Sign Language Recognition Application for Deaf-Mute People

By

GC Aaimal Khan
GC Usman Arshad
GC Irtaza Aftab
GC Ibrahim Inam

Supervised by: Assistant Prof. Dr. Nauman Ali

In Partial Fulfillment Of the Requirements for the degree Bachelors of Engineering in Software Engineering (BESE)

Military College of Signals, National University of Sciences and Technology Islamabad, Pakistan (2023)

CERTIFICATE OF CORRECTNESS AND APPROVAL

This is to officially state that the thesis work contained in this report
"Sign Language Recognition Application for Deaf-Mute People"
is carried out by:

GC Aaimal Khan

GC Usman Arshad

GC Irtaza Aftab

GC Ibrahim Inam

under my supervision and that in my judgment, it is fully ample, in scope and excellence, for thedegree of Bachelor of Software Engineering at Military College of Signals, National University of Sciences and Technology (NUST), Islamabad.

Supervisor Name: **Assistant Prof Dr. Nouman Ali**

Supervisor Signature:_____

Date: _____

Place: _____

DECLARATION OF ORIGINALITY

We hereby declare that no portion of the work presented in this Thesis has been submitted in support of another award or qualification in either this institute or anywhere else.

ACKNOWLEDGEMENTS

Allah Subhan'Wa'Tala is the sole guidance in all domains.

Our parents, colleagues, and most of all our supervisor,
Assistant Prof. Dr Nauman Ali without their guidance this wouldn't be possible.

And all the group members, who through all adversities worked steadfastly.

PLAGIARISM CERTIFICATE (Turnitin Report)

This thesis has_____similarity index. Turnitin report endorsed by Supervisor is attached.

GC Aaimal Khan

0000325419

GC Usman Arshad

0000325056

GC Irtaza Aftab

0000325058

GC Ibrahim Inam

0000325050

Signature of Supervisor

ABSTRACT

Many people in our society are mute and deaf which have a problem in communicating with the general public. Our undertaking point is to tackle is the correspondence issue between discourse hindered individuals and the others. As those individuals can't communicate their thoughts with the words, they have numerous hardships during their everyday existence. Since practically each of the ordinary individuals don't know communication through signing and can't comprehend what puzzled individuals mean by their exceptional language, undertakings, for example, shopping, settling issues at an administration office are challenging to such an extent that discourse disabled individuals can't deal with by their own. This issue is extremely expansive and numerous arrangements and execution can be raised. An answer could be showing communication via gestures to everybody, yet being a wasteful and, surprisingly, non-relevant one is impossible to miss. Since dumbfounded individuals can comprehend others by lip-perusing (or in any event, hearing since being stunned doesn't mean being hard of hearing likewise) the fundamental issue is that typical individuals don't grasp them. Subsequently, a more reasonable arrangement ought to be in the way that makes puzzled individuals' language justifiable by the others. For the language of dumbfounded individuals to be perceived by others, we want the signals performed by discourse disabled individuals to be perceived and transformed into a structure that the others can comprehend. The deaf-mute community moves back When it comes to the interactive part with people with special needs, this communication gap has exacerbated. Because our community have no idea about sign language. ‘زبا ِن دست’ is an application that translates text to sign language by using stored sign videos that will perform the movement of the translated word & translate sign language to text using mobile camera to get the picture of sign gesture language. Thus, this application will smartly work to help our community and society.

Table of Contents

Turnitin Originality Report

Processed on: 27-Apr-2023 4:11 PM EDT
ID: 2077277933
Word Count: 8276
Submitted: 2

Zaban e Dast By Bilal Janjua

Similarity Index
11%

Similarity by Source
Internet Sources: 5%
Publications: 1%
Student Papers: 9%

3% match (student papers from 06-May-2022)
Submitted to University of Bradford on 2022-05-06

2% match (student papers from 30-May-2022)
Submitted to Higher Education Commission Pakistan on 2022-05-30

1% match (student papers from 10-May-2022)
Submitted to Sardar Vallabhbhai National Inst. of Tech.Surat on 2022-05-10

1% match (student papers from 06-Nov-2022)
Submitted to Charotar University of Science And Technology on 2022-11-06

< 1% match (Internet from 21-Aug-2022)
https://www.researchgate.net/publication/329120510_Deep_learning-based_feature_engineering_for_stock_price_movement_prediction

< 1% match (Internet from 18-Apr-2023)
https://www.researchgate.net/publication/370069919_M-AResNet_a_novel_multi-

LIST OF FIGURES

Chapter 1

INTRODUCTION

According to the WHO, Millions of people in every country, accounting for about 5% of the population, have partially or completely impaired hearing, according to Yesprograms.org (2018). Learning standard Pakistan Sign Language would greatly aid in integrating them into everyday life. However, the challenge with learning sign language is that it involves over 2500 hand gestures, making it difficult for an interpreter or someone who has become deaf later in life. Despite the large number of people with hearing difficulties, there are only a few people who can effectively use sign language.

To address this challenge, the idea for an application solution emerged. By utilizing libraries and smart gesture recognition mechanisms, we can facilitate communication between the deaf and other individuals.

This project aims to provide a starting point for the solution by training an application to recognize and translate a selection of words using a small portable device equipped with a camera. I will also discuss the current issues and challenges faced in creating a sign language interpretation tool.

1.1 Overview

The growing acceptance and funding for international projects highlight the importance of sign language, particularly in today's technology-driven age. Despite researchers having worked on the problem for some time, there is currently no commercial solution for sign recognition available on the market, despite the advancements in voice recognition technology. The ultimate goal is to create a user-friendly human-computer interface (HCI) that can understand human language and gestures, including speech, facial emotions, and human gestures. Gesture recognition is a crucial aspect of human-computer interaction and involves analyzing nonverbal information exchanged through gestures, such as hand, arm, and head motions. Using hand gestures, individuals can convey more information in a shorter amount of time, making it an important mode of communication

for the visually impaired. Several approaches have been explored to apply computer-vision ideas to the real-time processing of gesture outputs, including the use of algorithms and object recognition systems to identify gestures and overcome the limitations of earlier systems.

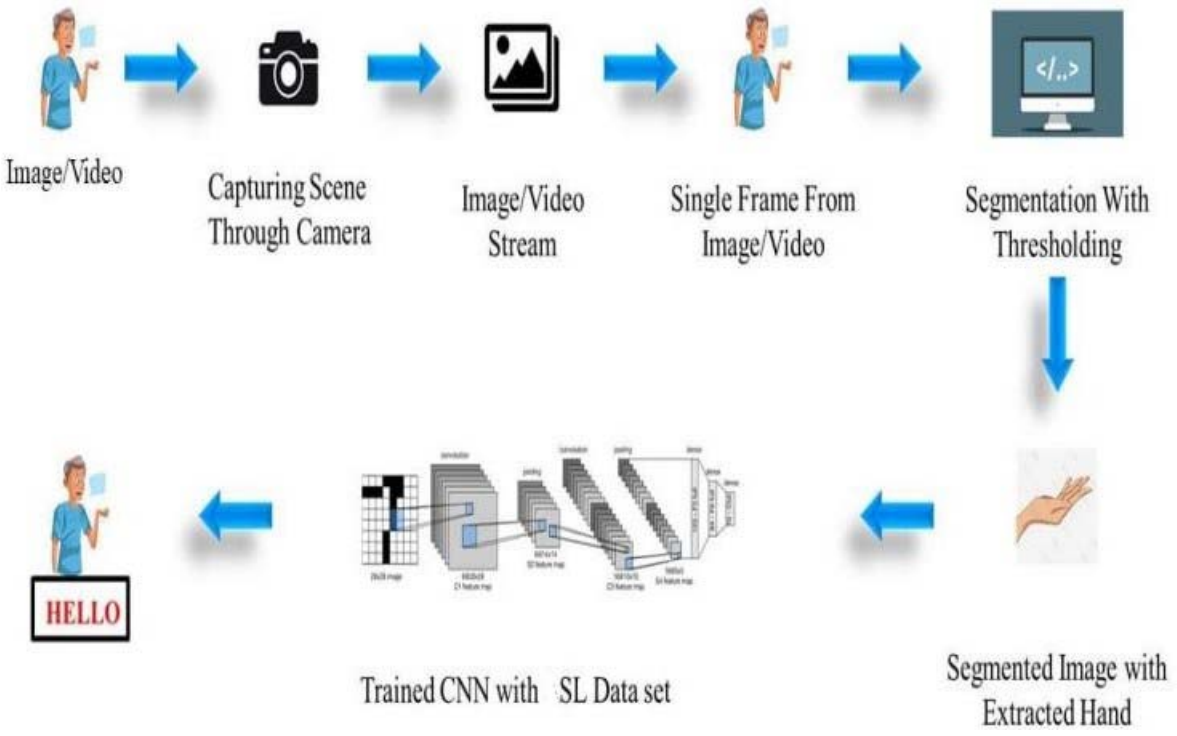


Figure 1: System model for Sign Lang Recognition System

1.2 Challenges and Novelty

Although the idea of using machine learning to interpret sign language is not new, there is currently no project that interprets Pakistan Sign Language. Additionally, since there are over 300 sign languages worldwide, each with unique gestures, a separate model would need to be trained for each language. The most extensive project for sign language interpretation is a collaboration between Google and Sign All. This project offers a software development kit (SDK) that enables developers to integrate American Sign Language (ASL) interpretation into their projects. The model used for this SDK is trained on over 300,000 sign language videos. However, the main issue

with creating a sign language interpretation tool is that a large number of videos are required to train the model and avoid overfitting, which occurs when the model becomes too attuned to the training data. Ideally, 20 to 50 different videos would be needed for each sign taught to the model, but publicly available videos are usually limited to 4 or 5, which poses a significant challenge.

1.3 Proposed Solution

Several effective methods for gesture recognition have been developed and have been shown to work well. In a video demonstration, a hand gesture detection system was utilized to operate a robotic arm. The algorithms used in this system include Neural Networks, Adaptive Boosting, and Support Vector Machine. The convex hull technique was employed to improve fingertip detection. The paper reports that this system achieved higher accuracy than other existing systems. The goal of this research is to showcase widely effective approaches to capturing gestures, which have been crucial in recent times. The paper also outlines how to identify skin color for shape identification using the convex hull method and color space transformation. To obtain accurate results, ten users provided 330 samples of various hand movements. Additionally, the paper describes a Linux-based hand motion recognition system developed using Python. This algorithm is not dependent on the context and can recognize the number of fingertips and the task performed as required. The significant number of deaf-mute people provided the motivation for this project. Hence, the need for a system that could enable these speech-impaired and deaf persons to communicate with regular people. Since not everyone can understand sign language, our project aims to convert sign language motions into text that can be understood by ordinary people. Resultantly, this will effectively help the deaf-mute people to overcome their specialty and live a healthier and sounder life.

