Sign-to-Speech

A PROJECT REPORT

Submitted by

Aaditya More (21BAI10014)

Chhavi Mohitkar (21BAI10367)

Meet Joysar (21BAI10063)

Vitthal Dubey(21BAI10142)

Rishita Bansal (21BAI10336)

in partial fulfillment for the award of the degree
of

BACHELOR OF TECHNOLOGY
in
COMPUTER SCIENCE AND ENGINEERING-SPECIALIZATION IN AIML

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING

VIT BHOPAL UNIVERSITY

KOTHRIKALAN, SEHORE

MADHYA PRADESH - 466114

OCTOBER 2022

VIT BHOPAL UNIVERSITY, KOTHRIKALAN, SEHORE
MADHYA PRADESH – 466114

BONAFIDE CERTIFICATE

Certified that this project report titled “ **Sign-to-Speech**” **is the bonafide work of “** Chhavi Mohitkar (Register No :21BAI10367), Aaditya More (Register No :21BAI10014), Meet Joysar (Register No :21BAI10063), Vitthal Dubey(Register No :21BAI10142), Rishita Bansal (Register No :21BAI10336)” **who carried out the project work under my supervision. Certified further that to the best of my knowledge the work reported at this time does not form part of any other project/research work based on which a degree or award was conferred on an earlier occasion on this or any other candidate.**

6/10/22

PROGRAM CHAIR
Dr. Suthir Sriram, Senior Assistant Professor,
Program Chair (AIML)
School of Computer Science and Engineering
VIT BHOPAL UNIVERSITY

6/10/22

PROJECT GUIDE
Dr. Suthir Sriram, Senior Assistant Professor,
Program Chair (AIML)
School of Computer Science and Engineering
VIT BHOPAL UNIVERSITY

The Project Exhibition I Examination is held on 6th Oct, 2022.

ACKNOWLEDGEMENT

First and foremost, I would like to thank the Lord Almighty for His presence and immense blessings throughout the project work.

I wish to express my heartfelt gratitude to Dr. Suthir Sriram, Head of the Department, School of Computer & Engineering-AIML for much of his valuable support encouragement in carrying out this work.

I would like to thank my internal guide Dr. Suthir Sriram, for continually guiding and actively participating in my project, giving valuable suggestions to complete the project work.

I would like to thank all the technical and teaching staff of the School of Aeronautical Science, who extended directly or indirectly all support.

Last, but not least, I am deeply indebted to my parents who have been the greatest support while I worked day and night for the project to make it a success.

ABSTRACT

The purpose of this project is to enable video-to-audio conversion by helping-hard-to-speak people to communicate over voice calls. This project focuses on the use of Indian Sign Language rather than American sign Language so as to inculcate ease of speech among the citizens of the country. The methodology behind this project is the training of Machine Learning algorithms which will identify the gestures from phone camera and then join them to make sentences which will be delivered as output over the phone call. There are existing methods for video-to-text and text-to-speech conversions which have been proven useful to some extent but if we combine both of them, we can make a great impact on the society.

TABLE OF CONTENTS

CHAPTER-1:
PROJECT DESCRIPTION AND OUTLINE

1.1 Introduction

Indian Sign Language (ISL) is a sign language used by hearing and speech impaired people to communicate with other people. The research presented in this paper pertains to ISL as defined in the Talking Hands website [1]. ISL uses gestures for representing complex words and sentences. It contains 33 hand poses including 10 digits, and 23 letters. Amongst the letters in ISL, the letters ‘h’, ‘j’ are represented by gestures and the letter ‘v’ is similar to digit 2. The system is trained with the hand poses in ISL as shown in Fig. 1. Most people find it difficult to comprehend ISL gestures. This has created a communication gap between people who understand ISL and those who do not. One cannot always find an interpreter to translate these gestures when needed. To facilitate this communication, a potential solution was implemented which would translate hand poses and gestures from ISL in real-time. It comprises of an Android smartphone camera to capture hand poses and gestures, and a server to process the frames received from the smartphone camera. The purpose of the system is to implement a fast and accurate recognition technique. The system described in this paper successfully classifies all the 33 hand poses in ISL. For the initial research, gestures containing only one hand was considered. The solution described can be easily extended to two-handed gestures. In the next section of this paper, the related work pertaining to sign language translation is discussed.

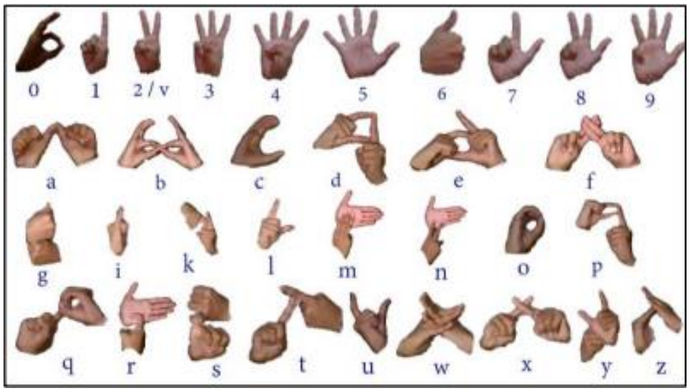


Fig.1 Alphabets in Sign-languages

1.2 Problem Statement

Difficulties faced by speech-impaired people to talk over a normal phone call.

1.3 Objective of the work

Sign language is the basic communication method used by hearing impaired people. Hand gestures are used for communication purpose and these people face problems in communicating with other people without a translator. The proposed system aims to fill the communication gap using machine learning by creating a system in which hand gesture can be converted into text using CNN algorithm.

