**A PROJECT REPORT**

**ON**

**CAMERA-BASED SIGN LANGUAGE RECOGNITION AND SPEECH GENERATION**

SUBMITTED TO THE SAVITRIBAI PHULE PUNE UNIVERSITY, PUNE

IN THE PARTIAL FULFILLMENT OF THE AWARD OF THE DEGREE

**OF**

**BACHELOR OF ENGINEERING**

**IN**

**INFORMATION TECHNOLOGY**

**BY**

**AYUSHI PATANI 71829052G**

**VARUN GAWANDE 71828724L**

**JASH GUJARATHI 71828800K**

**VEDANT PURANIK 71829113B**

**UNDER THE GUIDANCE OF**

**PROF. T. A. RANE**

****

**PUNE INSTITUTE OF COMPUTER TECHNOLOGY, PUNE**

**2020-21**

**CERTIFICATE**

This is to certify that the project report titled

**CAMERA-BASED SIGN LANGUAGE RECOGNITION AND SPEECH GENERATION**

**Submitted by**

**AYUSHI PATANI 71829052G**

**VARUN GAWANDE 71828724L**

**JASH GUJARATHI 71828800K**

**VEDANT PURANIK 71829113B**

is a bonafide work carried out by them under the supervision of Prof. T. A. Rane and it is approved for the partial fulfillment of the requirement of Savitribai Phule Pune University for the award of the Degree of Bachelor of Engineering (Information Technology)

This project report has not been earlier submitted to any other Institute or University for the award of any degree or diploma.

**Prof. T. A. Rane** **Dr. A. M. Bagade**

Internal Guide Head of Department

Department of Information Technology Department of Information Technology

**Dr. R. Sreemathy**

External Examiner Principal

Date:Pune Institute of Computer Technology

Place:

Date:

**ACKNOWLEDGEMENT**

We thank everyone who has helped and provided valuable suggestions for supporting us through the project.

We are very grateful to our guide, Prof. T. A. Rane, Head of Department Dr. A.M. Bagade and our principal Dr. P. T. Kulkarni. They have been very supportive and have ensured that all facilities remained available for smooth progress of the project.

We would like to thank Prof. Thamke and Prof. Buradkar for providing very valuable and timely suggestions and help. We would also like to thank the entire project staff team for providing valuable reviews and suggestions from time to time.

We would like to thank our entire department and college staff for the very valuable help and coordination throughout the duration of the project.

We would also like to thank our families and all our friends for the valuable support they provided, directly or indirectly, throughout the duration of the project.

Ayushi Patani

Varun Gawande

Jash Gujarathi

Vedant Puranik

LIST OF FIGURES

| **Figure Number** | **Figure Name** | **Page Number** |
| --- | --- | --- |
| 3.6.5 | Project Plan 2.0 Gantt Chart | 20 |
| 3.6.6 | PERT Diagram | 21 |
| 4.1 | Architecture | 22 |
| 4.2 | Data-Flow Diagram | 23 |
| 4.3.1 | Activity Diagram | 24 |
| 4.3.2 | Sequence Diagram | 25 |
| 4.3.3 | State Diagram | 26 |
| 4.5.1 | User Interface | 27 |
| 4.5.2 | Video Frame for ‘Please’ | 28 |
| 4.5.3 | Video Frame for ‘Help’ | 28 |
| 4.5.4 | Example Interface Output | 29 |
| 6.1.1.1 | VGG16 Architecture | 36 |
| 6.1.1.2 | LSTM Model Architecture | 37 |
| 6.1.2.1 | Loss Trend | 40 |
| 6.1.2.2 | Accuracy Trend | 40 |

LIST OF TABLES

| **Table Number** | **Table Name** | **Page Number** |
| --- | --- | --- |
| 3.1 | Division of Roles and Responsibilities | 19 |
| 6.1 | Dataset Description | 32 |
| 6.1.2.1 | Results for Varying Number of Words | 38 |
| 6.1.2.2 | Classification Report | 38 |

LIST OF ABBREVIATIONS

| **Term** | **Meaning** |
| --- | --- |
| USB | Universal Serial Bus |
| fMRI | Functional magnetic resonance imaging |
| WLASL | Word Level American Sign Language |
| OpenCV | Open Source Computer Vision |
| CNN | Convolutional Neural Network |
| GPT | Generative Pre-trained Transformer |
| GUI | Graphical User Interface |
| IDE | Integrated Development Environment |
| FPS | Frames per Second |
| LSTM | Long Short Term Memory |
| VGG | Visual Geometry Group |
| HOG | Histogram of Gradients |
| RNN | Recurrent Neural Network |

**CONTENTS**

| **CERTIFICATE** | 2 |
| --- | --- |
| **ACKNOWLEDGEMENT** | 3 |
| **LIST OF FIGURES** | 4 |
| **LIST OF TABLES** | 5 |
| **LIST OF ABBREVIATIONS** | 6 |
| **ABSTRACT** | 9 |
| **1. INTRODUCTION**  1.1 OVERVIEW  1.2 MOTIVATION | 10  10  10 |
| **2. LITERATURE REVIEW**  2.1 EXISTING METHODOLOGIES  2.2 PROPOSED METHODOLOGY | 11  11  13 |
| **3. REQUIREMENT SPECIFICATION AND ANALYSIS**  3.1 PROBLEM DEFINITION  3.2 CONCEPT  3.3 SCOPE  3.4 OBJECTIVE  3.5 PROJECT REQUIREMENTS  3.5.1 DATASETS  3.5.2 FUNCTIONAL REQUIREMENTS  3.5.3 NON-FUNCTIONAL REQUIREMENTS  3.5.4 HARDWARE REQUIREMENTS  3.5.5 SOFTWARE REQUIREMENTS  3.6 PROJECT PLAN  3.6.1 PROJECT RESOURCES  3.6.2 MODULE SPLIT-UP  3.6.3 FUNCTIONAL DECOMPOSITION  3.6.4 PROJECT TEAM ROLE AND RESPONSIBILITIES  3.6.5 PROJECT PLAN 3.0 | 14  14  14  14  15  15  15  15  16  16  17  17  17  17  18  19  20 |
| **4. SYSTEM ANALYSIS AND DESIGN**  4.1 ARCHITECTURE  4.2 DFD  4.3 BEHAVIORAL DIAGRAMS  4.3.1 ACTIVITY DIAGRAM  4.3.2 SEQUENCE DIAGRAM  4.3.3 STATE DIAGRAM  4.4 ALGORITHMS  4.5 UI | 21  21  22  23  23  24  25  26  27 |
| **5. IMPLEMENTATION**  5.1 STAGES OF IMPLEMENTATION  5.1.1 PREPARATION OF DATA  5.1.2 PRE-PROCESSING  5.2 IMPLEMENTATION ISSUES  5.2.1 COMPUTATIONAL RESOURCES  5.2.2 STORAGE ISSUES  5.3 TECHNIQUES  5.3.1 HAND DETECTION  5.3.2 TRANSFER LEARNING  5.3.3 LSTM  5.3.4 SPEECH GENERATION  5.4 SOFTWARE TOOLS | 30  30  30  32  33  33  33  33  33  34  34  34  35 |
| **6. RESULTS AND EVALUATION**  6.1 EXPERIMENTS  6.1.1 DETAILED DISCUSSION OF EXPERIMENTS  6.1.2 RESULTS  6.2 TESTING | 36  36  36  38  41 |
| **7. CONCLUSION**  7.1 CONCLUSION  7.2 LIMITATIONS  7.3 FUTURE SCOPE | 42  42  42  42 |
| **REFERENCES** | 43 |
| **APPENDICES**  Base Paper  Plagiarism Report  Semester 7 Documents  Semester 8 Documents | 46  46  47  48  56 |

ABSTRACT

People regularly face problems interpreting deaf-mute people, who primarily use sign language for communication amongst themselves and others. Despite efforts being conducted by different governments worldwide such as the provision of a sign language expert for interpreting and communicating all news to the impaired by the New Zealand media, active participation of the impaired is still at a very rudimentary stage.

Further, only a few people today are proficient in communicating via sign language and hence the majority of the population at large is still devoid of any understanding of the matter. This could be problematic for deaf-mute people especially during situations of distress like pain, fraud, or other emergency situations like murder, robbery, kidnapping, etc.

All these problems could be minimized substantially if this language barrier is effectively bridged. Our project aims to tackle this problem by setting up a sign-language interpreter that inputs sign language video from the signer and then processes it to output its equivalent spoken language meaning. Along with that it also generates an audio output to make sure that the illiterate and the blind can also understand what the signers are saying so that no section of the society remains left out.

# CHAPTER 1

**INTRODUCTION**

# 1.1 OVERVIEW

Our system works on a video input that is given by the user. This input could be either a pre-recorded video or a live video. Our project aims to interpret sign-language performed by the signers. When provided as a directory path, it takes in the videos individually and first performs a basic verification check of the file type. Once verified, it breaks the video into an image sequence. This image sequence is then checked for relevant content i.e. if the video submitted actually has a human or not. Once confirmed, hand recognition is performed on the system to extract only the hand movements in the video, hence removing all the unnecessary noise. The hand-extracted frames are then worked upon to extract features from them. This sequential data is then passed through a recurrent neural network to make predictions of the words performed by the signer. The output is then shown to the user as text and optionally as audio.

# 1.2 MOTIVATION

Mute people often face a challenge of actively communicating with the outside world as they are not able to speak and hence rely on sign language to express themselves. However, only a few people except the mute understand and are able to interpret sign language. Moreover, in times of distress such as kidnapping, robbery, etc., mute people could find it extremely difficult to find an interpreter to ask for help.

Therefore, to bridge the gap between the differently abled and the society, we propose a system that aims to interpret sign language and produce an output in textual and audio format, the latter being helpful for the people who are illiterate and/or blind.

CHAPTER 2

**LITERATURE REVIEW**

# 

# 2.1 EXISTING METHODOLOGIES

* **Camera-based image or video capturing :** One of the most extensively applied methods has been camera-based image or video capturing for sign language understanding. Capturing one hand, two hand signs as well as working on static and dynamic data have been different ways of going about the problem. Therefore the signs could be a sequence of images (generally for a word) or could be just an individual image (generally for a letter or digit). As far as videos are concerned, they are first split into images, which work as sequential data and then are worked upon by deep learning techniques. These techniques are then analyzed to obtain stable results.
* **Microsoft Kinect** **:** Researchers are also exploring Kinect as a viable alternative . Signers’ hand and body movements can be captured by Microsoft Kinect without much delay. Although the focus of Kinect’s audience has been video games, it [1] provides an uncanny edge over other image/video capturing techniques as it adds another dimension to the captured data, which is depth e.g. color depth, etc.. However, as the camera method is cheaper and has more portability, the Kinect approach is not as widely applicable as the camera approach.
* **Using Armband** **:** Electromyography (EMG) Signals provides a basis to the armband approach. These signals are generated in our muscles whenever there is any movement. Signals originating from this armband are assimilated[2] and then processed to recognize sign language. This approach has light intensity independent and hence has an edge with this respect over the camera and Kinect approaches. However, a major impediment this approach faces is the necessity to connect a number of wires to the system and hence it also hinders portability.
* **Using Cyber Glove :** Groundbreaking research of 1993 gave rise to another approach called the Cyber Glove[3]. Multiple sensors and a motion tracker [4] are attached to the glove which is supposed to be worn by the signers. The data obtained from these sensors is then sent to a computing device for further processing and subsequent interpretation.This method also faces a similar issue as kinect and armband of system configuration and setup. In real-life situations, this approach is hence not very viable and could be avoided. Moreover, it is also unable to capture facial features and symbols which can be easily done in camera-based systems.
* **Leap Motion Controller Based System** **:** An economical approach is using a low cost sensor based system called the Leap Motion Controller[5]. APIs are used to send hand and gesture movements to the computer, which are performed on top of a sensor that is horizontally placed. Finally the resultant data is forwarded to a computer. This is a cheaper alternative to the aforementioned techniques however still faces the hurdles as faced by the earlier methods.
* **Brain-Computer Interfacing** **:** One of the most advanced methods to detect and recognize sign language is brain-computer interfacing. Electroencephalogram [6] brain activities are obtained for the recognition of sign-language.This approach nullifies the need to capture hand movements and solely relies on brain waves which are captured and sent to a computing device for processing. fRMI[7] and Electrocorticography[8] are similar techniques but they all still require gear connected to a human brain to capture the signals and hence have portability issues.
* **Deep Learning:** Multiple approaches[19][20][21][22][23] with deep learning have also been tried out on private datasets and public datasets. Convolutional Neural Networks play an important role for image classification and they along with RNNs can be used for time series classification.

