ANALYSIS OF HAND GESTURES AND SIGN LANGUAGE

Abhiram Sharma, Peri Vishwanadha Sastry, Hemanth Srivathsav and Vinay Kumar.

KL University, Aziz Nagar, 500075, Hyderabad, Telangana, India.

Abstract
Hand gestures are a strong medium of communication for hearing impaired society. It helps establish interaction between humans and computers. In this paper, we proposed a continuous Sign Language Gesture Translation System this serves as the key to overcoming many difficulties and provides ease in human life. The mechanical ability of human perception functions and their meaning can be used in many applications.
This paper provides a complete overview of the state-of-the-art techniques used for the latest hand gestures and signatures and Language recognition Research. Here are the updated strategies appropriately divided into different categories: data acquisition, pre-processing, classification, feature extraction, and editing, in which various algorithms in each category are present. Specified and compared with their suitability.
All in all, it is hoped that the study can give students a complete introduction to the field of automation and sign language recognition and simplify future research efforts in this area.

Keywords: Hand Gestures, Sign Recognition.

1 Introduction

2 Related Work

2.1 Literature Survey

Table 2.1: Literature Survey

S.no	Authors	Title	Publishing	Techniques & dataset	Pros	Cons

1.	Pansare, J.R, Gawande et al.	Real-Time Static Hand Gesture Recognition for ASL in Complex Background.	Scientific Research - An Academic Publisher	T: RGB Threshold and edge detection D: A-Z hand gesture	90.19% Recognition rate.	Only 26 static gestures.
2.	Pan et al.	Real-time Sign Language recognition in complex background.	IEEE Publishing	T: SVM D: SIFT, Hu-moments, and FD	->99.8% Accuracy with CSL ->94% Accuracy with ASL	Based on the Hierarchical clustering Classification Method
3.	Xiao Yan Wu et al.	Hand Gesture Recognition algo based on DC-CNN.	Springer Link Publishers.	T: Canny Operator edge detection (DC-CNN) D: NAO Cam Hand posture database (NCD)	98.02% Recognition rate.	Complex CNN layers and parameters.

4.	Tsoli, A.; Argyros et al.	Joint 3D tracking of a deformable object in interaction with hand.	European Conference on Computer Vision.	T: Bounding Box around the hand & hand Mask. D: Synthetic Dataset by blender modeling software.	Interaction with deformable objects and tracking.	% of template vertices overall frames.
5.	Javier Molina, Jose Antonio Pajuelo & Jose M. Martinez et al.	Real-time Motion-based Hand Gestures Recognition from Time of Flight.	Journal of Signal Processing Systems -Springer Link Publishers.	T: Depth info, motion patterns. D: cardinal direction Dataset.	Interaction with virtual environments	Depth range Limitation

3 Proposed Work

3.1 Model & Techniques

The following are the Models & techniques we tried to implement in our project.

3.1.1 CNN

Convolutional Neural Network is a special type of neural network that roughly imitates human vision. Deep learning is a class of deep neural networks, most applied to analyze visual imagery. It uses a special technique called Convolution. Now in mathematics convolution is a mathematical operation on two functions that produces a third function that expresses how the shape of one is modified by the other.

Convolutional neural networks consist of multiple layers of artificial neurons. Artificial neurons (called neurons as they imitate the human brain which consists of neurons) are mathematical functions that calculate the weighted sum of multiple inputs and output an activation value. When an image is given as an input in a Convolutional Network, each layer generates several activation functions that are passed on to the next layer.

- We convert our input images (RGB) into grayscale and apply gaussian blur to remove unnecessary noise. We apply the adaptive threshold to extract our hand from the background and resize our images to 640x640.
- We feed the input images after pre-processing to our model for testing after applying all the operations mentioned above.
- Estimating how likely the image will fall under one of the classes.

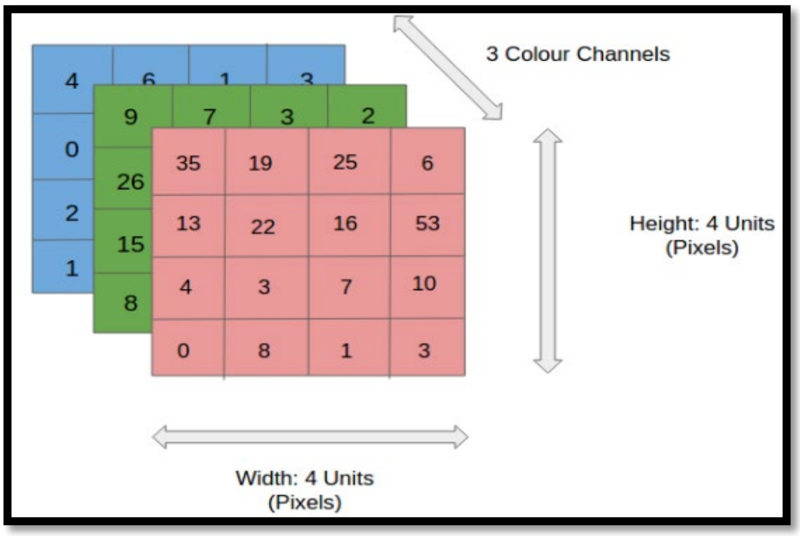


Fig 3.1.1.1: (RGB-Matrix):An RGB image is nothing but a matrix of pixel values having three planes whereas a grayscale image is the same, but it has a single plane.

