

# **SHIASH INFO SOLUTIONS**

## **PROJECT TITLE**

**HAND GESTURE RECOGNITION USING PRE - TRAINED  
MODEL**

**BATCH NO: 1**

**DOMAIN: DATA SCIENCE**

**DONE BY: HARISHVARAN M B.E(CSE)**

# CONTENTS

I	Abstract	3
1	Introduction	3
2	Background	4
3	Problem statement	5
4	Objectives	5
II	System architecture	6
III	Functionality	7
IV	Conclusion	8
V	Source code and Screenshot	9

# ABSTRACT

## INTRODUCTION

Hand gesture recognition is the process of interpreting and understanding the gestures and movements of a human hand to perform tasks or communicate with a computer system. This technology finds applications in various domains, including sign language interpretation, human-computer interaction, and virtual reality.

Pretrained models are neural networks that have been trained on extensive datasets for a specific task, such as image classification or object detection. Using pretrained models for hand gesture recognition is advantageous because they have already learned useful features and patterns from vast amounts of data, which can significantly speed up the development process and improve accuracy.

Hand gesture recognition is a fascinating and practical application of computer vision and deep learning. It involves recognizing and understanding hand movements and poses from images or video streams. Pretrained models have become instrumental in this field as they allow developers to leverage the knowledge gained from vast datasets and powerful neural networks.

In conclusion, hand gesture recognition using pretrained models is a powerful application of deep learning that can find utility in various domains. It involves collecting and preprocessing data, selecting appropriate pretrained models, fine-tuning them, and deploying them in real-world applications. With the advancements in deep learning and computer vision, this technology has the potential to revolutionize human-computer interaction and accessibility.

# BACKGROUND

Hand gesture recognition is a compelling field within computer vision and artificial intelligence that has garnered significant attention in recent years. The ability to interpret and understand hand movements and gestures has numerous applications in various domains, ranging from human-computer interaction to healthcare and gaming. This project aims to leverage the power of pretrained deep learning models to develop an accurate and real-time hand gesture recognition system.

The motivation behind this project is driven by the need for more intuitive and natural interfaces between humans and machines. Traditional input methods, such as keyboards and mouse devices, can be limiting and less accessible for certain user groups. Hand gesture recognition offers a more natural and inclusive way for users to interact with computers and other devices. Additionally, the widespread adoption of smartphones, augmented reality (AR), and virtual reality (VR) technologies has created a demand for robust and efficient hand gesture recognition systems.

## **Deep Learning:**

Deep learning techniques, specifically convolutional neural networks (CNNs), are at the core of hand gesture recognition. These neural networks can automatically learn and extract meaningful features from images, making them well-suited for image-based recognition tasks.

# OBJECTIVE

The primary objective of the hand gesture recognition project utilizing pretrained models is to develop a robust and efficient system capable of accurately interpreting and responding to hand gestures in real-time. This system aims to achieve a high level of accuracy in recognizing a variety of hand gestures, ensuring that it can generalize well to different hand shapes and poses. It should offer a user-friendly interface to facilitate seamless interaction, even for non-technical users.

Additionally, the system must demonstrate resilience to environmental challenges such as varying lighting conditions and occlusions. Multiclass recognition, low hardware requirements, and comprehensive documentation are crucial aspects of this project. Furthermore, the project aims to prioritize user privacy and data security. Continuous improvement and scalability, along with rigorous testing and evaluation, are integral to ensuring the system's reliability and adaptability to various applications and usage scenarios.

# PROBLEM STATEMENT

The problem at hand is to develop a hand gesture recognition system using pretrained deep learning models to enhance human-computer interaction and accessibility in diverse applications. This system should be capable of accurately recognizing a range of hand gestures in real-time, accommodating variations in hand poses, lighting conditions, and backgrounds, while ensuring low latency and efficient use of computing resources. The goal is to create a user-friendly and versatile system that can be easily integrated into different platforms, from smartphones to augmented reality environments, with a focus on privacy and data security.

# SYSTEM ARCHITECTURE

Data Collection

Data Preprocessing

Feature Extraction

Machine Learning Model

Model Training

Model Testing

Model Accuracy

# FUNCTIONALITY

## **Gesture Detection:**

Capture input from a camera or sensor to continuously monitor the user's hand movements. Integrate the pretrained deep learning model, such as a convolutional neural network that has been fine-tuned for hand gesture recognition.