# 2.2 PROPOSED METHODOLOGY

Our project aims to capture sign language performed by signers on a real-time basis and interpret the language to produce textual and audio output for the illiterate. For this, a camera-based approach will be made use of, owing to the ease of portability and movement that the camera-based method offers over other techniques. The video of the signer will be first captured by a camera-enabled device or can be also given as a series of videos in a directory. This video will then be processed by our application in which the video would be divided into a number of frames which will convert the video into a raw image sequence. Then the presence of a human is checked to make sure the video actually has a person. This image sequence is then worked upon to identify the hands in every frame. Only hands from every frame will be considered for the representation of the sign language words.

These hand-extracted frames will be then passed through a state-of-the-art model to extract features from these hands hence making use of transfer learning. These extracted features will then be passed through a recurrent neural network to make predictions about the specific sign being performed. All the videos for the words have been obtained from the WLASL dataset.

Finally, this output will be sent through an audio generator to generate speech from the same. This provides support for illiterate people and/or blind people, who are not able to understand written text.

# 

# 

# CHAPTER 3

**REQUIREMENT SPECIFICATION AND ANALYSIS**

# 3.1 PROBLEM DEFINITION

To build a system that can interpret sign language performed by signers on a real-time basis (or a prerecorded video) and translate sign language into its word equivalents and provide textual and audio output to the user to bridge the communication gap between the mute people and the Society.

# 3.2 CONCEPT

A live or a pre-recorded video will be provided by the user of the system. Image frames will be extracted from this video and verification of the same will be done. This will be followed by hand detection. Then we perform feature extraction using a CNN based transfer learning model which is followed by the use of a RNN to determine the corresponding English word for the sign performed. The output generated will then be presented to the user in textual and audio format.

# 3.3 SCOPE

* The system will be capable of interpreting the American sign language only, as only one language was to be worked on and sufficient open-source data was only available for this language.
* For the sake of research, only a limited set of words will be used for recognition initially.
* The signer should necessarily stand exactly in front of the video-recording camera.
* It performs only one-way communication i.e. sign language to spoken language.

# 3.4 OBJECTIVES

* Capture raw image sequence from camera.
* Use face recognition to verify that the video actually has a person.
* Use hand recognition to extract only hand data for all the video frames.
* Use a convolutional neural network for extracting features from the video frames.
* Classification of signs into words based on summation of obtained information using a recurrent neural network.
* Generating speech from the generated text.

# 3.5 PROJECT REQUIREMENTS:

## 

## 3.5.1 DATASETS

* Upon extensive research into possible datasets for our Deep learning Model, we have chosen to work on a large video dataset for World Level American Sign Language.
* It includes a json file with links to 20000+ videos hosted on multiple different websites including youtube, asl.org and Facebook.
* These videos refer to about 2000 gestures making it the largest dataset publicly available for research and academic purposes.
* These videos are generally 2-4 second snippets of a sign language expert enacting the signs in front of a blank/easily distinguishable background.
* We have also recorded additional videos for balancing the number of videos per word and boosting the amount of training data for the model.

## 

## 3.5.2 FUNCTIONAL REQUIREMENTS

* The proposed system must perform boundary identification on the generated raw image sequence and demarcate the areas for hands.
* The proposed system should extract features from the recognized hands by using convolutional neural networks.
* The proposed system must then classify the gesture as a known gesture for which it has been trained.
* The proposed system should then display the identified gesture as text and play audio for the same.

## 3.5.3 NON-FUNCTIONAL REQUIREMENTS

* The proposed system must be able to accept data as real time video feed from a camera as well as an uploaded video.
* Given the lack of verifiability in real time use and critical nature of the proposed system, the system must be able to identify the performed gesture with a very low false positive rate.
* It could be acceptable if this was achieved with a relatively higher false negative rate.
* The User Interface must be visually intuitive, aesthetically pleasing and follow standard UI development principles and norms.

## 3.5.4 HARDWARE REQUIREMENTS

* Camera Specifications -
  + 0.307 MegaPixel still image camera (basic)
  + 640 X 480 at 24fps (preferred)
  + Front mounted
* Processing Power requirements
  + 4GB RAM (minimum), 8GB RAM (preferred)
  + Intel i3 processor (minimum), i5/i7 (preferred)
  + Discrete GPU with 15GB dedicated memory (for model training) (preferred)

## 

## 3.5.5 SOFTWARE REQUIREMENTS

* Operating System - Linux, Windows, Mac OS
* Support for executing Python 3.x
* Libraries/Dependencies :
  + OpenCV
  + TensorFlow
  + Keras
  + Numpy
  + Sci-Kit Learn

# 3.6 PROJECT PLAN

## 3.6.1 PROJECT RESOURCES

* Google Colab/Jupyter Notebook as the environment for the model training and testing.
* Python Tkinter for platform development.
* Google Drive as a file system to store uploaded files.
* Python 3.x to be used as the default programming language.
* TensorFlow and OpenCV packages will be used for the Development of CNNs and RNNs

## 3.6.2 MODULE SPLIT-UP

* **User-Interface**: The GUI that the user interacts with. It bears the responsibility of communicating the results of operations with the User, regardless of the status of the result is a Success or a Failure (e.g., errors and exceptions). It provides the user with an interface to upload videos or provide a Live Video stream. It validates some details of the video and lets the user know if some video details are invalid. Finally, it also communicates the results of the translation to the user.
* **Video Pre-processor:** Undertakes the task of breaking the input video into a set of image sequences holding potential meaning.
* **Face Recognizer:** Performs a validation check on the video to determine if the video has a human or not by using the Haar Cascade classifier.
* **Hand Gesture processor:** Extracts images out of the given image sequence in terms of meaningful Hand Gestures.
* **Feature Extractor:** Extracts features from the video frames given to it as input by using a state-of-the-art convolutional neural network architecture.
* **Classifier:** Module that takes the input of the well-defined features and classifies the images sequences into words using a recurrent neural network.
* **Text-to-speech:** Generates audio for the output text.

## 3.6.3 FUNCTIONAL DECOMPOSITION

* The user is provided with the option of either uploading a video file or starting a live stream through the attached camera.
* If the user uploads a video then the video is initially validated for its format.
* Then the video data is passed onto the main module which initially performs pre-processing on the data such as conversion to an image sequence, hand recognition.
* If the preprocessor fails to find relevant data (For eg. the video does not have a human subject) then it notifies the user of the same, and the user is expected to act on the error and restart the process.
* Once the pre-processor demarcates the hand from the image sequence data, the hand recognized data is passed through a CNN model to extract features from it.
* Features are sent back to the main module which then passes the data to the primary classifier that classifies each frame sequence into a word.
* If the gesture cannot be identified then the user is prompted about the lack of identification. The user can then retry the process or exit the process.
* The text is displayed on the User interface and the text is also converted into audio to be played through the speakers ensuring effective communication.

## 3.6.4 PROJECT TEAM ROLE AND RESPONSIBILITIES

| **Member Name** | **Role** | **Responsibility** |
| --- | --- | --- |
| Ayushi Patani | Data Scientist, Full Stack Platform Developer, Tester | Web Scraping and data preprocessing, Model Development, Model Testing, Platform Design and Development, Recording Results and documentation |
| Varun Gawande | Data Scientist, Full Stack Platform Developer, Tester | Web Scraping and data preprocessing, Model Development, Platform Design and Development, Recording Results and documentation. |
| Jash Gujarathi | Data Scientist, Full Stack Platform Developer, Tester | Web Scraping and data preprocessing, Model Development, Model Testing, Tuning and optimization, Integration and Live Testing |
| Vedant Puranik | Data Scientist, Full Stack Platform Developer, Tester | Web Scraping and data preprocessing, Model Development, Tuning and optimization, Integration and Live Testing |

## Table 3.1: Division of Roles and Responsibilities

## 

## 3.6.5 PROJECT PLAN 3.0

* We have divided our project into multiple small steps and created a timeline accordingly.
* This timeline has been depicted in the following Gantt chart (Refer fig 3.6.5)
* The timeline has been decided by taking into consideration the current pandemic situation.

