1DCNN, for training or testing the dataset.

1. Feature Extraction based on LDA

The LDA method is employed in this stage to extract broad features from each image. The original hand image matrix is converted using the LDA approach into a reduced dimensional space to simplify the extraction of broad properties.

As a result, the highlighted aspects of each image were determined as a vector of row values, where the features of each alphabetic sample were represented by the columns. Three steps must be followed to achieve the objective of feature extraction. The inter-class matrix or distinction is the first stage; it is used to measure disengagement between different classes (the distance between their average values). The second stage was to determine the variance within each class, which is the difference between each sample's hand image and the class average. Finally, the procedure of developing the lower dimensional space reduces within-class variance while increasing between-class variance.

The final result is a vector database reflecting the extracted features, from which various images of text and faces can be retrieved. Every dataset is saved as an Excel file with the extension ".csv" where the rows represent images of the alphabet, words, or facial gesture features, and the columns reflect the features of each gesture.

2. Feature Extraction based on CNN

In this step, a CNN is applied to extract high-level features that are more specific and focused than the low-level features obtained from the previous stage. The low-level features are extracted using the LDA algorithm, which helps deduce the most important features for the subsequent CNN layers during training. Here is a breakdown of this step:

Low-level feature extraction using LDA: The LDA algorithm, as mentioned earlier, is applied to the dataset to extract low-level features. LDA analyzes the dataset and identifies the most discriminative features that contribute to class separability. These features are selected based on their ability to distinguish between different classes or categories within the dataset. By using LDA, the system obtains

a reduced-dimensional representation of the data that retains the most relevant information for classification or other tasks.

Applying a network to extract high-level features: Once the low-level features have been extracted, a network, typically a CNN, is employed to further process the data and extract high-level features. CNNs are designed to automatically learn hierarchical representations from the input data. The convolutional layers in a CNN perform local feature extraction, capturing patterns and structures at different levels of abstraction. As the data passes through these layers, higher-level features are progressively extracted, focusing on more specific and relevant aspects of the input. These high-level features are representations that capture meaningful characteristics of the gestures, which can aid in subsequent classification or analysis.

During the training phase, the network learns to optimize its internal parameters, adjusting the convolutional filters and other components to extract the most discriminative high-level features for the given task. This process involves iteratively feeding the training data through the network, calculating the loss, and updating the network parameters through backpropagation.

By combining the low-level features extracted using LDA with the subsequent high-level features extracted through the CNN layers, the system aims to capture and represent the most important and informative aspects of the gestures, enhancing the performance of subsequent classification or analysis tasks.

It's important to note that the specific network architecture, hyperparameters, and training procedures can vary depending on the specific implementation and requirements of the gesture recognition system being developed.

3.3.4 The Recognition Phases

In this phase, structural approaches are utilized to carry out the classification process. The proposed models for deep learning are trained on a dataset consisting of feature vectors derived from both images and videos, rather than directly on the video or images themselves. The key consideration is performance speed, as traditional approaches tend to have slow training and performance times. Hence, new classification models are proposed and presented for detection and classification, based on the One-Dimensional Convolutional Neural Network (1DCNN). Describe the list of all the models in Figure 3.6, as follows:

1. **Classification and Recognition of ArSL**

    A. LDA-CNN of Alphabet Model
    B. LDA-CNN of Word Model
    C. LDA-ML of Alphabet & Word Model
    D. Face-CNN Model
    E. Hybrid CNN-ML Word Model

2. **VGG-16 Model for ArSL**

    A. VGG-16 Model of One Dimensional
    B. VGG-16 Model of Two Dimensional

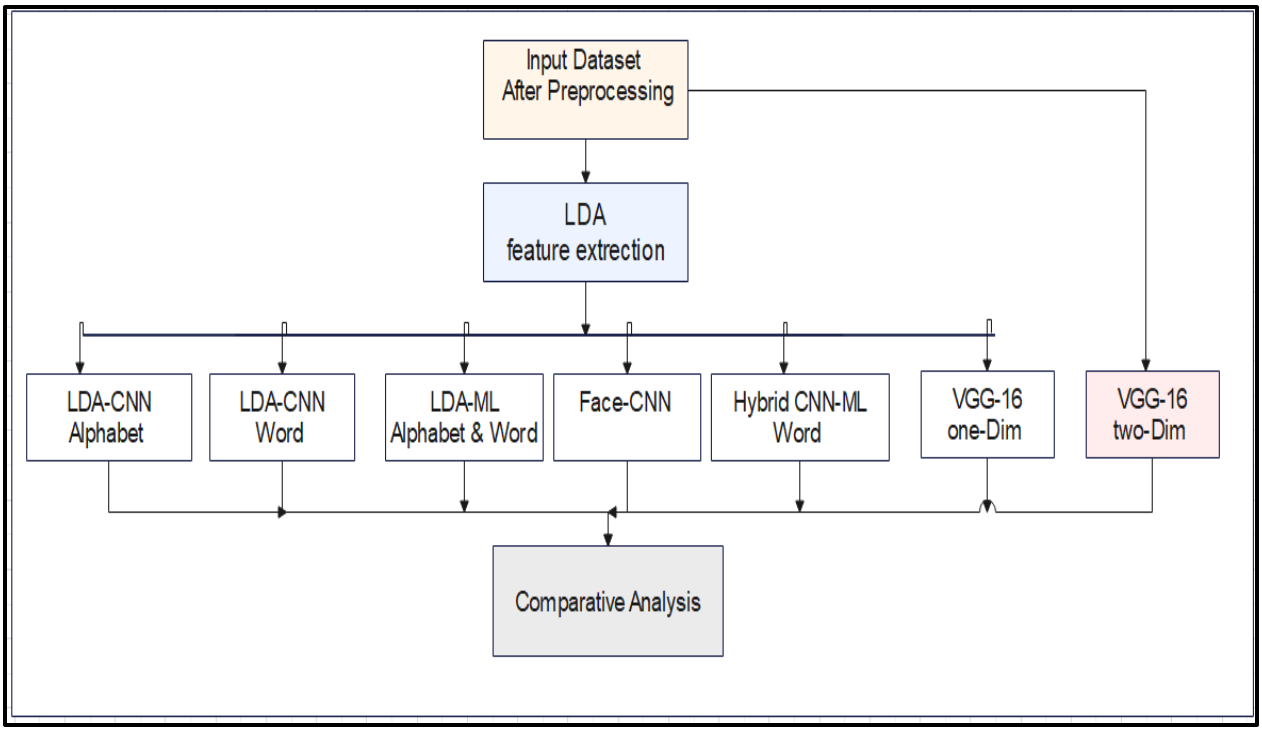


Figure 3.6: A Blok Diagram for All Models.

3.3.4.1 Classification and Recognition ArSL

A. LDA-CNN Alphabet Model

This model proposes a new structure for hand gesture recognition based on feature extraction using LDA and classification using a 1DCNN. The model aims to achieve high accuracy in recognizing hand gestures, particularly in the context of the ArSL alphabet.

This model consists of two main stages: feature extraction and classification. Each stage has its own set of layers with specific responsibilities and goals.

Feature extraction stage: In this stage, output from LDA is used for feature extraction. Each image is transformed into a vector of features, and these feature vectors serve as the input for the subsequent classification stage.

Classification stage using 1DCNN: The feature vectors obtained from the previous stage are fed into a 1DCNN for hand gesture classification. The 1DCNN is specifically designed to detect and recognize hand gestures, with a focus on comparing them to the ArSL alphabet. The architecture of the 1DCNN is illustrated in Figure 3.7, providing a visual representation of the network structure.

The proposed model includes additional details such as the arrangement of layers, their responsibilities, and the output shape and quantity of parameters (weights) for each layer. These specifics are summarized in 1DCNN Table 3.1, providing a comprehensive overview of the model's architecture.

1DCNNs are well-suited for real-time hardware implementation and are cost-effective due to their simple and compact design. This suggests that the proposed model is designed to be efficient in terms of both computational resources and real-time performance.

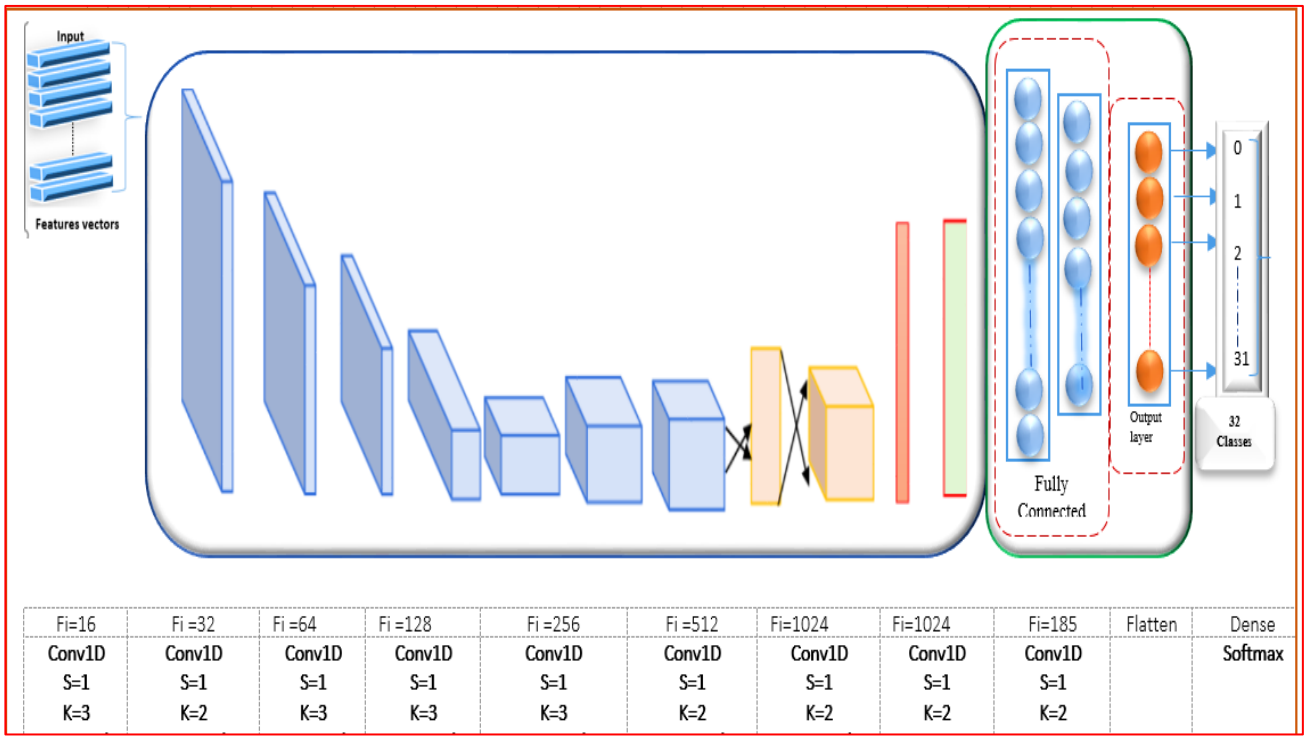


Figure 3.7: The proposed architecture of the LDA-CNN alphabet Model

The classification step begins with the activation function being applied to two fully connected layers, also known as Dense layers; the third fully connected layer, also known as the output layer or decision-making layer, implements a sigmoid function to provide the final class of the sequences; and the final layer is the output layer or decision-making layer.

The suggested model has seven convolutional layers, each of which is followed by a nonlinear layer and a pooling layer. Convolution layers of 16, 32, 64, 128, 256, 512 and 1024 convolution filters are utilized. When using these filters, a unique kernel is applied to each window to achieve the desired effect. With regards to the non-linear layer used in 1DCNNs and generic convolutional neural networks to distinguish between signals and meaningful information on each hidden layer, the use of CNN as a non-linear layer, typically with "leaky Rectified Linear Units" (LeakyReLU) is an activation function utilized for all levels of the model and a sigmoid for the last dense layer," can be used to construct a wide range of functions (output layer).

The fully connected layer, also known as the output layer or decision-making layer, is the last in a neural network architecture. It consists of three neurons and uses a SoftMax function to determine the final sequence of classification.

Table 3.1: A Summary of components of the proposed model's architecture.

Block no.	Layer type	Output shape	Params no.
1	Conv_1(Conv1D-size(3,1))	(None,32,16)	64
	leaky_re_lu_1(LeakyReLU)	(None,32,16)	0
	pooling1d_1(MaxPooling1)	(None,14,16)	0
2	conv1d_2 (Conv1D-size(3,1))	(None,12,32)	1568
	leaky_re_lu_2 (LeakyReLU)	(None,14,16)	0
	pooling1d_2 (MaxPooling1)	(None,6,32)	0
3	conv1d_3 (Conv1D)	(None, 4, 64)	6208
	leaky_re_lu_3 (LeakyReLU)	(None, 4, 64)	0
	pooling1d_3 (MaxPooling1)	(None, 4, 64)	0
4	conv1d_4 (Conv1D)	(None, 4, 128)	8320
	leaky_re_lu_4 (LeakyReLU)	(None, 4, 128)	0
	pooling1d_4 (MaxPooling1)	(None, 4, 128)	0
5	conv1d_5 (Conv1D)	(None, 4, 256)	33024
	leaky_re_lu_5 (LeakyReLU)	(None, 4, 256)	0
	pooling1d_5 (MaxPooling1	(None, 4, 256)	0
6	conv1d_6 (Conv1D)	(None, 4, 512)	131584
	leaky_re_lu_6 (LeakyReLU)	(None, 4, 512)	0
	pooling1d_6 (MaxPooling1	(None, 2, 512)	0
7	conv1d_7 (Conv1D)	(None, 2, 1024)	525312
	leaky_re_lu_7 (LeakyReLU)	(None, 2, 1024)	0
8	(Dense)	(None, 2, 512)	524800
	(Dense)	(None, 2, 256)	131328
	(Dense)	(None, 2, 256)	65792
	(Dense)	(None, 2, 128)	32896
	conv1d_8 (Conv1D)	(None, 2, 185)	23865
9	flatten_1 (Flatten)	(None, 370)	0
10	Fc-out(Dense)	(None, 32)	11872
Total params:		**1,496,633**	
Trainable params:		**1,496,633**	
Non-trainable params:		**0**	

B. LDA-CNN Word Model

The suggested model proposes a new network for classification and labeling over 80 words for each gesture in ArSL. The model leverages dynamic images and introduces a structure based on features extracted from these images. To classify the ArSL word, the model combines the output of the LDA algorithm, which generates a vector of image features with a one-dimensional CNN.

1DCNNs are particularly suitable for implementation due to their minimal processing requirements. Their simple and compact configuration, which involves only 1D convolution operations, also contributes to their cost-effectiveness. The detailed architecture of the suggested CNN model is provided in Table 3.2. These visual aids specify the layers of the model's layers architecture.

Block 1:

The first convolutional layer (Conv_1) performs 1D convolution with a kernel size of (3,1), resulting in an output shape of (None, 32, 16). It has 64 trainable parameters.

The LeakyReLU activation function is applied to introduce non-linearity. It was used because the features extracted from the LDA are far from zero, so the approximation is better using LeakyReLU.

The MaxPooling1D layer reduces the spatial dimensions of the output by selecting the maximum value in each neighborhood. It transforms the output shape to (None, 14, 16).

Block 2:

The second convolutional layer (Conv1D_2) has a kernel size of (3,1), resulting in an output shape of (None, 12, 32). It has 1568 trainable parameters.

The LeakyReLU activation function is applied.

The MaxPooling1D layer further reduces the spatial dimensions to (None, 6, 32).

Block 3:

The third convolutional layer (Conv1D_3) has an output shape of (None, 4, 64) and 6208 trainable parameters.

The LeakyReLU activation function is applied.

The MaxPooling1D layer maintains the same output shape of (None, 4, 64).

Block 4:

The fourth convolutional layer (Conv1D_4) produces an output shape of (None, 4, 128) with 8320 trainable parameters.

The LeakyReLU activation function is applied.

The MaxPooling1D layer preserves the output shape of (None, 4, 128).

Block 5:

The fifth convolutional layer (Conv1D_5) outputs (None, 4, 256) with 33024 trainable parameters.

The LeakyReLU activation function is applied.

The MaxPooling1D layer maintains the same output shape of (None, 4, 256).

Block 6:

The sixth convolutional layer (Conv1D_6) produces an output shape of (None, 4, 512) with 131584 trainable parameters.

The LeakyReLU activation function is applied.

The MaxPooling1D layer reduces the output shape to (None, 2, 512).

Block 7:

The seventh convolutional layer (Conv1D_7) outputs (None, 2, 1024) with 525312 trainable parameters. The LeakyReLU activation function is applied.

Dense Layers:

The first dense layer (Dense) has an output shape of (None, 2, 512) and 524800 trainable parameters.

The second dense layer (Dense) has an output shape of (None, 2, 256) and 131328 trainable parameters.

The third dense layer (Dense) has an output shape of (None, 2, 256) and 65792 trainable parameters.

The fourth dense layer (Dense) has an output shape of (None, 2, 128) and 32896 trainable parameters.

Conv1D_8 (Conv1D):

This convolutional layer has an output shape of (None, 2, 185) and 23865 trainable parameters.

Flatten_1 (Flatten):

The flatten layer reshapes the output from the previous layer into a 1D vector with a shape of (None, 370), preparing it for the final classification.

Fc-out (Dense):

The final dense layer (Fc-out) has an output shape of (None, 32) and 11872 trainable parameters.

In summary, the model consists of eight convolutional layers for 1D feature extraction, followed by seven MaxPooling1D layers to reduce spatial dimensions. Then, four dense layers are utilized before the final convolutional layer (Conv1D_8). The output from Conv1D_8 is flattened, and a fully connected layer (Fc-out) provides the final classification with an output shape of (None, 32). The total number of parameters in the model is 1,496,633, all of which are trainable.

Table 3.2: A summary of components of the LDA-CNN Word model's architecture.

Layer (type)	Output Shape	Param #
conv1d_1 (Conv1D)	(None, 27, 16)	64
leaky_re_lu_1 (LeakyReLU)	(None, 27, 16)	0
max_pooling1d_1 (MaxPooling1	(None, 13, 16)	0
leaky_re_lu_2 (LeakyReLU)	(None, 13, 16)	0
conv1d_2 (Conv1D)	(None, 11, 32)	1568
max_pooling1d_2 (MaxPooling1	(None, 5, 32)	0
conv1d_3 (Conv1D)	(None, 3, 64)	6208
leaky_re_lu_3 (LeakyReLU)	(None, 3, 64)	0
max_pooling1d_3 (MaxPooling1	(None, 3, 64)	0
conv1d_4 (Conv1D)	(None, 3, 128)	8320
leaky_re_lu_4 (LeakyReLU)	(None, 3, 128)	0
max_pooling1d_4 (MaxPooling1	(None, 3, 128)	0
conv1d_5 (Conv1D)	(None, 3, 256)	33024
leaky_re_lu_5 (LeakyReLU)	(None, 3, 256)	0
max_pooling1d_5 (MaxPooling1	(None, 3, 256)	0
conv1d_6 (Conv1D)	(None, 3, 512)	131584
leaky_re_lu_6 (LeakyReLU)	(None, 3, 512)	0
max_pooling1d_6 (MaxPooling1	(None, 2, 512)	0
conv1d_7 (Conv1D)	(None, 2, 1024)	525312
leaky_re_lu_7 (LeakyReLU)	(None, 2, 1024)	0
dense_1 (Dense)	(None, 2, 512)	524800
dense_2 (Dense)	(None, 2, 256)	131328
dense_3 (Dense)	(None, 2, 256)	65792
dense_4 (Dense)	(None, 2, 128)	32896
conv1d_8 (Conv1D)	(None, 2, 185)	23865
flatten_1 (Flatten)	(None, 370)	0
dense_5 (Dense)	(None, 30)	11130

Total params: 1,495,891
Trainable params: 1,495,891
Non-trainable params: 0

C. LDA-ML Alphabet &Word Model

The proposed framework for understanding ASL signs. The model uses machine learning techniques to identify and categorize ArSL. The goal of this task is to interpret ArSL into text while also interpreting hand images and videos. The primary procedures involved in carrying out the proposal are depicted in Figure 3.8. These are the strategies that are suggested:

- **Naive Bayes classifier (NB)**
- **Decision Tree (DT)**
- **Random forest (RF)**
- **K-Nearest Neighbor (k-NN)**
- **Stochastic Gradient Descent (SGD).**

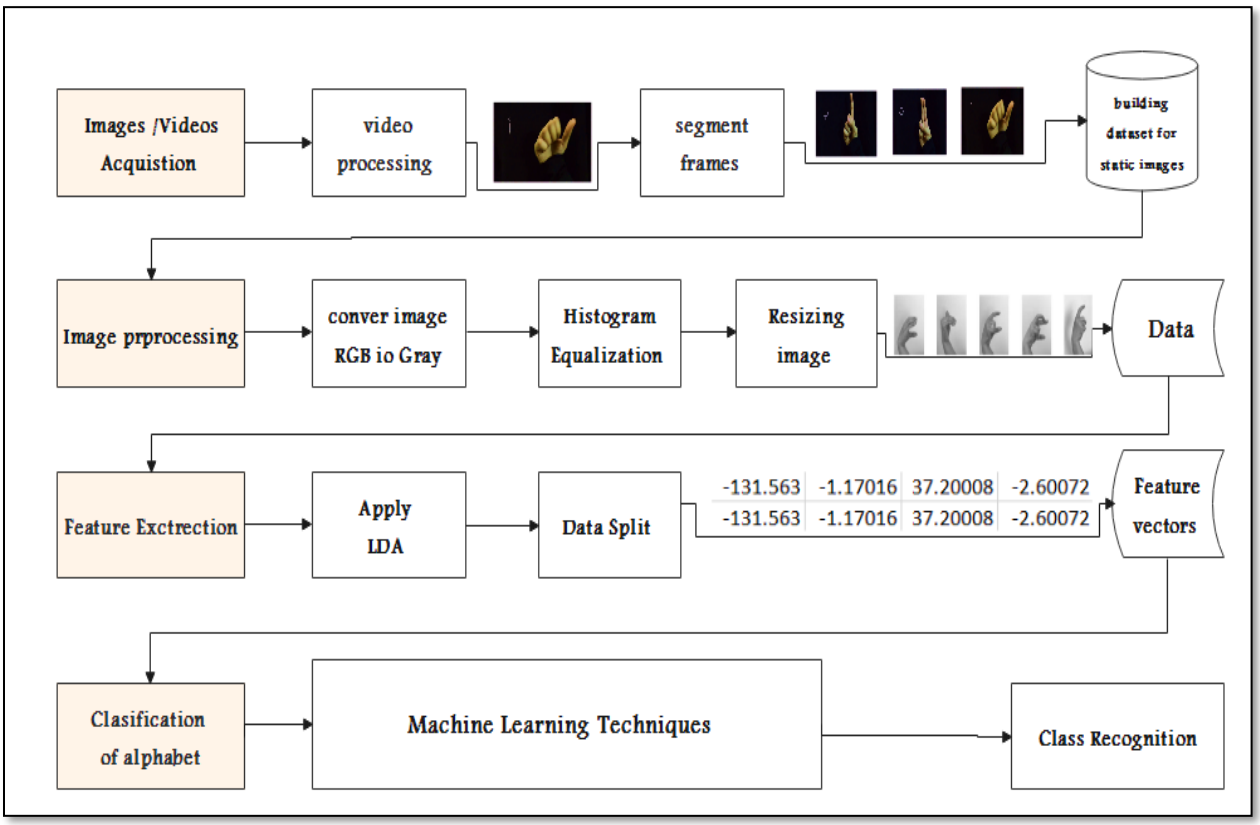


Figure 3.8: The Diagram of the LDA-ML model

The suggested technique employs machine learning techniques for the alphabetic and words signs classification. This model illustrates the implementation of five of the most popular machine-learning classification algorithms mentioned earlier. These algorithms are implemented on features extracted from images using the methods presented in the previous sections of the message to find the best accuracy for classification.