## **Gesture Classification:**

Employ the pretrained model to classify hand gestures into predefined categories or actions. This involves mapping the model's output of specific gestures. Support recognition of multiple hand gestures or signs, allowing for a wide range of interactions or commands.

## **Model Generalization:**

Ensure that the system can recognize gestures accurately across various hand shapes, sizes, orientations, and lighting conditions. Implement techniques like data augmentation during training to improve generalization.

## **Model Training and Retraining:**

Set up mechanisms for initial model training and periodic retraining. Continuous model updates ensure adaptability to evolving spam tactics.

# CONCLUSION

Hand gesture recognition represents a promising and rapidly evolving field within computer vision and artificial intelligence. This technology holds the potential to revolutionize how humans interact with machines and opens doors to a multitude of applications across various domains. By leveraging pretrained deep learning models and sophisticated algorithms, we can achieve accurate and real-time recognition of hand gestures, making interactions more intuitive and accessible.

As the field continues to advance, it is essential to focus on robustness, adaptability, and user-friendliness. Ensuring that recognition systems can handle diverse hand poses, lighting conditions, and backgrounds while protecting user privacy and data security is paramount. Additionally, the ability to continuously learn and adapt to user-specific gestures and preferences will further enhance the utility and acceptance of these systems.

Hand gesture recognition systems offer practical solutions for sign language translation, immersive gaming experiences, virtual reality interactions, and much more. These systems have the ability to bridge communication gaps, improve accessibility for individuals with disabilities, and enhance user experiences in interactive digital environments.