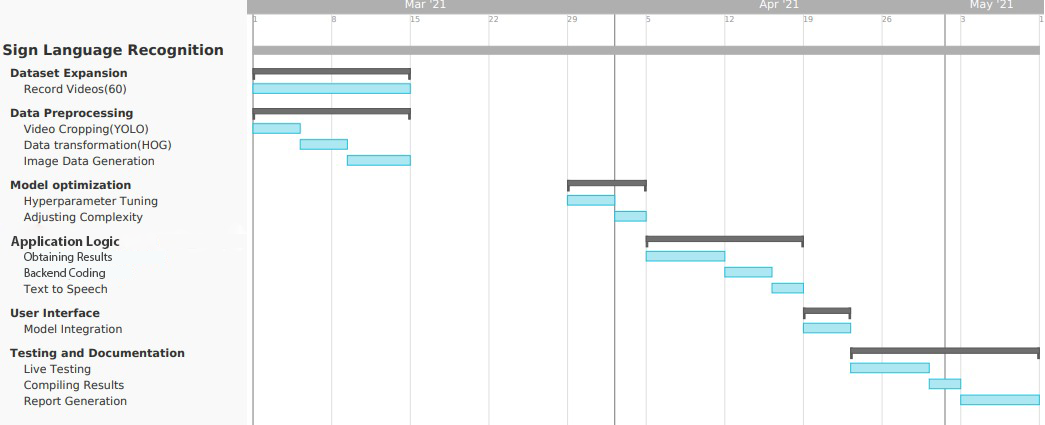


Figure 3.6.5: Project Plan 3.0 Gantt Chart

## 

**CHAPTER 4**

**SYSTEM ANALYSIS AND DESIGN**

# 4.1 ARCHITECTURE

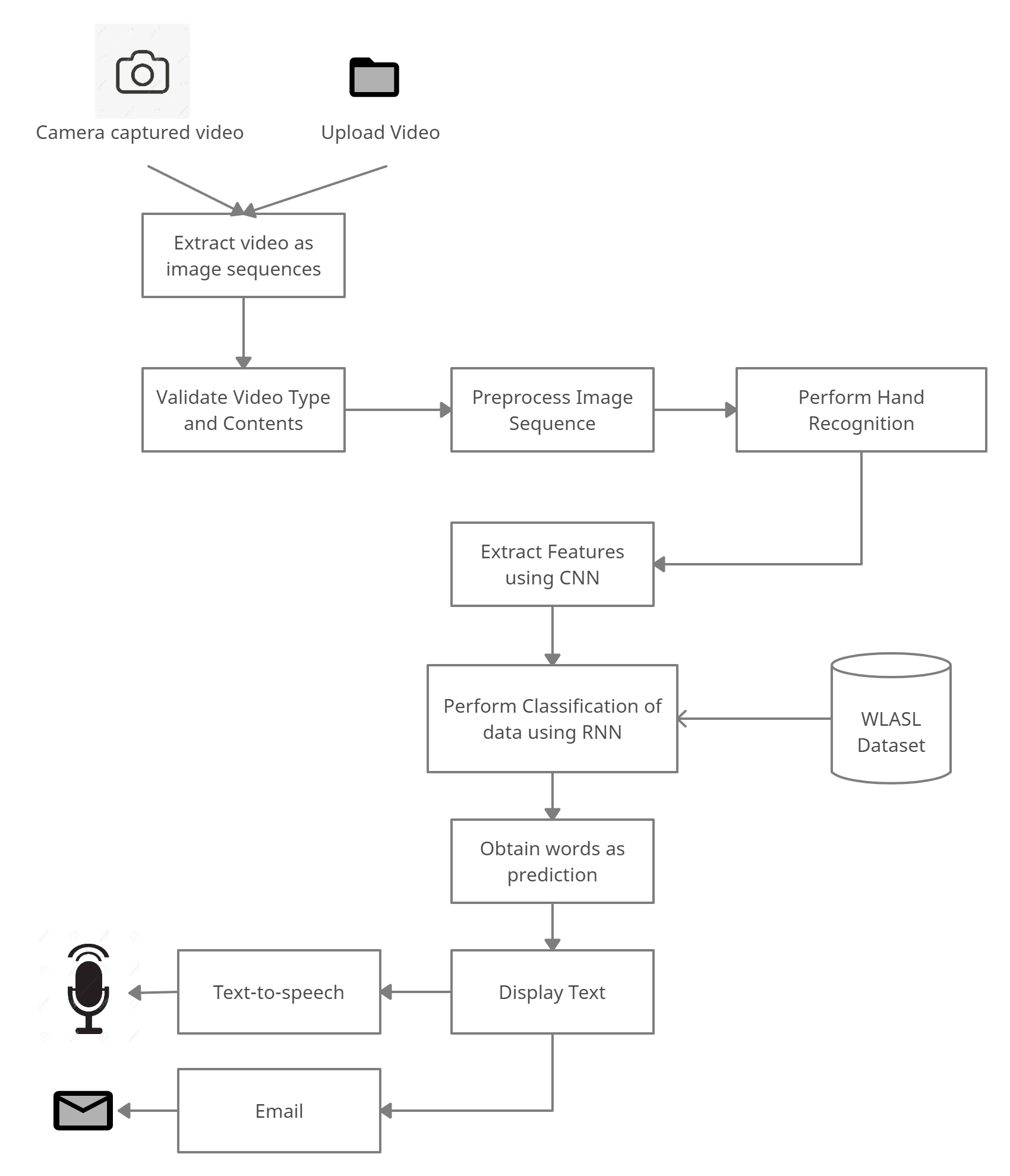


Figure 4.1: Architecture

# 4.2 DATA FLOW DIAGRAM (DFD)

# 

# 

Figure 4.2: Data-Flow Diagram

# 

# 4.3 BEHAVIORAL DIAGRAMS

## 4.3.1 ACTIVITY DIAGRAM

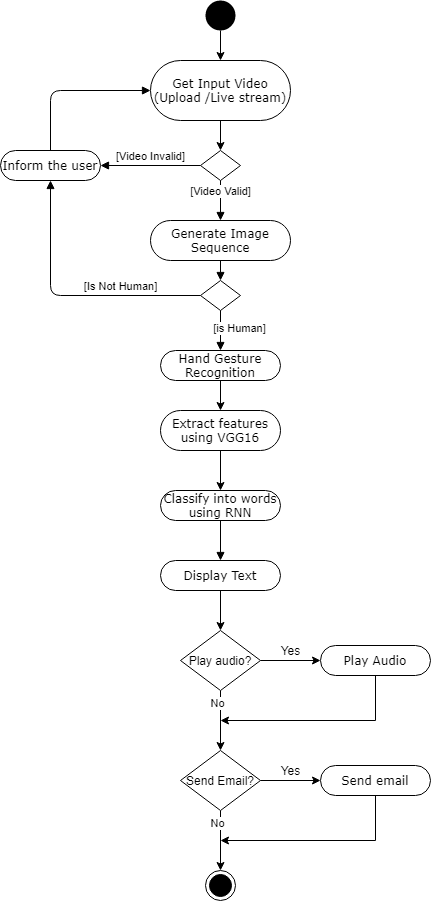


Figure 4.3.1: Activity Diagram

## 4.3.2 SEQUENCE DIAGRAM

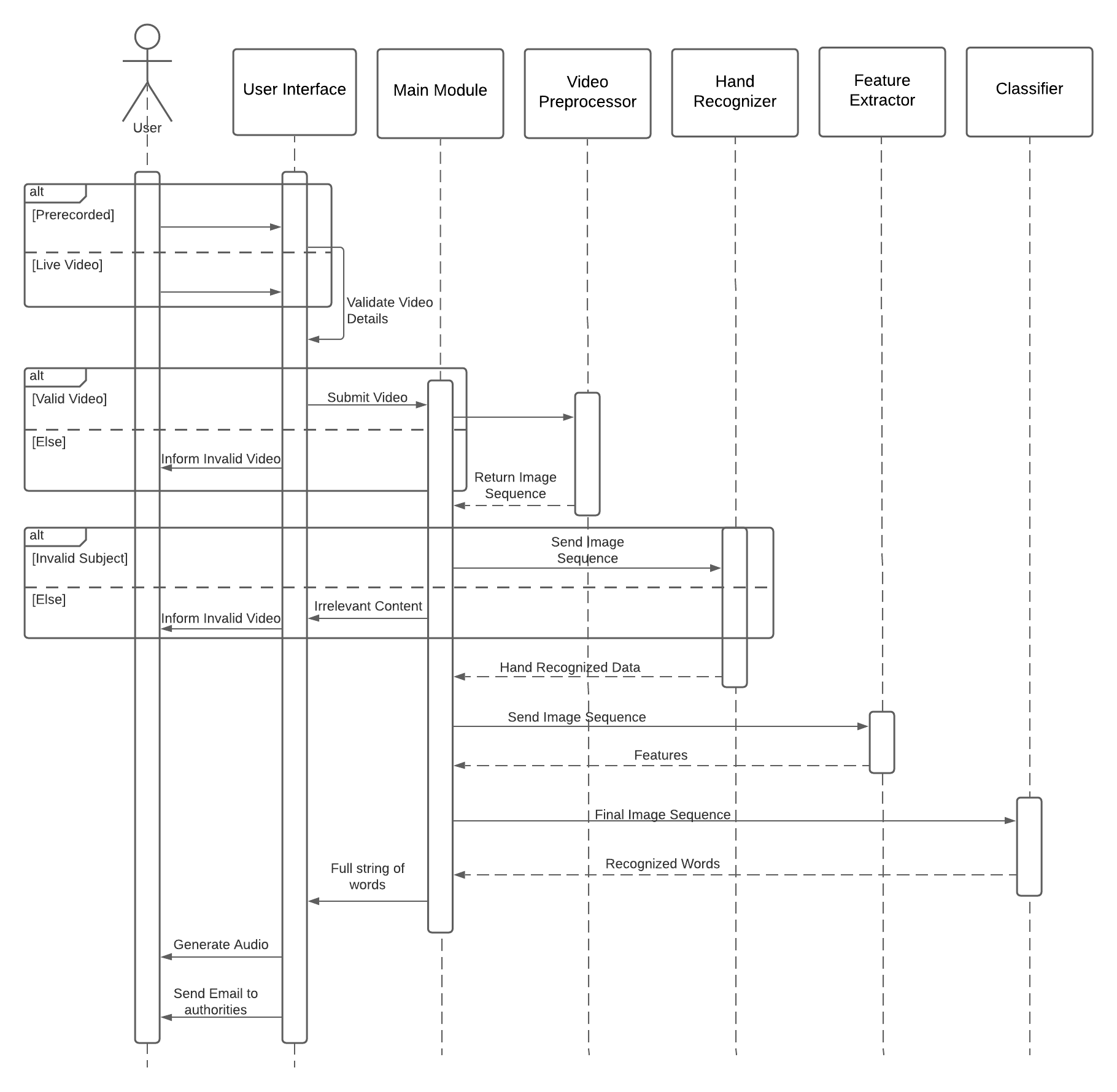


Figure 4.3.2: Sequence Diagram

## 

## 

## 4.3.3 STATE DIAGRAM

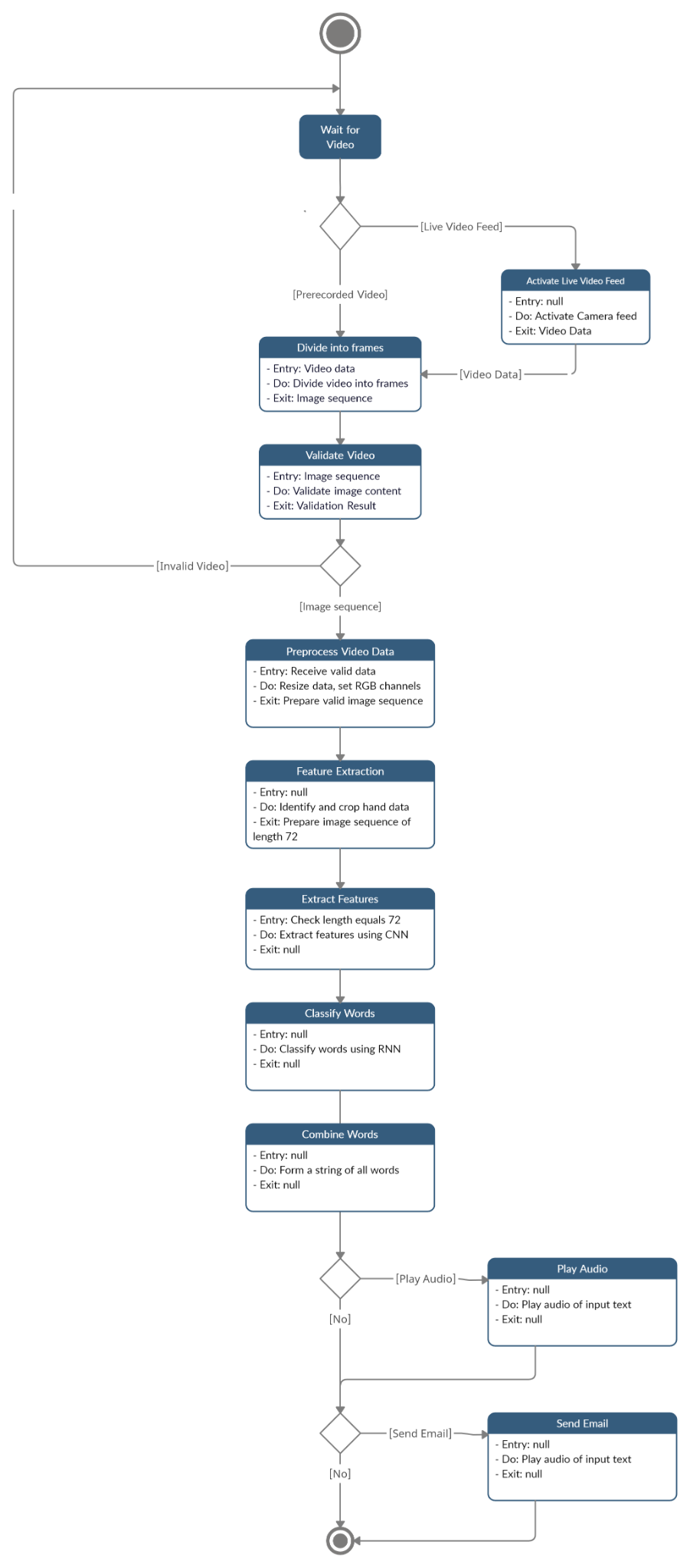


Figure 4.3.3: State Diagram

# 4.4 ALGORITHMS

* **Haar Cascade:** Once, the data is confirmed to be video, a face recognizer is applied on it to make sure that the video actually has a human being and is not some random data e.g. cat video. For this, a Haar Cascade classifier is used which classifies the video frames given to it as ones having human face and others as not. This is an extremely necessary step to verify that the data is actually valid and the model is not working on data that was either misplaced or was purposely given with malicious intent.
* **Hand Detection:** The video files, before being actually passed through the data generator and further processed by the neural network, is first run through a hand detection algorithm. Here, an inference graph, written in tensorflow and saved in a pickle file, is read into the module which efficiently detects hands in the video frames. Similarly, this algorithm is implemented for every frame of every video which is read into the module. The output of this module hence delivers a list of frames consisting of hand movements of the signer of size (64,64,3), which corresponds to the height, width and the number of channels of every frame. These frames are then passed through the data generator to increase the size of the dataset as well introducing variations in the angle, shear, brightness and zoom.
* **Transfer Learning (for feature extraction):**  Transfer learning is a machine learning method where a model developed for a task is reused as the starting point for a model on a second task. Although it can be used as a model, as our problem statement is a video classification task, we used transfer learning as a tool for extracting features from the frames, for which we used the state-of-the-art VGG16 model. Transfer Learning proved to be a far better approach than our previous approach of using ConvLSTM layers which replaces the matrix multiplication operations within LSTM with the convolution operation and thus allows working with sequential data. We tried other state-of-the-art transfer learning models like VGG19, MobileNet, etc. however VGG16’s performance was superior.
* **LSTM (Long Short Term Memory):**  Long short-term memory is a recurrent neural network architecture frequently used in deep learning. LSTM has feedback connections unlike conventional neural networks. We used LSTM for the prediction task as it can efficiently process sequences of data. A LSTM unit comprises an input gate, output gate, forget gate and a cell. The input and output gates control the information flow to and from the unit whereas the cell actually remembers that time series information. The forget gate is used to specify which information needs to be forgotten and which to be retained by the cell. We built a neural network using LSTM which was used for predicting the class i.e. the word being performed by the signer.

**4.5 USER INTERFACE**

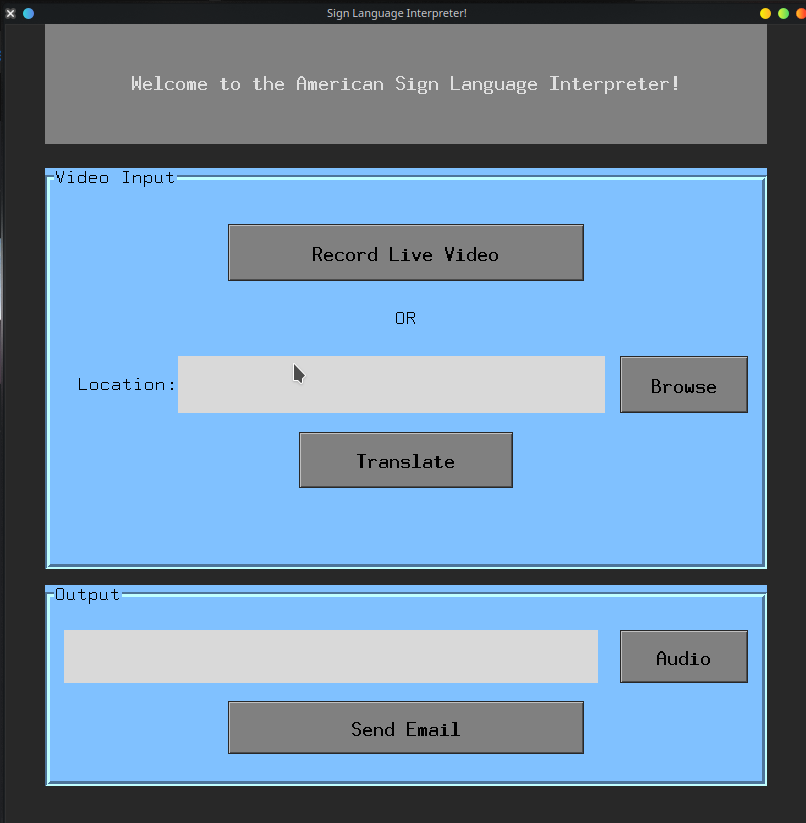
****

Fig. 4.5.1 User Interface

* The front-end of the application is a desktop application. We used Tkinter which is the standard Python binding to the Tk GUI Toolkit. Tkinter is inbuilt with all the leading Operating Systems like Linux, Mac and Windows and hence does not require any additional permissions or dependencies to work. As the main aim of this project is to help the disabled to communicate conveniently, the interface was kept simple and subtle.
* The user has 2 options for video data input : Record a live Video or upload a .mp4 file of a subject performing a Sign in American Sign Language from their local storage.
* After the video data has been selected, either recorded or uploaded , the User clicks translate. The system then runs the data through the Model to predict the signs performed.
* The predicted words are printed in the ‘Output’ in text format. The user can also click on the Audio button to hear the words in an audio format.
* In emergency situations, the speech-impaired cannot call or talk to the emergency authorities on the phone if there is no one near them to speak on the call. To overcome this pertinent challenge, we have added the ‘Send Email’ functionality, which enables the user to record their signs, which are translated by the model, and then send the translated text over Email to emergency authorities. This data is accompanied with the user’s location and picture of the face, cropped out from the video data.

**Actual Working Example**

Following is an example of our team member Vedant using the application and performing 2 words: namely “Please” and “Help”.

|  |  |
| --- | --- |
| Fig 4.5.2: Video frame of “Please” | Fig 4.5.3: Video Frame of “Help” |

The output of performing the live video above can be seen in the snapshot of the UI below:

# 

# Fig. 4.5.4 User Interface Output

CHAPTER 5

**IMPLEMENTATION**

# 5.1 STAGES OF IMPLEMENTATION

## 5.1.1 PREPARATION OF DATA

**5.1.1.1 Dataset Sources :**

* WLASL (Word-Level American Sign Language) Dataset :