1.4 Summary

In this project, we could achieve accurate prediction once we started testing using a white background. The other issue that people might face is regarding their proficiency in knowing the ISL gestures. Bad gesture postures will not yield correct prediction. This project can be enhanced in a few ways in the future, it could be built as a mobile application for the users to conveniently access the project, also, the existing project not only works for ISL, it can be extended to work for other native sign languages with enough dataset and training. This project implements a finger spelling translator, however, sign languages are also spoken in a contextual basis where each gesture could represent an object, verb, so, identifying this kind of a contextual signing would require a higher degree of processing and natural language processing (NLP).

CHAPTER-2:
RELATED WORK INVESTIGATION

2.1 Introduction

There has been considerable work in the field of Sign Language recognition with novel approaches towards gesture recognition. Different methods such as use of gloves or Microsoft Kinect sensor for tracking hand, etc. have been employed earlier. A study of many different existing systems has been done to design a system that is efficient and robust than the rest.

2.2 Existing Approaches/Methods

2.2.1 Approaches/Methods -1

A Microsoft Kinect sensor is used in [2] for recognising sign languages. The sensor creates depth frames; a gesture is viewed as a sequence of these depth frames.

2.2.2 Approaches/Methods -2

T. Pryor et al [3] designed a pair of gloves, called Sign Aloud which uses embedded sensors in gloves to track the position and movement of hands, thus converting gestures to speech.

2.2.3 Approaches/Methods -3

R. Hait-Campbell et al [4] developed Motion Savvy, a technology that uses Windows tablet and Leap Motion accelerator AXLR8R to recognize the hand, arm skeleton.

2.2.4 Approaches/Methods -4

Sceptre [5] uses Myo gesture-control armbands that provide accelerometer, gyroscope and electromyography (EMG) data for signs & gestures classification. These hardware solutions IEEE - 43488 9th ICCCNT 2018 July 10-12, 2018, IISC, Bengaluru Bengaluru, India provide good accuracy but are usually expensive and are not portable. Our system eliminates the need of external sensors by relying on an Android phone camera

2.2.5 Approaches/Methods -5

Now for software-based solutions, there are coloured glove based [6, 7] and skin colour-based solutions. R. Y. Wang et al [6] have used multi-coloured glove for accurate hand pose reconstruction but the sign demonstrator, while demonstrating the sign language, has to wear this each time

2.2.6 Approaches/Methods -6

Skin colour-based solutions may use RGB colour space with some motion cues [8] or HSV [9, 10, 11], YCrCb [12] colour space for luminosity invariance.

2.3 Issues/observations from investigation

All the existing methods are somewhere lacking precision therefore latest technologies need to be used to increase the precision percentage of trained models.

CHAPTER-3:
REQUIREMENT ARTIFACTS

3.1 Introduction

Creating a desktop application that uses a computer’s webcam to capture a person signing gestures for American sign language (ISL), and translate it into corresponding text and speech in real time. The translated sign language gesture will be acquired in text which is farther converted into audio. In this manner we are implementing a finger spelling sign language translator. To enable the detection of gestures, we are making use of a -nearest neighbour network (KNN). A NN is highly efficient in tackling computer vision problems and is capable of detecting the desired features with a high degree of accuracy upon sufficient training.

3.2 Hardware and Software requirements

deep learn-knn-image-classifier

Tensorflow

3.3 Specific Project requirements

3.3.1 Data requirement

The **startWebcam** function along with **trainingbtns** are used to collect data which is hereby the images we collect which are further given as input for KNN model.

3.3.2 Class requirement

The **Main** class is responsible for altering page elements on the user interface such as buttons, video elements, etc. It is also handling the training, prediction, and video call features.

The **PredictionOutput** class converts the predicted text passed by Main into text, image, and audio output. This class is also responsible for turning a caller's words into speech in video call mode.

3.3.4 Other Requirements

The KNN Classifier used for this project was created by Google TensorFlow.

The KNN classifier requires the computation of random numbers that is not readily available on JavaScript.

To accomplish this, the work of Johannes Baagøe on "implementations of Randomness in JavaScript" was used.

Additionally, usage of TensorFlow was learned from Abhishek Singh's "Alexa-sign- language-translator".

CHAPTER-4:

DESIGN METHODOLOGY AND ITS NOVELTY

4.1 Methodology and goal

Creating a desktop application that uses a computer’s webcam to capture a person signing gestures for Indian sign language (ISL), and translate it into corresponding text and speech in real time. The translated sign language gesture will be acquired in text which is farther converted into audio. In this manner we are implementing a finger spelling sign language translator. To enable the detection of gestures, we are making use of a Convolutional neural network (KNN). A KNN is highly efficient in tackling computer vision problems and is capable of detecting the desired features with a high degree of accuracy upon sufficient training.