D. **Face-CNN Model**

To improve the suggested model's categorization of head motion and word discrimination, a CNN architecture is used for training and testing using a standard dataset for face discrimination. Figure 3.9 illustrates the framework of the face motion classifier, including its feature extraction and classification phases. Face movement classification using a convolutional neural network consists of two stages: feature extraction and classification.

A total of three dense layers and one FCL (the output layer) make up the classification phase. Dense layers can make use of the "LeakyReLU" activation function, while the final layer (the output layer) makes use of the SoftMax activation function. This includes the layering of each layer, the shape of the output, the number of parameters (weights), and an overview of the proposed model as a whole. Table 3.3 shows the summary of 1DCNN to classification landmarks for lapis.

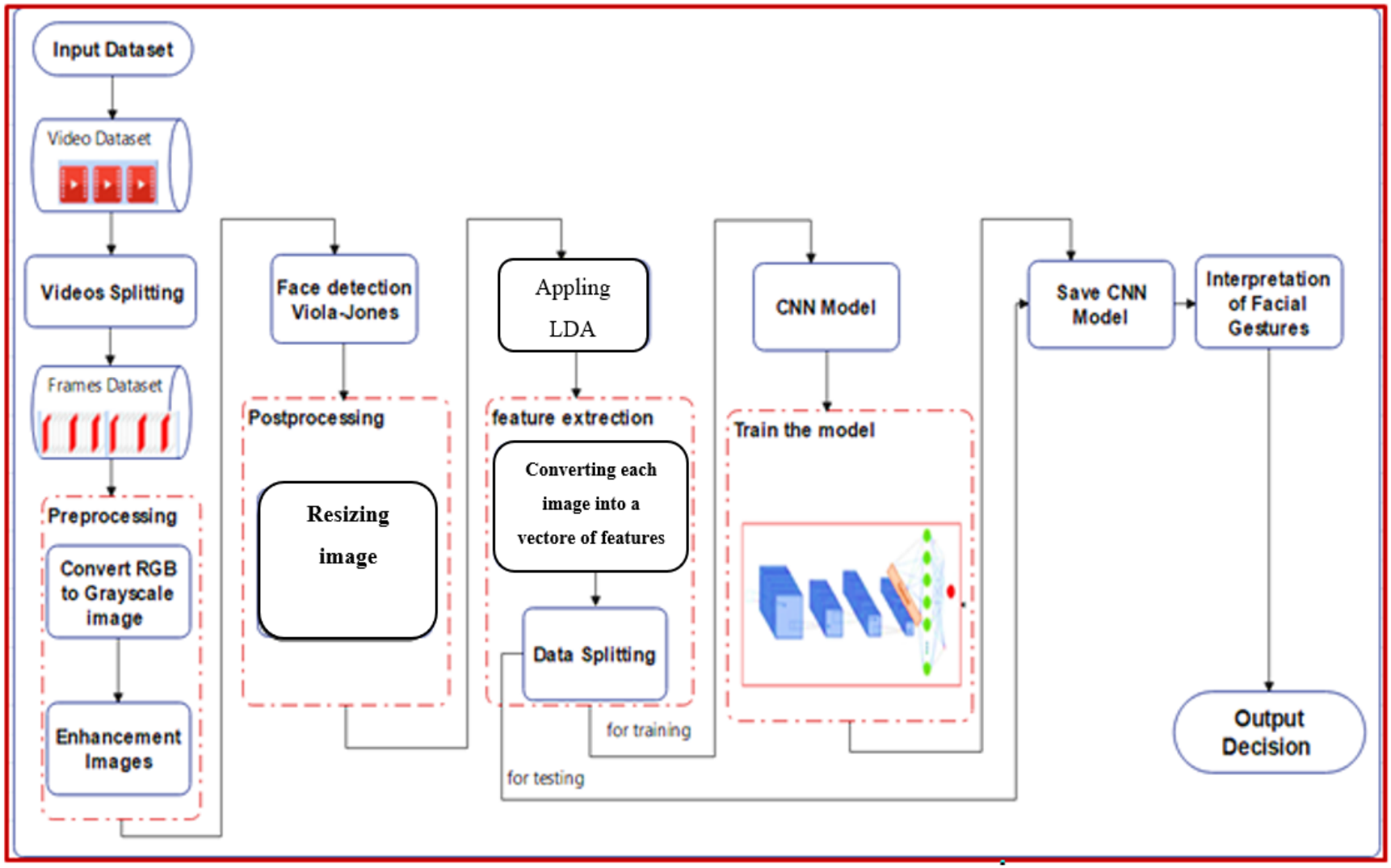


Figure 3.9: The diagram of the Face-CNN Model

Table 3.3: Summary of components for the proposed Face-CNN

Layer (type)	Output Shape	Param #
conv1d_1 (Conv1D)	(None, 15, 16)	64
leaky_re_lu_1 (LeakyReLU)	(None, 15, 16)	0
max_pooling1d_1 (MaxPooling1	(None, 15, 16)	0
leaky_re_lu_2 (LeakyReLU)	(None, 15, 16)	0
conv1d_2 (Conv1D)	(None, 13, 32)	1568
max_pooling1d_2 (MaxPooling1	(None, 13, 32)	0
conv1d_3 (Conv1D)	(None, 11, 64)	6208
leaky_re_lu_3 (LeakyReLU)	(None, 11, 64)	0
max_pooling1d_3 (MaxPooling1	(None, 11, 64)	0
conv1d_4 (Conv1D)	(None, 9, 128)	24704
leaky_re_lu_4 (LeakyReLU)	(None, 9, 128)	0
max_pooling1d_4 (MaxPooling1	(None, 9, 128)	0
conv1d_5 (Conv1D)	(None, 7, 256)	98560
leaky_re_lu_5 (LeakyReLU)	(None, 7, 256)	0
max_pooling1d_5 (MaxPooling1	(None, 7, 256)	0
conv1d_6 (Conv1D)	(None, 5, 512)	393728
leaky_re_lu_6 (LeakyReLU)	(None, 5, 512)	0
max_pooling1d_6 (MaxPooling1	(None, 5, 512)	0
conv1d_7 (Conv1D)	(None, 3, 1024)	1573888
leaky_re_lu_7 (LeakyReLU)	(None, 3, 1024)	0
conv1d_8 (Conv1D)	(None, 3, 1405)	1440125
flatten_1 (Flatten)	(None, 4215)	0
dense_1 (Dense)	(None, 18)	75888

Total params: 3,614,733
Trainable params: 3,614,733
Non-trainable params: 0

E. Hybrid CNN- ML Model

The proposed new method utilizes a combination of convolutional neural networks (CNNs) and machine learning algorithms to classify and recognize ArSL gestures from videos. The primary goal of the Hybrid CNN-ML model is to use CNN to make predictions and classifications about the deaf's state based on the detection of hands over time.

This hybrid model combines the CNN model and machine learning techniques. The LDA-CNN Word model is pre-trained on the word dataset and is used in the CNN implementation stage. In the first step, the output features of vectors (one vector for each class) from LDA are split into (70%) training and (30%) testing data as input to a one-dimensional CNN model. The next step is saving the model after completing the training process. When video translation was implemented, there was a delay in the translation process compared to real-time video. To solve this issue by building this hybrid model, it was decided to combine the features produced by this model with machine learning techniques.

In the hybrid model, Naive Bayes, Decision Trees, and Random Forest were the classifiers used in this model to categorize the CNN test input features for each gesture. These machine-learning techniques utilize information extracted from the image to categorize the Arabic words for sign language. The proposed system applied Dataset2 and Dataset3 for Arabic Sign Language gestures.

The CNN structure in Table 3.4 shows a summary of the convolutional network, which contains the input data, and the number and size of filters (often referred to as convolution kernels or feature detectors) that traverse data covering 30 words are specified in the dataset.

Table 3.4: 1DCNN Layers for Hybrid CNN-Ml Model

Layer (type)	Output Shape	Param #
conv1d_1 (Conv1D)	(None, 27, 16)	64
leaky_re_lu_1 (LeakyReLU)	(None, 27, 16)	0
max_pooling1d_1 (MaxPooling1	(None, 13, 16)	0
leaky_re_lu_2 (LeakyReLU)	(None, 13, 16)	0
conv1d_2 (Conv1D)	(None, 11, 32)	1568
max_pooling1d_2 (MaxPooling1	(None, 5, 32)	0
conv1d_3 (Conv1D)	(None, 3, 64)	6208
leaky_re_lu_3 (LeakyReLU)	(None, 3, 64)	0
max_pooling1d_3 (MaxPooling1	(None, 3, 64)	0
conv1d_4 (Conv1D)	(None, 3, 128)	8320
leaky_re_lu_4 (LeakyReLU)	(None, 3, 128)	0
max_pooling1d_4 (MaxPooling1	(None, 3, 128)	0
conv1d_5 (Conv1D)	(None, 3, 256)	33024
leaky_re_lu_5 (LeakyReLU)	(None, 3, 256)	0
max_pooling1d_5 (MaxPooling1	(None, 3, 256)	0
conv1d_6 (Conv1D)	(None, 3, 512)	131584
leaky_re_lu_6 (LeakyReLU)	(None, 3, 512)	0
max_pooling1d_6 (MaxPooling1	(None, 2, 512)	0
conv1d_7 (Conv1D)	(None, 2, 1024)	525312
leaky_re_lu_7 (LeakyReLU)	(None, 2, 1024)	0
dense_1 (Dense)	(None, 2, 512)	524800
dense_2 (Dense)	(None, 2, 256)	131328
dense_3 (Dense)	(None, 2, 256)	65792
dense_4 (Dense)	(None, 2, 128)	32896
conv1d_8 (Conv1D)	(None, 2, 185)	23865
flatten_1 (Flatten)	(None, 370)	0
dense_5 (Dense)	(None, 30)	11130

Total params: 1,495,891
Trainable params: 1,495,891
Non-trainable params: 0

Algorithm 3.2 explains the process of the hybrid model. The Train Model function takes the model, input data x, and labels y as parameters. It sets the optimizer (opt) as Adam with a learning rate of 0.001. Then, it compiles the model using the categorical_crossentropy loss function, optimizer opt, and metrics as accuracy. Finally, it trains the model using the fit() function with specified parameters.

The ExtractFeatures function is responsible for extracting features from the trained model. It iterates over the layers in the model and prints the layer index and layer details. The function getFeature is defined using K.function(), which takes model.layers[0].input as the input and K.learning_phase() as the learning phase. This function is used to extract the features from the model by applying it to the input data x.

The ExtractFeatures function applies getFeature to the input data x and stores the result in exTrain3000. It then reshapes exTrain3000 to match the desired shape and stores it in arr_reshaped1. The shape and values of arr_reshaped1 are printed.

In the Main section, the input data x and labels y are set. The model is also set as the constructed CNN model. The TrainModel function is called with the model, input data x, and labels y as arguments to train the model. The ExtractFeatures function is called with the trained model and input data x to extract the features. The extracted features are reshaped and stored in arr_reshaped1. The shape and values of arr_reshaped1 are printed to display the extracted features.

Algorithm 3.2: Hybrid CNN-ML Model

Input: Dataset features of vectors that are obtaind from LDA

Output: classification gestures for ArSL words

Function LoadData(filepath):

Begin

Read CSV file from filepath into inpdata

Set df_norm as inpdata columns 1 to 30

Set the target as inpdata column 'class'

Set df as a concatenation of df_norm and the target

Return x, y

end

Function PreprocessData(x, y):

Begin

Standardize the values of x using StandardScaler

One-hot encode y using LabelBinarizer

Reshape x to match the CNN input shape

Return x, y

End

Function ConstructCNNModel():

Begin

Create a Sequential model

Add Conv1D layers with specified parameters and activation functions

Add MaxPooling1D layers for downsampling

Add Dense layers and a final Conv1D layer with softmax activation

Return the model

End

Function TrainCNNModel(model, x, y):

Begin

Compile the model with specified optimizer, loss function, and metrics

Train the model using fit() function with x and y

```
    Return the trained model
  End
  Function ExtractFeatures(model, x):
    Begin
      Create a function getFeature using a specific layer of the trained model
      Apply the getFeature function to x to extract features
      Reshape the extracted features
      Return the reshaped features
    End
  Function MLClassification(xx, yy):
    Begin
      Scale xx using MinMaxScaler
      Split xx and yy into training and testing sets
      Train several classifiers (GaussianNB, SGDClassifier, RandomForestClassifier)
      Predict labels for the testing data using the trained classifiers
    End
Main:
  Begin
  Set filepath as the path to the input CSV file
  Set data as LoadData(filepath)
  Set x, y as PreprocessData(data)
  Set model as ConstructCNNModel()
  TrainedModel = TrainCNNModel(model, x, y)
  ExtractedFeatures = ExtractFeatures(TrainedModel, x)
  MLClassification(ExtractedFeatures, y)
End
```

3.3.4.2 VGG-16 Model for ArSL

Recently, the use of CNN for various applications has significantly increased in the computer vision field. The first and simplest deep CNN architecture after Alex Net was the VGG-Net. Later, a 22-layer architecture known as Google Net was introduced. Although Google Net has more layers than VGG-Net, it requires fewer parameters due to the use of the "inception module" as a building block. Inception modules perform convolution at different scales and spatial pooling in parallel. These modules are also optimized with convolutions (cross-channel pooling) to reduce dimensionality and eliminate or reduce redundant layers while keeping the network size manageable.

A. Two-Dimensional VGG-16 Model

The effectiveness of these models was compared using the model VGG16 for current deep learning and the suggested techniques. This network is used to process color images from video or images in the dataset. A 224*224-pixel image with three color channels (R, G, and B) is assumed as input by some of the many possible network architectures. Each pixel's RGB values are not normalized, but other than that, no preprocessing is performed. The uniformity of its structure makes it seem like a less complicated version of a deep convolutional neural network, which is appealing to a newbie. The framework of the network is depicted in Figure 3.10. There are only two guidelines to follow when using VGG-16:

1. A window size of 3*3, stride of 1*1, and padding of the same are used in each convolutional layer's setup. The number of filters is the sole distinguishing feature.

2. For each Max Pooling layer, the configuration window size is 2*2, and the stride is 2*2. This means that after each pooling layer, the image is twice as large. Layers used in this work that were trained with VGG-16 are listed in Table 3.5.

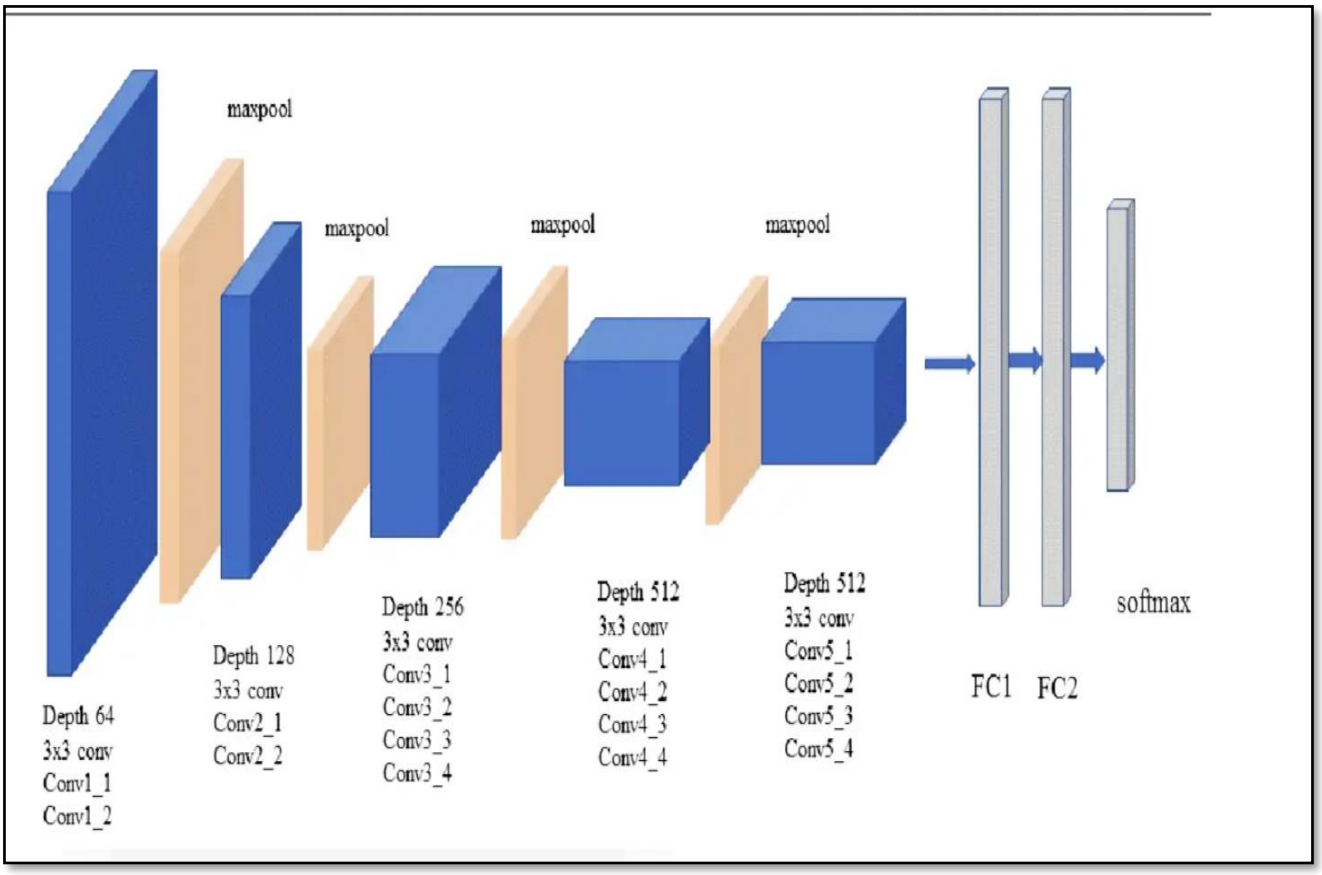


Figure 3.10: Two-Dim VGG-Net layers

Table 3.5: VGG-16 layers Model

Layer (type)	Output Shape	Param #
conv2d_1 (Conv2D)	(None, 20, 20, 64)	1792
conv2d_2 (Conv2D)	(None, 20, 20, 64)	36928
max_pooling2d_1 (MaxPooling2	(None, 10, 10, 64)	0
conv2d_3 (Conv2D)	(None, 10, 10, 128)	73856
conv2d_4 (Conv2D)	(None, 10, 10, 128)	147584
max_pooling2d_2 (MaxPooling2	(None, 5, 5, 128)	0
conv2d_5 (Conv2D)	(None, 5, 5, 256)	295168
conv2d_6 (Conv2D)	(None, 5, 5, 256)	590080
conv2d_7 (Conv2D)	(None, 5, 5, 256)	590080
max_pooling2d_3 (MaxPooling2	(None, 2, 2, 256)	0
conv2d_8 (Conv2D)	(None, 2, 2, 512)	1180160
conv2d_9 (Conv2D)	(None, 2, 2, 512)	2359808
conv2d_10 (Conv2D)	(None, 2, 2, 512)	2359808
max_pooling2d_4 (MaxPooling2	(None, 1, 1, 512)	0
conv2d_11 (Conv2D)	(None, 1, 1, 512)	2359808
conv2d_12 (Conv2D)	(None, 1, 1, 512)	2359808
conv2d_13 (Conv2D)	(None, 1, 1, 512)	2359808
max_pooling2d_5 (MaxPooling2	(None, 1, 1, 512)	0
flatten_1 (Flatten)	(None, 512)	0
dense_1 (Dense)	(None, 4096)	2101248
dense_2 (Dense)	(None, 4096)	16781312
dense_3 (Dense)	(None, 31)	127007

Total params: 33,724,255
Trainable params: 33,724,255
Non-trainable params: 0

B. One Dimensional VGG-16 Model

A convolutional neural network optimized for a single dimension, 1D-VGG16 is based on the more complex 2D-VGG16 model. Thirteen convolution layers, five maximum pooling layers, a linear layer, and three SoftMax layers make up the model. Miniature convolution kernels improve the model's capacity for learning. To avoid model overfitting, the original entire connection layer is omitted. The LeakyReLU activation function is not used. For 1D-VGG16 to work with experimental data, we must first perform LDA to transform the input one-dimensional matrix into a dataset of feature vectors. Figure 3.11 shows VGG-16 layers. Table 3.6, summarizes the individual parts that make up the VGG-16 one-dim model.