## WLASL is the largest video dataset for Word-Level American Sign Language (ASL) recognition, which features 2,000 common different words and 20000+ videos in ASL. The team carried out web-scraping by developing and deploying python scripts that pull official WLASL videos from YouTube and other sources into local storage. These videos are of varying lengths, quality and formats. The WLASL dataset acts as our primary reference dataset.

## Self Recorded Videos :

The WLASL dataset, despite being one of the largest of its kind, has a variable number of videos per word. Hence we decided to record videos of our team members and their family members performing the signs for the following reasons :

1) To balance the number of videos we had per word.

2) To add variety and generalize the data.

3) To boost the amount of training data we had to achieve better results.

## 

**5.1.1.2 Selecting words :**

As the scope of our project is limited to only a small batch of words due to limitations on our computational resources we wanted to work on basic signs that would be used in emergencies situations by the speech impaired community. We decided to work in batches of words with increasing sizes to ensure that our approaches were scalable and robust. Initially working with 9 words, then 15 words and finally 20 words. Listed below are the words we decided to work with and the number of videos we had for each.

| **Word** | **Number of Videos** |
| --- | --- |
| cold | 37 |
| crash | 37 |
| doctor | 34 |
| give | 40 |
| medicine | 36 |
| no | 41 |
| police | 35 |
| woman | 39 |
| yes | 40 |
| animal | 37 |
| child | 37 |
| danger | 33 |
| help | 44 |
| home | 39 |
| kill | 35 |
| please | 40 |
| rob | 39 |
| Send | 33 |
| sick | 38 |
| want | 39 |

Table 5.1.1.2 Dataset Description

## 5.1.2 PREPROCESSING

**5.1.2.1 Frame Duplication, Video Cropper and Resizing**

For our RNN model to train and process our dataset, every video needs to have the same amount of frames. After considering all our signs and collected data, we decided that every video must have 72 frames i.e 3secs video at 24fps, with each frame being of 64\*64 size.

As not every video is exactly 3 seconds long, they needed to either be cropped to 72 frames or have certain frames duplicated to make each video exactly 72 frames. Each video longer than 72 frames had a starting frame assigned to it , which signified the start of the section of the video that had the important gesture in it. For videos which had less than 72 frames, our scripts uniformly duplicated the frames till the video had exactly 72 frames. Each frame is resized to a 64\*64 size.

**5.1.2.2 Data Generator**

The Data Generator performs combinations of scaling, shearing, rotation, and brightness adjustment on all frames of a video to produce similar videos as the original. This was done for the following purposes :

1) It boosts the amount of data the models had to train on.

2) To generalize the model towards variations in camera angles and environment changes.

**5.2 Implementation Issues**

**5.2.1 Computational Resources**

As none of the team members possessed a system with a powerful GPU, which was extremely essential for efficient and fast training of the models, the building, training and testing of the model was executed completely on Google Colab. Google Colab provides a GPU and a 12 GB RAM free of cost and hence the development of the model was possible. However, these resources were still not enough when we tried executing more complex architectures and would run into resource exhaustion problems. Upon availability of more computational resources, there was still some scope for betterment of results.

**5.2.2 Storage Issues**

All the videos, required to be processed by the model, were stored on Google Drive. Google Drive provides a storage space of maximum 15 GB to every individual. A fee needs to be charged for buying more storage. Hence, as a lot of storage was not available, the attempt of scaling the project further was halted as the data would not fit in the available space.

**5.3 Techniques**

**5.3.1 Hand Detection:**

The video files, before being actually passed through the data generator and further processed by the neural network, is first run through a hand detection algorithm. Here, an inference graph, written in tensorflow and saved in a pickle file, is read into the module which efficiently detects hands in the video frames. Similarly, this algorithm is implemented for every frame of every video which is read into the module. The output of this module hence delivers a list of frames consisting of hand movements of the signer of size (64,64,3), which corresponds to the height, width and the number of channels of every frame. These frames are then passed through the data generator to increase the size of the dataset as well introducing variations in the angle, shear, brightness and zoom.

**5.3.2 Transfer Learning (Feature Extraction):**

Transfer learning is a machine learning method where a model developed for a task is reused as the starting point for a model on a second task. Although it can be used as a model, as our problem statement is a video classification task, we used transfer learning as a tool for extracting features from the frames, for which we used the state-of-the-art VGG16 model. Transfer Learning proved to be a far better approach than our previous approach of using ConvLSTM layers which replaces the matrix multiplication operations within LSTM with the convolution operation and thus allows working with sequential data. We tried other state-of-the-art transfer learning models like VGG19, MobileNet, etc. however VGG16’s performance was superior.

