Hand Gesture Recognition

M Hemanth Kumar
Roll no. 211AI025
IT Dept.(AI)
NIT Surathkal
Surathkal, Karnataka
India

G V Ravi Ram
Roll no. 211AI018
IT Dept.(AI)
NIT Surathkal
Surathkal, Karnataka
India

***Abstract*—Hand gesture recognition (HGR) is a significant area of research in computer vision and human-computer interaction. This paper presents a deep learning-based approach for recognizing hand gestures. We propose a system that uses convolutional neural networks (CNNs) to extract features from hand images and a long short-term memory (LSTM) network for gesture classification. The system is trained on a dataset of hand gestures, and the performance is evaluated on a held-out test set which is hidden. The system is designed to be real-time, making it suitable for use in interactive applications. This research contributes to the development of more natural and intuitive human-computer interfaces.**

Keywords- abstraction, CNN, Hand Gesture Recognition, mediapipe, tensors, volume

I. INTRODUCTION

This Hand gesture recognition (HGR) is an intuitive method for human-computer interaction (HCI).After investigating a wide variety of input devices and techniques, we came across a skeleton-based HGR which has also been a popular choice due to its robustness to background and light variations. Many of skeleton-based HGR systems rely on depth sensors, such as RGBD cameras, which are not nearly as common as RGB cameras on mobile devices.On the other hand, only a single RGB camera is sufficient for our HGR. It is carried out by taking key skelton points from image and then passing it through a gesture classifier.

II. TOOLS USED

2.1 Medapipe

Mediapipe is an Open-Source collection of pretrained CNN models to recognize a variety of objects like hands, pose, face etc and return mathematical coordinates in form of tensors to handle them. Here to make our hand-gesture volume controller lighter, we would be using Mediapipe-Hands' Hand-Landmark model.

2.2 OpenCV

Open CV is a library of python that tackle PC vision issue. It is an open-source computer vision library that provides a wide range of image processing and computer vision algorithms. It is often used in conjunction with TensorFlow and other machine learning libraries to build computer vision applications. It is used to detect various objects like hands, face etc which is done using the machine learning. It also performs object detection and motion detection. It also support several type of operating system.

So, here in order to identify and isolate the hand, we will be using OpenCV.

2.3 Tensorflow

TensorFlow is an open-source library for ML and DL that can run on a variety of platforms, including desktops, servers, and mobile devices and also a powerful library for building and executing computations that involve tensors, which are multi-dimensional arrays of data and provides a variety of tools for building, training, and deploying machine learning models, including support for various types of neural networks such as feedforward, convolutional, recurrent, and generative models and also support for distributed computing, allowing users to train large models on clusters of machines, thereby making training models on large-scale datasets feasible.

TensorFlow library provides us with a number of pre-built and pre-trained models for instance, image classification and object detection which we would be using.It also provides a high-level API called Keras, which provides a simple and intuitive interface for building and training models that we would be using in implementation.

Since we have been using CNN for analyzing large pixel data, Tensorflow will be used to make the utility lighter, faster and more compatible.

III. METHODOLOGY

3.1 Volume control using hand gesture

3.1.1 Abstraction of Mediapipe

We would be using Hand-Landmark detection in Mediapipe.. Here, CNN is pre-trained on a dataset of hand images and is used to extract features from new images. These features are then passed through a long short-term memory (LSTM) network, which is trained to classify the features as belonging to a specific hand.

Input layer, the first layer of the CNN, receives the raw image data as input. The image data is typically represented as a 2D array of pixels, where each pixel has a value between 0 and 255. The input layer normalizes the pixel values and converts the image data into a 3D tensor that can be handled using tensorflow, which is then passed to the hidden layers.

The hidden layers of a CNN are where the majority of the computation takes place. These layers consist of a combination of convolutional layers, pooling layers, and fully connected layers. Convolutional layers extract features from the input image. These layers consist of a set of filters, which are convolved with the input image to extract features. The filters are learned during the training process, and they are designed to detect specific patterns in the image data.