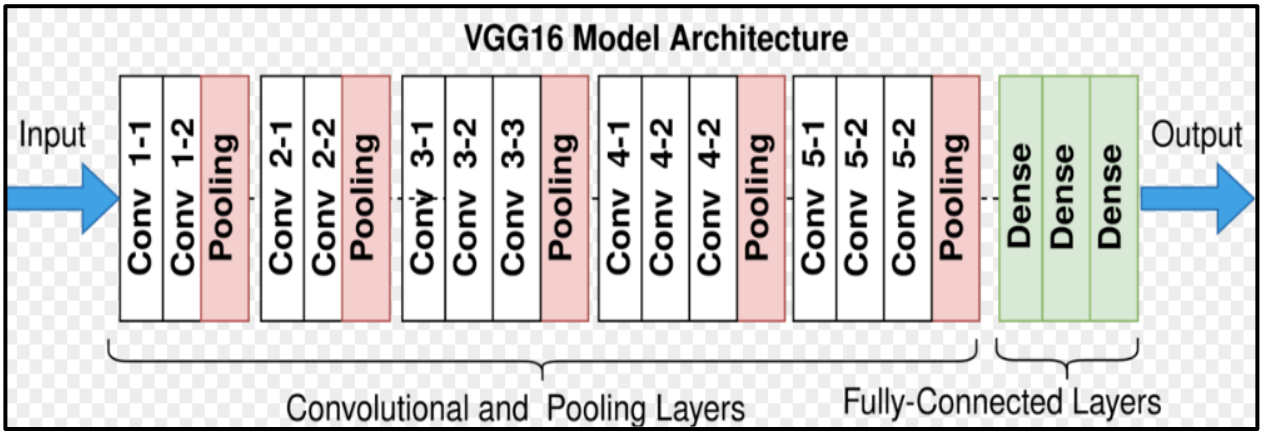


Figure 3.11: One-Dim VGG-Net layers

Table 3.6: Summary of components for VGG-16 One Dim model's architecture

Layer (type)	Output Shape	Param #
conv1d_1 (Conv1D)	(None, 30, 64)	256
conv1d_2 (Conv1D)	(None, 30, 64)	12352
max_pooling1d_1 (MaxPooling1	(None, 15, 64)	0
conv1d_3 (Conv1D)	(None, 15, 128)	24704
conv1d_4 (Conv1D)	(None, 15, 128)	49280
max_pooling1d_2 (MaxPooling1	(None, 7, 128)	0
conv1d_5 (Conv1D)	(None, 7, 256)	98560
conv1d_6 (Conv1D)	(None, 7, 256)	196864
conv1d_7 (Conv1D)	(None, 7, 256)	196864
max_pooling1d_3 (MaxPooling1	(None, 3, 256)	0
conv1d_8 (Conv1D)	(None, 3, 512)	393728
conv1d_9 (Conv1D)	(None, 3, 512)	786944
conv1d_10 (Conv1D)	(None, 3, 512)	786944
max_pooling1d_4 (MaxPooling1	(None, 1, 512)	0
conv1d_11 (Conv1D)	(None, 1, 512)	786944
conv1d_12 (Conv1D)	(None, 1, 512)	786944
conv1d_13 (Conv1D)	(None, 1, 512)	786944
max_pooling1d_5 (MaxPooling1	(None, 1, 512)	0
flatten_1 (Flatten)	(None, 512)	0
dense_1 (Dense)	(None, 4096)	2101248
dense_2 (Dense)	(None, 4096)	16781312
dense_3 (Dense)	(None, 31)	127007

Total params: 23,916,895
Trainable params: 23,916,895
Non-trainable params: 0

3.4 Constructing a Dictionary for Words Gestures

The dictionary database used in both the Hybrid CNN-ML and Face-CNN models was constructed on the ArSL Dictionary, which includes a collection of all the gestures that were interpreted in the system's real-time video implementation stage. The ArSL Dictionary is a comprehensive database of Arabic Sign Language (ArSL) that includes information on the signs, meanings, and usage of different ArSL words and phrases. The dictionary was used to train the models and to map the input signs to their corresponding words or phrases. During the implementation stage, the models used this dictionary to recognize the gestures performed by the user and to translate them into text output.

A dictionary is a data structure that stores key-value pairs, where each key maps to a corresponding value. In the context of storing words with their equivalent representations from a dictionary, a dictionary data structure would be a useful choice. Here are some of the attributes that such a dictionary might have: It contains one table, which has the following attributes:

- ID: a unique ID for each character, which represents many classes
- Character: familiar words with the Arabic language.
- Gesture: the equivalent representation in sign language for each word

3.5 Decision-Making to the Prediction Classification phase

This hybrid model includes models for prediction where two neural network models collaborate to make a classification decision and are executed in parallel: the proposed Hybrid CNN-ML model for prediction to word classification and the proposed Face-CNN detection model for identifying features of head movement associated with a word.

In this phase, two convolution neural network models work together to make a classification decision, which forms the basis for the classification decision that is produced. Decision partnership is one of the common strategies for multi-model cooperation. Each model's output is activated, indicating that selecting a certain filter represents a "decision".

Depending on all activations, the final decision is determined using a specific partnership approach. Both models are simultaneously candidates and decisions. The output activations of the models that make up the collaborative framework are partnership offerings. In the real-time video execution step, this hybrid model is used to interpret the words and gestures into text.

CHAPTER FOUR

Experiment Results and Discussion

4.1 Overview

This chapter provides a clearer explanation of the results and compares them to other models on the datasets used in this study. This is in addition to a summary of the proposed technical results. The best model for the ArSL dataset is determined by analyzing its behavior using multiple datasets. The following sections outline the datasets and system requirements that were used with the proposed system. The results of the proposed system are described in each section.

4.2 System Requirements

The software or program needs a computer system with enough processing power to perform machine learning and deep learning on any dataset. So, the proposed system is implemented using the following:

4.2.1 Hardware:

1. Processor: Intel Core i7-10750H CPU @ 2.60GHz 2.59 GHz.
2. Central Processing Unit (CPU): Intel Core I i7-7700HQ CPU @ 2.80 GHz.
3. RAM: 16.0 GB.
4. Hard Disk: 2 TB + 256 GB.
5. Graphics Processing Unit (GPU): NVIDIA GeForce RTX 3060 6GB GDDR6, Boost Clock 1425 / 1702MHz, TGP 130W.

4.2.2 Software:

1. Operating System: Windows 10 and above 64-bit.
2. Programming Language: Python 3.6.10, TensorFlow 2.1.0 version, C++, NetBeans Java, Oracle.

4.3 The Dataset Description

An outline of the datasets used, as indicated in section 3.3.1. When it comes to letter signs and word gestures, three different types of datasets are used, and these are represented in Table (4.1). Moreover, the FaceDataset is a specialized database for face recognition.

The content of BArDataset1 for the alphabet is shown in Table (4.2), and the components of Dataset 2 and Dataset 3, which are related to word gestures, are shown in Table (4.3) and Table (4.4). Some challenges are taken into account in these datasets, such as variations of the backgrounds of both images and videos, the difference in the position, the orientation of the hand, hand pose, and in the speed of the hand movement with different colors of the hand, etc. This dataset will be suited for research and will be available for free to all researchers.

Table 4.1: The Dataset of Arabic Sign Language

Dataset	Description	Samples
Dataset1	ArSL [2018] for deaf language Arabic alphabet gestures is made with hand sign on a static background, as well as a gray image 	54,000 Static Images
FaceDataset	The MUCT dataset contains the face gestures over a variety of backgrounds and persons	3755 unique human faces

Dataset 2	The dataset contains the Arabic sign language words, created with a sign hand over a variety of backgrounds, angles and lights 	2580 images
Dataset 3	The dataset contains the Arabic sign language words, hands with various backgrounds, locations, and illumination collected from the video are included in the dataset. 	100 video images

The database is divided into 32 classes, each class representing a letter from the Arabic sign language. This is the case with the Arabic sign language character database. As for the words in Arabic sign language, they are divided into 80 categories, where each category represents a word. These expressions contain verbs with one hand as well as verbs with two hands and face gestures. This data set consists of 50 categories of common and widely used words and 30 categories of isolated hand gestures that represent the governorates of Iraq.

Table 4.2: Pronunciation of each alphabet with a corresponding number of images.

#	Letter name in English Script	Letter name in Arabic script	# No. of Images	#	Letter name in English Script	Letter name in Arabic script	# No. of images
1	Alif	أ (أَلِف)	1672	17	Zā	ظ (ظَاء)	1723
2	Bā	ب (بَاء)	1791	18	Ayn	ع (عَين)	2114
3	Tā	ت (تَاء)	1838	19	Ghayn	غ (غَين)	1977
4	Thā	ث (ثَاء)	1766	20	Fā	ف (فَاء)	1955
5	Jīm	ج (جِيمْ)	1552	21	Qāf	ق (قَاف)	1705
6	Hā	ح (حَاء)	1526	22	Kāf	ك (كَاف)	1774
7	Khā	خ (خَاء)	1607	23	Lām	ل (لَامْ)	1832
8	Dāl	د (دَالْ)	1634	24	Mīm	م (مِيمْ)	1765
9	Dhāl	ذ (ذَال)	1582	25	Nūn	ن (نُون)	1819
10	Rā	ر (رَاء)	1659	26	Hā	ه (هَاء)	1592
11	Zāy	ز (زَاي)	1374	27	Wāw	و (وَاو)	1371
12	Sīn	س (سِينْ)	1638	28	Yā	ئ (يَا)	1722
13	Shīn	ش (شِينْ)	1507	29	Tāa	ة (ةَ)	1791
14	Sād	ص (صَادْ)	1895	30	Al	ال (ال)	1343
15	Dād	ض (ضَاد)	1670	31	Laa	لا (لا)	1746
16	Tā	ط (طَاء)	1816	32	Yāa	يَاء (يَاء)	1293

Table 4.3: Pronunciation of each word with a corresponding number of images.

#	Word Name in Arabic Script	Word Name in English Script	No. of images	#	Word Name in Arabic Script	Word Name in English Script	No. of images
1	الحمد لله	Thank God	120	16	حزين	Sad	123
2	السلام عليكم	Peace be upon you	154	17	وفاء	Loyalty	112
3	صباح الخير	Good morning	130	18	اتفاق	Agreement	189
4	مساء الخير	Good evening	140	19	كتب	Books	167
5	كيف الحال	How are you	114	20	اين المكان	Where is the place	139
6	العائلة	family	170	21	تعاون	Cooperation	134
7	جيد	good	180	22	تواصل	Contact	142
8	شكرا	Thank you	140	23	جاهز	Ready	178
9	لا استطيع	I can't	130	24	حتى الان	Yet	123
10	انتهى	finish	120	25	خيانة	Betrayal	156
11	سعيد	happy	145	26	دعم	Support	176
12	مساعدة	help	167	27	غدر	Treachery	145
13	يحب	like	189	28	غير مهم	Unimportant	143
14	سيء	bad	178	29	كامل	Complete	150
15	مهم	important	145	30	مبروك	congratulations	180

Table 4.4: Pronunciation of the governorates of Iraq with a corresponding number of images.

#	Word Name in Arabic Script	Word Name in English Script	No. of images
1.	بغداد	Baghdad	120
2.	أربيل	Erbil	154
3.	دهوك	Dohuk	130
4.	موصل	conductor	140
5.	كركوك	Kirkuk	114
6.	الانبار	Anbar	170
7.	كربلاء	Karbala	180
8.	نجف	Najaf	140
9.	بابل	Babylon	130
10.	الديوانية	Diwaniyah	120
11.	السليمانية	Sulaymaniyah	145
12.	البصرة	Basra	167
13.	المثنى	Double	189
14.	ديالى	Diyala	145
15.	ذي قار	Dhi Qar	187
16.	صلاح الدين	Salahaddin	153
17.	ميسان	Maysan	143
18.	واسط	Wasit	139

4.4 The Preprocessing Results

The preprocessing stages are three basic stages that were fully explained in the third chapter. These pre-processing steps are commonly used to improve image quality, contrast, and compatibility for further analysis or classification tasks. The specific results and their impact on the accuracy and quality of the system depend on the data set, the specific pre-processing techniques applied, and subsequent steps in the system or algorithm as a whole. Figure 4.1 shows an example of the images that were produced after the steps of pre-processing words and face recognition.

- Converting an image from color to grayscale removes color information and represents the image using shades of grey. The resulting grayscale image will have a single intensity channel, where each pixel represents the overall brightness of the corresponding pixel in the original image. This step can simplify subsequent processing steps and reduce computational complexity.
- Histogram equalization redistributes the pixel intensities in an image to enhance contrast. It stretches the intensity range to cover the full available range, which can bring out finer details and improve the visibility of objects or features. The resulting image will have improved contrast and may appear visually more appealing.
- Resizing images involves changing the dimensions of the image while maintaining the aspect ratio. The specific results will depend on the resizing method used and the desired output dimensions. Still images are converted to size 20*20, while images excerpted from the video are converted to size 64*64, and this is the appropriate size for the requirements of the designed models.

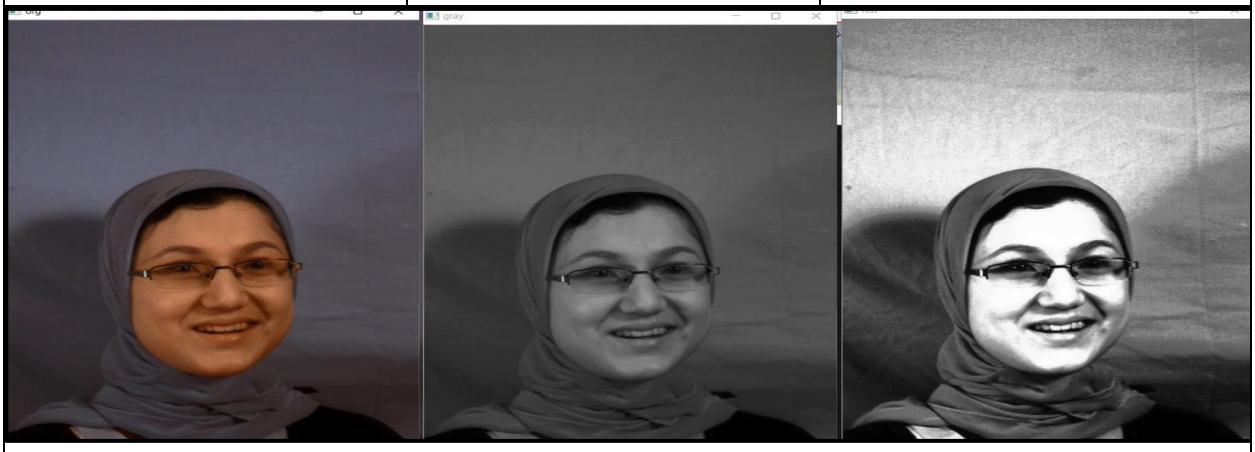
Original image	*Grayscale image*	*Histogram image*
		

Figure 4.1: preprocessing steps on a word image.

4.5 The outcome of Feature Extraction

Extracting interested features is an important phase in the recognition and classification of the system as discussed in chapter three, in which images are analyzed and features extracted to use as inputs to the models, and based on these features a decision is taken. This stage is implemented by using the LDA algorithm. The output is a vector of features for each image. Table 4.5 shows a brief sample of the results of the process of extracting features from images by applying the LDA algorithm, where the column represents the category number, the row represents the value of the features, and each line represents an image from the images within the specified category.

Table 4.6 shows a subtracted sample of the features extracted from the segmented frames of the video after applying the LDA algorithm.