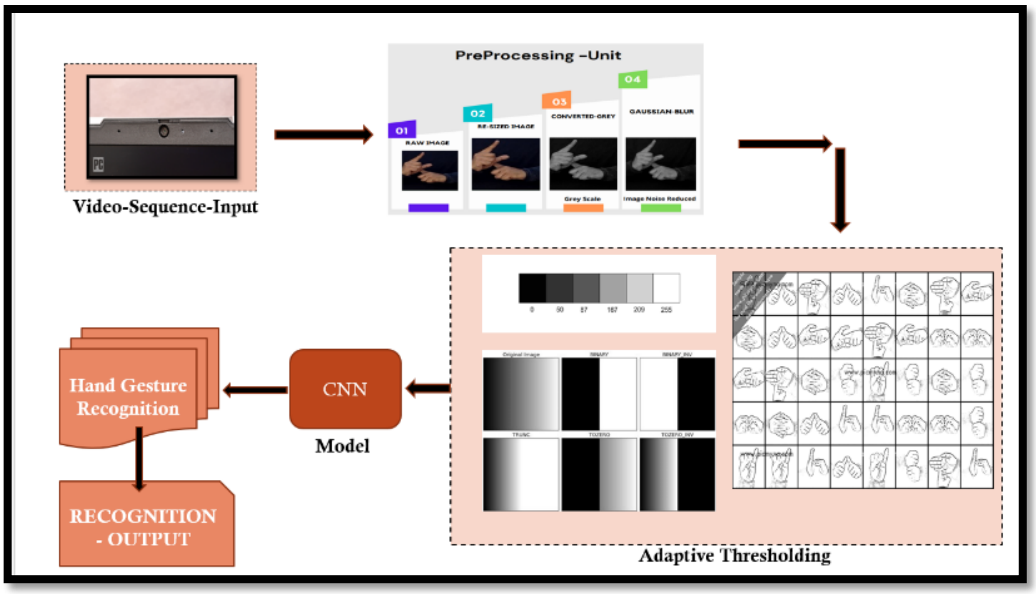


Fig 3.1.1.2: (CNN Model Flow Chart)

3.1.2 KNN

- K-NN is one of the simplest Machine Learning algorithms based on the Supervised Learning technique.
- K-NN algorithm assumes the similarity between the new case/data and available cases and put the new case into the category that is most similar to the available categories.
- K-NN algorithm stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a good suite category by using K- NN algorithm. This part of the K-NN algorithm is useful for gesture recognition.

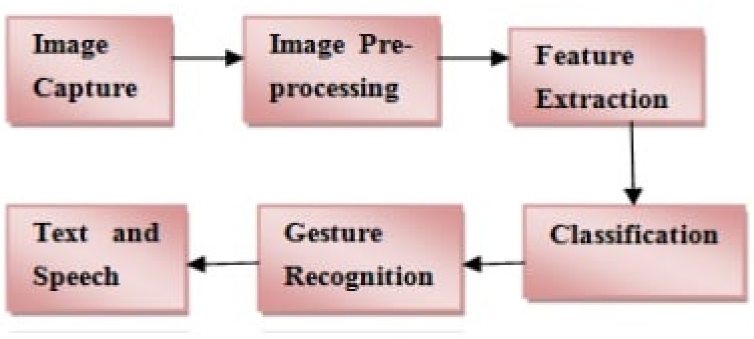


Fig 3.1.2.1: (KNN Model Flow Chart)

3.2 Model Tuning

Tuning is the process of maximizing a model's performance without overfitting or creating too high of a variance. In machine learning, this is accomplished by selecting appropriate "hyperparameters."

3.2.1 KNN Model

- K-NN algorithm stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a good suite category by using K- NN algorithm.

```
import {

  KNNImageClassifier
}
 from 'deeplearn-knn-image-classifier';
import * as dl from 'deep learn';
// 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 signaling when to start the prediction
// The stop gesture is for signalling when to stop prediction
var words = ["start", "stop"]; initializeTranslator() {
    this.startWebcam();
    this.loadKNN();
  }
// Initializing the kNN model to none.
this.knn = null;
/* Initializing the previous kNN model that we trained when training the current model
is stopped or prediction has begun. */
this.previousKnn = this.knn;
// Initalizing kNN model to none.
this.knn = null;
/* Initializing the previous kNN model that we trained when training the current model
is stopped or prediction has begun. */
this.previousKnn = this.knn;
//This function sets up the webcam
  startWebcam() {
    navigator.mediaDevices.getUserMedia({
        video: {
          facingMode: 'user'
        },
        audio: false
      })
      .then((stream) => {
        this.video.srcObject = stream;
        this.video.width = IMAGE_SIZE;
        this.video.height = IMAGE_SIZE;
        this.video.addEventListener('playing', () => this.videoPlaying = true);
        this.video.addEventListener('paused', () => this.videoPlaying = false);
      })
  }
//This function loads the kNN classifier
loadKNN() {
  this.knn = new KNNImageClassifier(words.length, TOPK);
  // Load knn model
  this.knn.load().then(() => this.initializeTraining());
```

3.2.2 CNN Model

We convert our input images (RGB) into grayscale and apply gaussian blur to remove unnecessary noise. We apply the adaptive threshold to extract our hand from the background and resize our images to 640x640.