Pooling layers are used to reduce the size of the feature maps produced by the convolutional layers. This is done by applying a pooling operation; here, max pooling, which takes the maximum value of a group of adjacent pixels. This also

helps us to reduce the computational cost and prevent overfitting.

Fully connected layers classify the features extracted by the convolutional and pooling layers. These layers take the flattened feature maps from the previous layer and pass them through a set of fully connected neurons. Each neuron in a fully connected layer is connected to all neurons in the previous layer.

The output layer produces the final classification of the input image. The output layer typically consists of a single neuron, which produces a probability score for each class in the dataset. The class with the highest probability score is the final classification of the image.

Hence, Mediapipe-Hands' Hand-Landmark Identification model is primary CNN which takes in an input of RGB image in form of pixels, thereby 3d tensors which after passing through several convulsed layers; gives 21 key-point locations on hand as output.

Various keypoint indices are shown in figure 1.

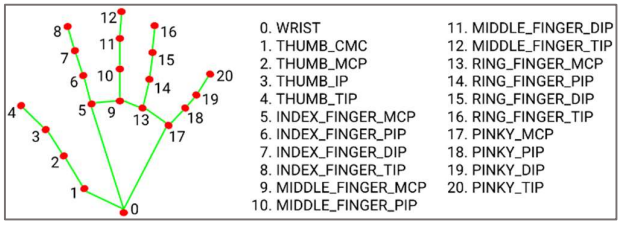
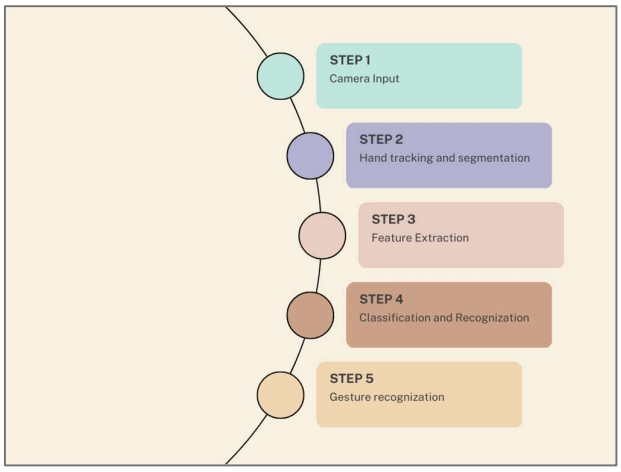


Figure 1

3.1.2 Implementation

There are 21 specific landmarks that can be detected and each having unique index. Now that we have obtained those key-point locations, in our model we use index-4(x1,y1) and index-8(x2,y2) for the volume control ,the length between these two finger tips in the hand determines the volume.

Len = sqrt.[((x2-x1)^2) + ((y2-y1)^2)]



Flowchart of execution

This length is scaled to the volume of the chosen device in order to maintain precision. Inorder to link the run time output to the system specifications related to aduio volume we are using pycaw module in python. PyCAW (Python Core Audio Windows) is a Python module that allows access to the Windows Core Audio API. It provides a simple, high-level interface to the Core Audio API and allows developers to control and monitor the audio devices on a Windows system. PyCAW can be used for tasks such as adjusting the volume, muting, and changing the default audio device. It is a free and open-source library.

We are using mpdraw function to plot the volume level on the basis of user input and modify the volume based on the level.

The image (ref. Figure 2) shows working of volume control using hand gestures.