Table 4.5: Sample of Result of LDA apply images

class	a1	a2	a3	a4	a5	a6	a7	a8	a9	a10	a11	a12	a13	a14	a15	a16	a17	a18	a19	a20	a21	a22	a23	a24	a25	a26	a27	a28	a29	a30	a31
1	-29.559	7.28407	-23.471	1.69525	16.3583	13.0072	18.1649	15.4101	-9.0918	4.40249	31.2651	19.2949	3.24988	28.8145	-8.4338	26.5856	5.33459	-10.251	-6.4525	7.10332	15.4039	-14.475	-4.5716	-2.0855	9.88703	17.9802	-21.071	11.3145	43.676	15.4372	13.3359
1	-37.216	3.56388	-23.998	4.47909	-8.0618	-11.553	5.44411	31.7122	-9.7155	5.02968	14.3199	24.0982	4.73882	11.8768	-8.2401	16.8013	-13.434	-5.9709	-1.741	-7.5181	11.6481	2.12296	-15.513	11.7815	6.5664	28.3322	-14.702	17.5574	35.6576	16.1638	3.07927
1	-33.834	-11.805	8.46409	-3.8372	19.93	10.6951	-4.7428	5.81491	12.4835	-10.727	-9.0119	7.51517	-3.9209	18.4741	3.85756	4.42778	-5.026	-1.2193	-22.941	5.64595	-3.7813	10.3814	-2.4044	-13.783	3.2933	1.63861	9.68968	4.46588	-16.459	-10.234	4.28746
2	25.9974	-18.721	-5.622	28.9502	45.8817	31.9572	-88.908	34.0268	-32.393	-17.911	25.6666	14.5087	50.8498	-14.689	8.35091	12.2698	6.93334	-0.5405	9.48235	10.532	-17.394	8.44342	20.1819	0.76079	-14.6	20.7523	-1.9458	15.8414	5.13143	2.2592	2.3524
2	22.4747	-23.301	-4.8446	26.0388	45.6194	14.5447	-80.406	40.2171	-31.045	-13.195	28.4303	19.7554	48.2034	-17.68	17.1871	20.8809	7.47898	-3.5531	14.0493	3.44104	-14.825	-6.7677	13.5295	12.8563	-23.499	20.8477	-10.317	2.66678	1.54658	-1.134	-0.8584
2	31.9718	-23.315	-7.7922	27.9477	40.6396	27.9585	-74.384	26.1787	-21.729	-9.919	25.1457	10.9548	49.6659	-17.154	8.34969	4.50374	5.73106	-3.1342	14.0799	9.97069	-16.601	-1.3207	7.08215	14.0356	-18.182	6.3941	-5.5301	18.0988	-12.571	-5.5608	5.37159
2	27.3918	-24.485	-2.3311	19.1587	40.7363	21.8131	-69.667	30.74	-27.109	-13.444	17.626	17.6657	55.7068	-16.221	11.3314	5.16802	7.51089	-4.6486	12.8023	-4.2712	-4.8398	-2.3943	0.49888	8.62224	-20.988	9.26303	-9.1378	13.1336	-9.5726	-7.1227	3.6419
2	27.6682	-19.563	0.793	41.5688	37.5579	27.0767	-86.381	20.066	-29.036	-13.869	20.2419	14.5496	41.6244	-16.386	4.27218	4.39032	7.08561	-3.6745	12.5375	22.8582	-6.0184	-0.1983	12.0811	4.97187	-22.065	27.2794	-6.1812	11.3435	-7.0337	-2.8718	7.9798
2	37.4396	-20.099	-7.2866	20.1528	27.7279	26.8709	-74.289	27.024	-25.387	-8.5625	25.1865	6.22716	49.1546	-11.807	9.05519	-3.0506	4.08752	-10.692	8.53247	11.0649	-6.1605	10.0877	4.92527	12.8065	-15.301	18.1548	-9.5559	6.92135	-13.568	2.90706	14.4444
2	31.4439	-35.828	10.8532	-4.0414	39.4167	34.9707	-25.433	-4.483	-1.756	-12.588	-22.289	16.2937	-22.379	-5.6875	-24.643	-12.222	-28.392	-3.4303	-21.984	3.68761	24.9012	2.94186	1.05524	6.48165	-17.59	-16.09	-19.083	23.6309	-28.466	0.10038	-10.482
2	32.8154	-21.889	2.99359	10.6319	38.5132	23.5356	-69.083	20.9966	-24.501	-9.8067	15.4019	18.1955	51.8879	-13.277	16.2279	2.36542	5.78289	0.94698	12.5528	12.3313	-6.7632	4.93166	9.26393	16.1172	-12.768	11.2465	-11.691	6.66873	-15.241	0.79456	1.15293
3	24.7907	-4.3162	-9.567	-11.924	0.34666	-8.5256	19.8947	-13.532	25.358	-0.2255	-11.203	3.2449	12.5722	20.2252	-17.601	17.9042	11.1152	1.73075	2.7016	8.68657	7.36935	-6.3495	5.05911	-8.1724	11.1452	5.37094	-10.681	14.8095	12.0189	10.9769	-11.689
3	-8.9901	-2.8274	2.3823	23.5139	20.4435	-20.062	-10.492	-16.509	-12.331	2.14301	-2.1252	-0.9842	4.8338	5.37411	-7.6818	17.6292	-5.6487	-18.418	-6.838	13.4296	9.00113	18.8172	5.59933	-25.849	3.55697	-17.51	-18.698	18.106	-1.7524	-2.0307	14.172
3	24.9803	-6.4605	-13.935	-5.9155	3.639	-8.5461	17.2776	-9.0877	24.4847	7.46761	-3.3624	4.41965	-3.8617	18.713	-17.651	19.4864	15.1205	-8.1588	-2.4475	13.9119	12.334	2.88677	4.36761	-5.1155	12.6619	-3.1828	-7.7466	18.8272	11.3809	4.29879	-8.6821
3	28.8232	-6.2479	-19.503	10.6004	0.61547	-3.8054	24.2621	-2.7953	25.3271	1.1129	-11.419	6.39896	-14.12	12.8886	-18.326	17.911	18.6756	-9.7269	-4.9386	1.91217	3.51331	1.75057	-1.2853	-3.2889	8.20689	20.9066	5.99964	18.7933	8.64495	-0.8563	-14.164
3	24.6119	-11.958	-11.036	-7.4041	0.51103	-9.9944	28.6709	-13.453	14.4326	13.1069	-7.6305	-3.0622	-23.232	24.6245	-15.694	15.4723	18.8492	-7.5147	-13.22	1.06836	7.44576	8.57514	-2.5222	-8.8627	7.9236	7.32772	17.3166	21.8143	10.3014	8.22599	-21.845
4	42.0409	-17.698	7.87101	13.5403	-2.8134	-7.2594	12.1944	-25.991	2.94528	-31.687	17.99	-0.0385	-5.6467	-22.654	17.5421	-7.3453	17.6999	-1.707	6.50672	4.4013	-15.535	-9.5872	-13.896	11.9023	-7.2285	-17.844	-4.7022	-18.436	-6.0701	-4.1633	10.9744
4	42.0409	-17.698	7.87101	13.5403	-2.8134	-7.2594	12.1944	-25.991	2.94528	-31.687	17.99	-0.0385	-5.6467	-22.654	17.5421	-7.3453	17.6999	-1.707	6.50672	4.4013	-15.535	-9.5872	-13.896	11.9023	-7.2285	-17.844	-4.7022	-18.436	-6.0701	-4.1633	10.9744
4	5.23548	-22.575	-36.045	-32.963	7.89731	17.5551	25.8018	3.54059	-11.702	-7.1468	10.7835	-28.39	-4.9395	3.07763	3.02671	16.514	11.1824	-11.455	42.5413	-4.2262	0.63758	28.5586	-9.9921	30.3724	-18.442	-2.8105	-33.838	-24.291	-4.1179	-6.6055	-4.7845
4	5.64077	-15.243	-28.828	-32.704	1.00054	25.9249	23.612	-2.5061	-12.994	-8.9186	8.6671	-27.054	-7.4844	-4.8638	-1.9414	14.9838	12.4142	-8.0773	46.3128	0.01854	2.17747	36.8898	-9.2892	34.8042	-10.528	-8.1275	-23.704	-21.385	-10.849	-9.7888	-5.9978
5	3.50445	20.0316	-2.9608	-7.2766	6.00329	0.98107	9.56457	-17.742	45.0476	19.9187	24.4317	19.158	20.66	-15.946	12.5437	13.1956	-4.0624	26.2996	15.7055	-13.314	22.2937	-2.603	2.21376	-22.769	8.18082	14.2027	24.7165	27.4372	10.4099	7.05475	0.44944
5	8.07033	23.1459	-13.257	-2.8867	16.6922	5.28739	3.7121	-6.1182	11.1545	5.7435	27.342	19.8744	2.53562	-10.433	20.2706	-10.126	6.80087	13.564	26.7102	-14.886	-7.624	7.51023	-16.693	-3.1739	-6.1124	10.7466	10.446	11.2896	-1.9797	-0.1766	24.8694
5	-29.274	23.834	15.1134	-3.9035	1.41161	-0.7148	-10.747	-15.394	-8.3038	-20.009	14.3638	-2.2545	-7.7096	15.3573	3.82227	4.0896	-17.783	-17.834	-6.4927	-10.993	-7.819	4.3328	4.11762	-11.537	0.07159	-18.281	3.02516	5.05474	-6.7073	-3.0198	-8.9177
5	-23.647	26.7227	36.9581	4.37448	-0.3112	12.2986	-17.096	-15.063	-11.782	-16.103	22.7922	-4.1619	-15.664	-10.299	-2.5729	5.88465	-12.999	-29.497	7.40031	-19.363	-28.648	-8.9894	-14.504	-5.438	-3.4766	-33.54	12.0744	-5.4732	-29.5	-0.2511	-12.243
6	-8.8433	6.54597	3.86341	6.05153	4.88483	13.6571	-2.7221	-5.779	-2.3072	25.9492	-14.952	4.0835	-3.3293	15.0298	3.57368	5.33153	5.04071	-6.3847	1.13866	-10.932	0.05938	-4.9306	8.7902	-4.3881	10.4736	8.1902	2.97631	-8.5335	3.49575	8.7404	-21.644
6	-6.449	2.67127	-1.2636	5.94415	1.71397	25.687	5.35147	-7.6234	-7.2861	12.8706	-16.684	-4.241	-6.7283	9.81386	12.8148	8.29409	15.782	-7.8328	-1.2848	1.77928	-3.2704	-1.8405	3.70897	-1.8228	16.0831	11.6371	6.07281	-7.2295	2.79178	6.20043	-11.636
6	-11.819	-0.6462	1.90599	10.5954	5.41941	25.32	8.28718	-5.5956	-7.0109	17.7603	-13.998	-2.4798	-7.7412	11.6512	5.09059	8.32103	15.993	-2.0335	0.594	9.05763	-5.8538	-4.5494	-1.7158	-4.3255	14.4084	13.8237	10.8431	-7.3732	3.08451	6.36681	-17.856
6	-19.24	2.21065	6.10057	3.05501	12.0946	23.1831	1.99101	-12.665	6.04545	21.2784	-9.4021	1.62998	-8.3855	14.2793	1.4654	3.55495	7.8296	-8.1762	-12.658	-10.438	-7.3133	1.30863	23.6266	-13.273	24.1194	3.64712	-0.5659	-7.2475	-0.0682	17.3174	-20.49
6	-11.926	8.93233	10.4267	-2.3933	12.3698	30.1	11.3751	-17.744	2.26156	14.0029	-4.715	4.22912	-8.8998	11.4272	10.6261	5.92756	17.0177	-6.4844	-10.216	-2.7508	-7.2205	-3.2383	7.47147	-6.9365	20.9964	0.74322	-1.0449	1.90188	-2.4125	16.1958	-21.786
6	-12.812	-2.1567	3.66853	0.94286	12.4275	22.6481	9.51219	-8.7824	12.6619	15.5951	-7.6587	1.3982	-7.5159	12.7248	3.99723	-1.6017	14.8678	-8.0115	-18.217	-5.0328	-1.7968	6.3205	14.6214	-17.449	23.1411	9.50959	0.88224	-2.4228	-2.2674	18.1957	-11.506
6	-17.758	-3.9033	4.55262	7.35738	9.51518	11.8953	9.88557	-17.384	2.17234	18.7996	-14.854	0.35104	-13.092	3.9685	14.7542	4.91515	10.276	-7.45	-7.1602	-10.48	5.20144	5.05375	16.0345	-9.4262	20.3966	3.6705	-0.0299	1.57204	4.06062	19.1299	-6.0857
6	-13.696	-4.4532	0.41705	7.69725	9.04645	16.3837	11.3498	-11.096	8.55645	13.3764	-10.848	-2.0659	-0.2256	2.57318	8.47775	9.08161	21.6651	-1.7364	-9.0202	4.163	12.3493	12.8149	9.14449	-13.147	33.5696	1.04237	7.49917	-2.1601	-3.9546	13.3336	-3.7428
6	-6.3544	-17.961	0.80601	20.3482	6.73839	25.5195	12.4877	-12.745	8.90564	13.9965	-21.086	4.11886	-17.189	-9.8494	5.97388	19.5779	9.1959	-11.801	6.20096	-3.4677	12.5151	-4.095	13.2708	-3.3862	18.517	-0.0658	10.1405	-2.6142	-3.0807	4.04436	-15.062
6	0.96131	-22.217	-1.0588	-19.625	3.17658	-29.122	9.16421	-10.417	10.1404	8.25543	-13.988	-15.201	-11.103	0.90833	-22.905	4.15429	-19.582	9.45852	2.85244	-11.808	26.7986	-6.2905	10.4662	-17.201	-27.643	4.60516	-11.448	-8.6053	-20.647	6.8947	3.71289
6	-7.0348	-16.802	0.04363	13.6807	20.5358	17.3374	26.9758	-16.404	12.9476	17.7881	-17.78	-4.2194	-13.699	-11.235	19.4849	11.8062	19.3454	-8.8048	-0.0156	6.2666	12.6707	-18.482	5.99527	0.64897	21.1652	3.84548	13.0986	1.07258	0.50765	10.0793	-4.0372
6	-9.8858	6.17383	8.69655	-0.5614	13.0044	17.2361	9.56543	-8.3189	7.01606	11.3346	-13.12	-0.4522	2.54785	11.3244	9.45708	1.8234	8.6347	0.09828	-6.7579	-6.36	3.1672	-2.2884	3.13456	-12.077	19.2966	14.2271	8.81113	-0.1705	-8.3928	14.5355	-16.678
6	-7.0862	2.00842	20.8733	-2.972	11.0575	15.9632	7.28568	-12.623	5.01135	11.538	-12.227	-7.6756	-7.8188	12.2285	5.99786	-6.3694	5.94295	-8.2597	-5.5152	-0.3604	6.59395	-15.67	-6.144	-7.1257	25.5304	11.9473	1.81672	3.84879	-2.9128	17.1771	-16.524
6	-13.684	1.20921	11.2554	8.22983	17.1068	21.4234	13.8971	-12.674	-7.2127	17.1638	-12.339	-14.112	-1.0223	7.98187	11.1051	-14.222	4.3258	1.75218	-7.818	10.9408	-3.5622	-18.092	-13.281	-10.106	27.2175	10.7291	9.68251	-2.0572	-7.4434	13.6072	-15.779
6	-14.779	1.55959	15.1971	11.1758	6.07514	15.1353	13.532	-18.037	1.81128	19.3638	-20.72	-4.6895	-4.7472	1.17595	7.29637	-10.688	10.2374	8.52949	-4.6177	5.46889	9.70013	-13.862	-11.912	-9.2935	26.639	16.5178	-1.3506	9.6352	2.19664	15.6447	-16.066
6	-2.6792	8.90172	15.8292	-0.9678	6.29417	12.6507	15.5576	-9.5927	-8.3374	25.6545	-10.966	-2.8417	-4.1422	8.70198	8.77175	-14.835	10.9002	2.3497	2.86541	6.76326	3.63357	-13.019	-8.4261	-7.5176	22.4399	12.4241	-2.0718	5.11667	-5.7766	13.7144	-7.9434

Table 4.6: Sample of Result of LDA apply frame from videos

class	a1	a2	a3	a4	a5	a6	a7	a8	a9	a10	a11	a12	a13	a14	a15	a16	a17	a18	a19	a20	a21	a22	a23	a24	a25	a26	a27	a28	a29	a30	a31	a32
1	89.251	58.6446	-29.764	29.4672	26.4791	62.7427	-56.075	2.58974	-2.2149	-1.4591	-10.527	-8.7829	-32.373	15.8367	-38.06	-36.254	11.3357	-74.798	-23.886	23.5134	56.2456	-12.757	6.2458	4.33904	38.1889	-54.642	-5.3762	-4.8028	79.8352	11.1558	25.543	-10.527
1	85.4501	41.3989	-7.8119	35.539	-0.0359	20.0795	21.914	-16.453	35.063	21.3793	23.1532	-45.602	40.443	-7.6332	-18.544	37.3775	-5.6264	-50.543	1.59248	27.9391	-20.438	73.2519	19.4041	-3.136	7.21904	-23.457	44.6706	-6.7561	18.135	-11.031	-36.257	23.1532
1	80.8311	-19.558	-30.218	64.9276	-24.641	-37.454	-10.026	10.9335	18.9038	7.62385	3.71321	-13.653	9.83783	-16.089	-2.4179	-7.8899	-11.79	5.10304	-26.664	-5.2365	-20.062	17.9045	-1.0854	2.34393	23.2798	-24.495	27.5391	11.3138	9.96728	-13.615	15.158	3.71321
1	80.8311	-19.558	-30.218	64.9276	-24.641	-37.454	-10.026	10.9335	18.9038	7.62385	3.71321	-13.653	9.83783	-16.089	-2.4179	-7.8899	-11.79	5.10304	-26.664	-5.2365	-20.062	17.9045	-1.0854	2.34393	23.2798	-24.495	27.5391	11.3138	9.96728	-13.615	15.158	3.71321
10	39.6218	-7.6592	1.90982	-3.2166	14.0603	-8.4813	-13.074	-10.077	-5.3847	-4.0949	17.6702	1.39605	-1.5131	16.2232	2.10691	-7.8884	4.56736	-21.736	6.89092	7.78082	5.30258	-4.6228	-15.863	14.7351	19.1644	-10.436	-4.4884	-11.829	4.15954	29.8653	-11.291	17.6702
10	39.6135	-31.283	-24.559	-17.8	-3.8803	-21.837	20.0316	7.65548	-12.96	-20.192	12.4862	-26.78	18.0618	6.95218	29.0587	-12.112	11.4911	-22.622	18.0705	29.5706	10.0294	4.93454	5.70187	-2.0943	4.02788	-19.09	1.62102	9.26022	-6.3463	14.4313	4.01463	12.4862
10	39.6122	-14.829	-9.6219	5.29308	30.371	-26.418	11.3053	-3.585	-23.432	-19.181	-22.294	-24.795	6.36453	-11.925	5.57839	-1.2276	-11.528	19.8213	-7.2303	-13.507	-4.3423	12.2547	-24.357	-21.487	13.696	-3.6343	-7.1702	11.9631	23.2639	11.2084	11.0804	-22.294
10	39.6104	-3.0729	-12.896	-10.416	19.0067	8.2801	-0.3413	1.9078	-15.059	1.32296	-5.8544	7.11509	-18.07	-8.2132	-7.4014	-30.959	0.21012	4.16511	-7.1204	-6.2128	-14.085	-0.6059	-12.19	15.0077	-6.0802	1.24569	7.1022	9.02812	-8.2295	-8.5667	2.99649	-5.8544
10	39.6072	-33.006	8.17448	20.2863	28.9101	-5.3358	-22.642	-6.1586	-21.778	5.0378	6.74293	-37.39	-0.1503	-4.806	8.55657	-18.908	20.8089	-29.122	53.0579	7.28584	-8.9509	-15.613	38.464	35.0009	-4.9577	-16.22	-2.1653	17.4173	-3.517	11.4506	22.1216	6.74293
10	39.5996	22.0198	-8.383	4.61198	-76.44	-35.894	-40.776	20.3722	-1.6359	-36.812	63.6574	13.1221	5.24517	-3.3192	-38.577	8.07825	8.79588	-46.436	13.598	38.4651	-35.76	9.45698	6.57409	26.6517	-21.472	4.43404	-14.97	-1.2007	2.22397	12.8328	-0.6909	63.6574
10	39.5943	44.491	22.5909	-9.2223	-3.4064	21.4659	10.6369	-25.73	-28.666	8.43972	-3.1501	-9.7711	5.27729	12.1583	-32.455	-11.531	18.7374	-3.9275	-42.302	2.43655	0.2579	-2.1245	-27.972	40.0071	13.1138	-16.793	-27.016	-17.368	0.64024	7.57884	15.1834	-3.1501
10	39.5911	-24.592	-0.7841	-11.293	-26.321	-1.8833	-4.0396	28.8989	4.45743	0.22002	-0.5308	-6.1908	27.4069	-5.1261	-15.184	-5.8716	-12.632	12.6617	7.88155	17.858	5.81996	-0.3087	0.69761	1.39529	-25.993	-22.317	1.54396	0.01816	-8.9186	9.53043	-3.0965	-0.5308
10	39.5885	-18.578	-5.5931	-40.276	-0.3596	-6.4311	-8.1371	0.57109	13.1246	16.7401	-8.4245	8.09329	-2.3359	20.4487	-11.521	-3.4458	-8.0277	-5.8967	-0.1569	-9.1036	-4.9403	17.1841	-11.91	0.05286	27.9094	-12.07	-1.0236	-4.6162	7.41116	9.29864	0.67914	-8.4245
10	39.5857	-7.6649	-18.884	-35.327	14.2397	-35.724	7.29889	5.76122	-7.4317	1.14354	-26.74	26.2129	-11.921	7.80344	9.7793	21.8976	-7.712	-8.6592	-3.1712	-9.1831	7.75328	8.92394	-12.321	17.1499	-3.4907	-1.0286	11.5555	6.95331	11.7365	-15.88	2.23277	-26.74
11	32.3499	47.4715	12.2907	-10.852	18.5377	-16.215	-0.5201	11.5613	1.43521	-26.536	-20.005	-3.2314	10.614	21.5701	-13.534	-0.602	11.1373	14.115	-24.295	6.88224	-3.0412	-2.785	-32.451	11.7348	-1.6595	3.27097	-9.6013	-18.147	19.4611	-1.8969	6.83748	-20.005
11	32.3497	43.6796	0.54057	-29.343	-3.2269	-21.194	20.5474	-11.181	-9.4021	-29.923	-7.4343	-6.9364	-10.553	18.7982	-9.8383	-1.955	8.0814	38.2647	-25.622	4.57732	16.3715	-5.7827	-25.359	12.3476	-25.339	-0.5991	1.66403	-4.7555	-8.7603	-15.091	6.34183	-7.4343
11	32.3453	48.1805	-10.083	-9.6259	-13.04	2.86058	-0.1636	15.279	11.614	-3.8585	-25.547	13.4889	-1.4225	7.38498	-3.5353	0.22263	2.92876	2.28548	3.83288	-5.0068	-13.025	2.64658	-11.302	-1.7508	-3.6676	3.16115	-11.435	-14.569	-10.341	11.5967	3.92721	-25.547
11	32.3414	-50.717	2.9854	0.41546	9.38047	25.2195	20.0279	14.9033	-32.289	-16.516	33.4331	-25.344	6.19395	-6.3689	-3.7917	22.7527	5.79914	9.41828	-29.566	-38.573	13.3085	-14.003	-9.2869	-28.163	13.7784	-12.933	15.2617	-8.3877	16.4532	3.91224	17.5302	33.4331
12	25.8524	-11.293	-8.3401	6.68005	1.41357	-3.6518	2.16746	-4.4217	-33.061	-1.1108	26.5649	10.4348	3.99899	12.7442	-5.4452	-12.984	-3.9146	7.39255	-19.867	-27.383	-11.017	-1.4591	-0.463	-14.158	-5.2502	7.84477	-2.8986	-3.3621	0.40407	-0.5434	-19.959	-0.9018
12	25.8489	-9.4024	48.4727	-13.403	-66.348	-12.819	8.7704	-13.915	17.3748	-14.827	-31.146	11.6148	-40.725	8.68188	35.0652	-40.476	39.1137	-19.361	1.4643	-10.855	-13.034	-14.67	4.69473	6.48837	22.0585	5.6143	6.04814	2.98625	0.5264	-0.3792	-7.8893	-6.2623
12	25.8486	31.5225	4.31061	19.9053	34.6723	22.5533	22.2087	-2.1832	-20.256	-32.99	-10.671	36.6535	14.4361	0.18458	6.18695	-2.6064	34.4569	25.4676	7.05479	-3.6103	8.41311	14.4817	-15.188	-3.0584	-3.6613	8.33104	3.02164	-8.1959	1.13958	3.72638	-23.291	8.74765
12	25.847	18.1247	5.04887	4.8811	6.36719	8.12088	21.0912	1.85913	4.92971	-7.8864	-7.7117	0.69592	2.0903	1.39785	12.0628	-20.455	-9.9872	3.98354	3.55214	-4.2343	13.9894	-1.7948	20.3795	23.5064	-6.6866	12.0608	-20.049	20.9758	5.9458	-4.8641	5.8989	26.9518
12	25.8461	42.3543	27.277	0.80473	5.63607	7.16921	2.09544	14.3405	9.01743	-22.821	1.15932	6.01448	11.7302	-7.048	-6.8619	4.09823	-6.774	-11.116	-0.5441	18.3717	-15.206	-10.266	15.1598	-2.4217	4.01254	-1.5818	-4.5267	-20.008	9.90186	-0.1298	-19.974	-23.078
12	25.843	18.463	3.23606	8.07672	12.3536	-10.631	1.9515	18.4514	35.0162	23.2935	8.08075	3.68455	3.5858	-2.9434	6.41058	-9.1176	-23.012	-16.656	7.57015	-25.049	-26.147	-13.692	-0.1874	-1.0123	7.40964	15.1351	-51.081	-11.727	-15.557	-34.959	5.6818	24.6108
13	22.07	36.7128	-38.533	-6.2668	-31.349	-7.8679	27.9543	18.7528	-7.3873	19.2142	38.1718	12.7476	13.3828	36.0042	25.4429	-2.5543	14.5749	-25.947	-3.5322	2.6004	-1.4792	-6.2272	-1.8055	-1.6629	-10.027	8.39008	16.6281	0.46199	-7.1799	-13.833	-19.807	-1.7822
13	22.064	20.4001	16.5565	2.03269	-1.6414	9.09054	-12.172	33.1848	-0.7835	1.59078	20.1081	1.79999	0.81798	14.3562	6.65506	0.39672	6.34054	-15.302	1.30388	-4.58	13.9644	-3.6876	-8.0726	-9.1733	-1.83	8.4369	-4.4572	4.65807	-6.3365	10.077	-2.6537	21.8554
13	22.0611	-26.608	-18.512	-20.224	1.37532	19.9086	24.9967	3.72538	-18.173	3.91255	-6.5742	-6.9545	-4.4302	-6.3618	2.01138	-18.307	5.13261	-13.981	-2.7224	12.5304	-17.126	8.8664	-0.9931	15.2047	-11.933	24.3376	2.05545	24.154	4.69495	9.33512	-2.643	33.0242
14	19.0139	11.6431	-49.198	26.9736	-12.452	13.9004	-8.4749	-3.6159	-10.129	31.5999	9.40845	7.57257	2.4433	-2.2242	-5.272	-14.284	10.9814	0.38834	-45.257	-7.4825	-4.0628	-15.752	0.77448	16.2944	-17.695	-12.213	3.20837	0.82761	-14.234	11.0087	-21.927	11.867
14	19.0139	11.6431	-49.198	26.9736	-12.452	13.9004	-8.4749	-3.6159	-10.129	31.5999	9.40845	7.57257	2.4433	-2.2242	-5.272	-14.284	10.9814	0.38834	-45.257	-7.4825	-2.8795	11.4072	8.93399	7.86592	-21.018	28.1124	2.59877	-10.604	13.0822	-1.2805	-14.812	-6.8096
14	19.0139	11.6431	-49.198	26.9736	-12.452	13.9004	-8.4749	-3.6159	-10.129	31.5999	9.40845	7.57257	2.4433	-2.2242	-5.272	-14.284	10.9814	0.38834	-45.257	-7.4825	-9.0809	-1.5329	-14.914	-6.112	-11.634	2.64746	-20.906	-12.145	-6.136	-10.704	-25.578	-7.9325
14	19.0139	11.6431	-49.198	26.9736	-12.452	13.9004	-8.4749	-3.6159	-10.129	31.5999	9.40845	7.57257	2.4433	-2.2242	-5.272	-14.284	10.9814	0.38834	-45.257	-7.4825	7.78494	-17.89	-12.352	8.0212	15.845	11.9157	-16.115	-17.346	-4.2639	-9.4774	-16.836	16.8993
14	19.0139	11.6431	-49.198	26.9736	-12.452	13.9004	-8.4749	-3.6159	-10.129	31.5999	9.40845	7.57257	2.4433	-2.2242	-5.272	-14.284	10.9814	0.38834	-45.257	-7.4825	12.4111	7.33462	-0.6912	7.95533	4.01975	14.0912	14.2048	15.3617	31.0787	14.1896	0.71142	9.53334
14	19.0113	-8.3679	4.41055	-1.9719	16.8838	4.52316	-2.8271	12.0116	-11.515	-0.6753	-11.339	-9.9349	11.4764	6.83066	5.69065	-1.6521	-3.0998	-6.0732	10.0603	-10.616	-17.55	4.51922	3.26969	9.36214	-2.276	0.54638	2.6475	-21.337	3.79676	-7.1353	-20.405	0.47884
15	19.0073	-1.6361	-20.562	10.2561	-11.025	1.50436	33.7855	9.87522	-27.847	9.81773	-6.2591	9.60814	-22.098	8.16719	18.72	11.6318	18.1103	10.3704	19.0427	2.38623	-14.564	-2.7701	-20.361	5.62437	-0.973	10.6406	2.67496	1.79778	12.2853	-15.344	9.48715	-6.6764
15	18.9992	20.9781	10.365	-26.971	-6.8476	-2.6189	-22.156	-8.1014	6.65504	-0.3497	-7.0948	0.32055	17.3041	10.5404	16.4894	-1.6145	15.3111	-4.1315	26.6295	5.67383	-0.141	2.19781	-14.634	-4.9768	-10.088	-2.889	-14.001	50.0301	0.09947	5.56756	-9.0625	-4.3297
15	18.9939	2.78241	-1.2844	12.7524	7.23406	-8.5239	-7.8331	5.53152	-1.7314	19.9277	-2.2689	12.4301	-0.7478	11.4716	-6.2612	9.99336	0.35964	-8.7571	-4.3776	-9.8443	-0.0898	-3.5679	-2.9056	-14.841	-8.071	-8.8266	25.0764	21.3733	12.919	2.7711	0.50901	5.99712
15	18.9905	8.46054	-5.8364	-23.672	15.6147	-16.729	8.60567	5.20789	5.2011	4.20017	7.02963	1.65304	11.9668	3.62423	8.81205	5.51811	-10.092	-0.8404	-2.459	-8.2944	14.718	8.35661	9.59224	3.71507	-1.3356	11.7219	-2.1845	-6.0905	-5.4796	5.33029	-3.3265	-13.387
15	18.9899	8.49313	5.36315	-12.699	-6.6716	-5.4882	-17.298	-8.2234	1.65985	-5.0003	-8.5309	-9.3027	2.30403	2.12915	-3.387	19.4551	-8.8222	-5.6697	-13.569	-3.4441	-6.8931	-14.785	-15.889	-19.461	3.31316	3.10566	-6.4755	-1.8421	-5.7405	4.23061	-6.3994	-2.2088
15	18.9894	-0.1036	17.908	19.9426	-9.8841	-5.3496	-31.68	10.4508	3.64228	-1.1855	22.3637	-12.681	-4.487	-4.0285	18.6319	1.08232	11.9595	-0.7142	6.33862	-13.705	-22.475	20.7442	-7.7266	-3.3521	-0.129	-9.1876	15.9368	-14.615	-0.2444	18.0178	-2.9362	9.21447
15	18.9894	4.48535	5.39191	17.6185	4.07116	-16.097	11.0484	9.43218	27.5487	-1.8044	-4.8806	-1.3636	12.2565	-7.0067	-23.447	9.8267	8.4732	9.4965	19.109	-4.342	4.42	0.68414	7.55584	-8.3401	20.7062	5.25663	11.0347	7.1637	8.26249	2.61968	-0.3116	-16.122