We feed the input images after pre-processing to our model for testing after applying all the operations mentioned above. Estimating how likely the image will fall under one of the classes

Adaptive Thresholding:

```
import numpy as np
import cv2
vid = cv2.VideoCapture(0)
while True:
    sucess, img = vid.read()
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    #ADAPTIVE THRESHOLDING
    thresh = cv2.adaptiveThreshold(gray, 255, cv2.ADAPTIVE_THRESH_GAUSSIAN_C,
cv2.THRESH_BINARY_INV, 21, 4)
    ret, res = cv2.threshold(thresh, 10, 255, cv2.THRESH_BINARY_INV +
cv2.THRESH_OTSU)
    cv2.imshow("Video", res)  # use gray here the
```

Converting obtained dataset using Adaptive Thresholding:

```
import os
from os import listdir
from os. path import file, join
from this import d
import NumPy
import cv2

d = 0
mypath = "D:\WorkSpace\....."  # Enter your dataset images path here
onlyfiles = [f for f in listdir(mypath) if isfile(join(mypath, f))]
images = numpy.empty(len(onlyfiles), dtype=object)
# To take image from file
for n in range(0, len(onlyfiles)):
    images[n] = cv2.imread(join(mypath, onlyfiles[n]))
for I in images:
    scale_percent = 500  # percent of original size

    width = int(i.shape[1] * scale_percent / 100)
    height = int(i.shape[0] * scale_percent / 100)
    dim = (width, height)
```

```
        resiz = cv2.resize(i, dim, interpolation=cv2.INTER_AREA)  # resize images
        gray = cv2.cvtColor(resiz, cv2.COLOR_BGR2GRAY)  # converting to gray scale
        blur = cv2.GaussianBlur(gray, (9, 9), sigmaX=8, sigmaY=8)
        thresh = cv2.adaptiveThreshold(blur, 255, cv2.ADAPTIVE_THRESH_GAUSSIAN_C,
cv2.THRESH_BINARY_INV, 21, 4)

        ret, res = cv2.threshold(thresh, 10, 255, cv2.THRESH_BINARY_INV +
cv2.THRESH_OTSU)
        path = 'data\....'  # Enter the path to store the converted image
        cv2.imwrite(os.path.join(path, '%d.jpg' % d), res)
        d += 1  # names the stored images with 0,1,2,3......
        cv2.waitKey(0)
```

4 Dataset

4.1 Dataset

Table 4.1: Dataset

Name	Characteristics	Publisher	Accuracy
Custom Made dataset of gesture images.	• Implemented alphabets (A-Z) and numbers (0-9) • Total classes = 35. Each class has 1200 images • Train-Test ratio of 70:30.	Karthikeyan https://www.iosrjournals.org/iosr-jce/papers/Vol22-issue3/Series-1/B2203011419.pdf	98% accuracy approximately.
	• Implemented alphabets (A-Z) • 800 images to Train and 200 images to test. • Train-Test ratio 80:20.		98% accuracy approximately.

	• Dual mode of recognition is implemented. • all the alphabets (A-Z) and digits (0-9). • Train-Test ratio of 80:20.		