4.2 Functional modules design and analysis

i. Initialize Translator: This function starts the webcam and initial training process. It also loads the KNN classifier.

ii. Startwebcam: This function sets up the webcam

iii. initial Training: This function initializes the training for Start and Stop Gestures. It also sets a click listener for the next button.

iv. loadKNN: This function loads the KNN classifier.

v. initial Gestures: This creates the training and clear buttons for the initial Start and Stop gesture. It also creates the Gesture Card.

vi. setupTrainingUI: This function sets up the custom gesture training UI.

vii. createTrainingBtns: This creates the training and clear buttons for the new gesture. It also creates the Gesture Card.

viii. initialize Training: This function starts the training process.

ix. Train: This function adds examples for the gesture to the KNN model.

x. createTranslateBtn: This function creates the button that goes to the Translate Page. It also initializes the UI of the translate page and starts or stops prediction on click.

xi. setUpTranslation: This function stops the training process and allows users to copy text on the click of the translation text.

xii. Predict: This function predicts the class of the gesture and returns the predicted text if it’s above a set threshold.

xiii. pausePredicting: This function pauses the predict method.

xiv. stopTraining: This function stops the training process.

xv. createVideoCallBtn: This function displays the button that start video call.

xvi. setStatusText: This function sets the status text.

xvii. populateVoiceList: Checks if speech synthesis is possible and if selected voice is available.

xviii. textOutput: This function outputs the word using text and gesture cards.

xix. copyTranslation: It copies the translated text to the user's clipboard.

xx. Speak: This function speaks out the user's gestures. In video call mode, it speaks out the other user's words.

4.3 Software Architectural designs

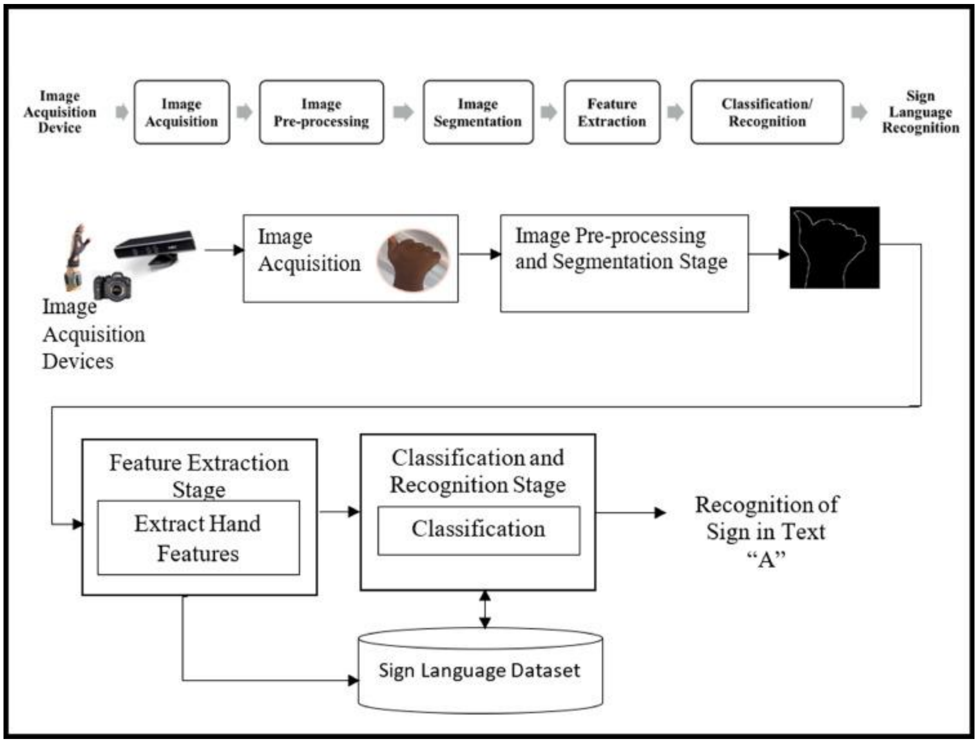
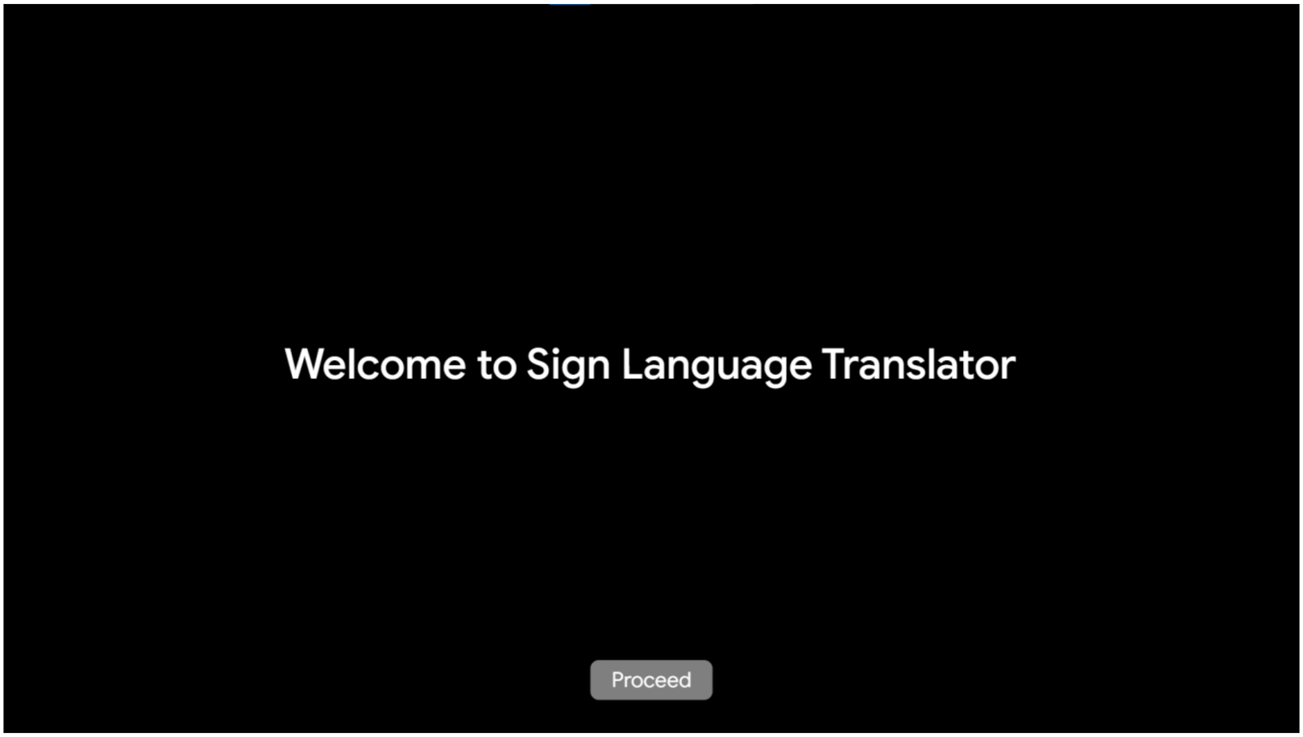