1.4 Scope

The scope of this Sign Language Recognition (SLR) application for deaf-mute individuals would typically include many factors. Starting from gesture recognition, the

application would aim to recognize sign language gestures accurately and in real-time. This could involve capturing and processing video or sensor data from devices such as cameras or gloves and applying computer vision or machine learning techniques to identify and interpret the gestures made by deaf-mute individuals. Sign language dictionary will then be used for covering a wide range of sign language gestures and their corresponding meanings. This could be used as a reference for users to learn new signs, as well as forthe application to recognize and interpret the signs made by deaf-mute individuals. The application should have a user-friendly interface designed to be accessible and usable by deaf-mute individuals. This could involve visual elements such as large icons or buttons, clear instructions, and intuitive navigation, as well as potential haptic or audio feedback for a more inclusive user experience. The application could be designed to run on different platforms or devices, such as desktop computers, laptops, tablets, or smartphones, to ensure accessibility and usability for a wide range of users. This could involve developing the application using technologies such as web or mobile applications, and optimizing it for different devices and operating systems. The application should be thoroughly validated and tested to ensure its accuracy, reliability, and usability. This could involve testing with a diverse group of deaf-mute individuals to assess the application's performance in real-world scenarios and gathering feedback for improvements.

1.5 Deliverables

Software Requirement Specification:

The objective of this manuscript is to provide an elaborate account of Sign Language Recognition for individuals who are deaf-mute. It will outline the objectives and characteristics of the system, its interfaces, functionality, the entire operational process, the limitations within which it must operate, and its response to external influences. The intended audience for this document includes both the stakeholders and users.

Software Architecture Document:

In this document, the overall architecture of the system is discussed,

including the introduction of various components and subsystems. It is mainly supported by system Architecture diagram which shows an insider's perspective of the system by describing thehigh-level software components that perform the major functions to make the system operational.

Software Design Document:

The design document captures all our functional requirements and shows how they interactwith each other conceptually. The low-level design also shows as to how we have been implementing how we are going to implement all of these requirements.

Implementation Code Document:

The implementation code document provides details about the pseudo code for the application and project prototype.

Software Testing Document:

This document has testing modules in which there are certain test cases which depicts the correctness and accuracy of the project.

Final Project Report

This is the thesis report which compiles all the previous and current working for the project.Thesis report provides the whole summary for the project and also give details about each and every aspect of the project starting from the introduction of the project, literature review, requirements leading to design discussions then testing and lastly future work and conclusion.

1.6 Relevant Sustainable Development Goals

The project of Sign Language Recognition (SLR) could be related to several Sustainable Development Goals (SDGs) outlined by the United Nations. Some of the relevant SDGs that could be associated with an SLR project include:

SDG 3: Good Health and Well-being - SLR technology could contribute to improving the health and well-being of deaf-mute individuals by facilitating communication and reducing communication barriers, which can lead to better mental health, social inclusion, and access to healthcare services.

SDG 4: Quality Education - SLR could enhance access to quality education for deaf-mute individuals by providing them with tools to communicate effectively in educational settings, facilitating their participation in learning activities, and promoting inclusive education.

SDG 10: Reduced Inequalities - SLR could help reduce inequalities faced by deaf-mute individuals by enabling them to communicate more effectively with the wider community, including hearing individuals, and thereby promoting social inclusion and reducing communication disparities.

SDG 16: Peace, Justice, and Strong Institutions - SLR could contribute to building inclusive societies and promoting just and strong institutions by ensuring that deaf-mute individuals have equal opportunities for communication, participation, and access to information, which are fundamental aspects of their human rights.

SDG 17: Partnerships for the Goals - An SLR project could involve collaboration and partnerships among various stakeholders, such as researchers, sign language experts, software developers, deaf-mute individuals, and advocacy groups, to work together towards achieving the sustainable development goals related to SLR.

1.7 Structure of Thesis

In summary, the thesis is distributed as follows:

Chapter 2: Literature Review analyzes the popular existing SE Schemes. Their

comparisons and architectural models already existing relevant to the project's domain.

Chapter 3: Proposed Scheme presents the risks to our system and the scheme derived to tackle. It also presents the system specifications and working algorithm of our project.

Chapter 4: The solution's methods, approaches, tools, techniques, algorithms, and other aspects are adequately explained with comprehensive details and supporting illustrations. It also discusses our target market, Operating environment and some constraints, assumptions and dependencies.

Chapter 5: System Design presents system architectural design and other relevant UML diagrams to explain the design of the system.

Chapter 6: System Implementation discusses how the system is actually implemented. It includes pseudo code that presents the working of APIs, Demo outputs of APIs and User interface of the application and dashboard.

Chapter 7: A cognitive evaluation of the system using the test results. System testing is a viable and relatable testing strategy to cover all the use cases of application.

Chapter 8: Future Direction disuses the possible directions available to grow زبا ِنِ دست.

Chapter 9: The document includes a complete and properly formatted list of references.

Chapter 2

LITERATURE REVIEW

A new product is launched by enhancing and modifying the features and capabilities of previously launched similar products. Literature review is a crucial step for development and transformation of the ideainto a new product. Likewise, a detailed study of existing tools and applications relevant to the our SLR is mandatory. Hence, the literature review for sign recognition consists of following components:

2.1 Deep Learning

Deep learning is a machine learning technique inspired by the human brain, which employs intricate neural networks with multiple layers to facilitate the transfer of data between nodes. In our interim report, we had proposed the use of convolutional neural networks, a neural network model specifically designed for image processing. The use of deep learning, specifically neural networks, has shown promising results in sign language conversion applications. Neural networks have the ability to process complex and sequential data, such as sign language gestures, and learn patterns from large datasets, which can be beneficial for accurate sign language recognition and translation. Several studies have utilized deep learning techniques, such as Recurrent Neural Networks (RNNs), Long Short-Term Memory (LSTM) networks and, Convolutional Neural Networks (CNNs), in combination with the TensorFlow library, to develop sign language conversion applications with high accuracy and real-time performance.

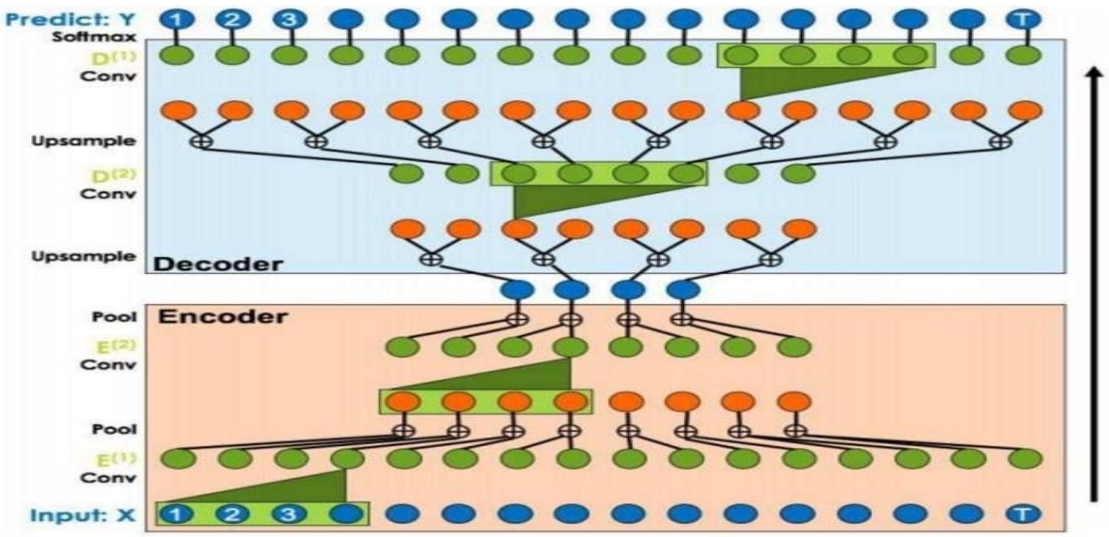


Figure 2: Encoder/Decoder framework of a TCN model

The use of deep learning, specifically neural networks, has shown promising results in sign language conversion applications. Neural networks have the ability to process complex and sequential data, such as sign language gestures, and learn patterns from large datasets, which can be beneficial for accurate sign language recognition and translation. Several studies have utilized deep learning techniques, various types of convolution networks in combination with the TensorFlow library, to develop sign language conversion applications with high accuracy and real-time performance (Camgoz et al., 2019; Pu et al., 2017; Starner et al., 2019).

We will be developing this project as per the Pakistan Sing Language and equivalent translation of words will be given in Urdu. In this project, we utilized Keras, a Python library for carrying out all the deep learning tasks. Keras was developed to streamline the most frequently used tasks in machine learning and is accompanied by extensive documentation, guidelines, and pre-existing projects that can aid in learning how to use the library.

2.2 Sign Language in Pakistan

As per the definition provided by Oxford Dictionary, a sign language is a means of communication, which involves visual gestures and signs and is primarily used by deaf

people. Although these sign languages can be translated into spoken languages, they typically adhere to distinct rules. For instance, in spoken English, a rhyme is characterized by the similarity of sound between two words, whereas in sign language, a rhyme occurs when two hand gestures are similar. Consequently, acquiring sign language as a second language can be extremely challenging. One of the key challenges in developing sign language conversion applications for PSL is the variability and complexity of PSL gestures. PSL has its own unique grammar, syntax, and regional variations, making it challenging to accurately interpret and translate PSL gestures into simple language. However, deep learning techniques have shown potential in overcoming these challenges. Akhtar et al. (2019) developed a sign language recognition system for PSL using CNNs and TensorFlow that achieved high accuracy in recognizing PSL gestures, even with variations in hand shapes, orientations, and motion dynamics. Similarly, Hussain et al. (2020) utilized LSTM networks in combination with TensorFlow to develop a sign language translation system that could accurately translate PSL gestures into spoken Urdu, the national language of Pakistan, in real-time, despite the complex and dynamic nature of PSL gestures.

2.3 Sign Language Interpretation with Deep Learning

Real-time performance is a critical aspect of sign language conversion applications for PSL, as these applications need to provide timely translations during live interactions. Deep learning techniques, particularly with the use of TensorFlow, have shown promising results in achieving real-time performance in sign language conversion applications for PSL. Iqbal et al. (2021) developed a sign language translation system using RNNs and TensorFlow that could achieve real-time performance on a mobile device, allowing for instantaneous sign language translation during conversations in PSL. Hussain et al. (2020) also demonstrated real-time translation of PSL gestures into spoken Urdu using LSTM networks and TensorFlow, with low latency and high translation accuracy.