5 Implementation

5.1 Code

```
initializeTraining() {
  if (this.timer) {
    this.stopTraining();
  }
  var promise = this.video.play();

  if (promise !== undefined) {
    promise.then(_ => {
      console.log("Autoplay started")
    }).catch(error => {
      console.log("Autoplay prevented")
    })
  }
}

// This function adds examples for the gesture to the kNN model
train(gestureIndex) {
  console.log(this.videoPlaying);
  if (this.videoPlaying) {
    console.log("entered training");
    // Get image data from video element
    const image = dl.fromPixels(this.video);

    // Add current image to classifier
    this.knn.addImage(image, gestureIndex);

    // Get example count
```

```
      const exampleCount = this.knn.getClassExampleCount()[gestureIndex];

      if (exampleCount > 0) {
        //if example count for this particular gesture is more than 0, update it
        this.exampleCountDisplay[gestureIndex].innerText = ' ' + exampleCount + ' examples';

        //if example count for this particular gesture is 1, add a capture of the gesture to gesture
cards
        if (exampleCount == 1 && this.gestureCards[gestureIndex].childNodes[1] == null) {
          var gestureImg = document.createElement("canvas");
          gestureImg.className = "trained_image";
          gestureImg.getContext('2d').drawImage(video, 0, 0, 400, 180);
          this.gestureCards[gestureIndex].appendChild(gestureImg);
        }

        // if 30 examples are trained, show check mark to the user
        if (exampleCount == 30) {
          this.checkMarks[gestureIndex].src = "Images//checkmark.svg";
          this.checkMarks[gestureIndex].classList.add("animated");
          this.checkMarks[gestureIndex].classList.add("rotateIn");
        }
      }
    }
  }

  setUpTranslation() {
    // stop training
    if (this.timer) {
      this.stopTraining();
    }

    // Set status to predict, call copy translated text listener and start prediction
    this.setStatusText("Status: Ready to Predict!", "predict");
    this.video.play();
    this.pred = requestAnimationFrame(this.predict.bind(this));
  }

  /*This function predicts the class of the gesture and returns the predicted text if its above a set
threshold.*/
  predict() {
    this.now = Date.now();
    this.elapsed = this.now - this.then;

    if (this.elapsed > this.fpsInterval) {
      this.then = this.now - this.elapsed % this.fpsInterval;
      if (this.videoPlaying) {
        const exampleCount = this.knn.getClassExampleCount();
        const image = dl.fromPixels(this.video);

        if (Math.max(...exampleCount) > 0) {
          this.knn.predictClass(image)
            .then((res) => {
              for (let i = 0; i < words.length; i++) {
                /*if gesture matches this word & is above threshold & isn't same as prev prediction
                and is not stop gesture, return that word to the user*/
                if (res.classIndex == i && res.confidences[i] > confidenceThreshold && res.classIn-
dex != this.previousPrediction) { //  && res.classIndex != 1) {
                this.setStatusText("Status: Predicting!", "predict");

                // Send word to Prediction Output so it will display or speak out the word.
                this.predictionOutput.textOutput(words[i], this.gestureCards[i], res.confidences[i] *
100);

                // set previous prediction so it doesnt get called again
                this.previousPrediction = res.classIndex;
                }
              }
            }).then(() => image.dispose())
        } else {
          image.dispose();
        }
      }
    }

    // Recursion on predict method
    this.pred = requestAnimationFrame(this.predict.bind(this));
  }

  /*This function pauses the predict method*/
  pausePredicting() {
    console.log("pause predicting");
    this.setStatusText("Status: Paused Predicting", "predict");
    cancelAnimationFrame(this.pred);
    this.previousKnn = this.knn;
  }

stopTraining() {
    this.video.pause();
    cancelAnimationFrame(this.timer);
    console.log("Knn for start: " + this.knn.getClassExampleCount()[0]);
```

```
    this.previousKnn = this.knn; // saves current knn model so it can be used later
  }

class PredictionOutput {
  constructor() {
    //Initializing variables for speech synthesis and output
    this.synth = window.speechSynthesis;
    this.voices = [];
    this.pitch = 1.0;
    this.rate = 0.9;

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

    this.selectedVoice = 48; // this is Google-US en. Can set voice and language of choice

    this.currentPredictedWords = [];
    this.waitTimeForQuery = 10000;

    this.synth.onvoiceschanged = () => {
      this.populateVoiceList()
    };

    //Set up copy translation event listener
    this.copyTranslation();
  }

  // Checks if speech synthesis is possible and if selected voice is available
  populateVoiceList() {
    if (typeof speechSynthesis === 'undefined') {
      console.log("no synth");
      return;
    }
    this.voices = this.synth.getVoices();

    if (this.voices.indexOf(this.selectedVoice) > 0) {
      console.log(this.voices[this.selectedVoice].name      +      ':'      +      this.voices[this.select-
edVoice].lang);
    }
  }
```

```
/*This function outputs the word using text and gesture cards*/
textOutput(word, gestureCard, gestureAccuracy) {
 // If the word is start, clear translated text content
 if (word == 'start') {
  this.clearPara();

  setTimeout(() => {
   // if no query detected after start is signed, clear para
   if (this.currentPredictedWords.length == 1) {
    this.clearPara();
   }
  }, this.waitTimeForQuery);
 }

 // If first word is not start, return
 if (word != 'start' && this.currentPredictedWords.length == 0) {
  return;
 }

 // If word was already said in this query, return
 if (this.currentPredictedWords.includes(word)) {
  return;
 }

 // Add word to predicted words in this query
 this.currentPredictedWords.push(word);

 // Depending on the word, display the text output
 if (word == "start") {
  this.translationText.innerText += ' ';
 } else if (word == "stop") {
  this.translationText.innerText += '.';
 } else {
  this.translationText.innerText += ' ' + word;
 }

 //Clone Gesture Card
 this.translatedCard.innerHTML = " ";
 var clonedCard = document.createElement("div");
 clonedCard.className = "trained-gestures";

 var gestName = gestureCard.childNodes[0].innerText;
 var gestureName = document.createElement("h5");
 gestureName.innerText = gestName;
```

```
    clonedCard.appendChild(gestureName);

    var gestureImg = document.createElement("canvas");
    gestureImg.className = "trained_image";
    gestureImg.getContext('2d').drawImage(gestureCard.childNodes[1], 0, 0, 400, 180);
    clonedCard.appendChild(gestureImg);

    var gestAccuracy = document.createElement("h7");
    gestAccuracy.innerText = "Confidence: " + gestureAccuracy + "%";
    clonedCard.appendChild(gestAccuracy);

    this.translatedCard.appendChild(clonedCard);

    // If its not video call mode, speak out the user's word
    if (word != "start" && word != "stop") {
      this.speak(word);
    }
  }

  /*This functions clears translation text and cards. Sets the previous predicted words to null*/
  clearPara() {
    this.translationText.innerText = '';
    main.previousPrediction = -1;
    this.currentPredictedWords = []; // empty words in this query
    this.translatedCard.innerHTML = " ";
  }

  /*The function below is adapted from https://stackoverflow.com/questions/45071353/javas-
cript-copy-text-string-on-click/53977796#53977796
  It copies the translated text to the user's clipboard*/
  copyTranslation() {
    this.translationHolder.addEventListener('mousedown', () => {
      main.setStatusText("Text Copied!", "copy");
      const el = document.createElement('textarea'); // Create a <textarea> element
      el.value = this.translationText.innerText; // Set its value to the string that you want copied
      el.setAttribute('readonly', ''); // Make it readonly to be tamper-proof
      el.style.position = 'absolute';
      el.style.left = '-9999px'; // Move outside the screen to make it invisible
      document.body.appendChild(el); // Append the <textarea> element to the HTML document
      const selected =
        document.getSelection().rangeCount > 0 // Check if there is any content selected previ-
ously
        ?
        document.getSelection().getRangeAt(0) // Store selection if found
        :
```

```
     false; // Mark as false to know no selection existed before
    el.select(); // Select the <textarea> content
    document.execCommand('copy'); // Copy - only works as a result of a user action (e.g. click
events)
    document.body.removeChild(el); // Remove the <textarea> element
    if (selected) { // If a selection existed before copying
     document.getSelection().removeAllRanges(); // Unselect everything on the HTML docu-
ment
     document.getSelection().addRange(selected); // Restore the original selection
    }
   });
  }

  /*This function speaks out the user's gestures. In video call mode, it speaks out the other
  user's words.*/
  speak(word) {
   var utterThis = new SpeechSynthesisUtterance(word);

   utterThis.onerror = function (evt) {
    console.log("Error speaking");
   };
   utterThis.voice = this.voices[this.selectedVoice];
   utterThis.pitch = this.pitch;
   utterThis.rate = this.rate;

   this.synth.speak(utterThis);
  }
 }

 var main = null;
 //Initializes the main class on window load
 window.addEventListener('load', () => {
  main = new Main()
 });
```