**5.3.3 LSTM (Long Short Term Memory):**

Long short-term memory is a recurrent neural network architecture frequently used in deep learning. LSTM has feedback connections unlike conventional neural networks. We used LSTM for the prediction task as it can efficiently process sequences of data. An LSTM unit comprises an input gate, output gate, forget gate and a cell. The input and output gates control the information flow to and from the unit whereas the cell actually remembers that time series information. The forget gate is used to specify which information needs to be forgotten and which to be retained by the cell. We built a neural network using LSTM which was used for predicting the class i.e. the word being performed by the signer.

**5.3.4 Speech Generation:**

This module was provided as aid to the illiterate who cannot read or write. The words predicted by the LSTM model were then passed through a script written in Python which used the playsound library to convert the text to audio. The user can optionally execute this functionality by clicking on the play sound button provided on the user interface.

# 5.4 SOFTWARE TOOLS

* **Frontend:** The front-end of the application is a desktop application. We used Tkinter which is the standard Python binding to the Tk GUI Toolkit. Tkinter is inbuilt with all the leading Operating Systems like Linux, Mac and Windows and hence does not require any additional permissions or dependencies to work. As the main aim of this project is to help the disabled to communicate conveniently, the interface was kept simple and subtle.
* **Backend:** Python provides an extensive amount of support for a vast array of uses from application development to front-end development to machine learning and AI libraries and does so with minimal development time. We have used Python as the backend coding language that is used to record videos, read videos, perform frame extraction and then execute the deep learning models to get predictions. Keras, OpenCV and Numpy were used for building the functionality.
* **Data Storage:** All models during the development phase were built, trained and evaluated on Google Colab. Google Colab was used primarily to leverage the GPU capacity of the platform. Subsequently, Google Drive was used as the data storage for all the videos from where they were read into the Jupyter notebook to process them.

CHAPTER 6

**RESULTS AND EVALUATION**

# 6.1 EXPERIMENTS

**6.1.1 Detailed Discussion of Experiments Carried Out**

* **Convolutional Neural Network**

The data that we obtained after the data generation model was first passed to through the VGG16 model, which is a state-of-the-art convolutional neural network used for image classification and feature extraction tasks. Our primary aim of using the VGG16 model was feature extraction as the VGG16 model itself is used for image classification while our problem required a sequential approach. We removed the last layer of the VGG16 model and replaced it with a Flatten layer that flattens the output to a vector of size 2048. We passed the frames of every video through the resultant model to get a 2048-sized vector from it. These vectors were subsequently used as data for training a stacked LSTM model for prediction of classes i.e. words.

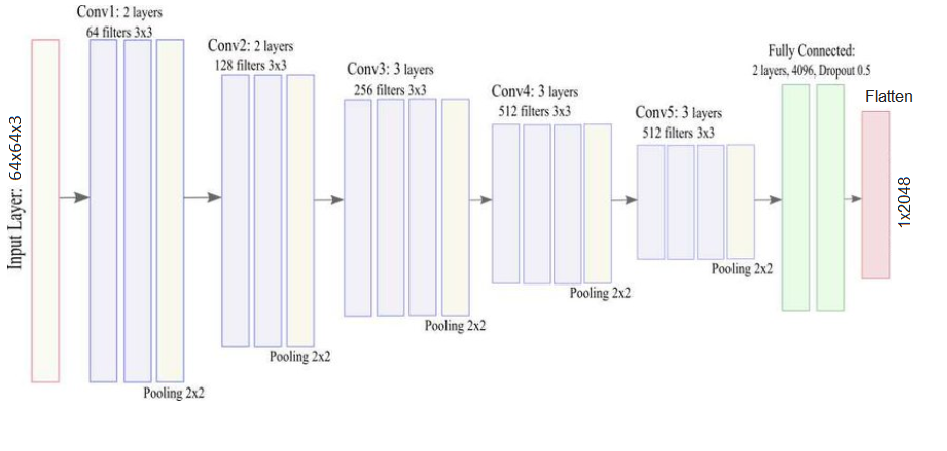


Fig. 6.1.1.1 VGG16 Model Architecture

* **Recurrent Neural Network**

We passed the output of the VGG16 model, which was a 2048-sized vector through a stacked LSTM architecture, consisting of 2 LSTM layers of 200 and 100 units each followed by a dense layer of 100 neurons. This neural network architecture culminated with a final dense layer of 20 neurons, each corresponding to a class in the gloss. The input shape given to this architecture was (72, 2048), where 72 corresponds to the number of frames in a video and 2048 is the size of the vector representation of each video frame. The LSTM model was trained for 50 epochs with a batch size of 8 using the Adam optimizer.

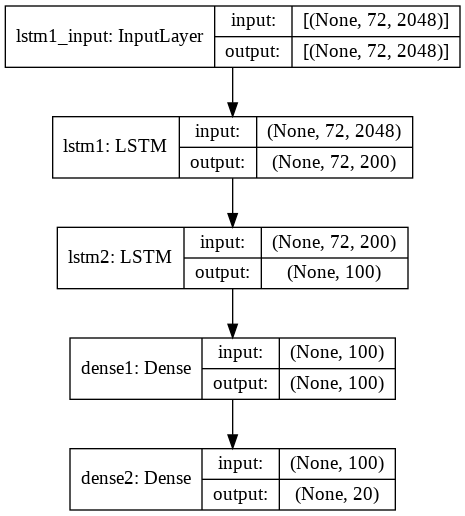


Fig. 6.1.1.2 LSTM Model Architecture

* **Varying number of Words**

Initially, we started working on 9 words that we thought to be the most important ones during times of distress for the disabled. Then we increased the count of words to 15 and ultimately we settled for 20 words. This approach followed to test the scalability of the project with the increase of data and the classes for prediction. Results obtained confirm the fact that the project is in fact scalable and only limited by the availability of computational resources.

**6.1.2 Results**

The results of the experiments carried out have been summarized in the table below. We have provided the result of our deep learning model by using various metrics such as train, test and validation accuracy. Along with that a classification report with the precision, support, f1 score and support for each word has also been given.

| **Number of Words** | **Train Accuracy** | **Validation Accuracy** | **Test Accuracy** |
| --- | --- | --- | --- |
| 9 | 100% | 93.06% | 94% |
| 15 | 100% | 92.16% | 91% |
| 20 | 100% | 90.11% | 89% |

Table 6.1.2.1 Results for Varying Number of Words

Although the training accuracy is 100%, the high values of the validation accuracy and test accuracy prove that overfitting has not occurred. As the accuracy values do not drop sharply after the increase in the number of classes, it can be stated that the model is pretty scalable.

The classification report for the model is as given below :

| **Word** | **Precision** | **Recall** | **F1-Score** | **Support** |
| --- | --- | --- | --- | --- |
| Cold | 0.96 | 0.78 | 0.86 | 32 |
| Crash | 0.97 | 0.97 | 0.97 | 31 |
| Doctor | 0.81 | 0.95 | 0.88 | 22 |
| Give | 0.82 | 0.78 | 0.80 | 36 |
| Medicine | 0.92 | 0.89 | 0.91 | 38 |
| No | 0.83 | 0.94 | 0.88 | 36 |
| Police | 0.80 | 0.77 | 0.78 | 26 |
| Woman | 0.88 | 0.88 | 0.88 | 26 |
| Yes | 0.83 | 1.00 | 0.91 | 24 |
| Animal | 0.93 | 0.90 | 0.91 | 29 |
| Child | 0.92 | 0.85 | 0.88 | 26 |
| Danger | 0.90 | 1.00 | 0.95 | 28 |
| Help | 0.92 | 0.88 | 0.90 | 41 |
| Home | 0.96 | 0.92 | 0.94 | 24 |
| Kill | 0.85 | 0.79 | 0.81 | 28 |
| Please | 0.89 | 0.96 | 0.92 | 25 |
| Rob | 0.94 | 0.94 | 0.94 | 34 |
| Send | 0.82 | 0.88 | 0.85 | 26 |
| Sick | 0.94 | 1.00 | 0.97 | 30 |
| Want | 1.00 | 0.84 | 0.92 | 32 |

Table 6.1.2.2 Classification Report

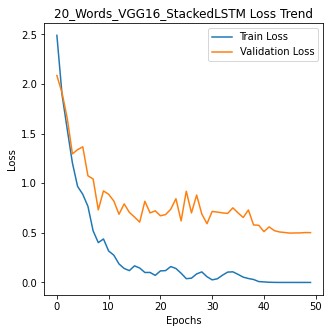


Fig. 6.1.2.1 Loss Trend

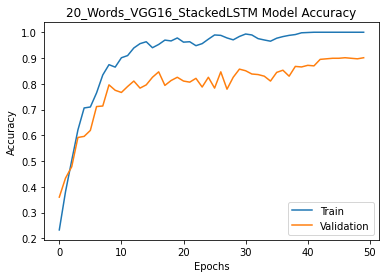


Fig. 6.1.2.2 Accuracy Trend

**6.2 Testing**

* **Checking the type of Uploaded File :**

If the user chooses to upload a file from his/her local storage, the system must validate the type of the file selected. The system in its current form can only work with .mp4 files. Hence, if the user selects any other type of file, there is an error prompt to the user.

* **Checking for presence of a human face in the Video Data :**

The Video data, either in the form of a live stream or .mp4 file selected from the local storage, must contain a human face for the entirety of the video. This ensures that a human is the subject of the video, and invalidates videos with incomprehensible subjects. It must be noted that the face detection is simply used to ensure the presence of a human, the model does not consider facial data in its processing as most of it is disregarded by the hand detection module. Failure to detect face is prompted to the user with an error message.

* **Checking for Hand Movement in the Video Data:**

The Video data, either in the form of a live stream or .mp4 file selected from the local storage, must contain at least a single frame with hands visible in it. This ensures that only data with hands which is used to predict the sign is presented to the model. If hands are not detected for the entire video, the user is prompted with an error message.

CHAPTER 7

**CONCLUSION**

# 7.1 CONCLUSION

We have proposed a system for the deaf and mute who primarily communicate using sign languages with others. This system records a video of the signer, segregates the video into word videos, extracts frames from the videos, detects hand movements and then predicts the word being said by the signer by using a Convolutional Neural Network followed by LSTM RNN. A total of 20 words can be identified by the system. The system also allows the user to generate audio output of the final word sequence that is obtained as the output of the video of the signer.

# 

# 7.2 LIMITATIONS

* Currently, the system will work only for the American Sign Language.
* Only 20 words can be successfully recognized by the system owing to the dearth of computational resources and data storage.
* The system facilitates only one-way communication - interpreting sign language and generating natural spoken language.

# 

# 7.3 FUTURE SCOPE

* This project can further be extended to facilitate two-way communication between the Mute and the Society - sign language to spoken language and vice versa.
* All the words of the sign language can be inculcated to institute a general conversation.
* An all inclusive platform can be made for all sign languages spoken worldwide to provide a single solution for effective communication.