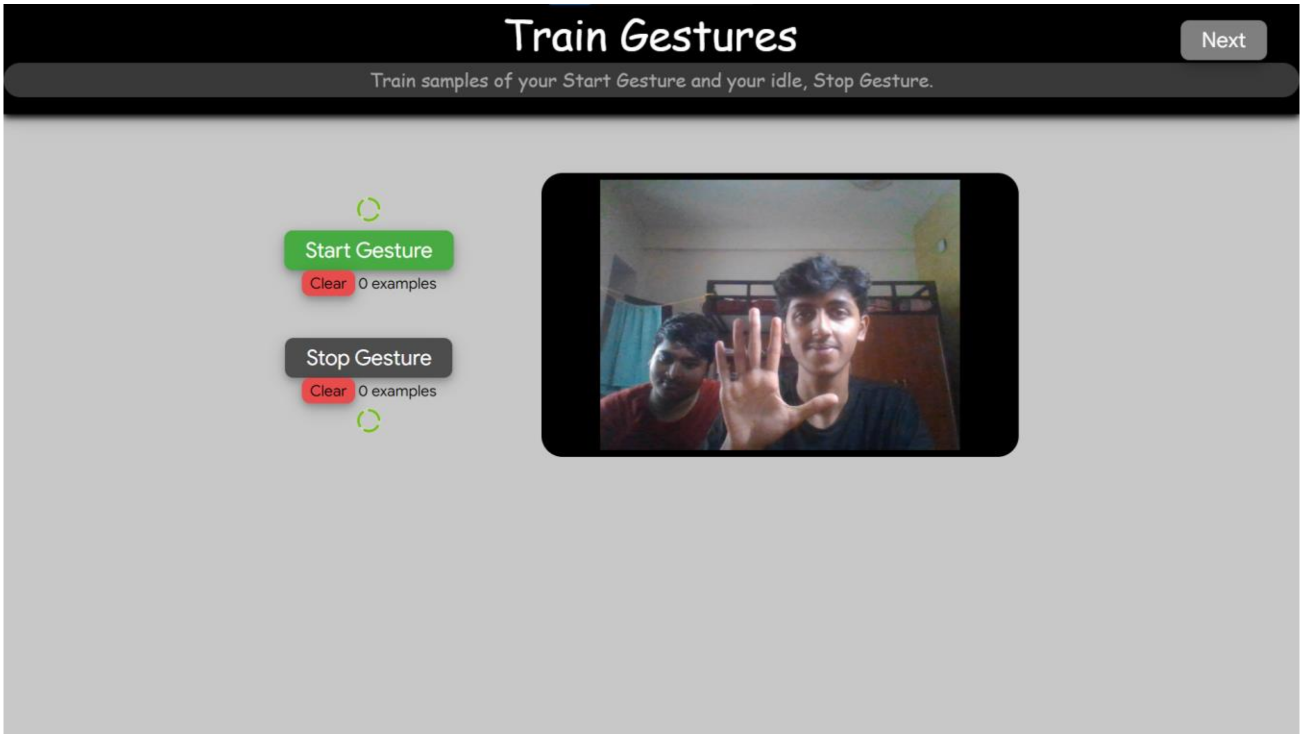
Fig.4 Software Architectural designs

4.4 User Interface designs



Welcome to Sign Language Translator

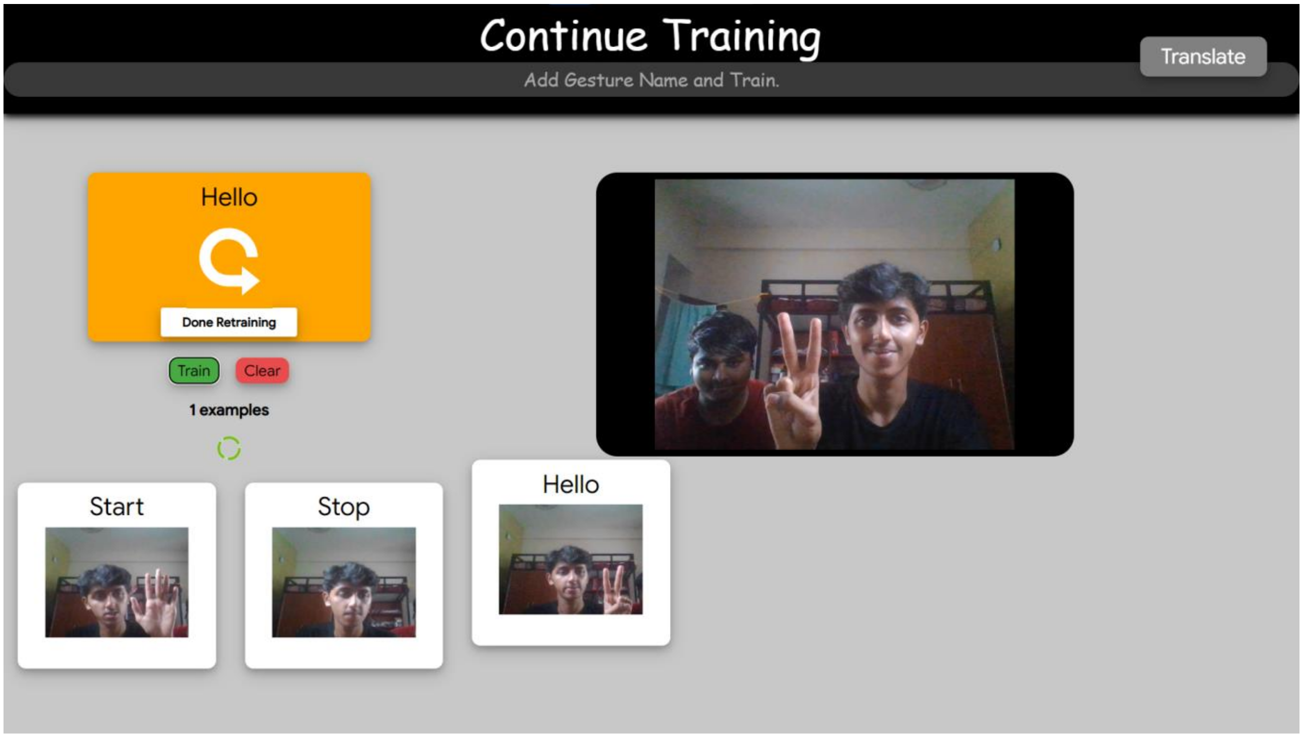
Fig.4a Homepage for Sign-to-speech



Train Gestures

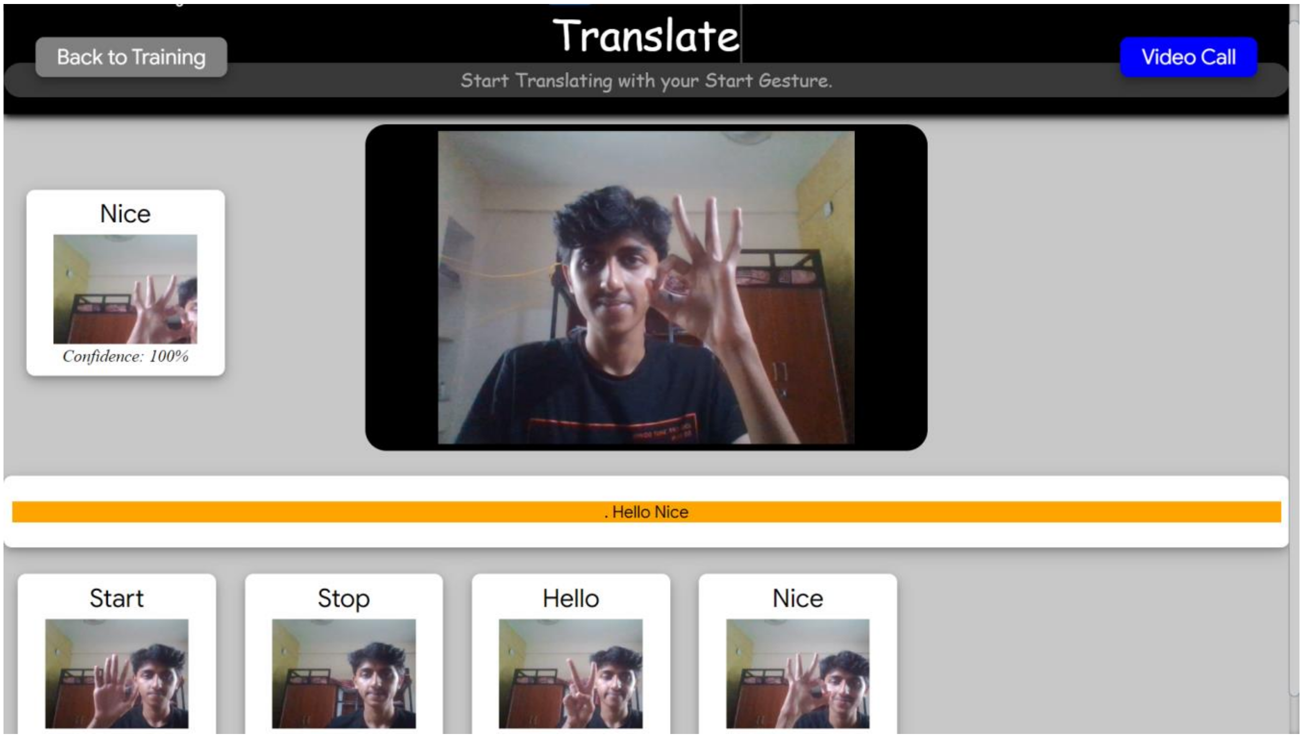
Train samples of your Start Gesture and your idle, Stop Gesture.

Fig.4b Start-Stop train gesture training page


Continue Training

Add Gesture Name and Train.

Fig.4c other signs and gesture training page


Translate

Start Translating with your Start Gesture.