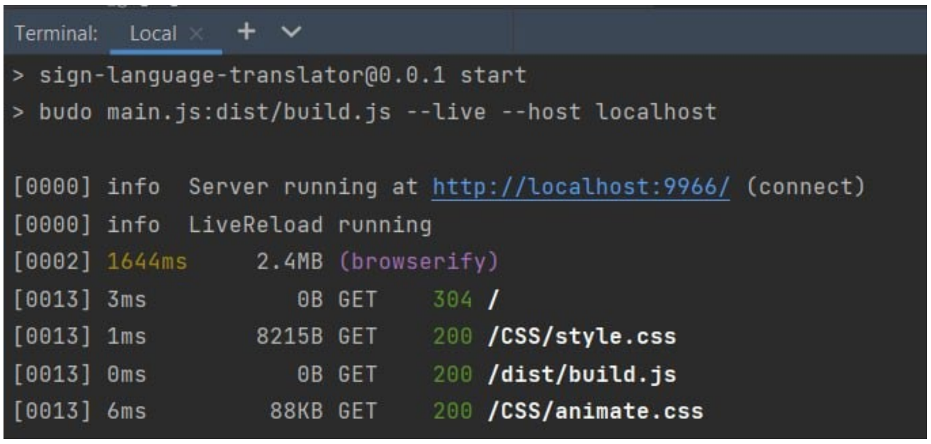
6 Result

6.1 Tasks Accomplished

- Hand Gesture Training and Classification
- Prediction works in varying Lighting Conditions
- Retrainable Image Classes
- Translated Text can be copied to the Clipboard
- Cards that display Information about each Gesture

- Text to Speech of translated text
- Minimal stress on memory
- Cohesive Text Styling
- Simple User Interface
- Comprehensive Commenting

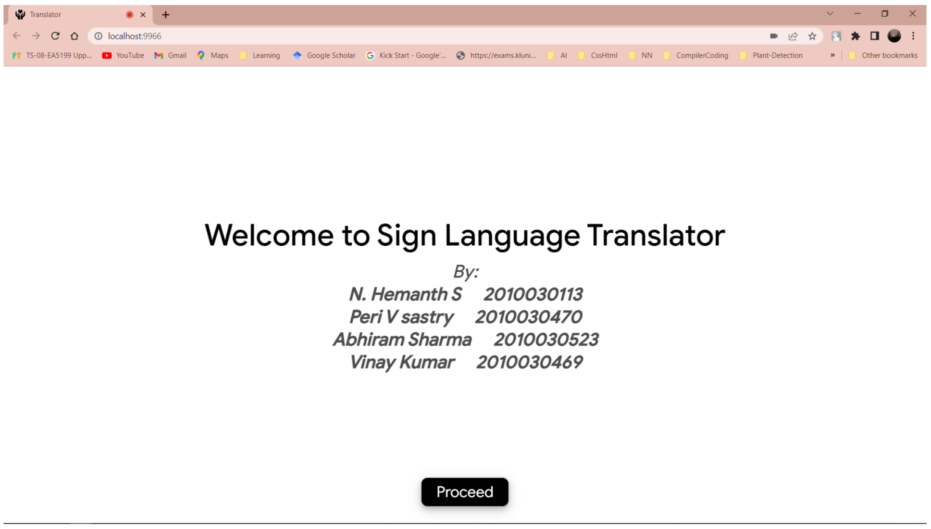
6.2 Outputs



```
> sign-language-translator@0.0.1 start
> budo main.js:dist/build.js --live --host localhost

[0000] info  Server running at http://localhost:9966/ (connect)
[0000] info  LiveReload running
[0002] 1644ms     2.4MB (browserify)
[0013] 3ms          0B GET    304 /
[0013] 1ms       8215B GET    200 /CSS/style.css
[0013] 0ms          0B GET    200 /dist/build.js
[0013] 6ms        88KB GET    200 /CSS/animate.css
```

Fig: 6.2.1



Welcome to Sign Language Translator

By:
N. Hemanth S   2010030113
Peri V sastry   2010030470
Abhiram Sharma   2010030523
Vinay Kumar   2010030469

Fig: 6.2.2

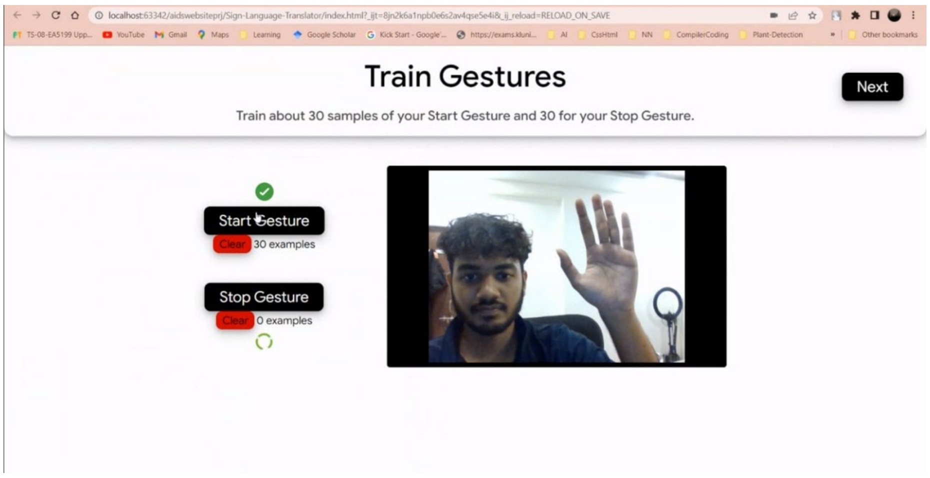

Train Gestures
Next
Train about 30 samples of your Start Gesture and 30 for your Stop Gesture.
Start Gesture
Clear
30 examples
Stop Gesture
Clear
0 examples