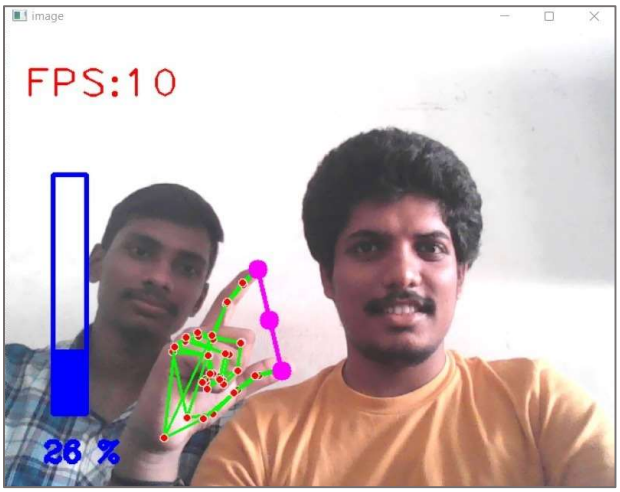
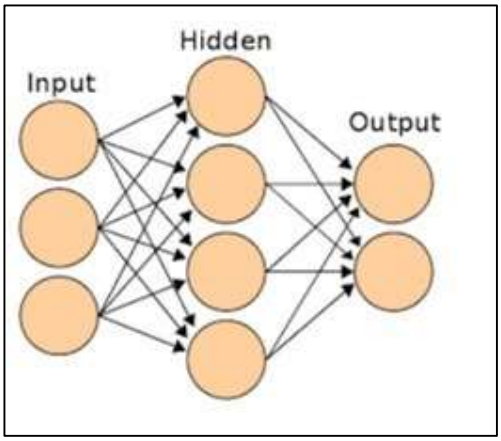


Figure 2

3.2 numbers recognition using hand gesture

3.2.1 Creating the Model

Basically we created our own customized model, which contains convultion layers ,pooling layers and fully connected layers.



Layers of Model

3.2.1.1 Convolution layers

There are two convolution layers in our model. For First convolution layer we used 32 filters each of kernel size (7,7) and for second convolution layer we used 16 filters each of kernel size (3,3)

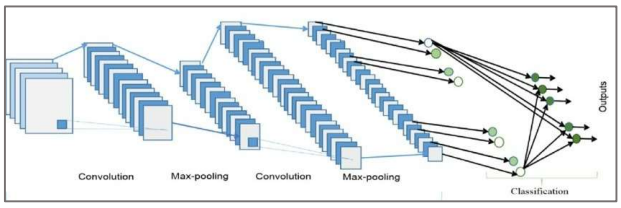
3.2.1.2 Polling layers

There are two polling layers after each convolution layer of of pool size (2,2). We are using ReLU(rectified linear unit) as an activator. ReLU (Rectified Linear Unit) is a widely used activation function in neural networks which maps all negative inputs to zero, and outputs the input if it is positive. It helps to speed up the training process and it works well on a wide range of problems.

3.2.1.3 fully connected layer

We are flattening the feature map that had been extracted in previous layers which acts as input layer.

We are using 3 hidden layers with 128,64,32 neurons in each layer respectively and activation function used is relu for all the hidden layers.



Layers of CNN

There are 20 different classes ranging from 0-19 that the model to classify so that for the output layer 20 neurons are used and activation function is softmax.

The softmax activation function is a multi-class classifier that converts a K-dimensional vector of real values to a probability distribution. It is commonly used in the output layer of neural networks for classification problems. It normalizes the input vector so that the sum of all outputs is 1, making it useful for probability prediction

3.2.2 Compiling the Model

Optimizer used is adam with the learning rate 10**-6 as the Adam is an optimization technique that is used to train neural networks. It is designed to adapt the learning rate during the training process, which helps to speed up convergence and improve performance. The algorithm uses moving averages of the gradients and squared gradients to adjust the learning rate on a per-parameter basis, which allows for more efficient training. It is a popular choice for large-scale machine learning problems due to its computational efficiency, the loss function is sparse categorical cross entropy asour dataset has large number of classes. It is a variant of categorical cross-entropy that is more computationally efficient when the classes are mutually exclusive and the labels are integers rather than one-hot encoded. It's used when the number of classes is large and the number of examples for each class is relatively small. and the metric used is accuracy in order to compile the model.

3.2.3 Fitting the Model

The training data and validation data are fit into the DL model created with 100 epochs .

3.2.4 Inferences

The images (ref. figure 3 and figure 4) depicts relation between accuracy and loss w.r.t. epochs. Clearly as number of epochs increase, the accuracy raised, but at the cost of time. Also Table (ref. Table 1) shows the final specifications followed (epoch) and obtained (Accuracies and Loss) of our model.