REFERENCES

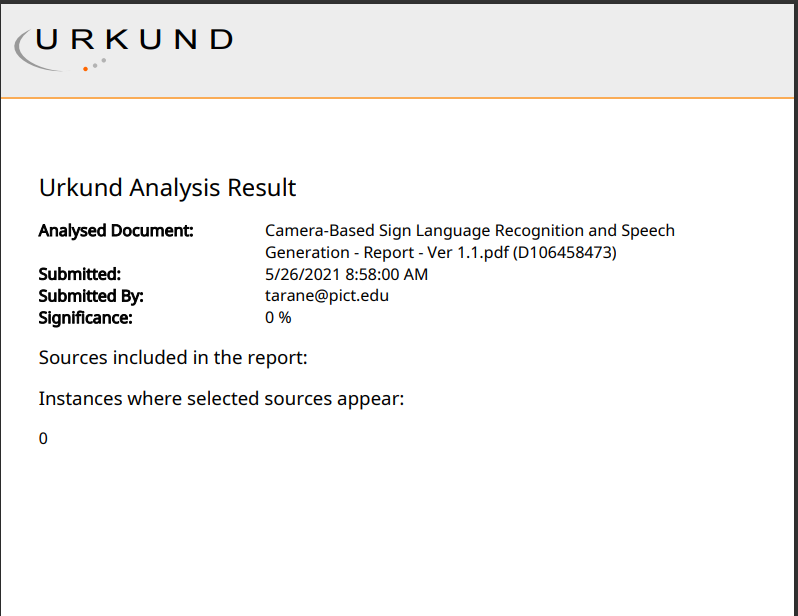
| [1] | Sun C, Zhang T, Bao BK, Xu C (2013a) Latent support vector machine for sign language recognition with Kinect. In: 20th IEEE international conference on image processing (ICIP), pp 4190–4194 |
| --- | --- |
| [2] | Savur C, Sahin F (2016) American Sign Language recognition system by using surface EMG signal. In: IEEE international conference on systems, man, and cybernetics (SMC), pp 002872–002877 |
| [3] | S. S. Fels and G. E. Hinton, “Glove-talk: a neural network interface between a data-glove and a speech synthesizer,” IEEE Transactions on Neural Networks, vol. 4, no. 1, pp. 2–8, 1993. |
| [4] | Oz C, Leu MC (2011) American Sign Language word recognition with a sensory glove using artificial neural networks. Eng Appl Artif Intell 24(7):1204–1213 |
| [5] | Chuan CH, Regina E, Guardino C (2014) American Sign Language recognition using leap motion sensor. In: 13th IEEE international conference on machine learning and applications (ICMLA), pp 541–544 |
| [6] | AlQattan D, Sepulveda F (2017) Towards sign language recognition using EEG-based motor imagery brain computer interface. In: 5th IEEE international winter conference on brain–computer interface (BCI), pp 5–8 |
| [7] | N. A. Mehta, T. Starner, M. M. Jackson, K. O. Babalola, and G. A. James, "Recognizing Sign Language from Brain Imaging, " in 2010 20th International Conference on Pattern Recognition (ICPR), 2010, pp. 3842-3845. |
| [8] | M. G. Bleichner and N. F. Ramsey, "Give Me a Sign: Studies on the Decodability of Hand Gestures Using Activity of the Sensorimotor Cortex as a Potential Control Signal for Implanted Brain Computer Interfaces, " in Brain-Computer Interface Research, C. Guger, T. Vaughan, and B. Allison, Eds. Springer International Publishing, 2014, pp. 7-17. |
| [9] | Wadhawan, A., Kumar, P. Sign Language Recognition Systems: A Decade Systematic Literature Review. *Arch Computational Methods Eng* (2019). https://doi.org/10.1007/s11831-019-09384-2 |
| [10] | Rewari, Hardik & Dixit, Vishal & Batra, Dhroov & Nagaraja, Hema. (2018). Automated Sign Language Interpreter. 1-5. 10.1109/IC3.2018.8530658. |
| [11] | Rajaganapathy, S. & Aravind, B. & Keerthana, B. & Sivagami, M.. (2015). Conversation of Sign Language to Speech with Human Gestures. Procedia Computer Science. 50. 10.1016/j.procs.2015.04.004. |
| [12] | Kaur, Sarabjeet and V. Banga. “Vision Based Static Hand Pose, Hand Movement Recognition System For Sign Language Using EigenVector Theory in MATLAB.” viXra(2014). |
| [13] | Ragab, Amira & Ahmed, Maher & Chau, Siu-Cheung. (2013). Sign Language Recognition Using Hilbert Curve Features. 7950. 143-151. 10.1007/978-3-642-39094-4\_17. |
| [14] | Tangsuksant, Watcharin & Adhan, Suchin & Pintavirooj, Chuchart. (2014). American Sign Language recognition by using 3D geometric invariant feature and ANN classification. 1-5. 10.1109/BMEiCON.2014.7017372. |
| [15] | M. M. Islam, S. Siddiqua and J. Afnan, "Real time Hand Gesture Recognition using different algorithms based on American Sign Language," 2017 IEEE International Conference on Imaging, Vision & Pattern Recognition (icIVPR), Dhaka, 2017, pp. 1-6, doi: 10.1109/ICIVPR.2017.7890854. |
| [16] | Oyedotun, Oyebade & Khashman, Adnan. (2017). Deep learning in vision-based static hand gesture recognition. Neural Computing and Applications. 28. 10.1007/s00521-016-2294-8. |
| [17] | Karayılan T, Kılıç Ö (2017) Sign language recognition. In: IEEE international conference on computer science and engineering (UBMK), pp 1122–1126 |
| [18] | Saha S, Lahiri R, Konar A, Nagar AK (2016) A novel approach to American Sign Language recognition using MAdaline neural network. In: IEEE symposium series on computational intelligence (SSCI), pp 1–6 |
| [19] | A. Neyra-Gutiérrez and P. Shiguihara-Juárez, "Feature Extraction with Video Summarization of Dynamic Gestures for Peruvian Sign Language Recognition," *2020 IEEE XXVII International Conference on Electronics, Electrical Engineering and Computing (INTERCON)*, 2020, pp. 1-4, doi:10.1109/INTERCON50315.2020.9220243. |
| [20] | K. Bantupalli and Y. Xie, "American Sign Language Recognition using Deep Learning and Computer Vision," 2018 IEEE International Conference on Big Data (Big Data), 2018, pp. 4896-4899, doi: 10.1109/BigData.2018.8622141. |
| [21] | N. Sarhan and S. Frintrop, "Transfer Learning For Videos: From Action Recognition To Sign Language Recognition," 2020 IEEE International Conference on Image Processing (ICIP), 2020, pp. 1811-1815, doi: 10.1109/ICIP40778.2020.9191289. |
| [22] | S. He, "Research of a Sign Language Translation System Based on Deep Learning," 2019 International Conference on Artificial Intelligence and Advanced Manufacturing (AIAM), 2019, pp. 392-396, doi: 10.1109/AIAM48774.2019.00083 |
| [23] | S. S Kumar, T. Wangyal, V. Saboo and R. Srinath, "Time Series Neural Networks for Real Time Sign Language Translation," 2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA), 2018, pp. 243-248, doi: 10.1109/ICMLA.2018.00043. |

**APPENDICES**

**Base Paper**

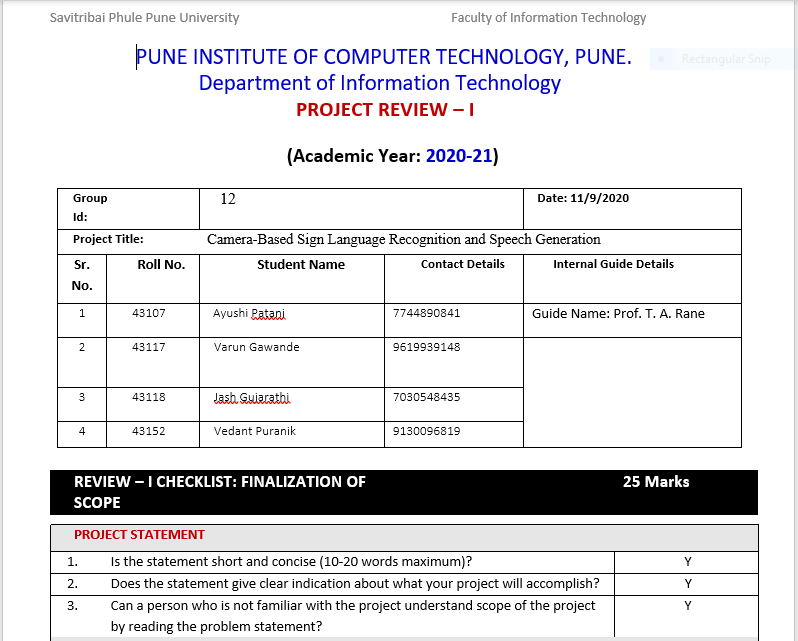
S. Demidenko et al., "Developing Automatic Markerless Sign Language Gesture Tracking and Recognition System," 2019 IEEE International Symposium on Haptic, Audio and Visual Environments and Games (HAVE), Subang Jaya, Malaysia, 2019, pp. 1-6, doi: 10.1109/HAVE.2019.8921358.

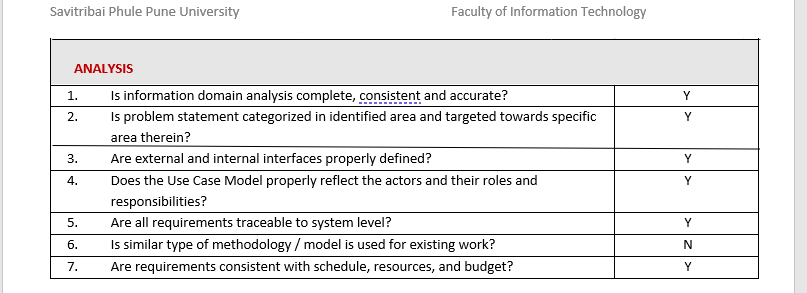
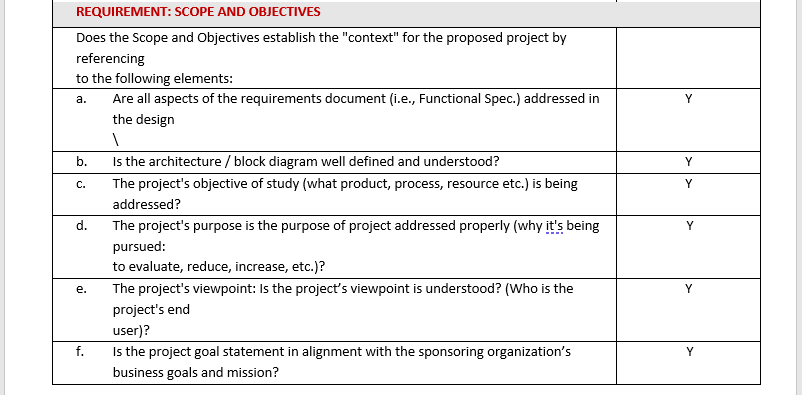
**Plagiarism Report**