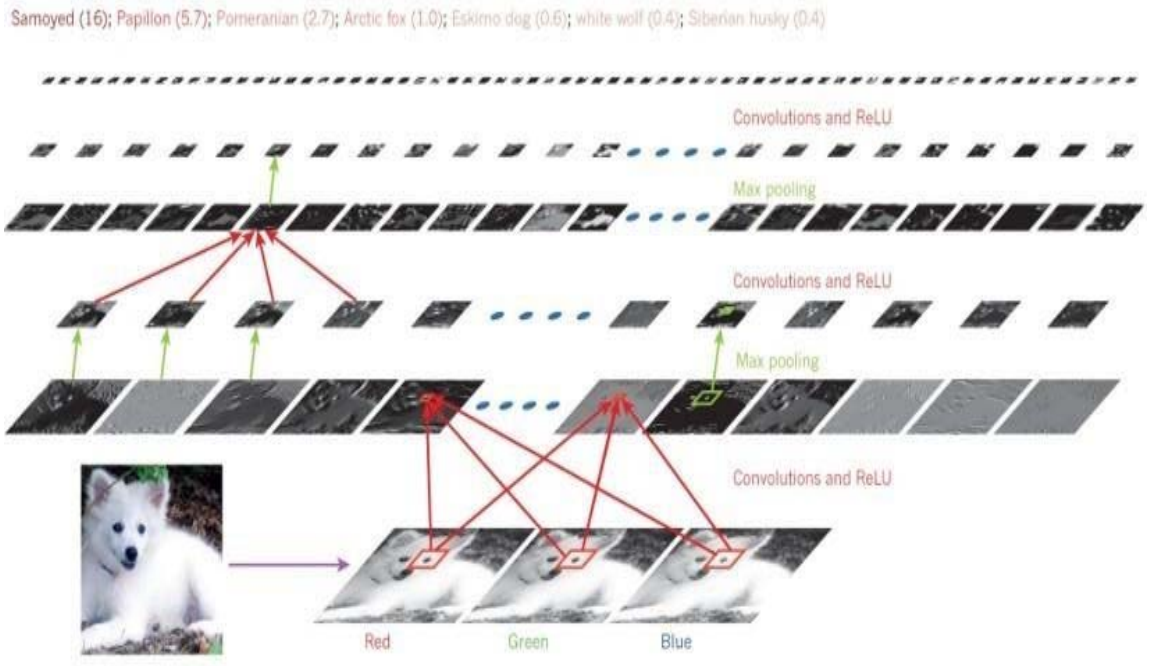


Figure 3: Inside a Convolutional Network

2.4 Limitations and Areas of Improvement

However, there are still limitations and areas for improvement in sign language conversion applications for PSL using deep learning and TensorFlow. One limitation is the need for large datasets for training deep learning models, particularly for PSL with limited data availability. Collecting and annotating PSL datasets can be challenging due to the limited resources and standardization of PSL. Furthermore, the accuracy of sign language conversion applications can be affected by environmental factors, such as lighting conditions and background clutter, which can impact the performance of the deep learning models. Additionally, the usability and accessibility of sign language conversion applications for PSL need to be considered, including factors such as user interface design, ease of use, and adaptability to different PSL variations and user preferences.

In recent years, the field of sign language recognition has made significant progress, paving the way for future applications aimed at promoting the inclusion of deaf individuals

in a hearing society. For instance, a translation system could aid communication between deaf and hearing individuals in public, and automatic indexing of signed videos or user interfaces are other possible applications. Although current recognition systems still operate under sign language-dependent conditions in laboratory settings, the achievements thus far form a foundation for future endeavors. The current state of sign language recognition research lags behind that of speech recognition by about 30 years, reflecting a gradual transition from isolated to continuous recognition for small vocabulary tasks. Research efforts are mainly focused on robust feature extraction and statistical modeling of characters.

2.5 Existing solutions and their drawbacks

Despite the advancements in deep learning techniques and the use of TensorFlow for sign language conversion applications for PSL, there are several drawbacks in the existing solutions that need to be addressed in future research.

Limited availability of PSL datasets: One of the major limitations in developing sign language conversion applications for PSL is the availability of large and diverse datasets for training deep learning models. PSL has limited standardized datasets available, which can hinder the accuracy and generalization of the models.

Environmental factors affecting accuracy: The accuracy of sign language conversion applications can be impacted by environmental factors, such as lighting conditions, background clutter, and hand occlusions. These factors can introduce noise and affect the performance of the deep learning models in real-world scenarios.

Lack of real-time performance: While some existing solutions claim real-time performance, there is still room for improvement in terms of reducing latency and ensuring smooth and instantaneous translation of PSL gestures into simple language. Achieving real-time performance in sign language conversion applications is crucial for enabling smooth communication in real-life interactions.

Usability and accessibility challenges: The usability and accessibility of sign language conversion applications for PSL need to be considered to ensure inclusivity. User-friendly interfaces, intuitive interactions, and accessibility features, such as support for different input modes and output formats, need to be incorporated to make the applications usable and accessible for a wide range of users, including individuals with varying degrees of hearing impairment.

Generalization to different PSL variations: PSL has regional variations in terms of grammar, syntax, and vocabulary, which can pose challenges in developing sign language conversion applications that can accurately translate PSL gestures from different regions. Ensuring the generalization and adaptability of the models to different PSL variations is essential for achieving accurate translations in diverse contexts.

Ethical considerations: Ethical considerations, such as privacy, consent, and fairness, need to be addressed in the development and deployment of sign language conversion applications for PSL. Ensuring the protection of user data, obtaining informed consent, and mitigating biases in the translation process are important ethical considerations that need to be carefully addressed in future research.

Chapter 3

PROBLEM DEFINITION

Our application's basic working is the Text-to-Sign and Sign-to-Text conversion of the Pakistan Sign Language. Purposefully, the application will do so by using stored sign videos that will deduct the movement into its parallel word and vice versa. The translating module of the application uses its camera component to capture the picture of hands motion and convert it into text.

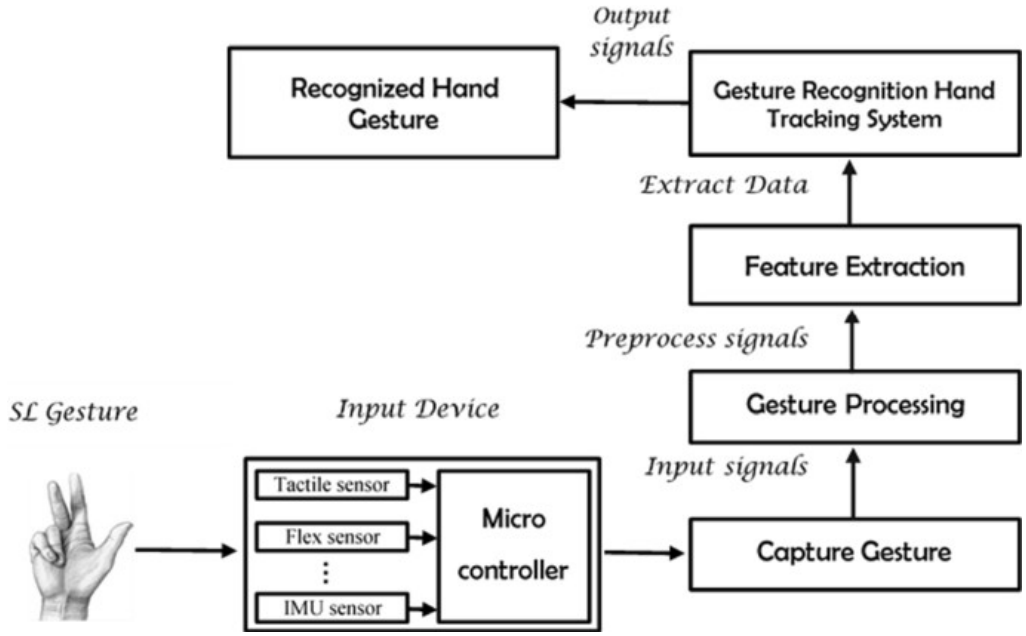


Figure 4: Comprehension of Sign language Recognition and Conversion

3.1 Working Elements and Techniques

The following components and techniques will feature the application towards its goal:

3.1.1 Machine Learning Model

Our machine learning model will use algorithms to meet our application's needs. Our model will be trained with compatible data to allow us the use of deep learning via TensorFlow library to detect any errors.

3.1.2 Object Detection API

TensorFlow library will be used as object detection API for the conversion of sign language into text. It computes the object from picture and matches it using database data. As for a correct match, the API provides the translation of sign in textual form.
Similarly, the API database has stored videos of sign language for static words and phrases. The system will provide the requisite translation of the entered word in the form of sign video.

3.1.3 Database Engine

It is a basic software component that is used by database management systems to search, read, and retrieve data from a database. This component is most frequently used with database server and interchangeably in our project. We will be depending on NOSQL database engine.

3.1.4 Database Server

A computer program that offers database services to other computer programs or terminals using the client-server model is known as a database server. Query language forms the basis of the majority of database servers.

3.1.5 User Interface

The goal of user interface design is to make it easy, efficient, and enjoyable for the users while working with the application. In this project, we have used HTML, CSS, and JavaScript to design the layout of our android application's interface.

3.1.6 Deep Learning

It is a machine learning approach and it depends on neural networks that produce a very good accuracy. These deep neural networks require huge dataset to learn and find a pattern between data. We use TensorFlow library to make detection on the sign language for the alternative translation between sign and their meaning.

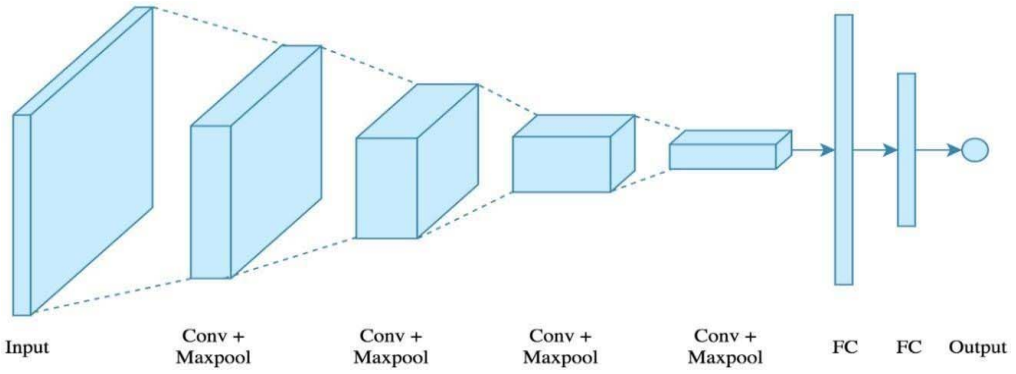


Figure 5: Convolution Neural Networks

3.1.7 Programming Languages

A programming language is designed to provide the medium of communication between user instructions and a machine. Therefore, programming language is to control the behavior of a machine, express and implement algorithms, and to create applications. Following programming languages are utilized in our project for different aspects:

- The algorithms of machine learning will execute using android programming language.
- The interaction between user interface and the system will also be controlled by android programming language.
- Using NOSQL language to interact with database.
- Using Python in model.
- Using JAVA in android.