4.6 The Implementation of the Models

The results of the ArSL image-based recognition system are explained in detail in this section. A variety of processes are carried out using the system models, and every phase is carefully explained and assessed. Testing can be performed to determine the way the suggested models perform and to confirm the viability of the concept.

4.6.1 The Results of the LDA-CNN Alphabet Model

The LDA-CNN model is implemented to find the best performance by selecting a different number of epochs that can be used to increase detection accuracy. Results are shown in Table 4.7. Although the accuracy is very close when the number of epochs is between 100 and 150 it is preferable to use the number of epochs equal to 150, which gives very good results.

Based on the results provided in Table 4.7, the performance of the LDA-CNN model is observed on two different datasets, namely Dataset1 and Dataset2, for different numbers of epochs. These datasets include for alphabet consisting of 32 letters. Dataset1 is standard data consisting of 5400 images, while Dataset2 is aggregated data consisting of 2600 images.

Table 4.7: The Effect of epoch numbers on the detection accuracy.

No. Epochs	LDA-CNN Model				
	Name of dataset	Processing	**Accuracy**	**Loss. function**	**process Time**
10	**Dataset1**	Training	0.67	1.158	26.0s
		Testing	0.70	1.065	
	Dataset2	Training	0.38	2.66	1.678s
		Testing	0.37	2.54	
50	**Dataset1**	Training	0.95	0.152	27s
		Testing	0.94	0.173	
	Dataset2	Training	0.95	0.105	1.766s
		Testing	0.95	0.099	
100	**Dataset1**	Training	0.96	0.103	26s
		Testing	0.97	0.0985	
	Dataset2	Training	0.94	0.075	1.635s
		Testing	0.95	0.074	
150	**Dataset1**	Training	**0.998**	**0.0035**	27s
		Testing	**0.999**	**0.0025**	

For the Dataset1, the model achieves an accuracy of 67% after 10 epochs during the training phase, with a corresponding loss function value of 1.158. In the testing phase, the accuracy improves to 70% with a reduced loss function value of 1.065. As the number of epochs increases to 50, the model's performance improves significantly, reaching an accuracy of 95% in both training and testing, with lower loss function values of 0.152 and 0.173, respectively. The accuracy further increases to 96% and 97% in the training and testing phases, respectively, after 100 epochs, with reduced loss function values of 0.103 and 0.0985. Finally, after 150 epochs, the model achieves an impressive accuracy of 99.8% during training and 99.9% during testing, with extremely low loss function values of 0.0035 and 0.0025.

For Dataset2, the LDA-CNN model exhibits lower performance compared to Dataset1. After 10 epochs, the model achieves an accuracy of 38% during training, with a loss function value of 2.66. In the testing phase, the accuracy slightly drops to 37% with a loss function value of 2.54. As the number of epochs increases, there is a gradual performance improvement. After 50 epochs, the accuracy increases to 95% in both training and testing, with reduced loss function values of 0.105 and 0.099, respectively. Similar trends can be observed after 100 epochs, where the model achieves an accuracy of 94% in training and 95% in testing, with lower loss function values of 0.075 and 0.074. Finally, after 150 epochs, the model achieves an accuracy of 99.4% during training and 99.5% during testing, with a loss function value of 0.075.

The summary, the results indicate that increasing the number of epochs generally leads to improved performance in terms of accuracy and reduced loss function values for both datasets. The model performs better on Dataset1 in comparison Dataset2, achieving higher accuracy and lower loss function values. It is also worth noting that

after 150 epochs, the model demonstrates impressive accuracy rates, particularly on Dataset1, suggesting that it has effectively learned the patterns and features of the sign language gestures in the images.

In deep learning, the learning rate is a hyperparameter that controls how much the weights of the model are updated with each iteration during training. It is an important parameter to tune, as a too-high or too-low learning rate can result in slow convergence or even divergence of the training process.

To determine the best learning rate for a specific model, it is common practice to perform multiple experiments with different learning rates and evaluate the performance of the model on a validation set. The learning rate that gives the best accuracy or lowest loss on the validation set is usually selected as the optimal learning rate.

Table 4.8 summarizes the findings of an experiment to determine the best learning rate for a particular model on two different datasets. It shows that a learning rate of 0.001 gave the best accuracy for both datasets. It is important to note that increasing the learning rate beyond this optimal value may result in a decrease in performance and longer run time.

Therefore, selecting the appropriate learning rate is crucial for achieving good performance of the model while avoiding excessive training time. The optimal learning rate can depend on various factors such as the dataset, the complexity of the model, and the optimizer used, and it is often determined through trial-and-error experimentation.

Table 4.8: The Testing Accuracy vs. Learning Rate.

Dataset		Initial Learning Rate		
		0.01	0.001	0.0001
Accuracy %	**Dataset1**	33	**96**	64
	Dataset2	31	95	**96**

Table (4.8) shows the results of implementing an LDA-CNN model for recognizing individual Arabic letters, trained for 100 epochs with a batch size of 1024. This table displays the accuracy of the model on each individual letter, as well as the processing time for each batch of 1024 letters.

The processing time of 0.635s refers to the time it takes to process one batch of 1024 letters. This is an important metric as it determines how quickly the model can be trained on large datasets. The processing time can depend on various factors such as the hardware used, the complexity of the model, and the size of the input data.

By analyzing the accuracy results in Table 4.9, it is possible to evaluate the performance of the LDA-CNN model for recognizing individual Arabic letters. It is important to display the results of a classification model on multiple classes. It shows the precision, recall, and F1-score for each class, as well as the support number of samples for each class, it is a count of the number of true instances of the class in the test set.

Table 4.9: An Individual Sign for Arabic Letters

CNN ALPHABET

Name of class	precision	recall	f1-score	support
أ	0.99	0.93	0.96	2114
ب	0.98	0.94	0.96	1343
ت	0.99	0.96	0.97	1672
ث	0.93	1.00	0.96	1791
ج	0.98	0.97	0.98	1634
ح	0.96	0.96	0.96	1723
خ	0.99	0.95	0.97	1670
د	0.97	0.98	0.98	1955
ذ	0.96	0.95	0.96	1705
ر	0.99	0.97	0.98	1977
ز	0.91	0.98	0.95	1592
س	0.95	0.98	0.97	1526
ش	0.93	0.99	0.96	1552
ض	0.99	0.97	0.98	1774
ص	0.95	0.96	0.96	1607
ط	0.98	0.97	0.98	1746
ظ	0.95	0.98	0.96	1832
ع	0.99	0.96	0.98	1765
غ	0.97	0.96	0.97	1819
ف	0.97	0.96	0.97	1659
ق	0.97	0.97	0.97	1895
ك	0.98	0.98	0.98	1638
ل	0.97	0.99	0.98	1507
م	0.90	0.98	0.94	1816
ن	0.96	0.98	0.97	1838
و	0.97	0.95	0.96	1766
ه	0.97	0.97	0.97	1582
ي	0.97	0.97	0.97	1791
ة	0.99	0.97	0.98	1371
ال	0.98	0.96	0.97	1722
لا	0.98	0.96	0.97	1293
ئ	0.98	0.96	0.97	1374
accuracy			0.97	54049

4.6.2 The Results of LDA-ML Alphabet/Words Model

Based on the information provided, it shows that Table 4.10 presents the results of testing various machine learning algorithms on the Alphabet dataset, and displays the F1 score, recall, precision, and accuracy for each algorithm across all classes. It is being used to compare the performance of each ML algorithm on the dataset and to determine which algorithm is the most effective.

Table 4.10: The LDA-ML model's implementation measurements.

Model	Accuracy	Precision	Recall	F1 score
DT	0.45	0.45	0.45	0.45
KNN	**0.86**	**0.87**	**0.86**	**0.86**
RF	0.78	0.78	0.78	0.78
NB	0.50	0.50	0.50	0.50
SGD	0.50	0.49	0.50	0.44

Table 4.11 and Table 4.12 present the results of accuracy and precision, respectively, for implementing machine learning algorithms on a different number of datasets. These data include various letters and words, and these images are either images or videos that have been pre-processed. Figure 4.2 and Figure 4.3 show the graphical representation of the accuracy and precision ratios, respectively. It is demonstrated that, out of the five possible algorithms, k-NN is the most effective.

Table 4.11: The LDA-ML model's accuracy.

No. of Dataset	Accuracy %				
	NB	k-NN	DT	RF	SGD
Dataset1	50	**86**	45	78	44
Dataset2	55	**89**	50	80	50
Dataset3	54	**87**	51	79	52

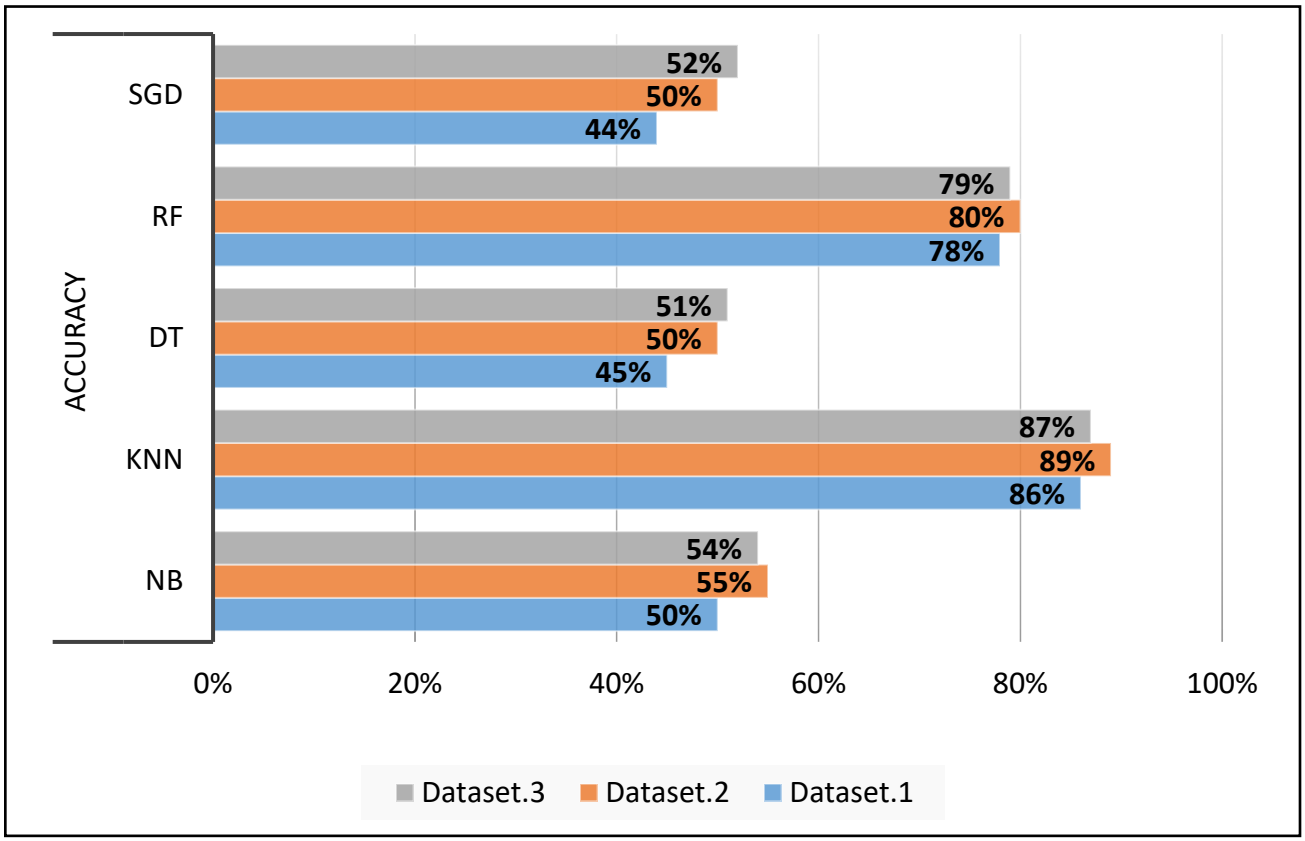


Figure 4.2: Accuracy of the LDA-ML Model

Table 4.12: The LDA-ML model's precision.