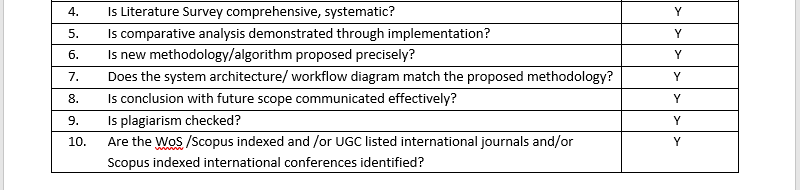
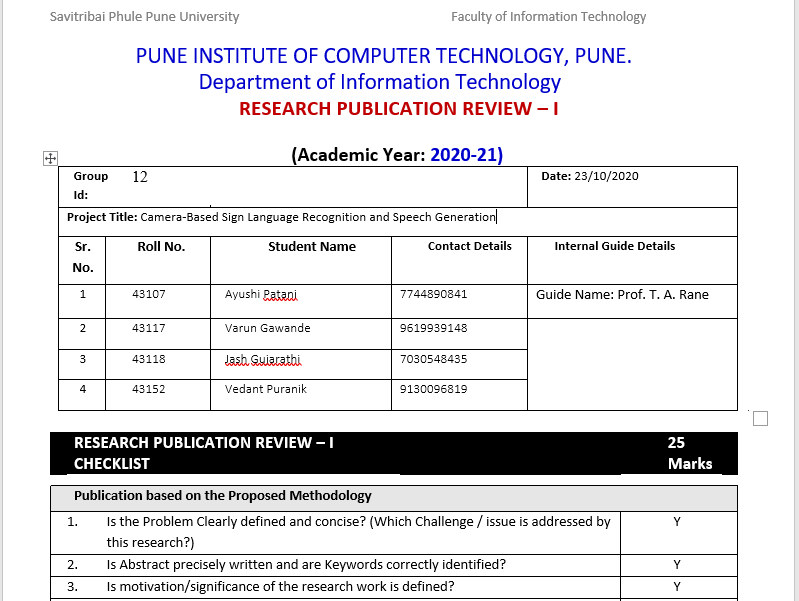
**Review Sheets (Semester 7)**

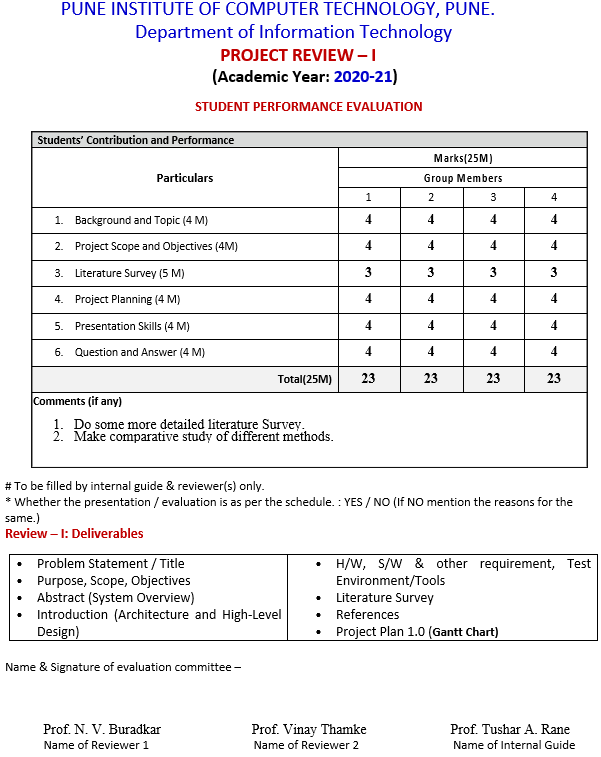
Project Review 1:

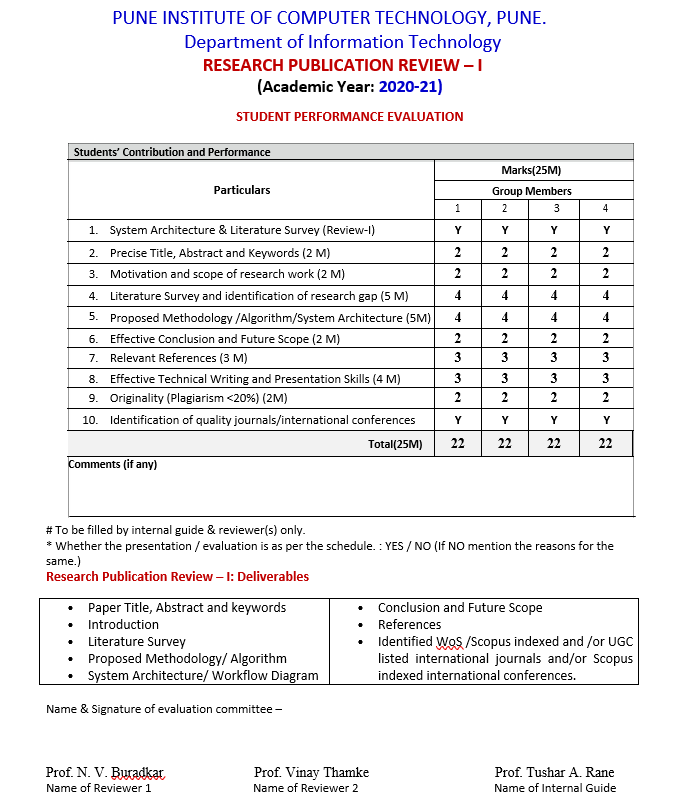




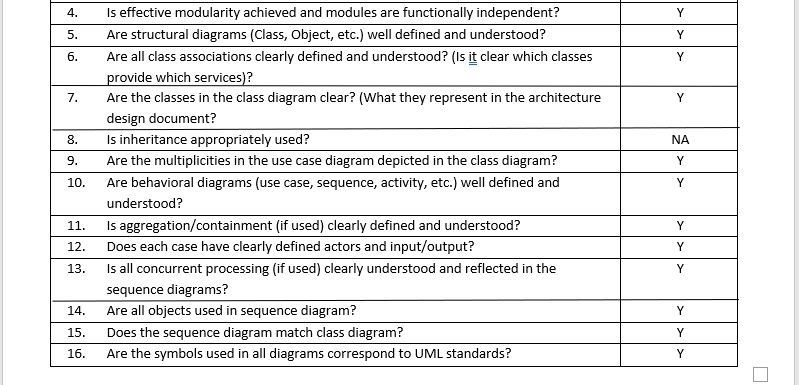
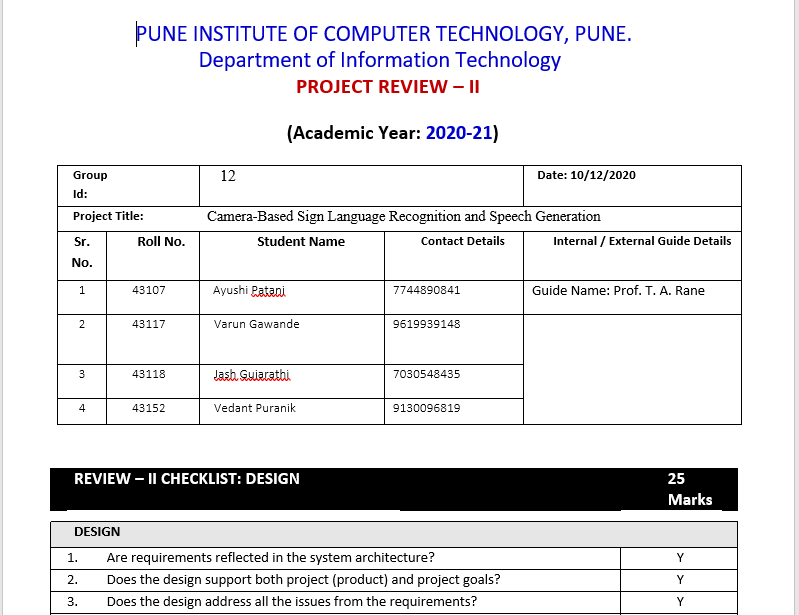
Paper Publication Review:

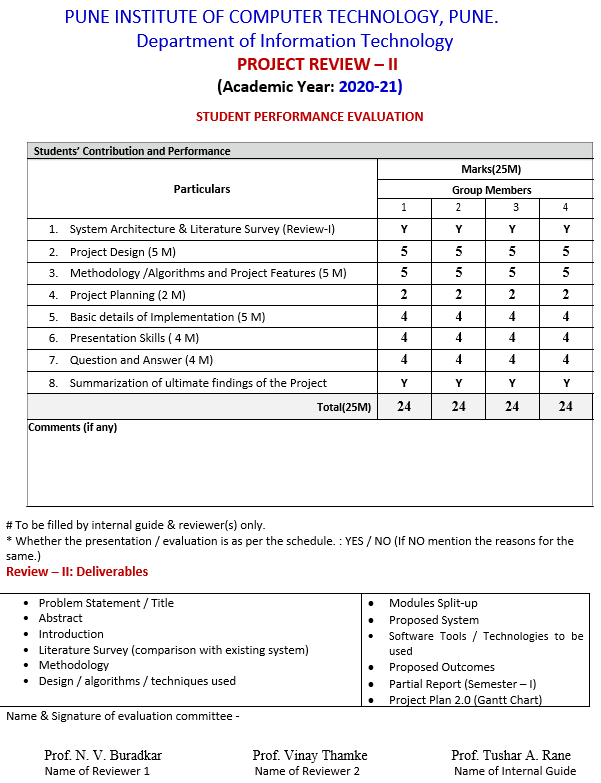






Project Review 2:





PUNE INSTITUTE OF COMPUTER TECHNOLOGY, PUNE.

Department of Information Technology

**(Academic Year: 2020-21)**

**Semester – I**

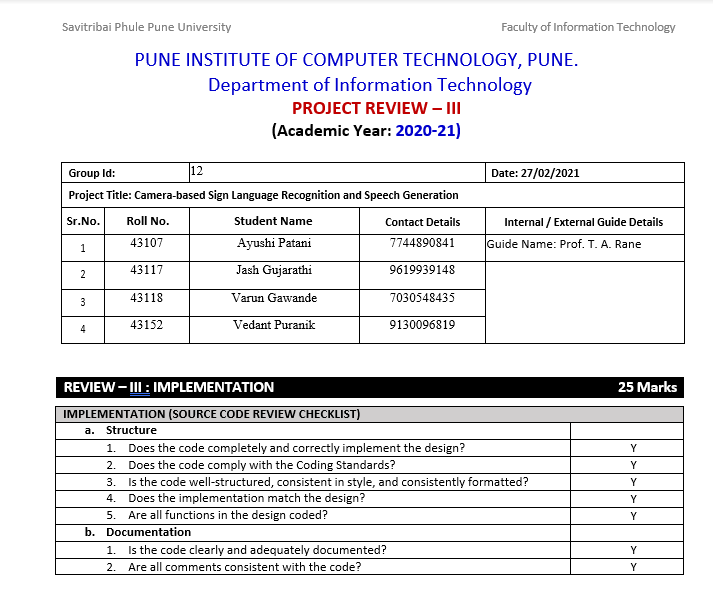
**Monthly Planning Sheet**

**Academic Year: 2020-2021**

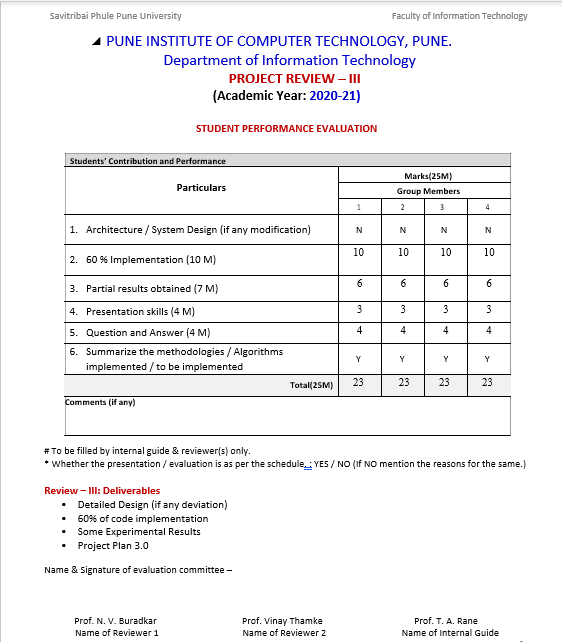
| **Week No.** | **Activity Planned** | **Activity Completed Status** | **Student Signature** | **Guide Signature** |
| --- | --- | --- | --- | --- |
| **Week 1** | To select project domain | Completed |  |  |
| **Week 2** | To approach companies for sponsorship | Completed |  |
| **Week 3** | To decide problem statement and its scope | Completed |  |
| **Week 4** | To deliver Project-Review 1 | Completed |  |
| **Week 5** | To collect project-related research papers | Completed |  |  |
| **Week 6** | To learn basic Python | Completed |  |
| **Week 7** | To deliver Research-Publication Review 1 | Completed |  |
| **Week 8** | To gather information from all research papers | Completed |  |
| **Week 9** | To identify necessary technologies | Completed |  |  |
| **Week 10** | To implement web scraping to collect data | Completed |  |
| **Week 11** | To write a survey paper based on the project topic | Completed |  |
| **Week 12** | To deliver Project-Review 2 | Completed |  |

