Hand Gesture Detection based Virtual Mouse using Transfer learning

Anmol Aafre
Computer Department
Sarvajanik College of Engineering and Technology
Surat, India
anmolaafre.co22d1@scet.ac.in

Aryan Nimavat
Computer Department
Sarvajanik College of Engineering and Technology
Surat, India
aryannimavat.co22d1@scet.ac.in

Nirmit Goti
Computer Departmernt
Sarvajanik College of Engineering and Technology
Surat, India
nirmitgoti.co22d1@scet.ac.in

Meet Goyani
Computer Department
Sarvajanik College Of Engineering and Technology
Surat, India
meetgoyani@scet.ac.in

***Abstract*—This paper presents a Convolutional Neural Network (CNN)-based virtual mouse system that utilizes hand gesture recognition for controlling standard mouse operations. The system captures real-time video through a webcam, detects and processes hand gestures using MediaPipe, and classifies them via a transfer learning model namely Resnet101. Mouse actions such as cursor movement, left/right clicks, scrolls, and screenshots are executed using PyAutoGUI based on the recognized gestures.**

Keywords—Virtual Mouse, Hand Gesture Recognition, Transfer Learning

I. INTRODUCTION

Rapid technological improvements over the last few years have caused technologies to evolve in input and output devices, such as the effort to substitute traditional mice. Although technologies like Bluetooth mice and touchscreens have been introduced to the market, full hardware replacement has not been possible. Traditional methods, like glove-based detectors and colored finger caps, failed to eradicate hardware reliance.

This paper is a virtual mouse system with hand gesture recognition supported by image processing and machine learning. It follows methods such as background subtraction, pre-processing techniques, and Convolutional Neural Network (CNN) model to classify gestures. The system feeds live webcam as input, performs hand gesture detection, and leverages the identified gestures to provide control over mouse operations, and it is an interactive, hardware-free solution.

II. METHODOLOGY

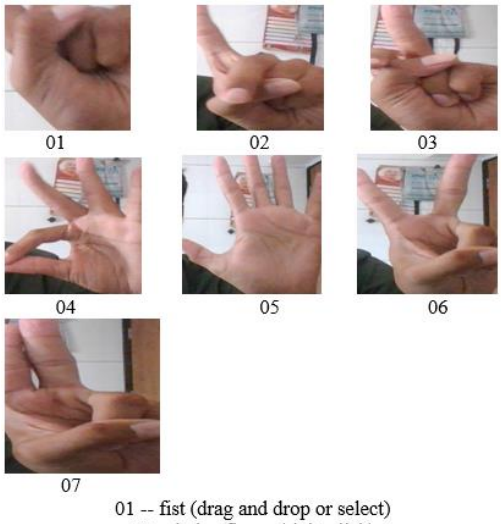
The envisaged virtual mouse system is constituted by three most important modules:
1.Capturing and pre-processing the real-time stream,
2.Gesture classification through a Convolutional Neural Network (CNN), and
3.Virtual control of the mouse depending on the anticipated gestures.
The system is deployed in Python via different libraries such as OpenCV, TensorFlow, Keras, NumPy, and PyAutoGUI. To accurately classify hand gestures, a custom dataset was created and used to train the model. Transfer learning was employed and the ResNet101 model was used for training, which significantly enhanced performance and reduced training time.

A. Capturing and pre-processing the real-time stream

The system captures live video from the webcam at a resolution of 1280 x 720 and flips each frame horizontally. Once a hand is detected, the frame is processed using Media Pipe's landmark detection, which identifies 21 hand key points. These landmarks are drawn on both the processed and displayed frames, with only one hand being tracked at a time to avoid confusion.

A white background array is created, and the landmark points are overlaid on it. These coordinates are normalized and saved for further use. The resulting image is resized and passed to the CNN model for gesture classification. The predicted gesture is shown on screen, and the 10th landmark (base of the middle finger) is used as the reference point for controlling the mouse pointer in real time.

The Figure 2. Shows the different gestures images from the dataset.



01 -- fist (drag and drop or select)
02 -- index finger (right click)
03 -- middle finger (left click)
04 -- nice (scrolling)
05 -- palm (halt)
06 -- peace (cursor movement)
07 -- two fingers (double click)

B. CNN ResNet Model

The CNN model which is trained is of sequential type. That means the model is built layer by layer. Training and testing batches are made and data augmentation is done with the help of ImageDataGenerator. The batch size is kept as 32 and it is shuffled. While pre-processing, the images are converted into arrays. The images are resized to 64 x64. The class mode is set to categorical and not binary as the images will classified into different classes.

The ResNet-101 architecture is a deep convolutional neural network introduced in the ResNet (Residual Networks) paper by He et al. in 2015. It is a deeper variant of ResNet that includes 101 layers, primarily using residual blocks to help train very deep networks effectively.

The ResNet-101 architecture is a deep convolutional neural network consisting of 101 layers, designed to effectively train very deep models using the concept of residual learning. It starts with an input layer that typically accepts images of size 224 × 224 × 3 (RGB). The initial layers include a 7×7 convolutional layer with 64 filters and a stride of 2, followed by a 3×3 max pooling layer with a stride of 2, which reduces the spatial dimensions. The core of ResNet-101 is built using multiple residual blocks that follow a bottleneck design. Each of these blocks includes a 1×1 convolution for dimensionality reduction, a 3×3 convolution for spatial processing, and another 1×1 convolution to restore dimensionality, followed by a skip connection and ReLU activation.

The architecture is organized into several stages: Conv2_x with 3 blocks, Conv3_x with 4 blocks, Conv4_x with 23 blocks, and Conv5_x with 3 blocks, making a total of 33 bottleneck blocks. Since each bottleneck contains three convolutional layers, this results in 99 layers, and including the initial convolution and the final fully connected layer, the model totals 101 layers. Following the residual blocks, a global average pooling layer is used, followed by a fully connected layer, typically with 1000 output nodes for classification tasks like ImageNet, and a softmax activation function. The key strengths of ResNet-101 lie in its use of skip connections to prevent vanishing gradients and its efficient bottleneck structure that maintains performance while reducing computational cost.

III. Accessing Mouse virtually

Before you begin to format your paper, first write and save the content as a separate text file. Complete all content and organizational editing before formatting. Please note sections A-D below for more information on proofreading, spelling and grammar.

Keep your text and graphic files separate until after the text has been formatted and styled. Do not use hard tabs, and limit use of hard returns to only one return at the end of a paragraph. Do not add any kind of pagination anywhere in the paper. Do not number text heads-the template will do that for you.

IV. RESULTS

This project has successfully implemented virtual mouse using convolutional neural network based on hand gesture recognition After training, the ResNet101 CNN model has obtained the training accuracy of 94.32% and testing accuracy of 96.875%

V. CONCLUSION

With technology advancing every day, improvements are made in each and every aspect. In this paper, we have developed a virtual mouse which is based on CNN. The user's hand is detected and the gestures of hand are identified by CNN and according to the gestures different operations are carried out. The detection of hand is not affected by the lighting conditions and background thus, overcoming the limitations of the previous systems. The developed system has quite high accuracy and is suitable for real-time use.

VI. APPLICATION AND FUTURE WORK

From future perspective, we can incorporate more mouse operations thus widening the scope of the project. By using different algorithms, we can also strive to increase the accuracy of the results. Apart from the virtual mouse system mentioned in the paper, this model can be combined with other technologies to provide support to differently-abled people. For e.g., sign language and communication. It can also be used in the gaming field. Further eye movements can be used to control the mouse which will be much more beneficial.