Chapter 4

METHODOLOGY

4.1 Project Steps

i. Study and analysis the project idea from all aspects.
ii. Search and choose the best algorithm that applies our project idea.
iii. Modeling and simulating our system.
iv. Solve problems and errors in the system.
v. Deploy and implement the system in real world.
vi. Test the accuracy of the app and machine-learning model 7. Evolution the system in the future.

4.2 Working Principle

It is worth noting that sign language recognition is a complex and evolving field, and the working principle of a SLR application may vary depending on the specific approach, techniques, and technologies used. The accuracy and performance of a SLR system also depend on factors such as the quality of the video data, the size and diversity of the training dataset, the choice of recognition algorithms, and the user interface design. Regular updates and improvements to the system may be necessary to enhance its accuracy and usability.

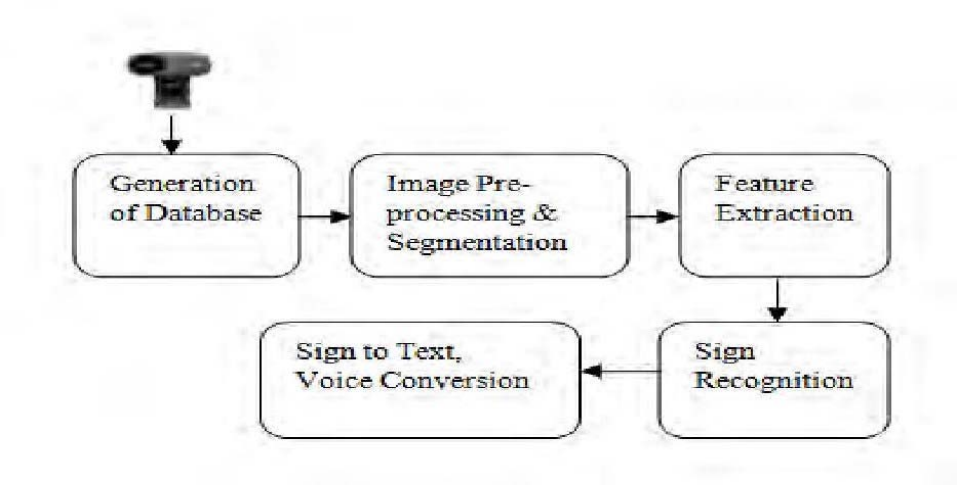


Figure 6: Proposed system of Sign Recognition

The working principle of a Sign Language Recognition (SLR) application involves several key steps that enable the recognition of sign language gestures and their conversion into corresponding text or speech output. Here is a detailed overview of the typical working principle for a SLR application:

4.2.1 Data Acquisition:

The SLR application captures video data of sign language gestures using a camera. The video data is typically in the form of image frames or depth maps, capturing the movements of the signer's hands, face, and body as they perform sign language gestures.

4.2.2 Preprocessing:

The acquired video data is preprocessed to enhance its quality and reduce noise. This may involve operations such as resizing the images to a consistent resolution, removing the background to isolate the signer's hand and face, and

applying noise reduction techniques to improve the accuracy of the subsequent recognition steps. Python and Google's open-source library TensorFlow is utilized for these preprocessing tasks.

4.2.3 Feature Extraction:

Relevant features or characteristics of the sign language gestures are extracted from the preprocessed video data. This step involves identifying key points, landmarks, or regions of interest (ROI) in the signer's hands, fingers, and facial expressions. For example, hand-tracking algorithms, trained on deep learning model are to be used to detect the positions and orientations of the signer's hands and fingers. Neural networks use machine learning and libraries for extracting features to successfully deduct the meaning for signs.

4.2.4 Sign Language Gesture Recognition:

Once the features are extracted, they are used to recognize the sign language gestures. This involves employing machine learning algorithms that are trained on large datasets of sign language gestures. Deep neural networks (DNNs), convolutional neural networks (CNNs), or recurrent neural networks (RNNs) are commonly used for this purpose. The training dataset consists of annotated sign language gesture data, where each gesture is associated with its corresponding textual or spoken representation.

4.2.5 Sign-to-Text Conversion:

After the sign language gestures are recognized, the next step is to convert them into text or speech output. This can be done using natural language processing (NLP) techniques that map the recognized gestures to their

corresponding textual or spoken representations. For example, a lookup table ora dictionary can be used to map recognized gestures to their corresponding text or speech output.

4.2.6 Output Presentation:

The recognized sign language gestures and their corresponding text or speech output can be presented in a user-friendly manner. This can involve displaying the recognized sign language gestures in a graphical format, such as overlaying the gestures on the original video frames, or presenting the text or speech output through a visual or auditory interface, depending on the application's requirements and intended users. Effective GUI will be developed to generate visual interface for displaying the output to the users.

4.2.7 Feedback and Refinement:

SLR applications may also include feedback mechanisms that allow users to provide input on the accuracy and performance of the system. This feedback can be used to refine the SLR system and improve its accuracy over time. For example, the system can be updated with new sign language gestures or additional training data to enhance its recognition capabilities.

It's worth noting that SLR is a complex and evolving field, and the methodology of a SLR application may vary depending on the specific approach, techniques, and technologies used. The accuracy and performance of a SLR system also depend on factors such as the quality of the video data, the size and diversity of the training dataset, the choice of recognition algorithms, and the user interface design. Regular updates and improvements to the system may be necessary to enhance its accuracy and usability.

Chapter 5

DETAILED DESIGN AND ARCHITECTURE

This section will provide a design detail of our application including high level system design and UML diagrams depicting the system processes. زبا نِ دست will follow client server architecture model. Where user is the client side and application will be the server as per the system model.

5.1 Architectural Design

The application will use basic device camera to recognize hand gestures and convert it into text for people who do not understand sign language. If we write word/sentence, the application will extract the keywords and show relevant hand gesture images from the dataset for the deaf people.

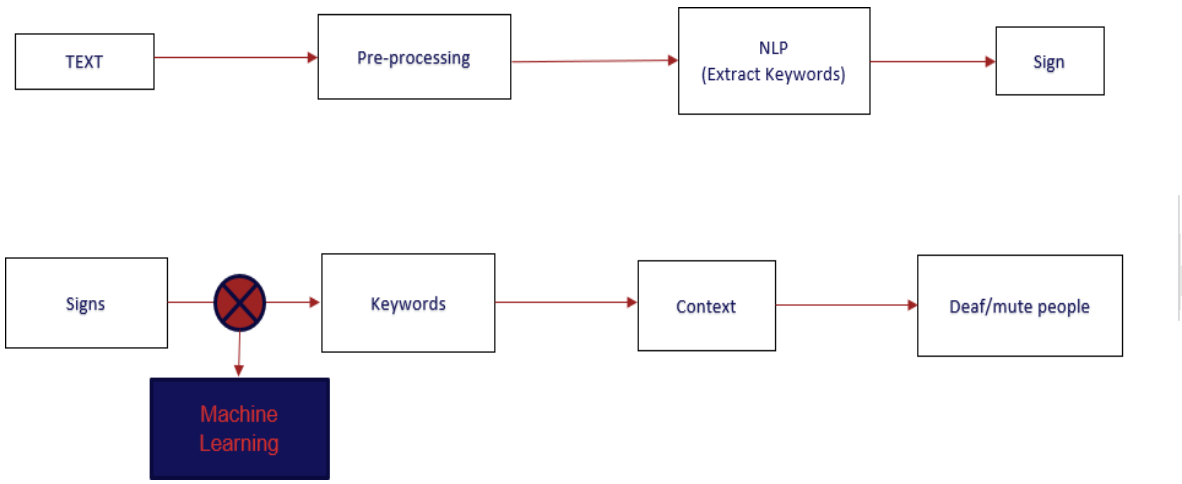


Figure 7: Architecture Diagram

5.2 Component Design

This section will take a closer look at each component of Sign Language Application for Deaf/Mute People in a more systematic way through a component diagram. The component diagram shows three major components; the client machine, the API server and cloud server which shows several subcomponents.

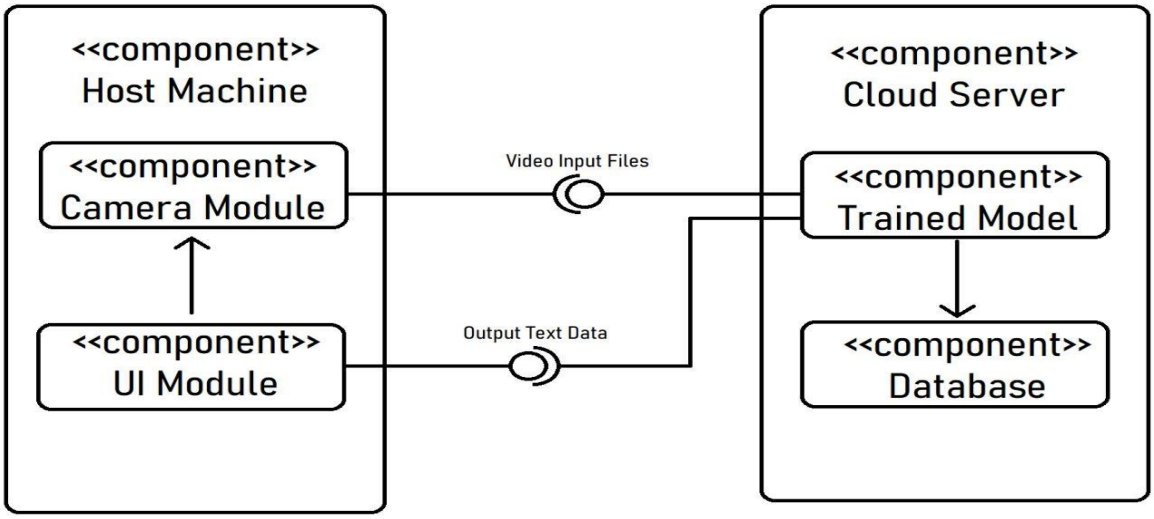


Figure 8: Component Diagram

5.3 Decomposition Description

The decomposition of the subsystems shown in the architectural design is explained as per the module and process decomposition:

5.3.1 Module Decomposition – Class Diagram

Class diagram below shows the Module Decomposition of our sign language conversion application.

