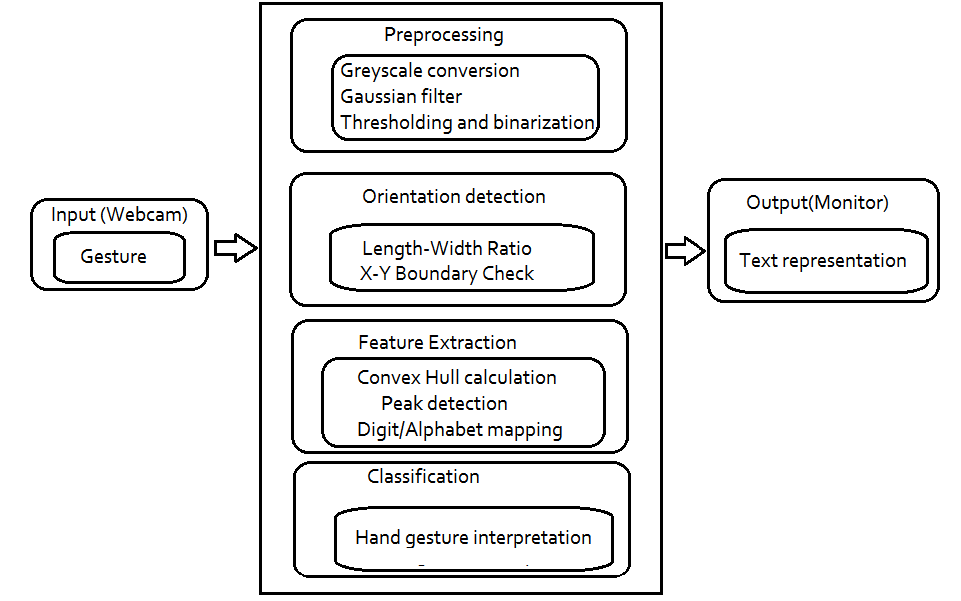
**Our Goal**

Our Goal Construct words for detecting American Sign Language(ASL) Fingerspelling gestures from a video input/Live Webcam.

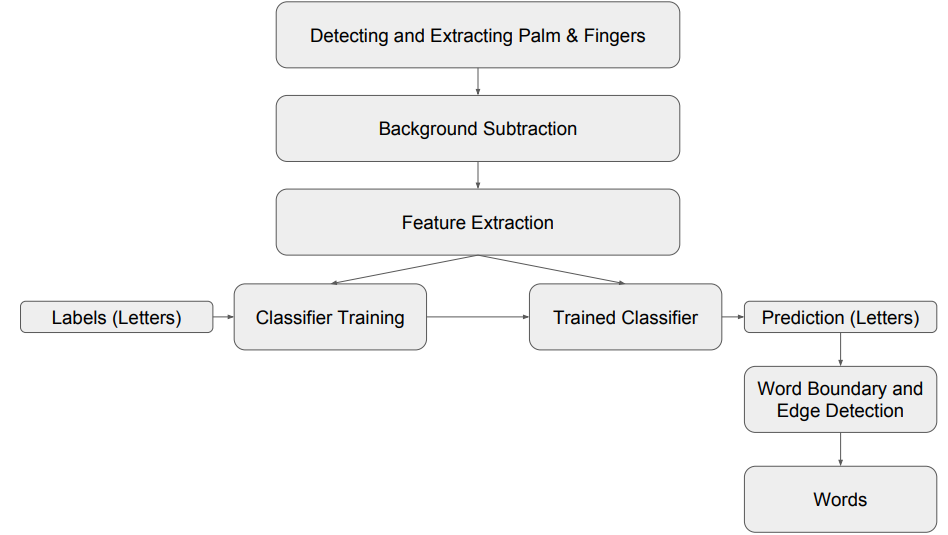
**Expected Outcome of the project :**

The deaf people and people who cannot speak can make use of this system, and communicate with the outside world with the help of sign language. Our main motivation in this research is to bridge the gap between human computer interaction between physically disabled people. Conventionally, mute people needed to type in order to interact with machines but now they can use their sign language and this will make tasks very easy and simple for them. The desired sign language messages is converted into a text message, which is further displayed on the screen.

**Architecture Design**



**DataFlowDesign**



**Modules Invloved in the Project**

1. **Setup webcam and canvas**

Setup our webcam and canvas to view the webcam stream in the web display. For that, we’re going to make use of the Webcam component that we installed and imported earlier. First, we need to create reference variables using the useRef hook, as shown in the code snippet below:

const webcamRef = useRef(null);

const canvasRef = useRef(null);

Next, we need to initialize the Webcam component in our render method. Using this, we can stream the webcam feed in the canvas, also passing the refs as prop properties. The coding implementation is provided in the code snippet below:

<div className="App">

<header className="App-header">

<Webcam

ref={webcamRef}

style={webcamstyle}

/>

Now, we need to add the canvas component just below the Webcam component. The canvas component enables us to draw anything that we want to display in the webcam feed. The canvas component, with its prop configurations, are provided below:

1. **Loading the Hand Pose model**

In this module, we’re going to create a function called runHandpose, which initializes the hand pose model using the load method from the handpose module. The overall code for this function is provided in the code snippet below:

const runHandpose = async () => {

const net = await handpose.load();

console.log("Handpose model loaded.");

};

In order to load the hand pose model upon starting the app, we’re going to call it inside the useEffect hook, as shown in the code snippet below:

useEffect(() => {

runHandpose()

}, [])

1. **Detect Hand Positions**

In this module, we’re going to create a function called detect, which handle the hand will pose detection. First, we detect the webcam and grab the video properties to handle the video adjustments, as directed in the code snippets below:

const detect = async (hand) => {

if (

typeof webcamRef.current !== "undefined" &&

webcamRef.current !== null &&

webcamRef.current.video.readyState === 4

) {

We then get and set the video properties using webcamRef that we defined earlier:

const video = webcamRef.current.video;

const videoWidth = webcamRef.current.video.videoWidth;

const videoHeight = webcamRef.current.video.videoHeight;

webcamRef.current.video.width = videoWidth;

webcamRef.current.video.height = videoHeight;

Then, we need to set the canvas width and height based on the dimensions of the video:

canvasRef.current.width = videoWidth;

canvasRef.current.height = videoHeight;

Then, we start estimating the hand pose using the estimateHands method provided by the hand module that takes video as a parameter, as shown in the code snippet below:

const result = await hand.estimateHands(video);

1. **HandePose Estimation**

Next, in this module we need to call this detect function inside the runHandpose method under the setInterval method. This enables the detect function to run every 10 milliseconds. The coding implementation is provided in the code snippet below:

const runHandpose = async () => {

const net = await handpose.load();

setInterval(() => {

detect(net);

}, 10);

};

1. **Convert user pose to text**

We need to update our detect function with a gesture detecting function. We’re going to make use of the GestureEstimator method from the fingerpose package in order to detect hand gestures. we apply the pose to text mapping as well as the confidence index to detect the accurate gesture and set the appropriate text to the end user.

**Conclusion**

In this project we have demonstrated an application of Hand-Gesture recognition by detecting hand poses of the user to specific set of words. Future work will primarily comprise of expanding the dataset to include more letters.