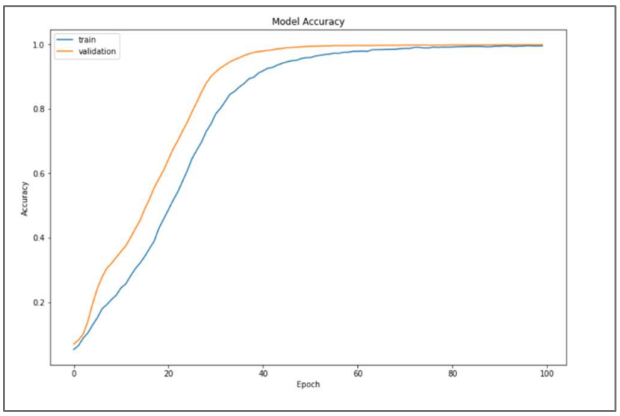


Figure 3

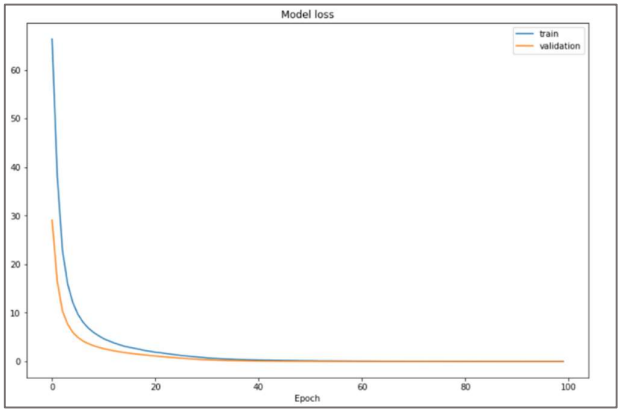


Figure 4

IV. CONCLUSION

Hand gesture numbers recognition is used to detect the numbers ranging from 0-19 we have got an accuracy of 99%. Hand gesture volume control is used to control the volume of the various electronics and an interesting development could be the integration of hand gestures control in the virtual agumented reality applications , where users could interact with the virtual environment in a more natural and intutive way.

Epochs	100
Train Accuracy	0.9935
Train Loss	0.0049
Test Accuracy	0.9986
Test Loss	0.0254

Table 1

V. APPLICATIONS

This character recognition model using DL can be used for various applications like ; creating a utility for dumb and deaf to understand sign language, making devices smarter using motion detection and gesture detection, analyzing CC-Footage to identify any suspicious communication using sign language.

VI. FUTURE WORKS

We would like to optimize the DL model of identifying digits to reduce time taken in order to integrate Computer Vision with the DL model we proposed to pass the frames of a video recorded live directly to identify the symbols, add support to standard hand gestures used by dumb and deaf in context of sign language; and deploy it on a website

VII. ACKNOWLEDGEMENT

We would like to express our deepest gratitude to Professor Dr.Dinesh Naik, Assistant Professor, Information Technology department, NITK, for their unwavering support and guidance throughout the completion of this project. His expertise and valuable insights have been instrumental in shaping the direction and outcome of our research. We are grateful for the endless hours of mentorship and the dedication he has towards the success of this project. His encouragement and support have been invaluable.

VIII.REFERENCES

[1] Li-juan Sun, Li-cai Zhang and Cai-long Guo, "Technologies of Hand Gesture Recognition Based on Vision [J]", Computer Technology and Development

[2] Gao Long, Research on Static Gesture Recognition Algorithm Based on Neural Network [D], Ning Xia University, 2017.

[3] Lin Guo, Zongxing Lu, Ligang Yao, "Human-Machine Interaction Sensing Technology Based on Hand Gesture Recognition: A Review", IEEE Transactions on Human-Machine Systems, vol.51, no.4, pp.300-309, 2021.

[4] Rachana R. Chhajed, Komal P. Parmar, Manvi D. Pandya, Neha G. Jaju, "Messaging and Video Calling Application for Specially Abled people using Hand Gesture Recognition", 2021 6th International Conference for Convergence in Technology (I2CT), pp.1-4, 2021.