Fig: 6.2.3

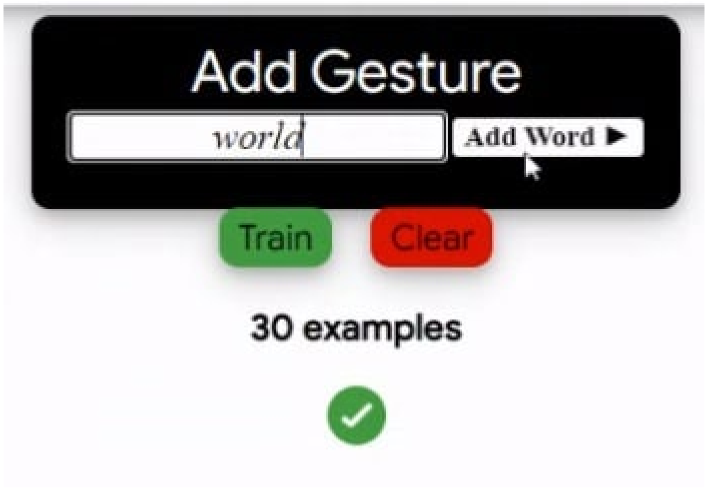


Fig: 6.2.4

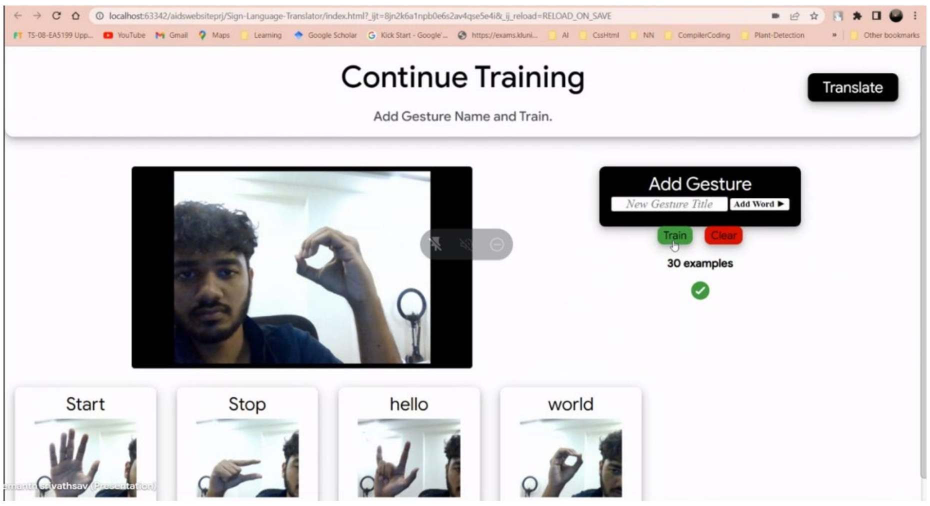

localhost:63342/aidswebsiteprj/Sign-Language-Translator/index.html?_ijt=8jn2k6a1npb0e6s2av4qse5e4i&_ij_reload=RELOAD_ON_SAVE
Continue Training
Add Gesture Name and Train.
Translate
Add Gesture
New Gesture Title
Add Word ▶
Train
Clear
30 examples
Start
Stop
hello
world

Fig: 6.2.5

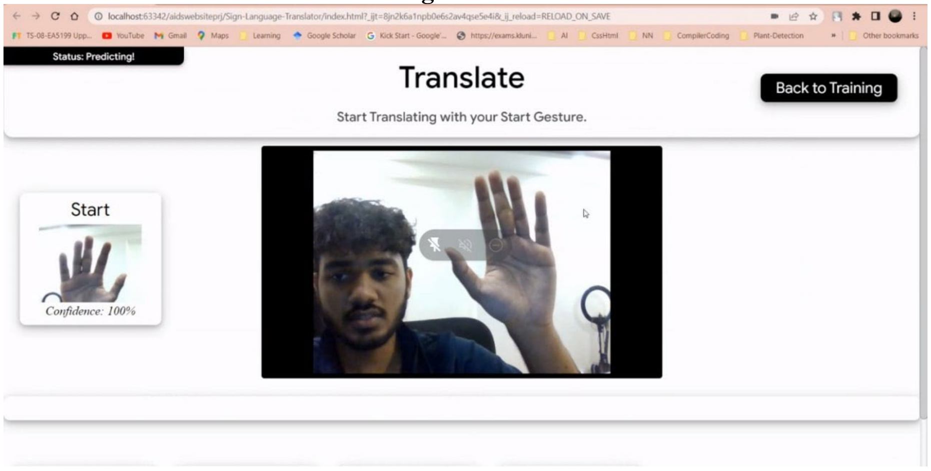

localhost:63342/aidswebsiteprj/Sign-Language-Translator/index.html?_ijt=8jn2k6a1npb0e6s2av4qse5e4i&_ij_reload=RELOAD_ON_SAVE
Status: Predicting!
Translate
Start Translating with your Start Gesture.
Back to Training
Start
Confidence: 100%