. Hello Nice

Fig.4d output page

CHAPTER-5:

TECHNICAL IMPLEMENTATION & ANALYSIS

5.1 Module Workflow & Explanations

An overview of TensorFlow.js APIs. TensorFlow.js is powered by WebGL and provides a high-level layer’s API for defining models, and a low-level API for linear algebra and automatic differentiation. TensorFlow.js supports importing TensorFlow SavedModels and Keras models.

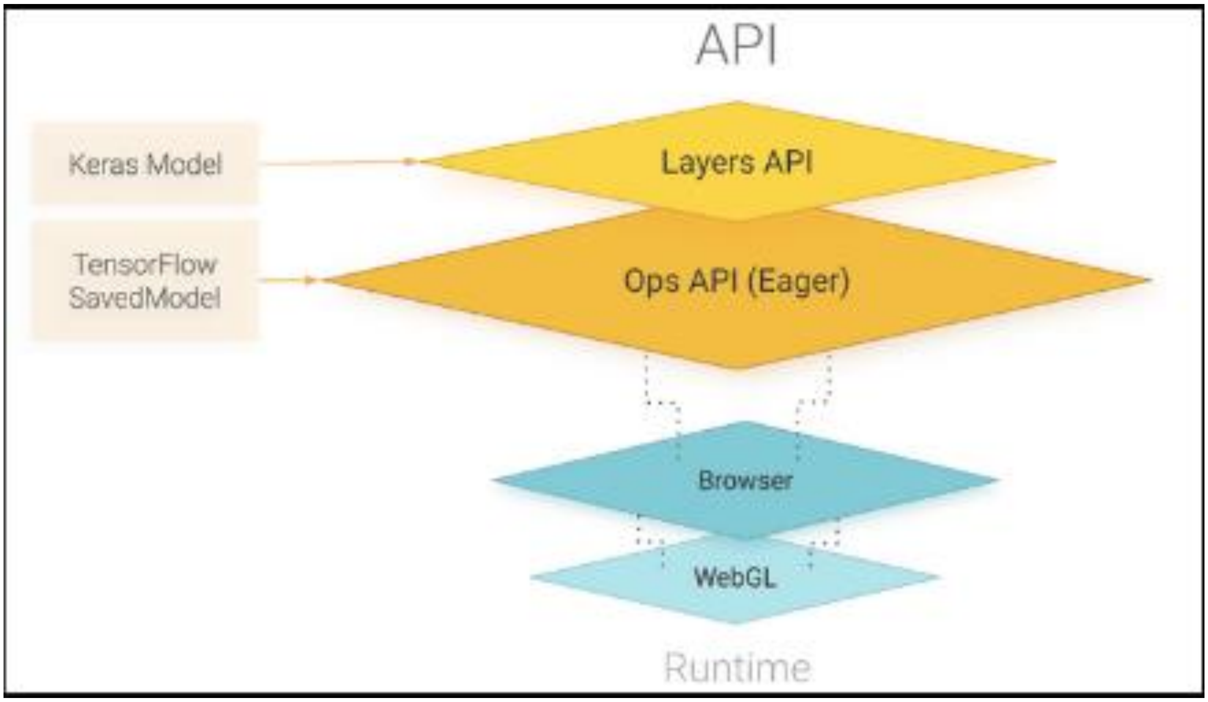


Fig.5a module workflow

5.2 Working Layout of forms

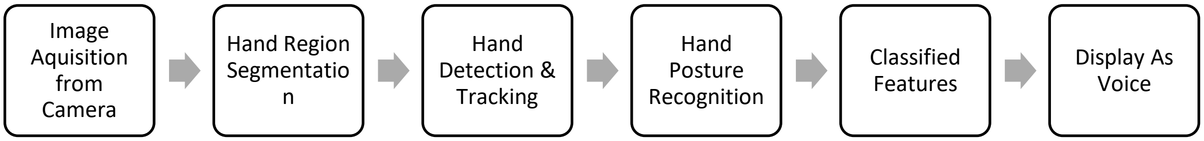


Fig.5b working layout model

5.3 Test and validation

Real time sign language conversion to text and Start:

S1: Set the hand histogram to adjust with the skin complexion and the lighting conditions.

S2: Apply data augmentation to the dataset to expand it and therefore reduce the overfitting.

S3: Split the dataset into train, test and validation data sets.

S4: Train the KNN model to fit the dataset.

S5: Generate the model report which includes the accuracy, error and the confusion matrix.

S6: Execute the prediction file – this file predicts individual gestures, cumulates them into words, displays the words as text