No. of Dataset	Precision %				
	NB	k-NN	DT	RF	SGD
Dataset1	55	**89**	50	76	40
Dataset2	55	**85**	50	78	50
Dataset3	54	**87**	51	79	52

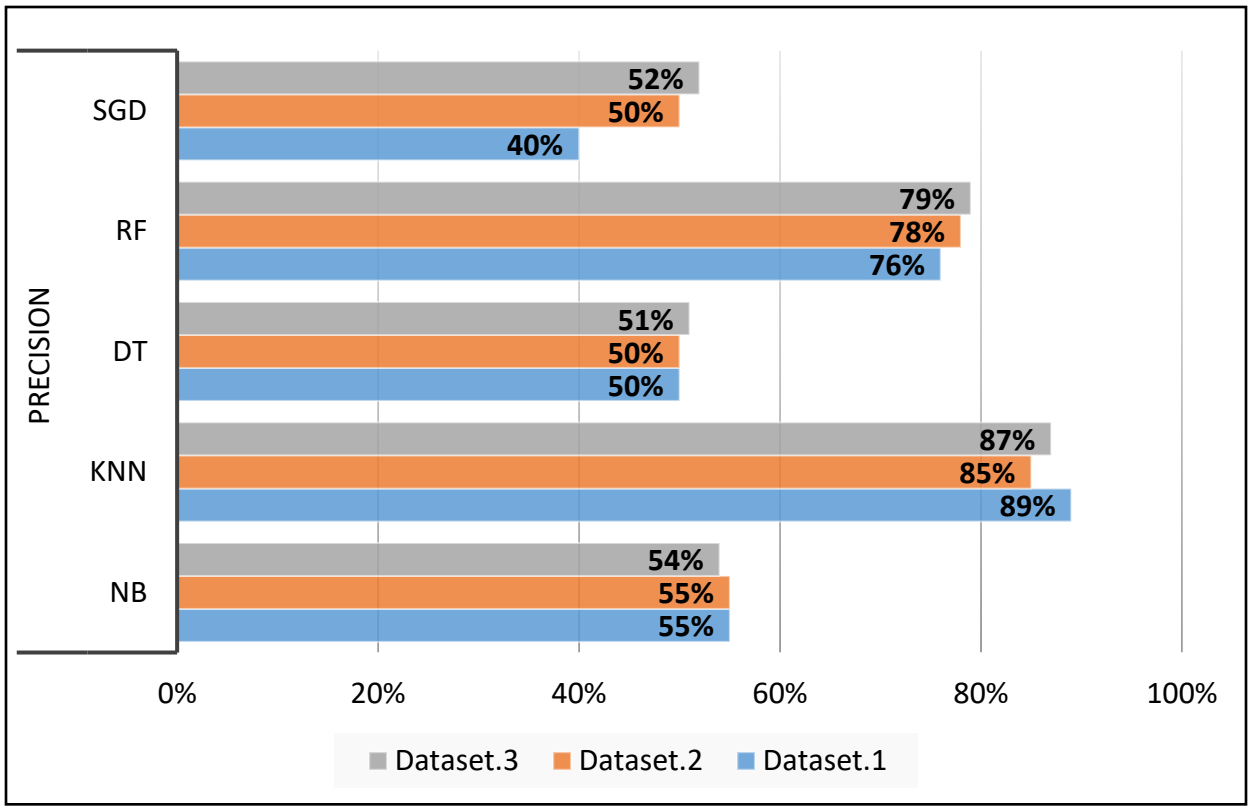


Figure 4.3: Precision of the LDA-ML Model

Through the previous results of implementing the LDA-ML model, it is noted that there are differences in the performance of the machine learning algorithms, because the image lighting, background, video quality, hand geometry, skin color, etc. are just a few of the variables that can affect the validity of the interpretation.

For this apply implemented Linear Discriminant Analysis for trait vector extraction (used as input for the five classification algorithms) because it is insensitive to differences in illumination, hand position, and hand rotation and rotation.

Based on the results provided in this table, we can observe the testing performance of different models, namely Decision Tree (DT), k-Nearest Neighbors (KNN), Random Forest (RF), Naive Bayes (NB), and Stochastic Gradient Descent (SGD). The evaluation metrics used to assess the models include accuracy, precision, recall, and F1 score.

The summary, the KNN model demonstrates the highest accuracy among the models, indicating its effectiveness in correctly classifying the test samples. The RF model also performs reasonably well with a balanced accuracy, while the DT, NB, and SGD models show lower accuracy scores, indicating the need for further improvement. It is important to note that the evaluation metrics may vary depending on the specific dataset and problem being addressed, so further analysis and comparison with other metrics would provide a more comprehensive understanding of the models' performance

4.6.3 The Results of Face-CNN Model

The 1DCNN model for classification and the facial features extracted using the Viola-Jones technique are combined in this model. Only the lips, which have 18 distinct features and are categorized according to how they express motions and words, are employed as special facial features.

Experiments revealed that by combining the 1DCNN model and the Viola-Jones algorithm, the new model is more effective and accurate at identifying the motions of words produced by the lips. It is crucial to keep in mind that the success of this strategy will depend on the caliber and volume of the training data as well as the

precise features that the Viola-Jones algorithm extracts. Table 4.13 shows the accuracy of the 1DCNN during training for several periods. Table 4.14 shows the feature-by-feature accuracy.

Table 4.13: The Face-CNN model's implementation measurements.

No. of Epoch	Model CNN for face			
	Accuracy	Precision	Recall	F1 score
10	78.0	0.83	0.78	0.79
50	96.9	99.8	90.0	95.0
100	**99.9**	**99.0**	**99.0**	**99.0**

Table 4.14 *The* Face-CNN model's

Landmarks for face 1DCNN

Features	precision	recall	f1-score	support
0	0.87	1.00	0.93	47
1	1.00	1.00	1.00	58
2	1.00	1.00	1.00	31
3	1.00	0.94	0.97	54
4	0.88	1.00	0.94	37
5	1.00	1.00	1.00	76
6	1.00	1.00	1.00	38
7	1.00	1.00	1.00	63
8	1.00	1.00	1.00	97
9	0.92	1.00	0.96	72
10	1.00	0.76	0.86	45
11	0.93	1.00	0.96	41
12	1.00	1.00	1.00	38
13	1.00	1.00	1.00	92
14	1.00	1.00	1.00	40
15	1.00	0.88	0.94	58
16	1.00	1.00	1.00	67
17	1.00	1.00	1.00	51
accuracy			**0.98**	**1036**

4.6.4 The Results of Hybrid CNN-ML Model

This hybrid model is a combination of 1DCNN implementation with machine learning algorithms. To clarify the results of the system, the implementation results are presented separately for each stage of the work. Table 4.15 shows the implementation of the results of the 1DCNN for gesture classification. Table 4.16 shows the results of implementing machine learning algorithms NB, DT, and RF in gesture classification.

Table 4.15: The Results of 1DCNN in the classification of gestures

No. of Dataset	Model CNN			
	Accuracy	Precision	Recall	F1 score
Dataset1	98.9	97.9	97.8	**98.8**
Dataset2	97.8	96.9	95.9	**96.3**
Dataset3	98.8	97.8	96.8	**97.2**

Table 4.16: The accuracy of Machine learning algorithms for gesture classification.

No. of Dataset	Accuracy %		
	NB	DT	RF
Dataset1	55	50	**76**
Dataset2	55	50	**78**
Dataset3	54	51	**79**

Table 4.15 represents the implementation of the 1DCNN on three types of data. The results are somewhat high-performance, but during processing in the testing phase there is a delay of approximately 0.26 ms, and this causes a problem during

the translation of the video, as the translation is not synchronized with the movement.

Table 4.16 shows the results of implementing the classification algorithms in which the accuracy is somewhat low, but the processing time is fast.

Therefore, the new results in Table 4.17 and Figure 4.4 are the outputs of the implementation of the hybrid model.

Table 4.17: The Hybrid CNN-ML model's implementation measurements.

No. of Dataset	F1 score %		
	CNN-NB	CNN-DT	CNN-RF
Dataset1	95.7	99.9	**99.9**
Dataset2	98.9	99.7	**99.8**
Dataset3	98.8	99.6	**99.8**

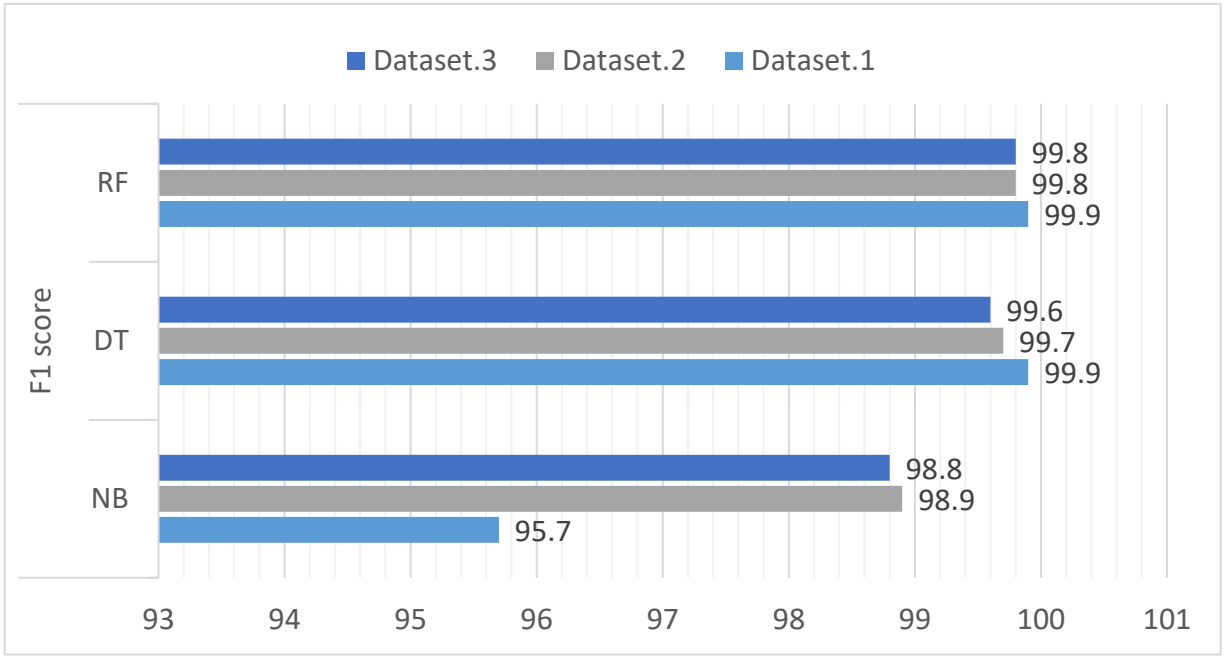


Figure 4.4: F1-score of the Hybrid CNN-ML.

Table 4.17 represents the implementation measurements of a Hybrid CNN-ML model using different machine learning algorithms (Naive Bayes, Decision Tree, and Random Forest) to classify the outputs of a CNN model. The F1 score, which is a harmonic mean of precision and recall, is used as the performance metric.

Let is discuss this table and its implications:

The different datasets represent the features of letters and words to which the Hybrid CNN-ML model was applied. This difference helps distinguish the different data sets used for testing and evaluation.

- **CNN-NB**: The Hybrid CNN-ML model with the Naive Bayes algorithm.
- **CNN-DT**: The Hybrid CNN-ML model with the Decision Tree algorithm.
- **CNN-RF**: The Hybrid CNN-ML model with the Random Forest algorithm.

It indicates that the **Hybrid CNN-ML** model performs well across all three datasets, with consistently high F1 scores for all machine learning algorithms. In Dataset1, the CNN-DT and CNN-RF combinations achieve particularly high F1 scores of 99.9%. Dataset.2 also demonstrates high F1 scores, with the CNN-DT and CNN-RF combinations performing well at 99.7% and 99.8% respectively. Dataset.3 maintains a similar pattern, with F1 scores ranging from 98.8% to 99.8%.

- **Overall Performance**: The hybrid model consistently performs well, with F1 scores ranging from 95.7% to 99.9%. These scores indicate that the model achieves a high balance between precision and recall for each dataset.
- **Comparison of Algorithms**: When comparing the machine learning algorithms, CNN-DT and CNN-RF consistently achieve higher F1 scores than CNN-NB. This suggests that decision tree and random forest algorithms

provide better performance when combined with the CNN model for these datasets.

- **Potential Overfitting:** It should be noted that the F1 scores are very close to 100% for some combinations of the hybrid model and machine learning algorithms. While this indicates high performance, it may also indicate the possibility of overfitting, as the model performs exceptionally well on training data but within epochs of 150 which is the stability limit of the network.

In summary, the **Hybrid CNN-ML model** achieves high F1 scores across the dataset, with the combination of CNN and decision tree or random forest algorithms demonstrating strong performance. However, it's important to further validate the model's generalization capability and guard against potential overfitting when applying it to new and unseen data. It is regarded as successful when the time it takes to recognize a gesture is between 0.15 and 0.25 seconds.

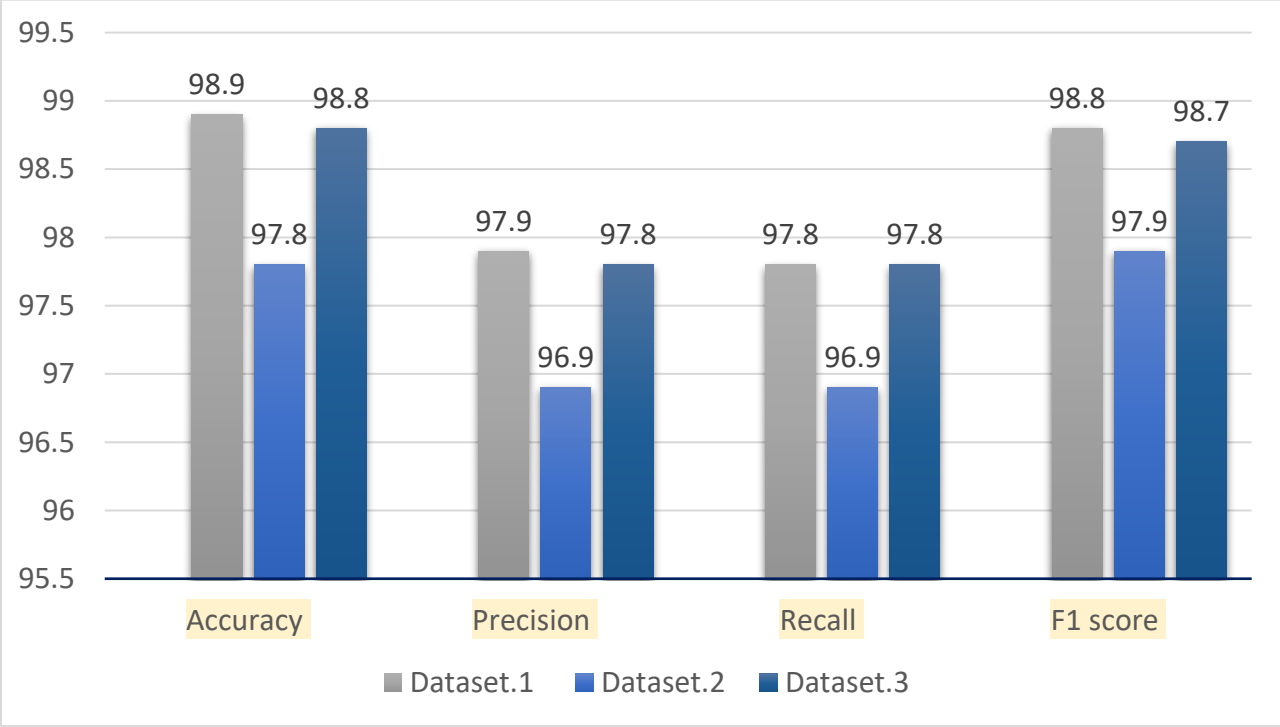


Figure 4.5: F1-score of the Hybrid CNN-RF.

Figure 4.5 shows the performance measures of the hybrid model CNN with RF that is specifically applied to the features of word gestures extracted from video clips.

Table 4.18 shows the performance evaluation during the implementation of the DT algorithm with CNN for each class, where each class represents a word of Arabic sign language. Table 4.19 displays the outcomes of using the algorithm NB and CNN. Table 4.20 illustrates RF and CNN algorithms' findings. This is contributing to distinguishing the classes in each gesture.

Table 4.18: The Outcomes of DT and CNN for each class in words dataset

CCN&DT

No. of class	precision	recall	f1-score	support
1	1.00	1.00	1.00	102
2	1.00	1.00	1.00	75
3	1.00	1.00	1.00	102
4	1.00	0.99	0.99	91
5	0.99	1.00	1.00	139
6	1.00	1.00	1.00	66
7	1.00	1.00	1.00	75
8	1.00	1.00	1.00	167
9	1.00	1.00	1.00	130
10	1.00	1.00	1.00	65
11	1.00	1.00	1.00	167
12	1.00	1.00	1.00	84
13	1.00	1.00	1.00	95
14	1.00	1.00	1.00	60
15	1.00	1.00	1.00	123
16	1.00	1.00	1.00	92
17	1.00	1.00	1.00	144
18	1.00	1.00	1.00	104
19	1.00	1.00	1.00	167
20	1.00	1.00	1.00	114
21	1.00	1.00	1.00	90
22	1.00	1.00	1.00	84
23	1.00	1.00	1.00	68
24	1.00	1.00	1.00	105
25	1.00	1.00	1.00	44
26	1.00	1.00	1.00	54
27	1.00	1.00	1.00	115

28	1.00	1.00	1.00	40
29	1.00	1.00	1.00	72
30	1.00	1.00	1.00	33
accuracy			1.00	2867

Table 4.19: The Outcomes of NB and CNN

CNN&NB

No. of class	precision	recall	f1-score	support
1	0.76	1.00	0.87	78
2	0.85	0.91	0.88	70
3	0.90	0.93	0.92	99
4	0.83	0.82	0.83	91
5	0.85	0.89	0.87	134
6	0.79	0.88	0.83	59
7	0.85	0.90	0.88	71
8	0.98	0.73	0.84	222
9	0.71	0.96	0.81	96
10	0.92	0.59	0.72	102
11	0.82	0.81	0.81	170
12	0.92	0.83	0.87	93
13	0.88	0.98	0.93	86
14	0.90	0.87	0.89	62
15	0.85	0.95	0.90	110
16	0.87	0.92	0.89	87
17	0.97	0.89	0.93	156
18	0.76	0.92	0.83	86
19	0.95	0.89	0.92	177
20	0.89	0.91	0.90	112
21	0.91	0.86	0.89	95
22	0.90	0.92	0.91	83
23	0.87	0.91	0.89	65
24	0.93	0.95	0.94	103
25	0.91	0.89	0.90	45
26	0.91	0.91	0.91	54
27	0.97	0.95	0.96	118
28	0.88	0.92	0.90	38
29	0.92	0.89	0.90	74
30	0.76	0.81	0.78	31
accuracy			0.88	2867

Table 4.20: The Outcomes of RF and CNN

CNN&RF

No. of class	precision	recall	f1-score	support
1	1.00	1.00	1.00	102
2	1.00	1.00	1.00	75
3	1.00	1.00	1.00	102
4	1.00	0.99	0.99	91
5	0.99	1.00	1.00	139
6	1.00	1.00	1.00	66
7	1.00	1.00	1.00	75
8	1.00	1.00	1.00	167
9	1.00	1.00	1.00	130
10	1.00	1.00	1.00	65
11	1.00	1.00	1.00	167
12	1.00	1.00	1.00	84
13	1.00	1.00	1.00	95
14	1.00	1.00	1.00	60
15	1.00	1.00	1.00	123
16	1.00	1.00	1.00	92
17	1.00	1.00	1.00	144
18	1.00	1.00	1.00	104
19	1.00	1.00	1.00	167
20	1.00	1.00	1.00	114
21	1.00	1.00	1.00	90
22	1.00	1.00	1.00	84
23	1.00	1.00	1.00	68
24	1.00	1.00	1.00	105
25	1.00	1.00	1.00	44
26	1.00	1.00	1.00	54
27	1.00	1.00	1.00	115
28	1.00	1.00	1.00	40
29	1.00	1.00	1.00	72
30	1.00	1.00	1.00	33
accuracy			1.00	2867

4.6.5 The Results of the VGG-16 Models

4.6.5.1 The Two-Dimensional VGG-16 Model

This section describes the results of the model for the building of the classification gestures for Arabic words using the VGG-16 for the two-dimensional model.

In this case, the data entered after the pre-processing process is used before the process of applying the LDA algorithm for feature extraction. The VGG-16 model receives images as input to the model to classify the extracted features into ASL.

Figures 4.6 and 4.7 depict the classification accuracy and loss rate report, respectively. As seen in the results, miss-classification is low leading to high accuracy. The performances in this section are obtained utilizing the data split into two parts during training: 70% and 30%. Training of the model is performed using 50 epochs. At the test phase, the obtained accuracy rate is 88% and the error is 0.335.

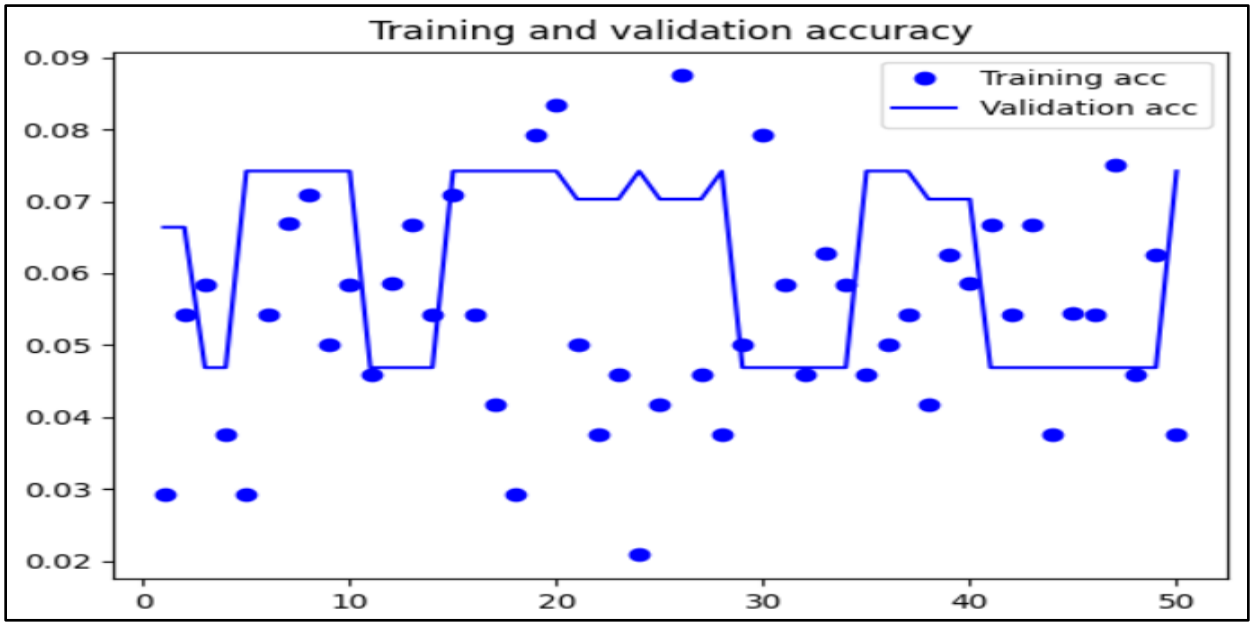


Figure 4.6: The accuracy for two dim VGG-16.