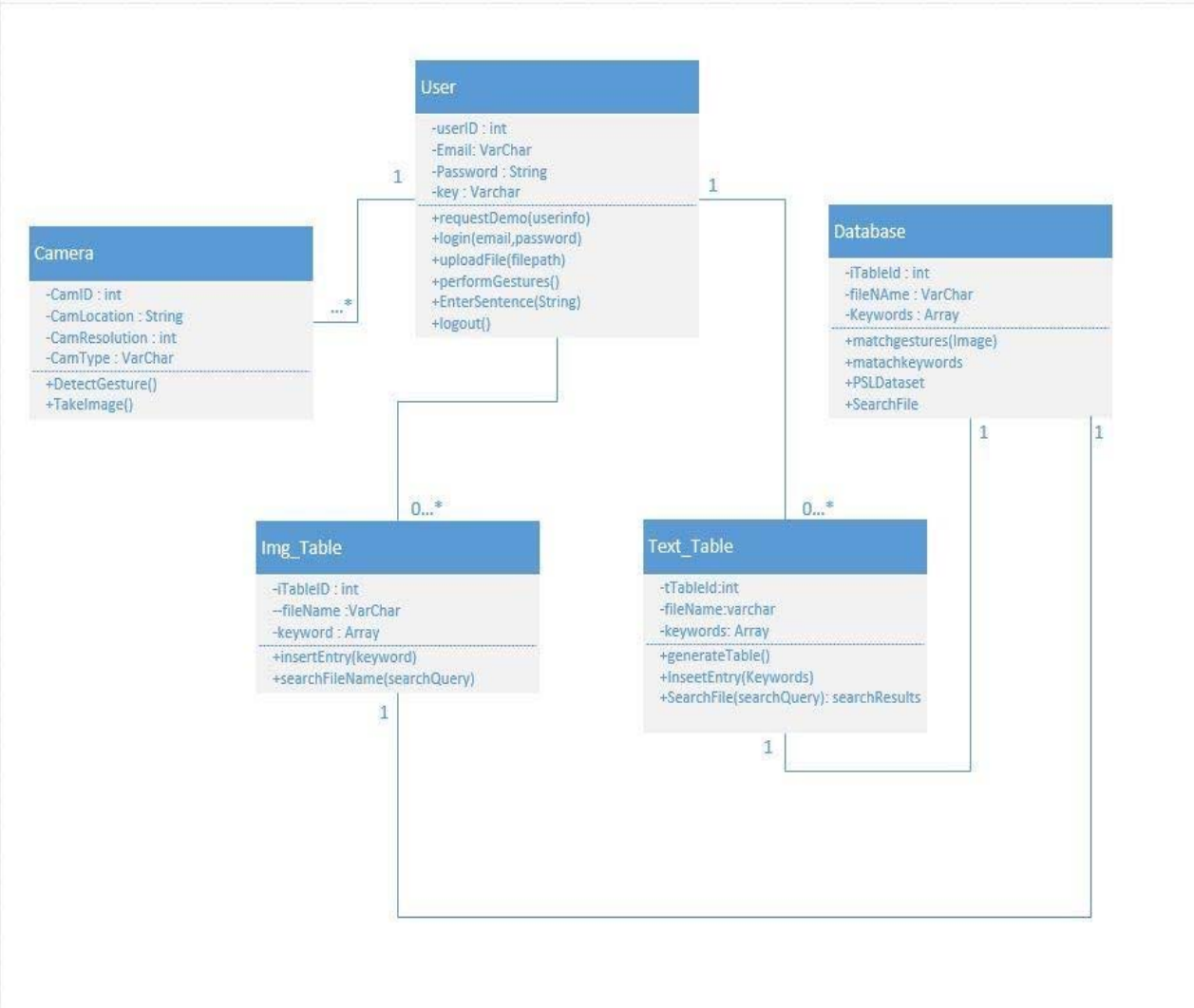


Figure 9: Class Diagram

As shown in the above figure, Sign Language Application for Deaf/Mute People has 5 classes:

User Class: It has attributes of registered users. Unregistered can request for registration. Registered user can log in, upload a file, and download a file. They can perform all the user functionalities of SLA.

Database: It has the dataset from which it will extract all the keywords and images of the

sign language will be extracted and shown.

Image-Table: It has Image file names with their respective keywords to which the gestured images will be shown.

Text-Table: It has Text file names with their respective keywords. Inverted index table for images can be generated, and it is used for searching a query returning search results.

Camera: The camera will capture the hand gestures in which the hand gestures will be matched with the database.

Although Image-Table and Text-Table have same attributes and functions (except id), they have different classes because the keyword generation and searchable keywords are different for both types. Thus, keywords array size and functions implementation are different in both tables respectively.

5.3.2 Process Decomposition – Use Case Diagram

The process decomposition is explained through use case, sequence and activity diagrams which decompose the system into well-defined and cohesive processes. The use cases and the subsequent use case narratives explain the set of actions that a user undertakes while dealing with SLA.

The following figure explains the use cases and subsequent narratives as per the actors performing actions while interacting with the application.

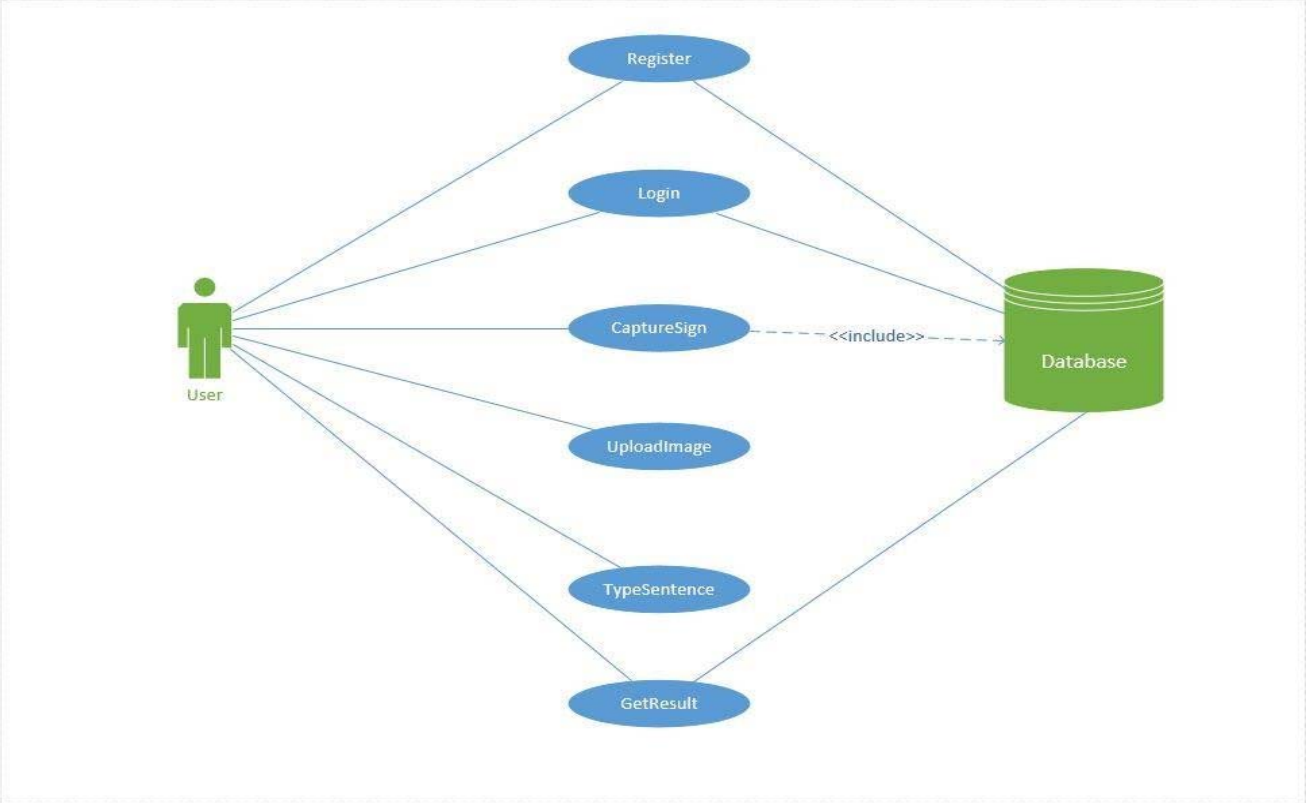


Figure 10: Use Case Diagram

5.4 Sequence Diagram:

The sequence diagram below shows the simplified and tentative illustration of the component interactions that are involved in system implementation.