S7 relays the voice output. Stop

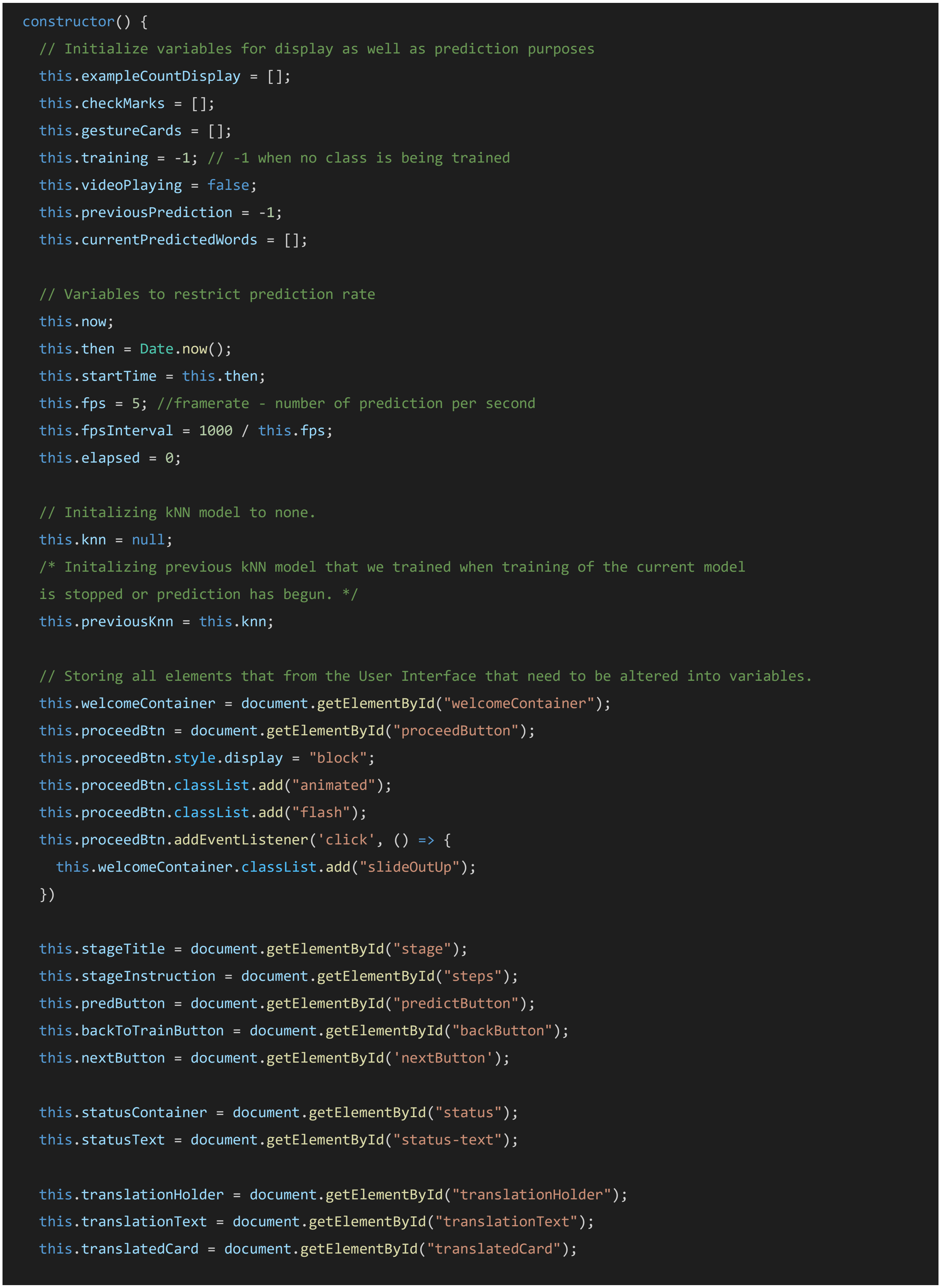
5.4 Code and Training Model:

```
// Importing the k-Nearest Neighbors Algorithm
import {
  KNNImageClassifier
} from 'deeplearn-knn-image-classifier';
import * as dl from 'deeplearn';

// Webcam Image size. Must be 227.
const IMAGE_SIZE = 227;
// K value for KNN. 10 means that we will take votes from 10 data points to classify each tensor.
const TOPK = 10;
// Percent confidence above which prediction needs to be to return a prediction.
const confidenceThreshold = 0.98

// Initial Gestures that need to be trained.
// The start gesture is for signalling when to start prediction
// The stop gesture is for signalling when to stop prediction
var words = ["start", "stop"];

/*
The Main class is responsible for the training and prediction of words.
It controls the webcam, user interface, as well as initiates the output of predicted words.
*/
class Main {
```



```
constructor() {
  // Initialize variables for display as well as prediction purposes
  this.exampleCountDisplay = [];
  this.checkMarks = [];
  this.gestureCards = [];
  this.training = -1; // -1 when no class is being trained
  this.videoPlaying = false;
  this.previousPrediction = -1;
  this.currentPredictedWords = [];

  // Variables to restrict prediction rate
  this.now;
  this.then = Date.now();
  this.startTime = this.then;
  this.fps = 5; //framerate - number of prediction per second
  this.fpsInterval = 1000 / this.fps;
  this.elapsed = 0;

  // Initalizing kNN model to none.
  this.knn = null;
  /* Initalizing previous kNN model that we trained when training of the current model
  is stopped or prediction has begun. */
  this.previousKnn = this.knn;

  // Storing all elements that from the User Interface that need to be altered into variables.
  this.welcomeContainer = document.getElementById("welcomeContainer");
  this.proceedBtn = document.getElementById("proceedButton");
  this.proceedBtn.style.display = "block";
  this.proceedBtn.classList.add("animated");
  this.proceedBtn.classList.add("flash");
  this.proceedBtn.addEventListener('click', () => {
    this.welcomeContainer.classList.add("slideOutUp");
  })

  this.stageTitle = document.getElementById("stage");
  this.stageInstruction = document.getElementById("steps");
  this.predButton = document.getElementById("predictButton");
  this.backToTrainButton = document.getElementById("backButton");
  this.nextButton = document.getElementById('nextButton');

  this.statusContainer = document.getElementById("status");
  this.statusText = document.getElementById("status-text");

  this.translationHolder = document.getElementById("translationHolder");
  this.translationText = document.getElementById("translationText");
  this.translatedCard = document.getElementById("translatedCard");
  this.initialTrainingHolder = document.getElementById('initialTrainingHolder');

  this.videoContainer = document.getElementById("videoHolder");
  this.video = document.getElementById("video");

  this.trainingContainer = document.getElementById("trainingHolder");
  this.addGestureTitle = document.getElementById("add-gesture");
  this.plusImage = document.getElementById("plus_sign");
  this.addWordForm = document.getElementById("add-word");
  this.newWordInput = document.getElementById("new-word");
  this.doneRetrain = document.getElementById("doneRetrain");
  this.trainingCommands = document.getElementById("trainingCommands");

  this.videoCallBtn = document.getElementById("videoCallBtn");
  this.videoCall = document.getElementById("videoCall");

  this.trainedCardsHolder = document.getElementById("trainedCardsHolder");

  // Start Translator function is called
  this.initializeTranslator();

  // Instantiate Prediction Output
  this.predictionOutput = new PredictionOutput();
}

/*This function starts the webcam and initial training process. It also loads the kNN
classifier*/
initializeTranslator() {
  this.startWebcam();
  this.initialTraining();
  this.loadKNN();
}
```