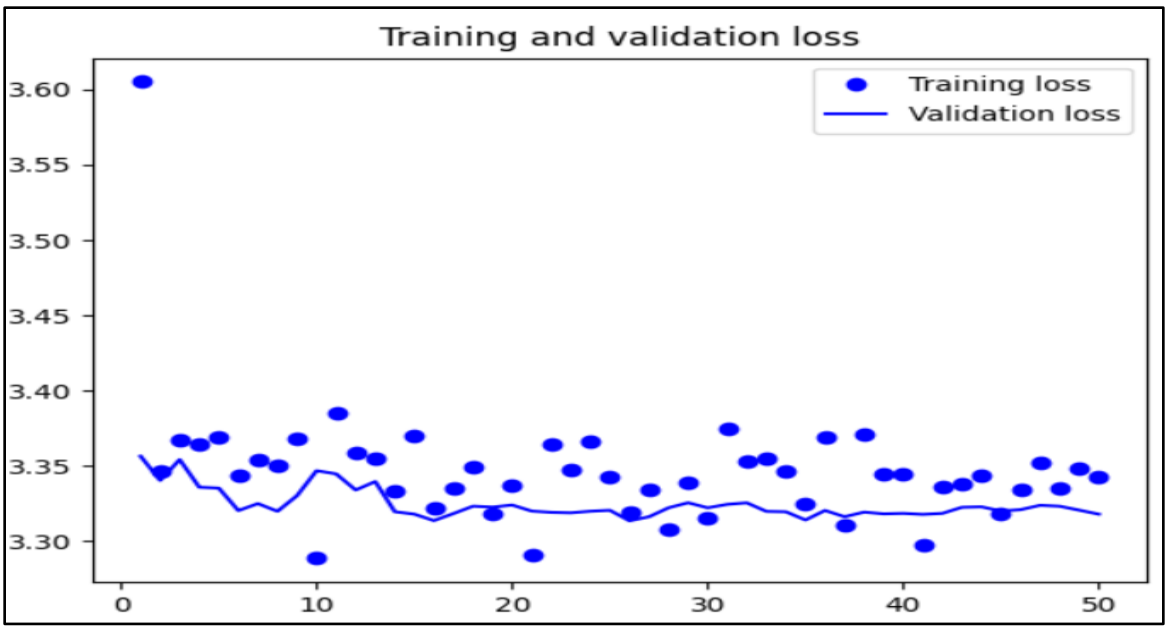


Figure 4.7: The Loss function for two dim VGG-16.

4.6.5.2 The One-Dimensional VGG-16 Model

In this part, an ASL words and letters dataset is utilized to construct the model, using a one-dimensional VGG-16 network. The model utilizes the features extracted from each hand image after applying the LDA algorithm. The VGG-16 network is modified to have a one-dimensional structure and is applied to two types of data: letter data and word data. The design of this model is similar to other models proposed earlier in the text. The model is trained for 100 epochs and the results are fixed after some processing.

In Table 4.21, the performance evaluation is shown for each category, where each category represents a letter of the Arabic sign language. Table 4.22 the results are shown for a dataset, where there is a clear decrease compared to the results of our previously designed models, in addition to the training time and the test time being longer.

Table 4.21: The Outcomes of the VGG-16 model

No. of Dataset	Model VGG-16 One Dim			
	Accuracy	Precision	Recall	F1 score
Dataset1 (Alphabets)	91.0	92.0	91.0	91.0
Dataset3 (Words)	44.0	36.0	44.0	39.0

Table 4.22: The Outcomes VGG-16 models for alphabet class

Name of class	precision	recall	f1-score	support
أ	0.83	0.94	0.88	195
ب	0.85	0.77	0.81	136
ت	0.95	0.90	0.93	183
ث	0.90	0.94	0.92	182
ج	0.98	0.94	0.96	187
ح	0.93	0.93	0.93	170
خ	1.00	0.96	0.98	144
د	0.80	0.66	0.72	178
ذ	0.89	0.78	0.83	147
ر	0.91	0.94	0.93	229
ز	0.96	0.89	0.93	152
س	0.97	0.94	0.95	149
ش	0.97	0.80	0.88	159
ص	0.94	0.94	0.94	180
ض	0.94	0.94	0.94	186
ط	0.82	0.99	0.89	168
ظ	0.97	0.89	0.93	165
ع	0.97	0.92	0.95	161
غ	0.88	0.86	0.87	162
ف	0.95	0.99	0.97	178
ق	0.95	0.76	0.84	188
ك	0.95	0.91	0.93	172
ل	0.90	0.95	0.92	157
م	0.92	0.96	0.94	170
ن	0.92	0.94	0.93	198
ه	0.83	0.92	0.87	179
و	0.96	0.99	0.97	151
ي	0.94	0.94	0.94	162
ة	0.99	0.99	0.99	153
ال	0.71	0.92	0.80	179
لا	0.91	0.99	0.95	139
ئ	0.97	0.99	0.98	146

4.7 Comparison of the Results of Various Models

It is important to note that the performance of these models can vary depending on the specific dataset used, the number of classes, and other factors such as lighting, hand position, and rotation. Therefore, it is important to carefully evaluate the results and consider the limitations of each study.

Table 4.23 shows a comparison of the accuracy rates achieved by different models for Arabic Sign Language (ArSL) recognition, using various datasets. The comparison may include models with their corresponding accuracy rates. This table may also indicate whether the models were tested on the standard ArSL2018 dataset or other datasets.

Table 4.23: Evaluation in light of prior research

Index	Model	Accuracy	Dataset for alphabets
1.	ResNet-18 [80]	93%	ArSL2018
2.	CNN [81]	96.4	ArSL2018
3.	CNN [32]	92.9	ArSL2018
4.	CNN [82]	88.87	ArSL2018
5.	**LDA-CNN Model**	**99.98%**	**ArSL2018**
1	CNN Model [83]	90%	Collected Dataset
2	PCNN [84]	90.4	Collected Dataset
3	ANFIS networks [85]	93.55	Collected Dataset
4	R-CNN [80]	93	Collected Dataset
5	CNN [83]	90	Collected Dataset
6	**LDA-CNN Model**	**98.57**	**Collected Dataset**

The second comparison involved experiments that also included ArSL words and extracted words from images or videos with LDA-ML and LDA-CNN models. The outcome is shown in Table 4.24. The goal of this comparison was to evaluate the performance of the LDA-ML and LDA-CNN models in recognizing ASL words from images or videos.

In this comparison, the LDA-ML model achieved an accuracy of 96.89%, while the LDA-CNN model achieved an accuracy of 98.57%. The results suggest that the LDA-CNN model is more effective in recognizing ASL words from images or videos compared to the LDA-ML model.

Table 4.24: The Evaluation in light of Arabic sign language words.

Index	Model	Accuracy	Dataset for words
1	CNN Model [19]	88.87	10 Arabic words
2	Geometric Features [11]	97.0	30 isolated words
3	Decision Tree & Bayesian Network [17]	93.7	42 Arabic gestures
6	**LDA-ML Model**	**96.89**	80 words
7	**LDA-CNN Model**	**98.57**	80 words

The third comparison is made between all models in this dissertation, depending on a Mean Squared Error, Root Mean Squared Error, and Mean Absolute Error. The error in statistical models is measured by a statistic called the mean squared error (MSE). To do this, it calculates the average squared deviation between the observed and projected values as shown in Table 4.25. The MSE is considered to be "zero"

when there are no mistakes in the model. The worth of a model's inaccuracy goes up as it grows larger. Table 4.26 explains the compression process time for all models during testing time.

Table 4.25: The Evaluation of models of Arabic sign language.

index	Model	MSE	RMSE	MAE
1.	**LDA-CNN of alphabet**	5.645	2.376	0.354
2.	**LDA-CNN of Word**	1.965	1.402	0.200
3.	**Hybrid CNN-ML**	0.055	0.236	0.009
4.	**Face-CNN**	1.885	1.373	0.159
5.	**VGG-16 ID**	11.806	3.436	1.314
6.	**VGG-16 2D**	**332.06**	**18.22**	**2.897**

Table 4.26: The Evaluation of the testing process.

index	Model	Testing time	Acc.
1.	**LDA-CNN of alphabet**	1.711ms	0.98
2.	**LDA-CNN of Word**	1.734ms	0.97
3.	**Hybrid CNN-ML**	**0.711ms**	0.99
4.	**Face-CNN**	**3.5ms**	0.97
5.	**VGG-16 ID**	**1.10ms**	0.94
6.	**VGG-16 2D**	**5.2ms**	0.88

4.8 Interfaces for System Implementation

As previously stated in Section 3.8, the implementation of the real-time system relies on two models: the hybrid CNN-ML model and the Face-CNN model. These models are utilized to achieve optimal performance in terms of accuracy and timing, with the system output dependent on the real-time video threshold.

"Threshold" refers to a value or threshold that is used to define the parameters for a particular event or process that occurs in processing. The threshold in this system is the point at which the system initiates a specific action or output, depending on the hybrid model or face model.

The resulting Figure 4.8 is based on the CNN-ML model, where the translation appears during the movement of the hands, while in Figure 4.9, the translation appears depending on the Face-CNN model before the movement of the hands.

Figure 4.8: The proposed system's GUI.

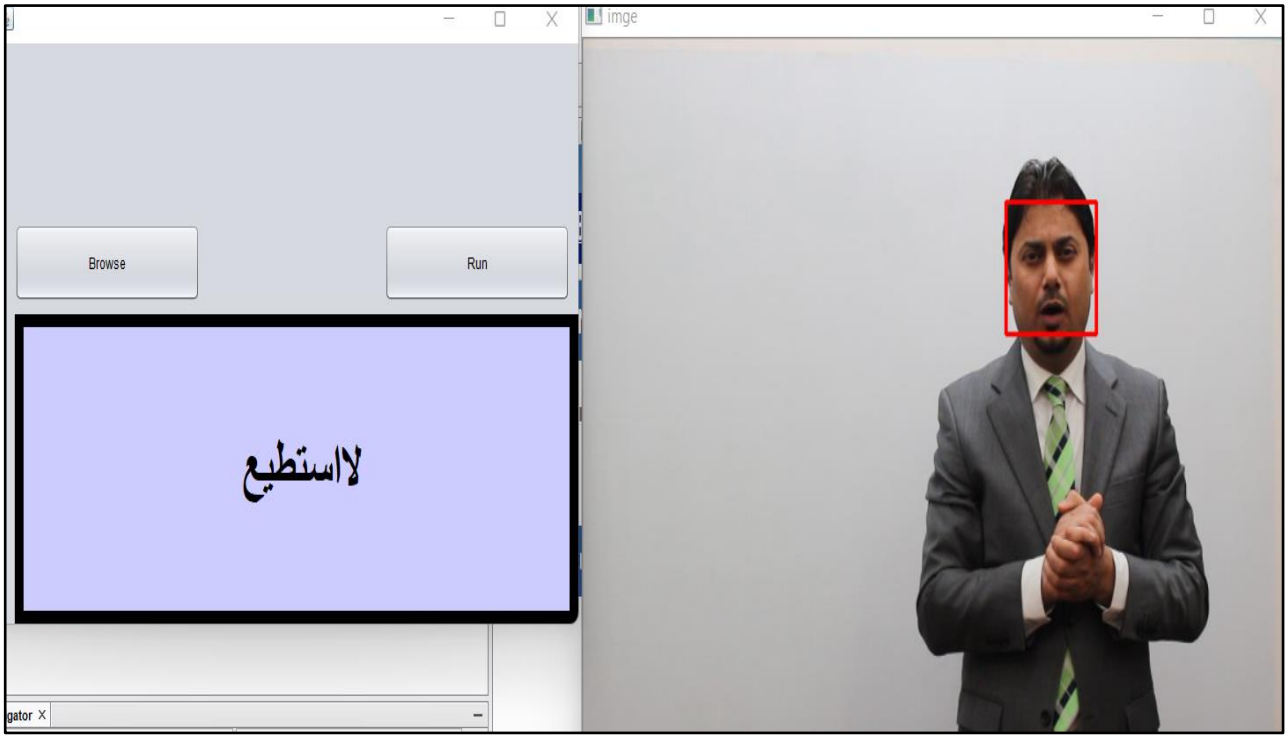


Figure 4.9: The proposed system's recognition phase.

CHAPTER FIVE

Conclusions and Future Work

Suggestions

5.1 Conclusions

In this dissertation, Convolutional Neural Networks, Machine Learning algorithms, and Viola-Jones algorithms were used to detect and classify Arabic sign language through the use of images and videos. Comparing the finding with other approaches using 1DCNN and machine learning, our final model achieved optimal performance and reached an accuracy of more than 99.9%. It is possible to reach 100% accuracy if the number of epochs is increased to 150, but this will increase the system compact. We conclude that 1DCNN will work perfectly for object detection and classification. In the light of the results obtained, the study concluded the following: -

1. The proposal for interpreting Arabic sign language using a one-dimensional convolutional network (1DCNN) system is effective in identifying and distinguishing words with high performance and fast execution time, through the combination of machine learning algorithms, deep learning, and an algorithm for recognizing facial cues.
2. This system employs a 1D convolutional network, which is well-suited for low-cost real-time applications due to its low computational demands. This is particularly advantageous for portable or mobile devices. In contrast, training deep 2D CNNs often necessitates specialized resources such as cloud computing or GPUs. However, training 1DCNNs with multiple hidden layers and neurons on a standard computer is both feasible and relatively fast.

3. The CNN component of the model is responsible for extracting relevant features from the input data. CNNs are particularly effective in image analysis tasks, as they can learn hierarchical representations of visual patterns. In this case, the CNN likely processes visual representations of Arabic alphabet signs and extracts meaningful features that capture important characteristics for classification. On the other hand, ML algorithms like decision trees or random forests are known for their classification capabilities. These algorithms use the extracted features from the CNN as input and make predictions about the class labels of the input data. The combination of CNN and ML in the Hybrid CNN-ML model leverages the speed of CNN for feature extraction and the efficiency of ML algorithms for classification. By combining the strengths of Convolutional Neural Networks (CNN) for feature extraction and Machine Learning (ML) algorithms like decision trees or random forests for classification, this model achieves high F1 scores across the dataset.
4. To preserve the temporal information of a video, it is crucial to avoid treating it as a single continuous stream of frames. Instead, it is recommended to implement segmentation based on specific features or characteristics of the video. For example, in the context of sign language, segmentation can be based on stop motion, which can be used as a transition point to another gesture while speaking. By dividing the video into smaller periods based on these transitions, the temporal information of the video can be preserved and analyzed more effectively. This approach allows for a more accurate analysis of changes over time and can provide insights into patterns and trends that may not be apparent when analyzing the video as a whole. Therefore, in this dissertation, segmentation based on stop motion was implemented as a way to represent the video as a time series and preserve its temporal information.

In conclusion, the identification of Sign Languages and Arabic Sign Language (ArSL) was studied, this was performed using several types of classifications as well as, using different gestures and signs in ArSL. The proposal focused on using various datasets, including letters and words, either as images or video clips. It also used multiple algorithms by combining them. It achieved results, high performance, and speed with good gesture translation. The output of the proposed system is an Arabic text obtained through the detection of Arabic sign language. Moreover, the solution presented here would be excellent for people with disabilities.

5.2 Future Work Suggestions

There are some future work suggestions to further develop the performance and efficiency of this work for Arabic sign language recognition and translation:

1. **Exploring multi-feature attributes**: Investigating additional sign language attributes beyond hand gestures, such as human body movements and facial expressions, can contribute to more accurate and nuanced sign language recognition. Continuous body motion, for example, can be utilized to determine the resolution of the signal and provide more context for accurate translation. Considering a multi-modal approach that combines various features and modalities can lead to a more comprehensive and robust system for sign language recognition and translation.
2. **Building a broader dictionary**: Expanding the dictionary to include a larger set of Arabic words and signs would enhance the translation capabilities of the model. Collecting data from video clips or images of different signs performed by a diverse group of sign language users can help in achieving a more comprehensive and representative dataset. This would allow the model to recognize and translate a wider range of Arabic sign language gestures and improve its usability for real-world applications.
3. **Online application with video broadcast**: Implementing the proposed model online with video broadcast capabilities would enable real-time sign language recognition and translation. This would make it accessible to a larger user base and provide a practical tool for communication between sign language users and non-sign language users in online platforms, video conferences, or other digital communication channels.

REFERENCES

[1] B. S. Parton, "Sign Language Recognition and Translation: A Multidisciplined Approach From the Field of Artificial Intelligence," *The Journal of Deaf Studies and Deaf Education,* vol. 11, doi.org/10.1093/deafed/enj003, pp. 94-101, 2006.

[2] M. Center, Deafness and hearing loss, Tech. rep.,445 World Health Organization, 2020????.

[3] Maha S. Altememe and Nidhal K. El Abbadi, "A Review for Sign Language Recognition Techniques," in *1st. Babylon International Conference on Information Technology and Science 2021*, Babil- IRAQ, 2021.

[4] L. Yang, J. Chen, and W. Zhu, "Dynamic Hand Gesture Recognition Based on a Leap Motion Controller and Two-Layer Bidirectional Recurrent Neural Network," *IEEE Sensor. J.*, vol. 20, 2106, p. 7, 2020.

[5] M.M. Reda and N.G. Mohammed, "SVBiComm: Sign-Voice Bidirectional Communication System for Normal, “Deaf/Dumb” and Blind People based on Machine Learning," in *1st International Conference on Computer Applications & Information Security (ICCAI),IEEE, pages 1-8*, 2018.

[6] Ghazanfar Latif, Nazeeruddin Mohammad, Jaafar Alghazo, Roaa AlKhalaf and Rawan AlKhalaf, "ArASL: Arabic Alphabets Sign Language Dataset, Data in Brief,," *ELSEVIER*, vol. 23, doi: org/10.1016/j.dib.2019.103777., 2019.

[7] M.A. Almasre and H. Al-Nuaim, "A comparison of Arabic sign language dynamic gesture recognition models," *Published by Elsevier Ltd.*, vol. 6, doi: 10.1016/j.heliyon.2020.e03554, p. 14, 2020.

[8] Sahoo, Ashok and Mishra, Gouri and Ravulakollu, Kiran, "Sing Inguage Recognition: State of the art," *ARPN Journal of Engineering and Applied Sciences,* vol. 9, pp. 116-134, 2014.

[9] S. Barnett, "Communication with deaf and hard-of-hearing people: a guide for medical education.," *Acad Med*, vol. 7, no. 77, doi: 10.1097/00001888-200207000-00009. PMID: 12114142., pp. 694-700, 2002.

[10] Nashwa El-Bendary, Hossam M. Zawbaa, Mahmoud S. Daoud, Aboul Ella Hassanien and Kazumi Nakamatsu, "ArSLAT: Arabic Sign Language Alphabets Translator," *International Journal of Computer Information Systems and Industrial Management Applications.*, vol. 3, pp. 498-506, 2011.

[11] M. Mohandes, S. Aliyu, and M. Deriche, "Arabic sign language recognition using the leap motion controller," in *2014 IEEE 23rd International Symposium on Industrial Electronics (ISIE)*, Istanbul, Turkey, 2014.

[12] Nada B. Ibrahim, Mazen M. Selim and Hala H. Zayed, "An Automatic Arabic Sign Language Recognition System (ArSLRS)," *Journal of King Saud University - Computer and Information Sciences,* vol. 30, doi.org/10.1016/j.jksuci.2017.09.007., pp. 470-477, 2018.

[13] P.K. Athira, C.J. Sruthi and A. Lijiya, "A Signer Independent Sign Language Recognition with Co-articulation Elimination from Live Videos: An Indian Scenario," *Journal of King Saud University - Computer and Information Sciences,* vol. 34, doi.org/10.1016/j.jksuci.2019.05.002, pp. 771-781, 2019.

[14] Basma Hisham and Alaa Hamouda, "Supervised learning classifiers for Arabic gestures recognition using Kinect V2," *SN Applied Sciences, Springer Nature Switzerland,* vol. 1, doi.org/10.1007/s42452-019-0771-2, p. 21, 2019.

[15] M. ElBadawy, A. S. Elons, H. A. Shedeed, and M. F. Tolba,, "Arabic sign language recognition with 3D convolutional neural networks," in *Eighth International Conference on Intelligent Computing and Information Systems (ICICIS)*, 2017.