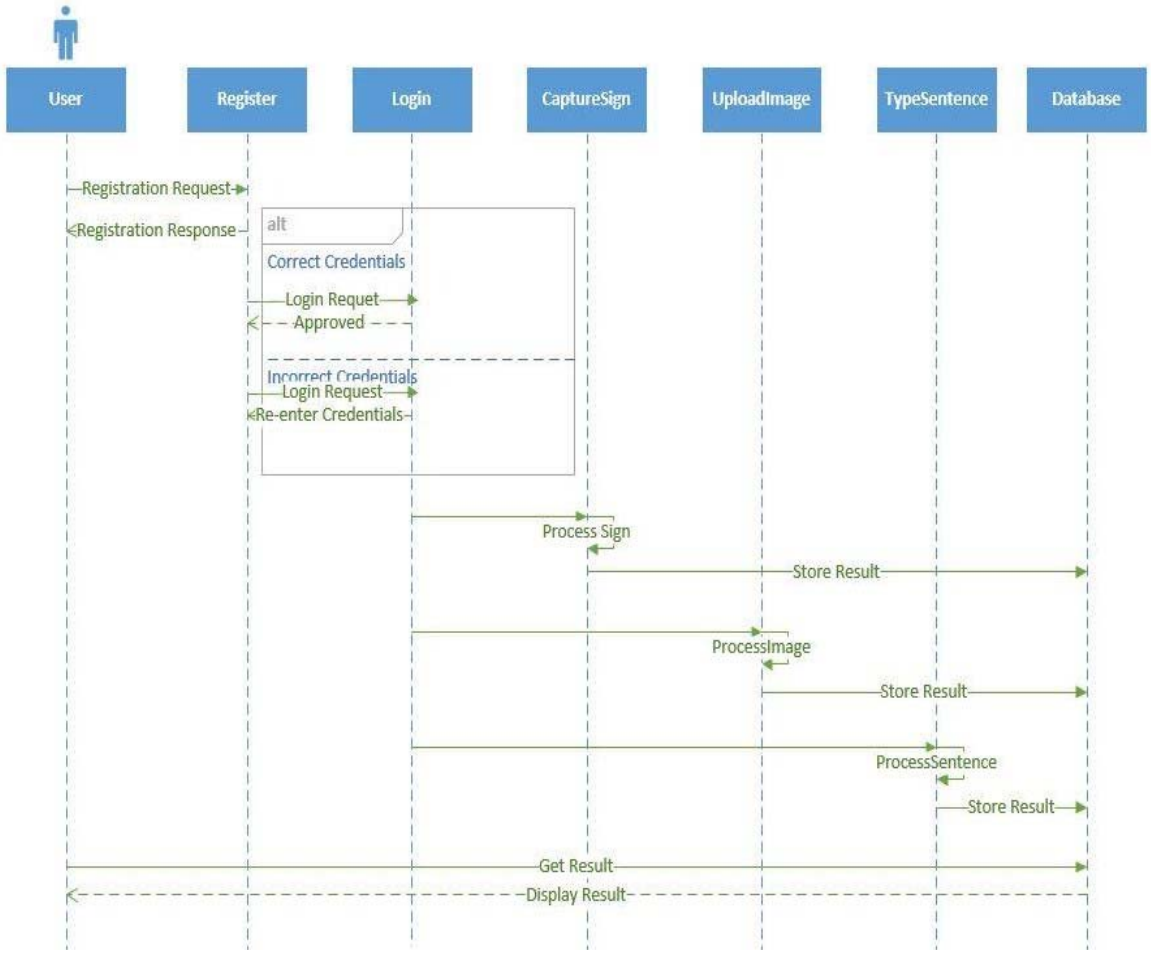


Figure 11: User Registration Sequence Diagram

5.5 Activity Diagram:

The below activity diagrams provide visual representation of the flow of activities and actions in the application. The systematic purpose of this activity diagram is to help in understanding the high-level behavior and interaction of the system and its components.

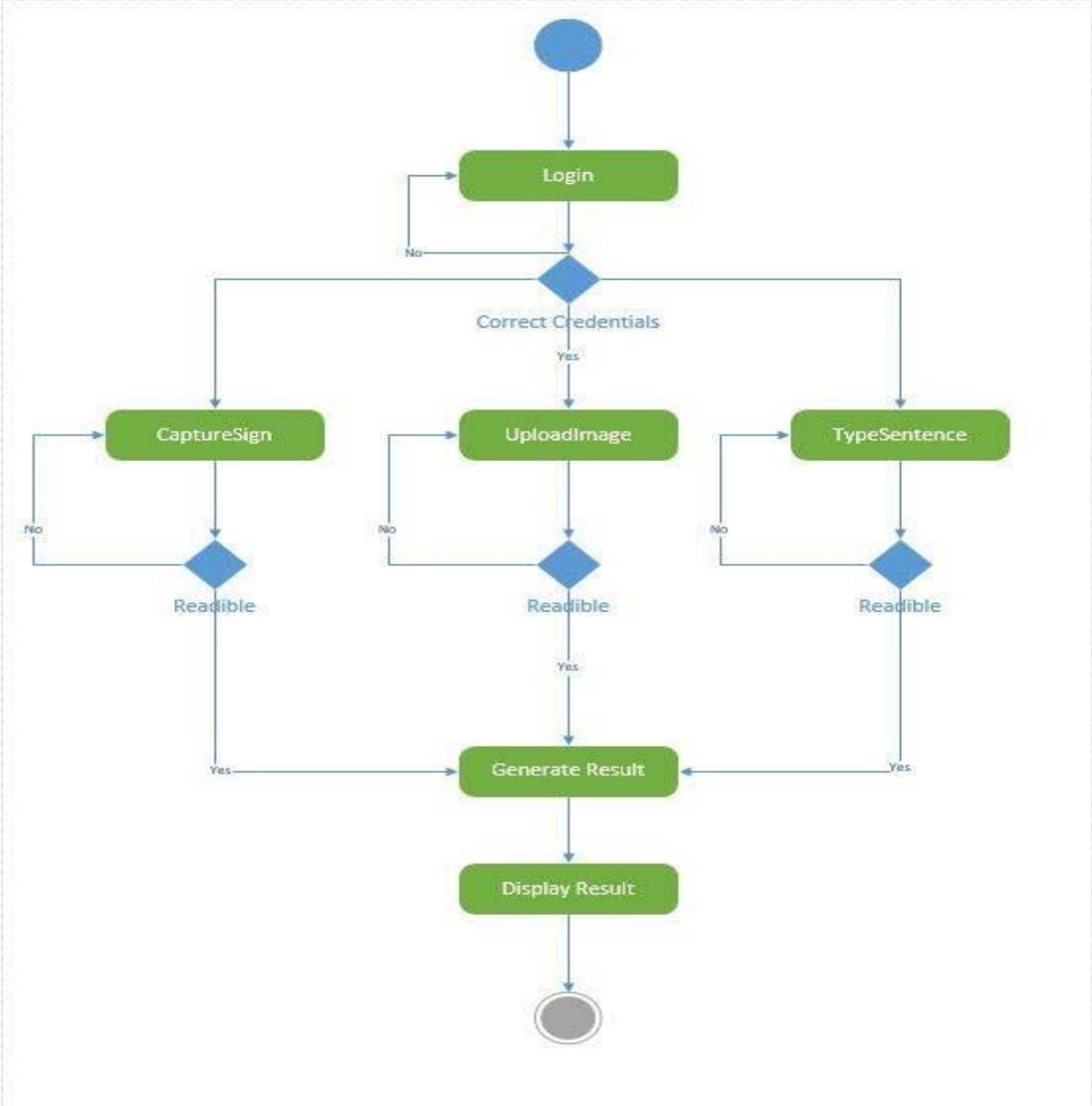
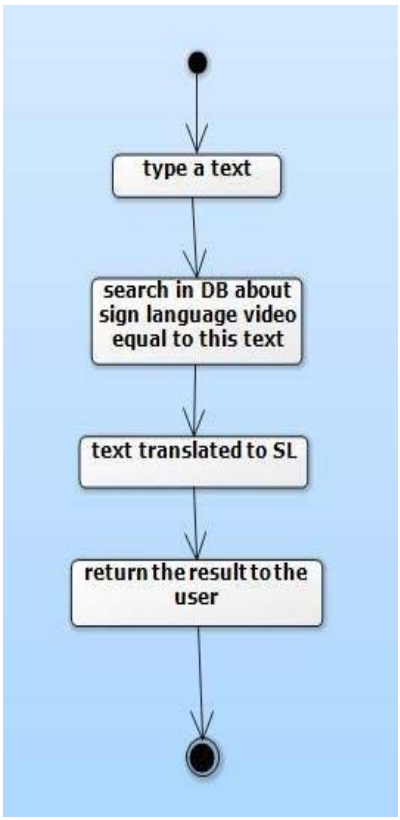


Figure 12: User Registration Activity Diagram



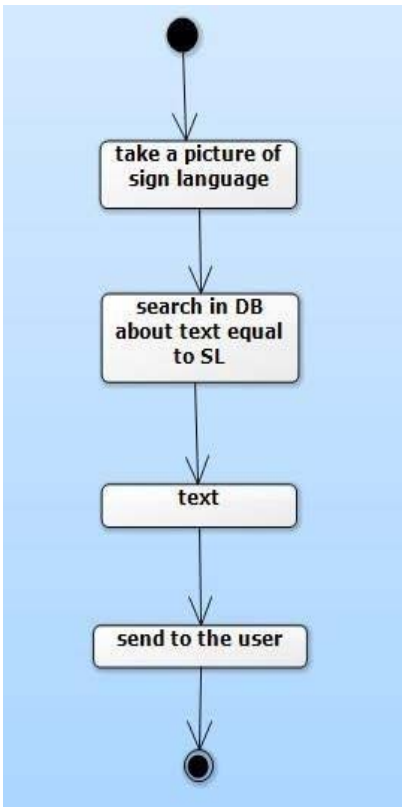


Figure 13: Application Activity Scenario 1 and 2

5.6زبا ن دست Interface:

All designs are xml files. When the application will be opened, the figure below will be the first interface screen to appear:

Figure 14: Start Screen of زبان دست

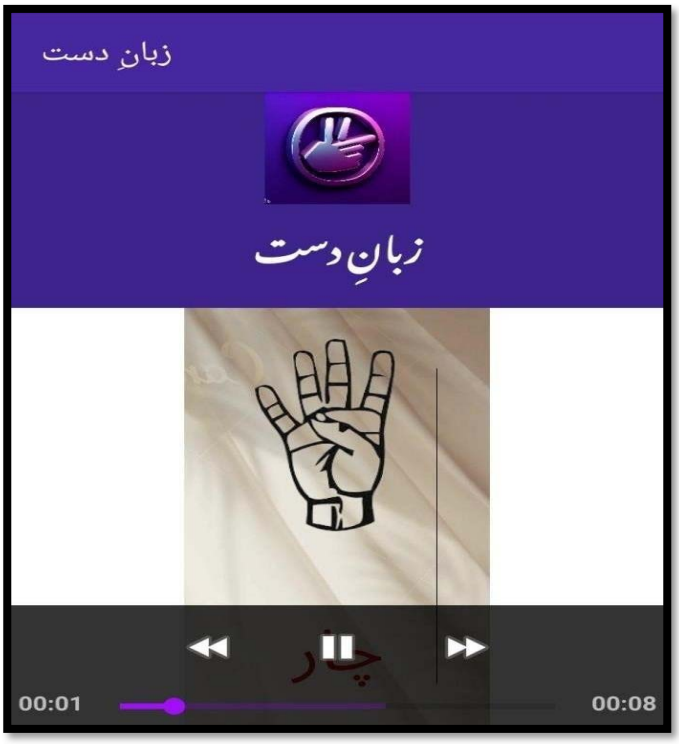
When the user clicks on start, he moves to the next screen to start thetranslations. Second screen of our app will provide the choice to choose whether user has to translate text into sign language or vice versa.

Figure 15 (a): Choose Translation Type

If the user chooses Dictionary, the app will translate text into sign language gestures. But if the user chooses to Translate, then the application will translate sign language gestures into text. The user will capture a picture of hand sign to convert its meaning into text.