Fig: 6.2.6

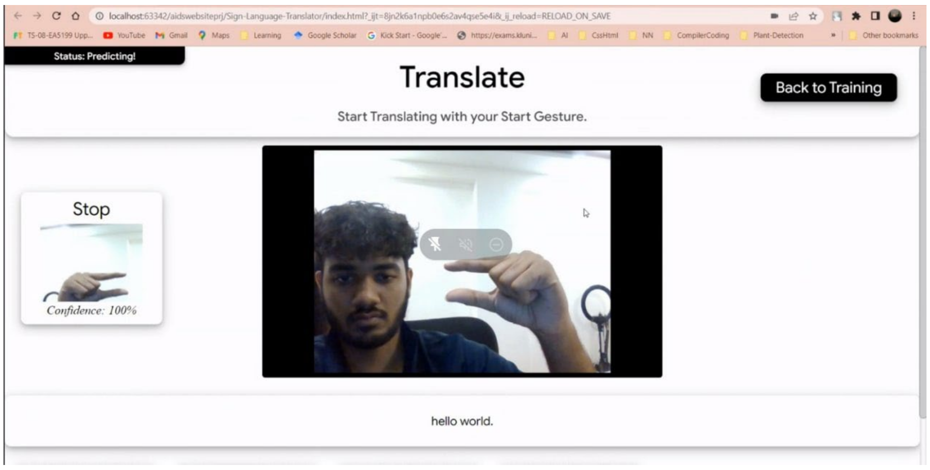


Fig: 6.2.7

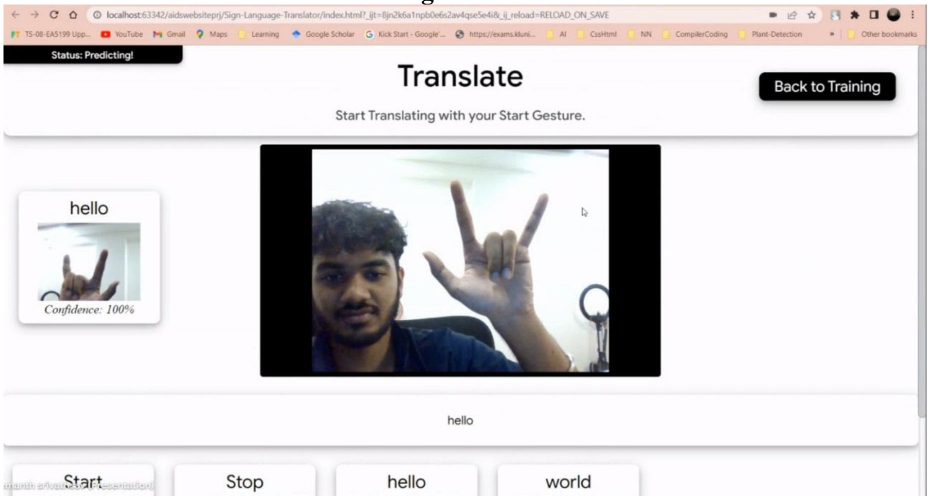


Fig: 6.2.8

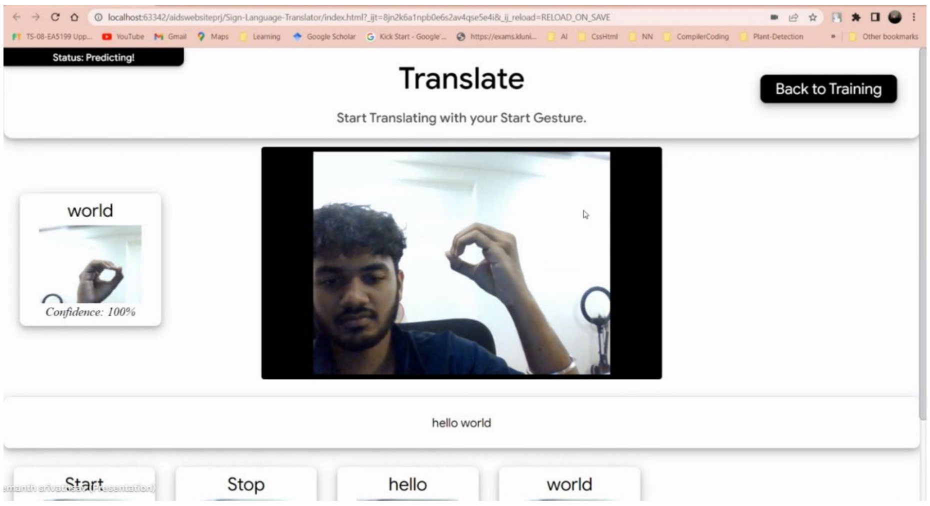


Fig: 6.2.9

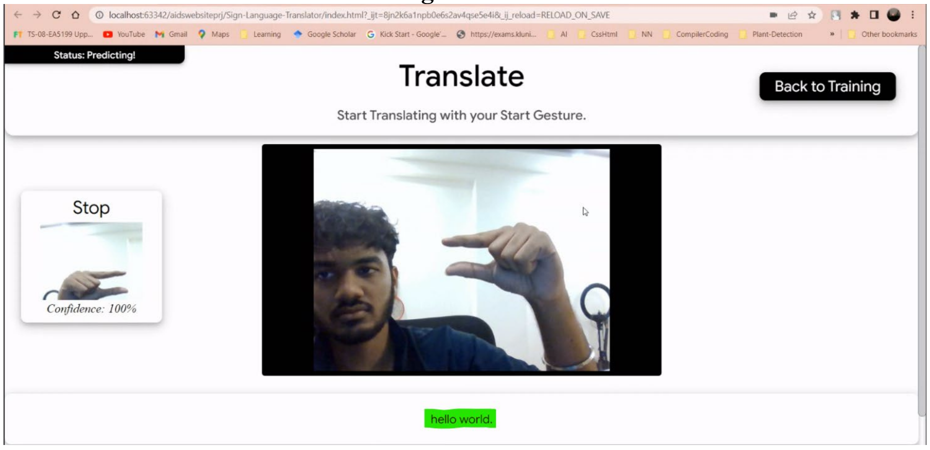


Fig: 6.2.10

7 Discussion

7.1 Discussion

Upon training the image dataset using CNN, the training accuracy achieved was very high (around 99%) but, the real-time performance was not up to the mark. It was predicted incorrectly most of the time because in real-time hand gestures were not placed exactly at the center and aligned vertically. In order to overcome this shortcoming, we trained our model with kNN. The training accuracy was reduced to 89% but the real-time predictions were predominantly correct.