CHAPTER-6:
PROJECT OUTCOME AND APPLICABILITY

6.1 Project applicability on Real-world applications

The project is a simple demonstration of how KNN can be used to solve computer vision problems with an extremely high degree of accuracy. A finger spelling sign language translator is obtained which has an accuracy of 95%.

The project can be extended to other sign languages by building the corresponding dataset and training the KNN. Sign languages are spoken more in context rather than as finger spelling languages, thus, the project is able to solve a subset of the Sign Language translation problem.

CHAPTER-7:

CONCLUSIONS AND RECOMMENDATION

7.1 Limitation/Constraints of the System

i. The thresh needs to be monitored so that we don’t get distorted grayscales in the frames.

ii. Proficiency in knowing the ISL gestures.

7.2 Future Enhancements

The main objective has been achieved, that is, the need for an interpreter has been eliminated. There are a few finer points that need to be considered when we are running the project. The thresh needs to be monitored so that we don’t get distorted grayscales in the frames.

If this issue is encountered, we need to either reset the histogram or look for places with suitable lighting conditions. We could also use white background to eliminate the problem of varying the signee.

In this project, we could achieve accurate prediction once we started testing using a white background. The other issue that people might face is regarding their proficiency in knowing the ISL gestures. Bad gesture postures will not yield correct prediction. This project can be enhanced in a few ways in the future, it could be built as a mobile application for the users to conveniently access the project, also, the existing project not only works for ISL, it can be extended to work for other native sign languages with enough dataset and training.

This project implements a finger spelling translator, however, sign languages are also spoken in a contextual basis where each gesture could represent an object, verb, so, identifying this kind of a contextual signing would require a higher degree of processing and natural language processing (NLP).

References

[1] TalkingHands.co.in, “Talking Hands,” 2014. [Online]. Available: http://www.talkinghands.co.in/. [Accessed: 21- Jul- 2017].

[2] A. Agarwal and M. K. Thakur, “Sign Language Recognition using Microsoft Kinect,” Sixth International Conference on Contemporary Computing (IC3), September 2013.

[3] MailOnline, ‘'SignAloud gloves translate sign language gestures into spoken English,” 2016. [Online]. Available: http://www.dailymail.co.uk/sciencetech/article-3557362/SignAloudgloves-translate-sign-language-movements-spoken-English.html . . [Accessed: 10- Feb- 2018].

[4] Alexia. Tsotsis, “MotionSavvy Is A Tablet App That Understands Sign Language,” 2014. [Online]. Available: https://techcrunch.com/2014/06/06/motionsavvy-is-a-tablet-app-thatunderstands-sign-language . [Accessed: 10 – Feb- 2018].

[5] P. Paudyal, A. Banerjee and S. K. S. Gupta, “SCEPTRE: a Pervasive, Non-Invasive, and ProgrammableGesture Recognition Technology,” Proceedings of the 21st International Conference on Intelligent User Interfaces, pp. 282-293, 2016.

[6] R. Y. Wang and J. Popovic, “Real-Time Hand-Tracking with a Color Glove,” ACM transactions on graphics (TOG), vol. 28, no. 3, 2009.

[7] R. Akmeliawati , M. P. L. Ooi and Y. C. Kuang, “Real-Time Malaysian Sign Language Translation using Colour Segmentation and Neural Network,” Instrumentation and Measurement Technology Conference Proceedings, 2007.

[8] F. S. Chen, C. M. Fu and C. L. Huang, "Hand gesture recognition using a real-time tracking method and hidden Markov models", Image and vision computing, vol. 21, pp. 745-758, 2003.

[9] M. A. Rahaman , M. Jasim, M. H. Ali and M. Hasanuzzaman, “RealTime Computer Vision-Based Bengali Sign Language Recognition,” 17th International Conference on Computer and Information Technology (ICCIT), 2014.

[10] S. Padmavathi, M. S. Saipreethy and V. Valliammai, “Indian Sign Language Character Recognition using Neural Networks,” IJCA Special Issue on Recent Trends in Pattern Recognition and Image Analysis RTPRIA, 2013.

[11] A. Chaudhary, J. L. Raheja and S. Raheja, “A Vision based Geometrical Method to find Fingers Positions in Real Time Hand Gesture Recognition,” JSW, pp. 861-869, 2012.

[12] A. B. Jmaa, W. Mahdi, Y.B. Jemaa and A.B. Hamadou, “Hand Localization and Fingers Features Extraction: Application to Digit Recognition in Sign Language,” International Conference on Intelligent Data Engineering and Automated Learning, pp. 151-159, 2009.