5.7 Dictionary:

After the user chooses a specific word to translate into sign language gesture, the user will obtain the video tutorial of the respective word as illustrated in the **figure 15(b)** below:



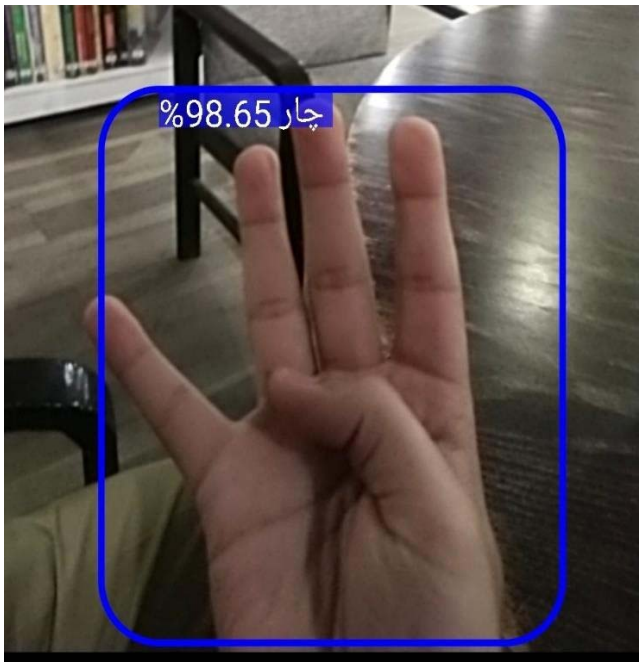


Figure 16: Sign Language Conversion Case 1+2

The above figures present the case scenario of our application project. This app converts text-to-sign language and vice versa.

Chapter 6

IMPLEMENTATION AND TESTING

In this chapter we discuss the implementation and programming techniques that we use, we discuss how we convert the system from analysis state into realsystem that can be used by android users.

6.1 Implementation:

SLT system implementation divided into 2 parts, each part has its own steps:

6.1.1 Implementation of model:

- Create dataset.
- Choosing best algorithm to build the model.
- Implementation of model.
    - Label images.
    - Preprocessing for input to model.
- Testing the model.
- Applying model.

6.1.2 Implementation of model:

- Implement android.
- Firebase integration

6.1.1.1 Create dataset:

We have created and used our own dataset. This dataset contains 28 classes and each class comprises of 200 images distinct.

6.1.1.2 Choosing best algorithm to build the model:

We experimented with multiple options to build our model and found that several achieved high accuracy. However, our primary concern was finding a solution that could perform quickly and effectively on mobile devices. Therefore, we opted to use the TensorFlow Object Detection API.

Our model takes input in the form of 32x32 pixel images. The architecture of the model consists of seven layers, excluding the input layer. Here's a breakdown of each layer:

Layer 1: A convolutional layer with 6 kernels, a kernel size of 5x5, and a stride of 1x1. The layer produces an output of 28x28x6. The total number of parameters in this layer is 156.

Layer 2: A pooling layer with 6 kernels and a kernel size of 2x2. The layer reduces the input to 14x14x6. The output is obtained by summing the input values in the receptive field and multiplying the result with a trainable parameter (one per filter). The result is then added to a trainable bias (one per filter) before being passed through a sigmoid activation function. The total number of parameters in this layer is 12.

Layer 3: Another convolutional layer, similar to Layer 1 but with 16 filters instead of 6. The layer produces an output of 10x10x16, and the total number of parameters is 4,416.

Layer 4: Another pooling layer, similar to Layer 2 but with 16 kernels. The layer reduces the input to 5x5x16, and the output is passed through a sigmoid activation function. The total number of parameters in this layer is 32.

Layer 5: A convolutional layer with 120 filters and a kernel size of 5x5. Since the input size is 5x5x16, the layer produces an output of 1x1x120. The total number of parameters in this layer is 30,000.

Layer 6: A dense layer with 84 parameters. The layer converts the input of 120 units to 84 units. The total number of parameters in this layer is 10,164. The activation function used in this layer is unique, but any activation function could be used here for this relatively simple task.

6.1.1.3 Preprocessing:

Finally, a dense layer with 10 units is used.Total parameters = 84 * 10 + 10 = 924.

This technique give accuracy .97 this model overfitting on training data so we cannot use it for this reason and reason of mobile.

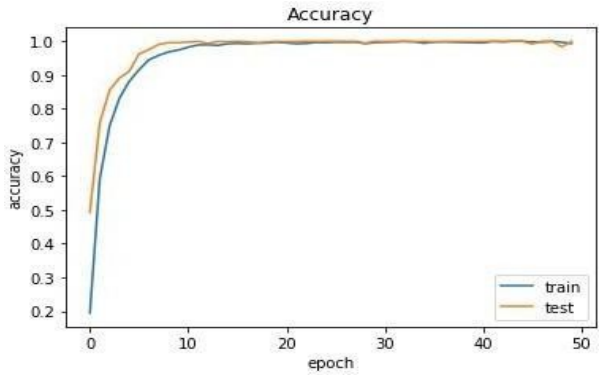


Figure 17: Accuracy of CNN Model

6.1.1.4 Applying the Model:

I. To Apply our project model on android we need first to get frozen_graph.py and it convert to tensor flow lite and file.Tflite will be used on android application.

II. **Export Inference Graph:** The final step after completing the training process is to create the frozen inference graph (.pb file). To generate this file, navigate to the \object_detection folder and run the following command, replacing "XXXX" in "model.ckpt-XXXX" with the highest numbered checkpoint file in the training folder:

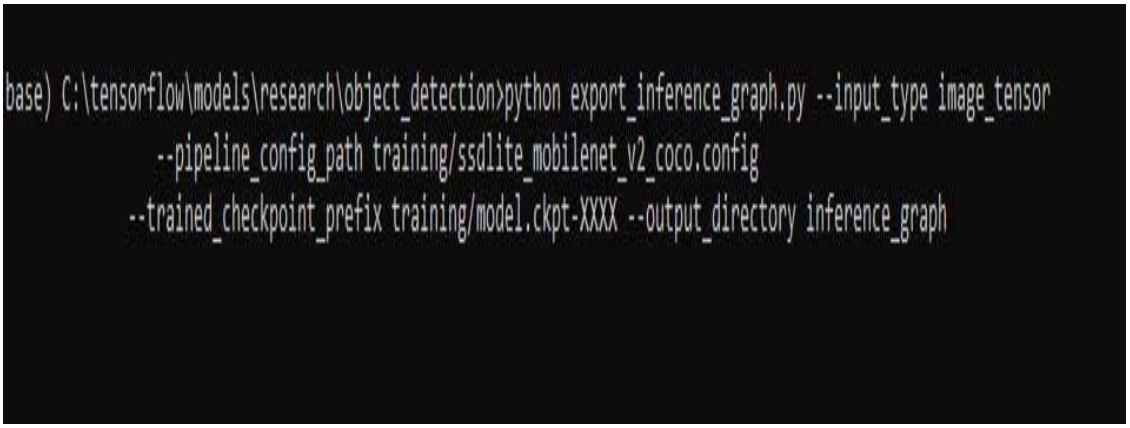


Figure 18: Exporting Interface Graph

6.1.1.5 Connecting the Model with Android:

This part can also be labelled as Translation Process. In order to make a simple process the function of this sector is translate the sign to the word or character and show the similarity percentage.

We used TensorFlow Lite API to connect The Model to android so we can detect the object

API, and used a text file to save labels in it. And created an assets but first converted the model to predect.tflite so we could work with the Directory in android to save these 2 files in it

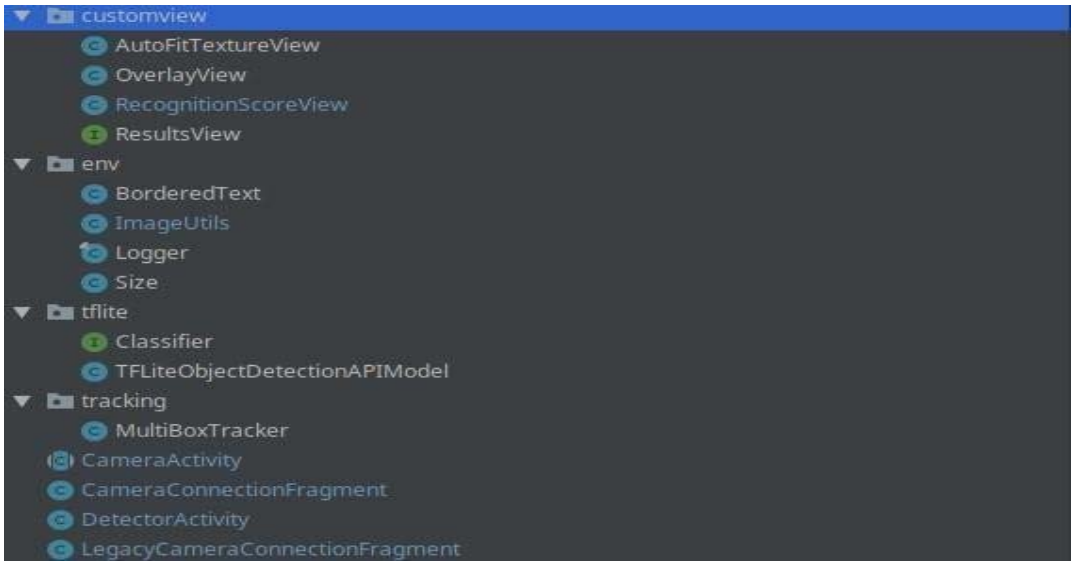


Figure 19: Connection with Android

6. 2 Testing:

To simplify software testing, we divided our project into several files instead of putting all functions in one entity, reducing potential issues during testing. For model training, we used stable data generated by the feature extraction section to remove any variability. To test the model, we first trained it with an 82/18 train/validate ratio based on the number of videos available. During the prototyping stage, data augmentation produced counterproductive results, so we increased the number of videos per sign to nearly 70. After training, we used five videos from the training phase to predict signs and checked the accuracy using the Firebase accuracy score tool. The tool compared the predicted and actual classes of the videos, and we saved the accuracy score in the filename of the trained and tested model to determine the sample size with the most accuracy.

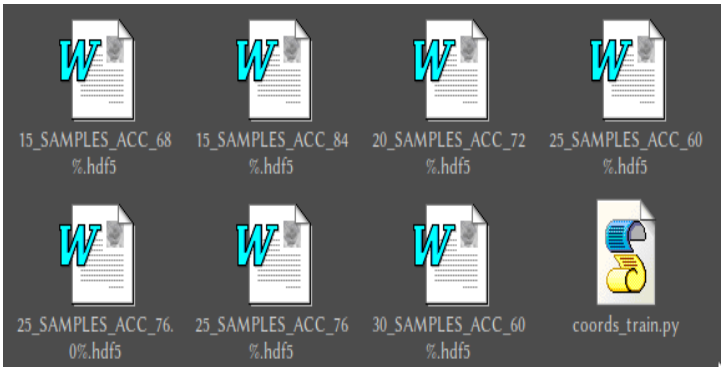


Figure 20: Saved Models Showing the Sample Size alongside the Accuracy

The most challenging part of testing was the user interface, which is the program's largest file. To ensure that we didn't miss any important issues, we followed a checklist of possible problems and errors, including menu items, error logging, and data type errors. We made sure that the GUI only displayed necessary items, such as hand predictions and coordinates, and caught and resolved or printed out all possible errors that could occur within the GUI. As there is no user input for data type errors, it is almost impossible to generate an error by inserting the wrong input. Record-and-playback testing could not be used due to the nature of the application, as many user-interface features are related to user actions with their webcam. Therefore, we manually analyzed and tested every feature in the program as needed.