**Review Sheets (Semester 8)**

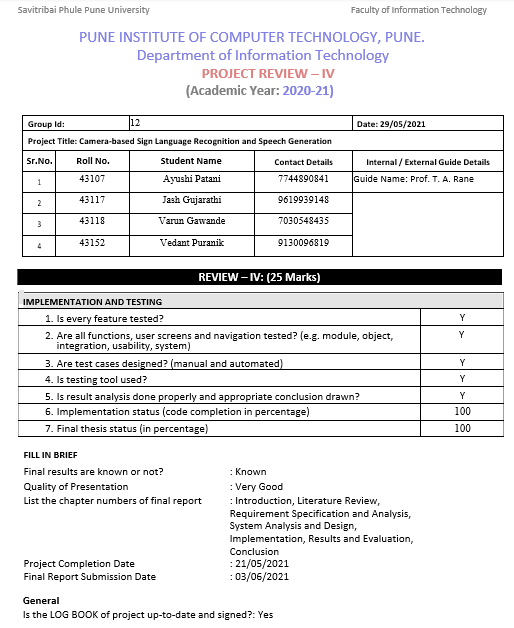
**Project Review 3:**

****

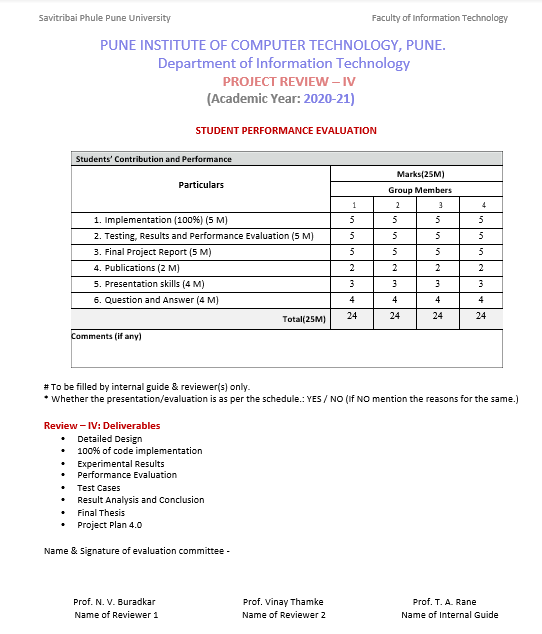
**Project Review 3 Performance Evaluation:**

****

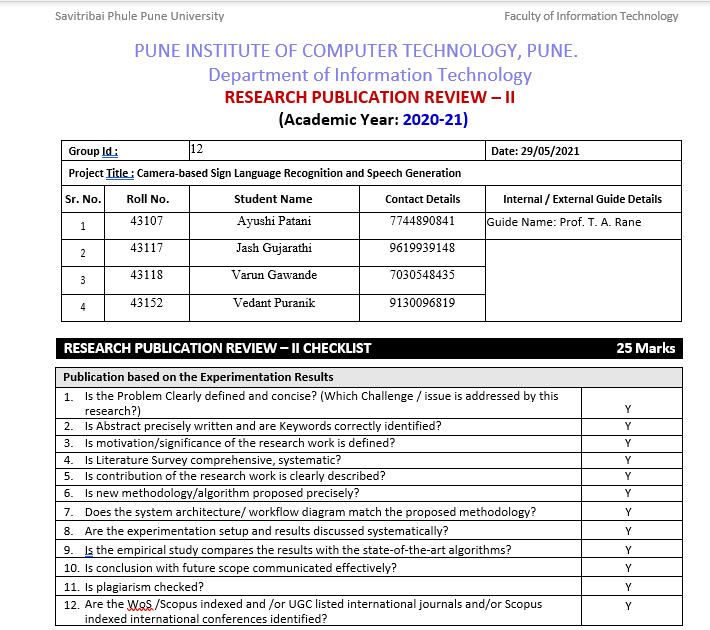
**Project Review 4:**

****

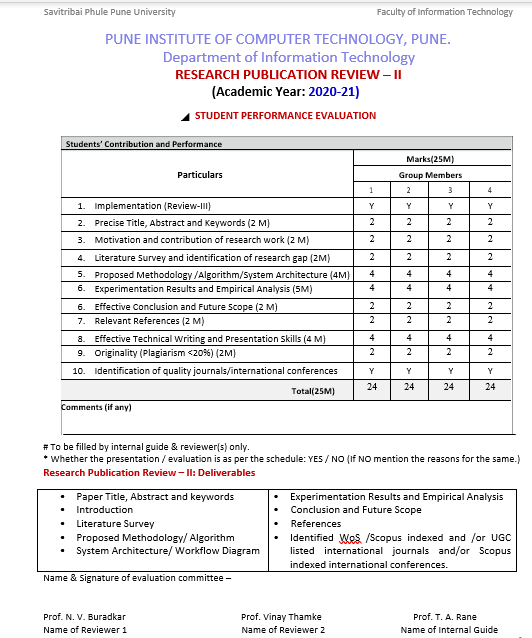
**Project Review 4 Performance Evaluation:**

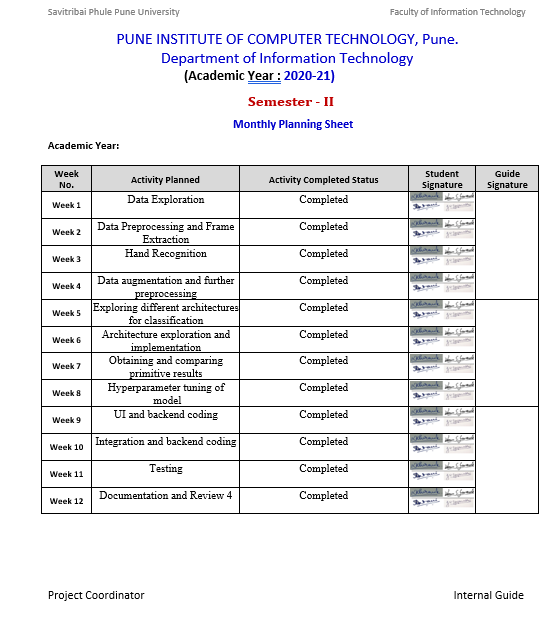
****

**Research Publication Review 2:**

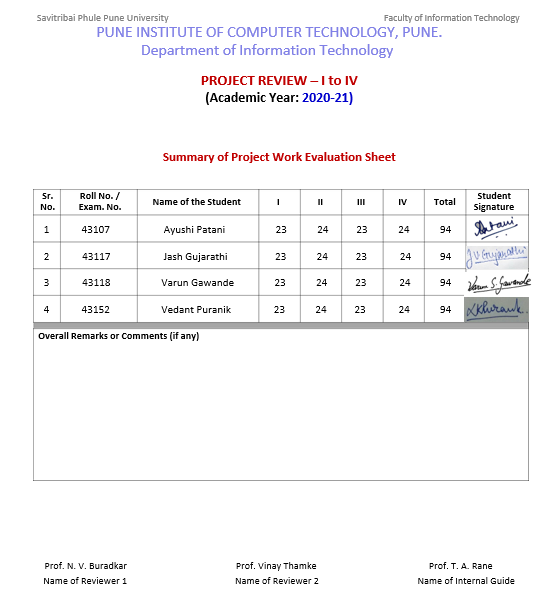
****

**Research Publication Review 2 Student Performance Evaluation:**

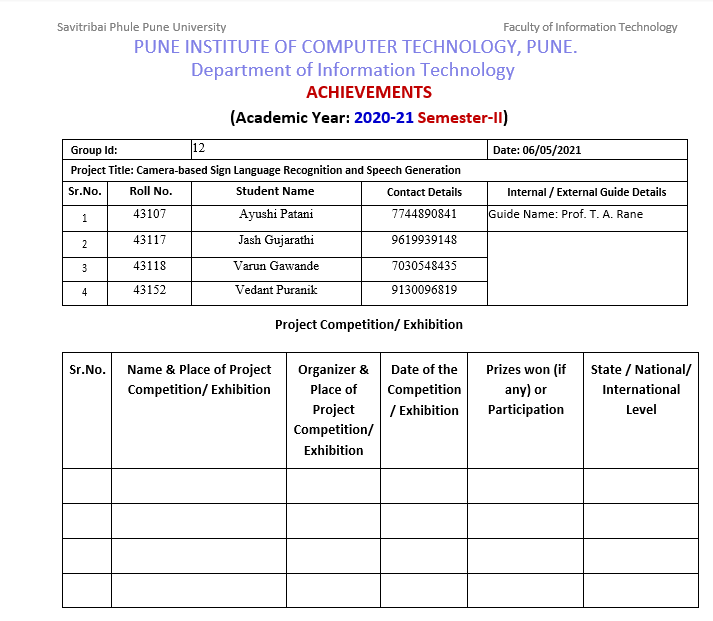
****

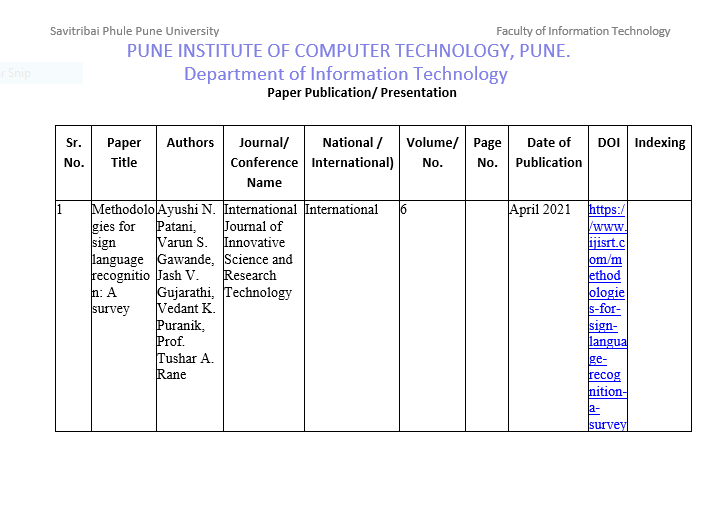
****

**Summary of Project Work:**

****

**Achievements:**

****

****

**Prof. T. A. Rane**

**Certificates**

****