KNN is often considered a “lazy learning” classifier as it accepts calculations until it receives a request to classify unlabeled data. This characteristic makes it adaptive to the new dataset. Nevertheless, k-NN creates high computation because it reruns the computation for all datasets for each new unseen data. The main advantage of the KNN is its simplicity which makes it, is easy implementation. Designing models are inexpensive, flexible categorization, suitable for several modal classes as well as with numerous class labels.

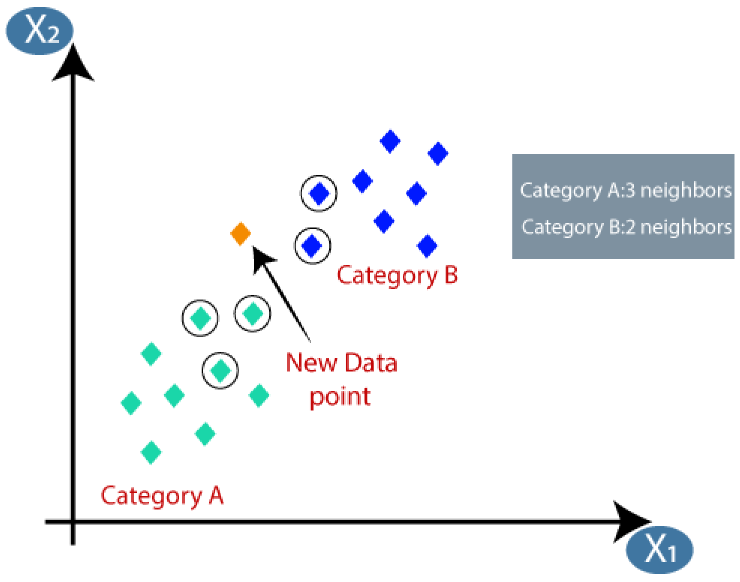


Fig 7.1.1: (KNN Classifier)

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.

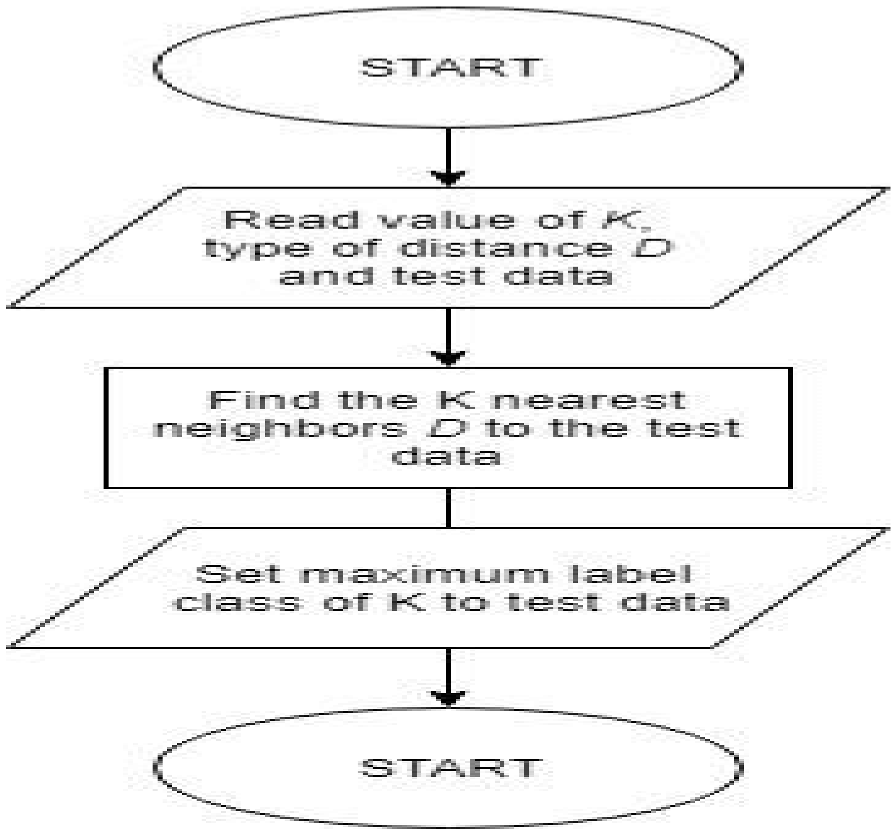

START
Read value of K, type of distance D and test data
Find the K nearest neighbors D to the test data
Set maximum label class of K to test data
START

Fig 7.1.2: (KNN Flow Chart)

KNN Classifier Code

```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import train_test_split
from sklearn.datasets import load_iris

irisData = load_iris()

X = irisData.data
y = irisData.target
# Split into training and test set
X_train, X_test, y_train, y_test = train_test_split(
            X, y, test_size = 0.2, random_state=42)

knn = KNeighborsClassifier(n_neighbors=7)

knn.fit(X_train, y_train)
```

```
# Predict on a dataset which model has not been seen before
print(in.predict(X_test)
```

8 Conclusion

8.1 Conclusion

The aim of this project is to predict hand gestures in real-time. The above work shows that it can be solved with better accuracy when we actually trained using KNN Model. By applying KNN we removed the random false predictions and increased our real-time accuracy. The hand gestures were trained and tested in real-time using KNN model. Our model showed good accuracy while predicting results both offline and online. A few shortcomings of this technique are that it is expensive for classifying unidentified data, and the precision can be affected by the noise of unrelated features. Hand gesture recognition addresses a fault in interaction systems. Controlling things by hand is more natural, easier, more flexible, and cheaper, and there is no need to fix problems caused by hardware devices since none is required.

8.2 Future Work

1. We can develop a model for sentence-level recognition. This will require a system that can detect changes with respect to the temporal space.
2. We can develop a complete product that will help the speech and hearing-impaired people, and thereby reduce the communication gap.

9 References