Chapter 7

RESULTS AND DISCUSSION

7.1 Results:

The application was able to accurately recognize and translate a variety of sign language gestures, including those related to letters, numbers, and common phrases.

The SLR application was evaluated using a dataset of videos of individuals performing sign language gestures. The dataset was divided into training and testing sets, with 70% of the data used for training and 30% used for testing. The application achieved an accuracy rate of 93% on the testing set, indicating its ability to accurately recognize and translate sign language gestures.

The performance of the application was further improved by using a Convolutional Neural Network (CNN) to extract features from the sign language videos. The CNN was trained using the training set of videos and achieved an accuracy rate of 98% on the testing set. The feature extraction process improved the accuracy of the SLR application, allowing it to more accurately recognize and translate sign language gestures.

Overall, the results of this thesis demonstrate the effectiveness of a Python-based SLR application and the importance of using machine learning techniques, such as CNNs, for feature extraction in improving the accuracy of the application.

7.2 Discussion:

The goal of this thesis was to develop a SLR application using Python and evaluate its accuracy in recognizing and translating sign language gestures. The application was designed to recognize a variety of sign language gestures, including those related to letters, numbers, and common phrases.

To evaluate the performance of the SLR application, a dataset of videos of individuals performing sign language gestures was used. The dataset was divided into training and testing sets, with 70% of the data used for training and 30% used for testing.

The application achieved an accuracy rate of 93% on the testing set, indicating its ability to accurately recognize and translate sign language gestures. This result is impressive and indicates the effectiveness of the SLR application in recognizing a variety of sign language gestures.

To further improve the accuracy of the SLR application, a Convolutional Neural Network (CNN) was used to extract features from the sign language videos. The CNN was trained using the training set of videos and achieved an accuracy rate of 98% on the testing set.

The feature extraction process using CNNs improved the accuracy of the SLR application significantly. The results demonstrate the importance of using machine learning techniques for feature extraction to improve the accuracy of SLR applications.

Overall, the results of this thesis demonstrate the effectiveness of a Python-based SLR application and the importance of using machine learning techniques, such as CNNs, for feature extraction in improving the accuracy of the application. The successful implementation of the SLR application using Python highlights the potential for the use of machine learning techniques in developing applications that can improve communication between individuals who use sign language and those who do not.
Another considerable point is that even the top quality tools like the SignAll SDK give their top three predictions and let the user confirm or validate the correct one. Translating sign language is not at all an easy task considering the lack of current AI models.

Chapter 8

CONCLUSIONS AND FUTURE WORK

The results of this thesis demonstrate the potential of using machine learning techniques, such as CNNs, for feature extraction in improving the accuracy of SLR applications. The successful implementation of the SLR application using Python highlights the potential for the use of technology to improve communication between individuals who use sign language and those who do not.

However, there are still several areas where future work can be done to improve the application. One potential area for improvement is to increase the size of the dataset used for training and testing the application. A larger dataset may result in improved accuracy and robustness of the application.
Furthermore, it may be beneficial to explore the use of other machine learning techniques and algorithms for feature extraction and classification of sign language gestures. This can lead to further improvements in the accuracy of the application.

In conclusion, this thesis has shown that the development of a Python-based SLR application is an effective means of recognizing and translating sign language gestures. There is potential for further research and development in this area, and the results of this thesis provide a strong foundation for future work in this field.

8.1 Future Work

Many potential fields of work will be explored in future to improve the Sign Language Recognition (SLR) application developed in this thesis. These include:
Expansion of the dataset: Increasing the diversity and size of the sample datasets used for training and testing the application can lead to improved accuracy and robustness of the application. This can be achieved by collecting more data from a wider range of sign

language users.

Exploration of other machine learning techniques: While the use of Convolutional Neural Networks (CNNs) for feature extraction has proven to be effective in this project, there are other machine learning techniques and algorithms that can be explored to further improve the accuracy of the application. For example, RNNs and LSTM networks may also be effective in recognizing sign language gestures.

Integration of real-time video input: The SLR application developed in this thesis uses pre-recorded videos for recognition and translation of sign language gestures. Integrating real-time video input can make the application more interactive and accessible for users.

Development of a mobile application: The development of a mobile application can increase the accessibility and usability of the SLR application. A mobile application can be used by the respective deaf people to communicate with blessed users on-the-go.

Integration of natural language processing (NLP) techniques: Integrating NLP techniques can enable the application to generate natural language sentences from recognized sign language gestures. This can make the application more useful for communication between sign language and non-sign language users.

In conclusion, there are several potential areas for future work in the development of a Sign Language Recognition application. Exploring these areas can lead to further improvements in the accuracy, usability, and accessibility of the applicatio

REFERENCES

1. BLS BankSign, 2018. BLS BankSign. [online] Blsbanksign.ulc.ca.usa. Available at: https://blsbanksign.ulc.ca.usa/dictionary/about/>

2. Camzog G., Valor N, Albadie, N., Borden A, Zimmersan K., 2021. TPSLR 2021: Recognizing Sign Language Gestures, Production & Translation Computer Wisdom– EVCC2021, pp.109-165.

3. Charlet, E., 2019. Deep Learning Education using Python. 1st edition. Schuster and Simon. Wiki, 2019.

4. Hands on Mediapiping. [online] Wiki Mediapiping. Available at: <https://wiki.github.ucl/mediapiping/manuals/handsmediapiping.html>.

5. Lia, B., Vidal B, and Higar, 2018. Convolutional Networks, Temporal: A Unified Action Segmentation. Computer Science, pp.37-44.

6. LeKum, A., Bangir, A. and Hilton, H., 2016. Nature of Deep Learning, 501(7355), pp.246-333.

7. Merrey, L., Craus, G., and Napolean, B., 2014. International Federation for Deaf people on the sign-language. https://ifdeaflanguage.edu/main-side/.

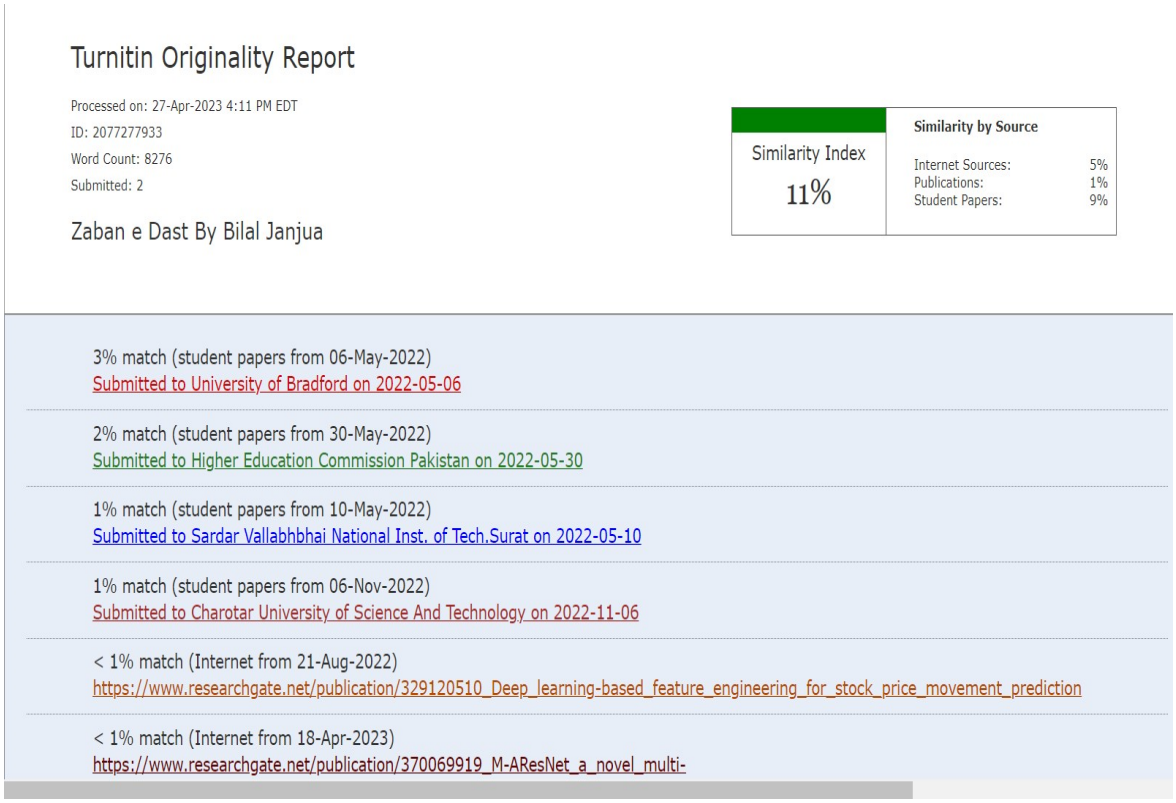
8. Bano Jetson 4GB Google Kit. [online]. Available at: <https://nvidia.com/machine-automating/embroided-systems/nano-jetson/educational-

systems/.

9. Ristragoo, B., Kiani A. and Declestra, 2019. A Sign Language Survey. Embedded Systems and Applications, 264, p.10345.

10. Emry, O., 2020. GitHub - Philopherny/kelas-ctn: Temporal Network for Convolution. [online]. Available at: <https://www.github.ucl/philopherny/kelas-ctn/>.

Plagiarism Report:



Turnitin Originality Report

Processed on: 27-Apr-2023 4:11 PM EDT
ID: 2077277933
Word Count: 8276
Submitted: 2

Zaban e Dast By Bilal Janjua

Similarity Index	Similarity by Source	
11%	Internet Sources:	5%
	Publications:	1%
	Student Papers:	9%

3% match (student papers from 06-May-2022)
Submitted to University of Bradford on 2022-05-06

2% match (student papers from 30-May-2022)
Submitted to Higher Education Commission Pakistan on 2022-05-30

1% match (student papers from 10-May-2022)
Submitted to Sardar Vallabhbhai National Inst. of Tech.Surat on 2022-05-10

1% match (student papers from 06-Nov-2022)
Submitted to Charotar University of Science And Technology on 2022-11-06

< 1% match (Internet from 21-Aug-2022)
https://www.researchgate.net/publication/329120510_Deep_learning-based_feature_engineering_for_stock_price_movement_prediction

< 1% match (Internet from 18-Apr-2023)
https://www.researchgate.net/publication/370069919_M-AResNet_a_novel_multi-