[16] Salma HayaniSalma HayaniMohamed BenaddyMohamed BenaddyO.E. MeslouhiO.E. MeslouhiMustapha Kardouchi, "Arab Sign language Recognition with Convolutional," in *2019 International Conference of Computer Science and Renewable Energies (ICCSRE)*, Agadir, Morocco, 2019.

[17] M. M. Kamruzzaman, "Arabic Sign Language Recognition and Generating Arabic Speech Using Convolutional Neural Network," *Wireless Communications and Mobile Computing,* vol. 2020, doi.org/10.1155/2020/3685614, p. 9, 2020.

[18] Hernandez V., Suzuki T. and Venture G., "Convolutional and recurrent neural network for human activity recognition: Application on American sign language.," *PloS one,* vol. 15, doi.org/10.1371/journal.pone.0228869, p. 12???, 2020.

[19] E. K. Elsayed, D. R. Fathy,, "Sign Language Semantic Translation System using Ontology and DeepLearning," *Journal of Advanced Computer Science and Applications,* vol. 11, DOI:10.14569/ijacsa.2020.0110118, pp. 141-147, 2020.

[20] Asma Althagafi Ghofran Althobaiti, Tahani Alsubait and Tahani Alqurashi, "ASLR: Arabic Sign Language Recognition Using Convolutional Neural Networks," *IJCSNS International Journal of Computer Science and Network Security,* vol. 20, doi: paper.ijcsns.org/07_book/202007/20200716.pdf, p. 7, July 2020.

[21] Z. Mohammed, A. Youse, D. Guo, Y. Mamun Elahi and M. Hu, "Sign Language Recognition for Arabic Alphabets Using Transfer Learning Technique," *Hindawi Limited*, vol. 11, no. 1, 2020., no. 1687-5265????, https://doi.org/10.1155/2022/4567989, 2020.

[22] Susan D. Fischer & Harry van der Hulst, Sign Language Structures, Oxford University Press, DOI: 10.1093/oxfordhb/9780199750986.013.0024, 2012.

[23] C. M., "Pattern Recognition and Machine Learning," *Springer*, vol. 7(4), 2017.

[24] A.-F. M., "Arabic Sign Language: A Perspective," *Journal of Deaf Studies and Deaf Education*, vol. 10, doi: 10.1093/deafed/eni007., pp. 212-221, 2005.

[25] Al-Ahdal, M. Ebrahim and Md Tahir Nooritawati, "Review in sign language recognition systems," *IEEE Symposium on Computers & Informatics (ISCI)*, no. doi: 10.1109/ISCI.2012.6222666, pp. 52-57, 2012.

[26] M. F. Tolba and A. S. Elons, "Recent developments in sign language recognition systems,," in *8th Int. Conf. Comput. Eng. Syst. ICCES,*, 2013.

[27] Reema Alzohairi, Raghad Alghonaim, Waad Alshehri and Shahad Aloqeely, "Image based Arabic Sign Language Recognition," *(IJACSA) International Journal of Advanced Computer Science and Applications*, vol. 9, 2018.

[28] Roaa AlKhalaf and Rawan AlKhalaf, " Arabic Alphabets Sign Language Dataset," *National Center for Biotechnology Information*, vol. 23,doi: 10.1016/j.dib.2019.103777, p. 10, 2019.

[29] A.-B. S., "Arabic Sign Language Recognition Using an Instrumented Glove," in *King Fahd University of Petroleum & Minerals*, Saudi Arabia, 2004.

[30] N. Tubaiz, T. Shanableh and K. Assaleh, "Glove-based continuous Arabic sign language recognition in user-dependent mode," *IEEE Trans Hum.-Mach. Syst.*, vol. 45, p. 526–533, Aug. 2015.

[31] Aliyu S., Mohandes M., Deriche M. and Badran S., "Arabie sign language recognition using the Microsoft Kinect," in *13th international multi-conference on systems, signals and devices (SSD). IEEE, Leipzig, pp 301–306*, 2016.

[32] Arslan Arif and Syed Tahir Hussain Rizvi and Iqra Jawaid and Muhammad Adam Waleed and Muhammad Raheel Shakeel, "An American Sign Language (ASL) Translator," in *International Conference on Control, Decision and Information Technologies (CoDIT)*, DOI:10.1109/CoDIT.2016.7593642, 2016.

[33] Cooper H., Holt B. and Bowden R., "Sign language recognition in Visual Analysis of Humans," *Springer*, doi.org/10.1007/978-0-85729-997-0_27, p. 539–562, 2011.

[34] Ayodeji Olalekan Salau and Shruti Jain, "Feature Extraction: A Survey of the Types, Techniques, Applications," in *Conference: 2019 International Conference on Signal Processing and Communication (ICSC)*, 2019.

[35] Abuzneid, Mohannad A. and Ausif Mahmood, "Enhanced human face recognition using LBPH descriptor, multi-KNN, and back-propagation neural network," *IEEE,* vol. 6, pp. 641-651, 2018.

[36] Wan Azani Mustafa and Mohamed Mydin M. Abdul Kader, "A Review of Histogram Equalization Techniques in Image Enhancement Application," in *1st International Conference on Green and Sustainable Computing (ICoGeS)*, doi :10.1088/1742-6596/1019/1/012026, 2017.

[37] C. M. Bishop, Pattern recognition and machine learning, 2006.

[38] Abimbola Ganiyat AKINTOLA, Abdul-hafiz Taiwo ONIYANGI and Muhammed Besiru JIBRIN, "Comparative Analysis Of Appearance-Based Feature Extraction Techniques In Face Recognition," *University Of Pitesti Scientific Bulletin: Electronics And Computers Science,* vol. 18, 2018.

[39] K. M., "Image feature extraction - an overview," in *Computer as a Tool, 2005. EUROCON 2005.The International Conference*, DOI: 10.1109/EURCON.2005.1629889, 2005.

[40] Li-Fen Chen, Hong-Yuan Mark Liao and Ming-Tat Ko, "New LDA-based face recognition system which can solve the small sample size problem," *Pattern Recognition,* vol. 33, pp. 713-726, 2000.

[41] M. Hassaballah, A. A. Abdelmgeid, and H. A. Alshazly, "Image Features Detection Description and Matching," *Springer,* vol. 630, 2016.

[42] E. Barshan, A. Ghodsi, Z. Azimifar, and M. Z. Jahromi, "Supervised principal component analysis: Visualization, classification and regression on subspaces," *Pattern Recognit.*, vol. 44, pp. 357-371, 2011.

[43] Tharwat A., Gaber T., Ibrahim, A. and Hassanien A.E, "Linear discriminant analysis: A detailed tutorial," *AI Commun,* vol. 30, pp. 169-190, 2017.

[44] P. V. Jones, "Rapid object detection using a boosted cascade of simple features," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2001.

[45] B. S. Celiktutan, "A comparative study of face landmarking techniques," *EURASIP J. Image Video Process*, vol. 13, 2013.

[46] Y. Wang and X. Tang, "Deep learning face representation from predicting 10,000 classes," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 1891–1898.*, 2014.

[47] B. Sun, S. Chen, J. Wang and H. Chen, "A robust multi-class AdaBoost algorithm for mislabelled noisy data," *Knowledge-Based Systems,* vol. 102, pp. 87-102, 2016.

[48] A. Parkhi, Omkar M., Vedaldi, Andrea and Zisserman, "Deep Face Recognition," *Vis. Geom. Group, Dep. Eng. Sci. Univ. Oxford*, 2018.

[49] Stephen Milborrow, John Morkel, and Fred Nicolls, "The MUCT Landmarked Face Database," at the University Of Cape Town., Leslie Social Sciences Building, December 2008.

[50] Paul Viola and Michael Jones, "Robust real-time object detection," *International Journal of Computer Vision*, 2002.

[51] S. N. Dhananjay Rathod, A. Vinay and S. Shylaja, "Facial Landmark Localization – A Literature Survey," *Int. J. Curr. Eng. Technol*, vol. 4, p. 901–907, 2014.

[52] A.I. Shahin and Sultan Almotairi, "Automated Arabic Sign Language Recognition System Based on Deep Transfer Learning," *IJCSNS International Journal of Computer Science and Network Security*, vol. 19, p. 10, 2019.

[53] Ala addin I. Sidig, Hamzah Luqman, and Sabri A. Mahmoud, "Arabic Sign Language Recognition Using Optical Flow-Based Features and HMM," in *Recent Trends in Information and Communication Technology*, doi:10.1007/978-3-319-59427-9_32, 2018.

[54] Batista G. and Monard M.C., "An Analysis of Four Missing Data Treatment Methods for Supervised Learning," *Applied Artificial Intelligence*, vol. 17,doi: 10.1080/713827181, pp. 519-533, 2007.

[55] S. Kotsiantis., "Supervised Machine Learning: A Review of Classification Techniques," *Informatica*, vol. 31,pp. pp 249-268, 2007.

[56] B. S., "Communication with deaf and hard-of-hearing people," *Academic medicine: journal of the Association of American Medical Colleges*, vol. 77,doi: 10.1097/00001888-200207000-00009., pp. 694-700, 2002.

[57] P. K. Pisharady and M. Saerbeck, "Recent methods and databases in vision-based hand gesture recognition," *computer Vis. Image Understand*, vol. 141,doi.org/10.1016/j.cviu.2015.08.004, p. 152–165, Dec. 2015.

[58] Hayani S, Benaddy M, El Meslouhi O, Kardouchi M., "Arab Sign Language Recognition with Convolutional Neural Networks.," in *In2019 International Conference of Computer Science and Renewable Energies (ICCSRE)* , IEEE, 2019 Jul 22.

[59] A. C., "Neural networks and deep learning," *Springer*, vol. 10, p. 973–978, 2018.

[60] Z. Yagang, Types of Machine Learning Algorithms, IntechOpen, 2010.

[61] M. S. ALtememe and N. K. El Abbadi, "Gesture Interpreting of Alphabet Arabic Sign Language Based on Machine Learning algorithms," in *Iraqi International Conference on Communication and Information Technologies (IICCIT)*, Basrah, Iraq, 2022.

[62] Rekha Nagar and Yudhvir Singh, "A literature survey on Machine Learning Algorithms," *International Journal of Emerging Technologies and Innovative Research (www.jetir.org)*, vol. 6, pp. 471-474, April 2019.

[63] X. Pan, Y. Luo and Y. Xu, "K-nearest neighbourbased structural twin support vector machine," *Knowledge-Based Systems*, vol. 88, pp. 34-44, 2015.

[64] Rajat Sharma et al., "Sign Language Gesture Recognition," *International Journal of Recent Research Aspects*, vol. 7, pp. 14-19, 2020.

[65] Roaa AlKhalaf, Rawan AlKhalaf, Ghazanfar Latif, Nazeeruddin Mohammad, Jaafar Alghazo, "Arabic Sign Language (ArSL) recognition system using convolutional neural network (CNN)," in *In 2017 Eighth international conference on intelligent computing and information systems (ICICIS) (pp. 66-71). IEEE*, 2018.

[66] Saleh Yaser and Issa Ghassan, "Arabic sign language recognition through deep neural networks fine-tuning," *International Association of Online Engineering*, vol. 16, p. 5, 2020.

[67] Fariha Raisa Alam, Muhaimin Bin Munir, Shadman Ishrak, Sonaila Hussain, Md. Salahuddin, Muhammad Nazrul Islam,, "A Machine Learning Based Sign Language Interpretation System for Communication with Deaf-mute People," in *21th International Conference on Human Computer Interaction (Interacción 2021)*, At: Málaga, España, 2021.

[68] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke and Andrew Rabinovich, "Going Deeper With Convolutions," in *the IEEE Conference on Computer Vision and Pattern Recognition (CVPR),*, Boston, 2015.

[69] S. Kiranyaz, T. Ince, R. Hamila and M. Gabbouj, "Convolutional Neural Networks for patient-specific ECG classification," *in Proc. Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. EMBS, doi:10.1109/EMBC.2015.7318926.*, 2015.

[70] S. Jansen, "Convolutional Neural Networks," in *Hands-On Machine Learning for Algorithmic Trading: Design and implement*, UK, packt publishing, 2018, p. 496.

[71] Ja Hyung Koo,Se Woon Cho and Na Rae Baek, "CNN-Based Multimodal Human Recognition in Surveillance Environments," *Sensors is the leading international*, vol. 18, doi.org/10.3390/s18093040, p. 9, 2018.

[72] S. Kiranyaz, A. Gastli, L. Ben-Brahim, N. Alemadi and M. Gabbouj, "Real-Time Fault Detection and Identification for MMC using 1D Convolutional Neural Networks,," *IEEE Trans. Ind. Electron.*, doi:10.1109/TIE.2018.2833045,2018.

[73] HOANG and Toan Minh, "Deep retinanet-based detection and classification of road markings by visible light camera sensors," *Sensors,* vol. 19, no. 281, p. 2, 2019.

[74] Krasimira Kapitanova, Sang Hyuk Son, "Machine Learning Basics," *Intelligent Sensor Networks: The Integration of Sensor Networks, Signal Processing and Machine Learning,* no. 9780429066962, p. 12, 2012.

[75] Visa S., Ramsay B., Ralescu A. L. and Van Der Knaap E., "Confusion matrix-based feature selection," *Maics,* vol. 1, pp. 120-127, 2011.

[76] M. S., "Accuracy measures," *theoretical and practical concerns. International journal of forecasting,* vol. 9, pp. 526-529, 1993.

[77] Al-Nuaim and M. A. Almasre and H., "Recognizing Arabic Sign Language gestures using depth sensors and a KSVM classifier," in *2016 8th Computer Science and Electronic Engineering (CEEC)*, Colchester, UK, doi: 10.1109/CEEC.2016.7835904., 2016.

[78] Sharma, Alok and Kuldip K., "Linear discriminant analysis for the small sample size problem: an overview," *International Journal of Machine Learning and Cybernetics,* vol. 3, no. 6, pp. 443-454, 2015.

[79] Bala, Raja, and Karen M. Braun., "Color-to-grayscale conversion to maintain discriminability," *International Society for Optics and Photonics,* vol. 5293, DOI:10.1117/12.532192, pp. 196-202, 2004.

[80] Rahaf Abdulaziz Alawwad, Ouiem Bchir and Mohamed Maher, "Arabic Sign Language Recognition using Faster R-CNN," *International Journal of Advanced Computer Science and Applications,* vol. 12, doi: 10.14569/IJACSA.2021.0120380, 2021.

[81] Mohamed Mohandes, Mohamed Abdelouaheb Deriche and Salihu Oladimeji Aliyu, "ARABC SIGN LANGUAGE RECOGNITION USING MULTI-SENSOR DATA FUSION," *U.S. PATENT DOCUMENTS,* vol. 62, p. 17, 2016.

[82] Salma A. Essam El-Din and Mohamed A. Abd El-Ghany, "Sign Language Interpreter System: An alternative system for machine learning," in *2nd Novel Intelligent and Leading Emerging Sciences Conference (NILES)*, DOI:10.1109/NILES50944.2020.9257958, 2020, pp(332-337).

[83] M. M. Kamruzzaman, "Arabic Sign Language Recognition and Generating Arabic Speech Using Convolutional Neural Network," *Hindawi Wireless Communications and Mobile Computing*, vol. 20, doi.org/10.1155/2020/3685614, p. 9, 2020.

[84] Tolba, Mohamed, Abdel-wahab, Mohamed, Aboul-Ela, Magdy and Samir Roshdy, "Image Signature Improving by PCNN for Arabic Sign Language Recognition," *Canadian Journal on Artificial Intelligence, Machine Learning and Pattern Recognition*, AM publisher Corporation canada, p. 8, 2014.

[85] Omar Al-Jarrah and Alaa Halawani, "Gestures in Arabic sign language using neuro-fuzzy systems," *Artificial Intelligence*, p. 117–138, 2001.

الخلاصة

لغة الإشارة هي لغة بصرية تستخدم مزيجًا من إيماءات اليد وتعبيرات الوجه ولغة الجسد لتوصيل المعنى. وفقًا لتقرير منظمة الصحة العالمية (WHO) ، وفقًا لإحصاءات عام 2021 ، يعاني ما يقرب من 13٪ من سكان العالم من شكل من أشكال فقدان السمع. وهذا يؤكد على أهمية رفع مستوى الوعي حول فقدان السمع ، وتعزيز تدابير الوقاية ، وضمان الوصول إلى خدمات الرعاية الصحية السمعية المناسبة.

يعد التعرف على لغة الإشارة مجالًا معقدًا من البحث يطرح العديد من التحديات ، مثل الحركة السريعة للأيدي والأجساد التي يمكن أن تغير المعنى المقصود للإشارات ، فضلاً عن تداخل العلامات. يعد بناء نظام ترجمة للغة الإشارة العربية يعتمد على التعلم العميق هدفًا طموحًا وقيّمًا. تُظهر طريقة الجمع بين التعلم الآلي والتعلم العميق نتائج واعدة في مجالات مختلفة ، بما في ذلك رؤية الكمبيوتر ومعالجة اللغة الطبيعية ، والتي يمكن الاستفادة منها في التعرف على لغة الإشارة.

تم تنظيم مراحل العمل في الأطروحة الحالية لإنشاء نظام متكامل لتصنيف وتفسير لغة الإشارة العربية. المرحلة الأولى هي عملية جمع البيانات. يجب أن تتضمن العديد من المواقع المختلفة ومجموعة واسعة من العلامات والتعبيرات. يجب تصنيف هذه الصور أو مقاطع الفيديو المجمعة وتصنيفها لكل كلمة أو حرف.

المرحلة الثانية هي المعالجة المسبقة. يتضمن ذلك تقسيم الإطار ، وتقليل الضوضاء ، وتحسين التباين ، وتحويل الصورة الملونة بتدرج الرمادي ، وتوحيد الحجم. المرحلة الثالثة هي مرحلة استخراج الميزة. هذه المرحلة هي الحصول على ميزات مهمة من الصور أو الفيديوهات. يتم ذلك باستخدام خوارزمية التحليل الخطي التمييزي (LDA) وخوارزمية فيولا وجونز ، حيث يتم استخدام LDA لاستخراج ميزات مهمة تتعلق باليد ، بينما يتم استخدام الخوارزمية الثانية لاستخراج ملامح الوجه وتعتبر خوارزميات لتحديد المنطقة المهمة في الصور.

في المرحلة الرابعة ، يتم التركيز على تصميم وبناء نماذج قادرة على تصنيف العلامات أو الحركات. هناك خمسة نماذج رئيسية يمكن استخدامها لهذا الغرض: نموذج LDA-CNN لنموذج Alphabet الذي يجمع بين طريقة LDA وتقنية CNN أحادية البعد لإنتاج دقة وكفاءة ممتازين على حساب الوقت. هذا النموذج هو عملية للتعرف على علامة الحروف. يستخدم نموذج LDA-CNN الخاص بـ Word مزيجًا من خوارزمية LDA لاستخراج الميزات و CNN لتصنيف وتمييز إيماءات كلمات لغة الإشارة بناءً على الصور الديناميكية. يُستخدم LDA-ML الخاص بنموذج الأبجدية والكلمة لتصنيف الأبجدية والكلمات في لغة الإشارة. تم تنفيذ نموذج Face-CNN لاستخراج وتحديد الوجوه. أخيرًا ، يجمع Hybrid CNN-ML Word Model ، وهو النظام النهائي للأطروحة ، بين نموذج اكتشاف الوجه والدقة العالية للنموذج الأول والسرعة العالية للنموذج الثاني لإجراء مكالمة نهائية حول كيفية التعرف على الحركات الفريدة للغة الإشارة العربية وتصنيفها. .

المرحلة الخامسة هي التدريب والتقييم. عادةً ما يتضمن تدريب نماذج التعلم العميق كمية كبيرة من مجموعة البيانات المصنفة. تنقسم مجموعة البيانات الخاصة إلى مجموعات التدريب والتحقق والاختبار. استخدام وظائف الخسارة المناسبة وتقنيات التحسين هو تدريب النموذج. نموذج الأبجدية LDA-CNN معدل دقة يبلغ 99.9٪ في تصنيف الحروف. وبالمثل ، فإن نموذج LDA-CNN Word يحقق دقة تصل إلى 98.9٪ في تصنيف الكلمات. من ناحية أخرى ، يتفوق نموذج Hybrid CNN-ML على كليهما بمعدل دقة أعلى يصل إلى 99.9٪ ، وكل ذلك مع الحفاظ على سرعة المعالجة الفعالة.

تكمن أهمية هذا العمل في لغة الإشارة العربية في قدرتها على تحسين إمكانية الوصول والدمج والتواصل للأفراد الصم وضعاف السمع. من خلال تطوير تقنية التعرف على لغة الإشارة ، فإنها تساهم في كسر حواجز الاتصال وتعزيز المشاركة المتساوية في مختلف جوانب الحياة.

جمهورية العراق
وزارة التعليم العالي والبحث العلمي
جامعة بابل
كلية تكنولوجيا المعلومات
قسم البرمجيات

تصنيف لغة الإشارة العربية على أساس تقنيات التعلم الآلى

أطروحة مقدم الى

مجلس كلية تكنولوجيا المعلومات ـجامعة بابل كجزء لمتطلبات نيل

درجة الدكتوراه في تكنولوجيا المعلومات ـ برمجيات

من قبل

مها صبري زباله شاكر التميمي

بأشراف

أ.د. نضال خضير العبادي

1444 هـ

2023م