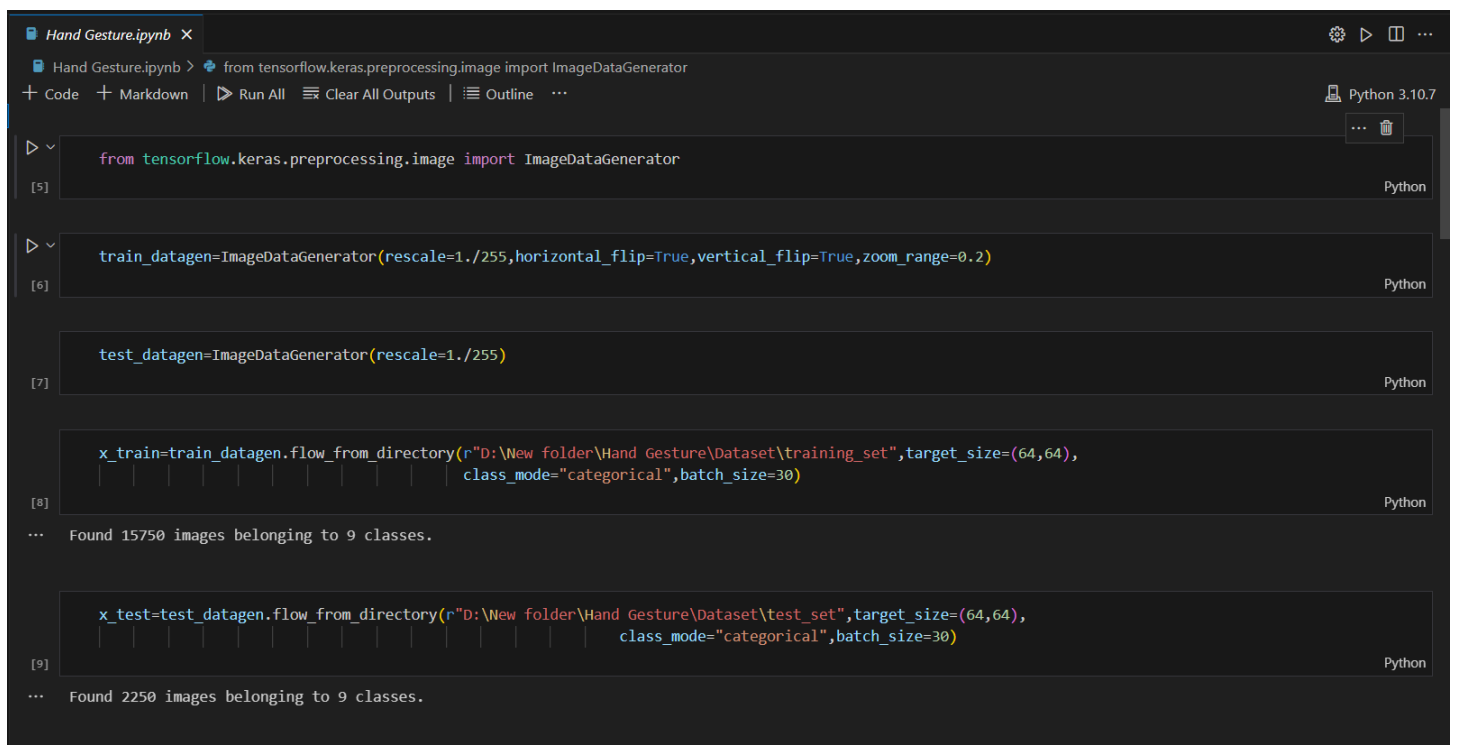


# SOURCE CODE

Here, I provided my GITHUB link contained the files of source code for the project with the Hand gesture recognition using pre trained model link

GITHUB LINK: [https://github.com/Harish290/IBM-Project-30646-1660151961/tree/main/Hand%20Gesture\(Harishvaran\)July-Sep%20\(12-1.30%20batch\)](https://github.com/Harish290/IBM-Project-30646-1660151961/tree/main/Hand%20Gesture(Harishvaran)July-Sep%20(12-1.30%20batch))

# SCREENSHOT



```
Hand Gesture.ipynb X
Hand Gesture.ipynb > from tensorflow.keras.preprocessing.image import ImageDataGenerator
+ Code + Markdown | ▶ Run All | ➡ Clear All Outputs | 📄 Outline ... Python 3.10.7

[5] from tensorflow.keras.preprocessing.image import ImageDataGenerator
Python

[6] train_datagen=ImageDataGenerator(rescale=1./255,horizontal_flip=True,vertical_flip=True,zoom_range=0.2)
Python

[7] test_datagen=ImageDataGenerator(rescale=1./255)
Python

[8] x_train=train_datagen.flow_from_directory(r"D:\New folder\Hand Gesture\Dataset\training_set",target_size=(64,64),
class_mode="categorical",batch_size=30)
Python

... Found 15750 images belonging to 9 classes.

[9] x_test=test_datagen.flow_from_directory(r"D:\New folder\Hand Gesture\Dataset\test_set",target_size=(64,64),
class_mode="categorical",batch_size=30)
Python

... Found 2250 images belonging to 9 classes.
```

Hand Gesture.ipynb

Hand Gesture.ipynb > from tensorflow.keras.preprocessing.image import ImageDataGenerator

+ Code + Markdown | ▶ Run All ≡ Clear All Outputs | ≡ Outline ...

Python 3.10.7

# Model Building

```
[10] from keras.models import Sequential
      from keras.layers import Dense
      from keras.layers import Convolution2D
      from keras.layers import MaxPooling2D
      from keras.layers import Dropout
      from keras.layers import Flatten
```

```
[11] model=Sequential()
```

```
[12] model.add(Convolution2D(32,(3,3),activation="relu",input_shape=(64,64,3)))
```

```
[13] model.add(MaxPooling2D(pool_size=(2,2)))
```

```
[14] model.add(Flatten())
```

```
model.add(Dense(200,activation='relu'))
```

Cell 1 of 29

```
Hand Gesture.ipynb X
Hand Gesture.ipynb > from tensorflow.keras.preprocessing.image import ImageDataGenerator
+ Code + Markdown | Run All | Clear All Outputs | Outline ... Python 3.10.7

[15] model.add(Dense(200,activation='relu')) Python

[16] model.add(Dense(9,activation="softmax")) Python

[17] model.compile(loss="categorical_crossentropy",metrics=["accuracy"],optimizer='adam') Python

[18] len(x_train) Python
... 525

[19] len(x_test) Python
... 75

[20] x_train.class_indices Python
... {'A': 0, 'B': 1, 'C': 2, 'D': 3, 'E': 4, 'F': 5, 'G': 6, 'H': 7, 'I': 8}
```

```
Hand Gesture.ipynb X
Hand Gesture.ipynb > from tensorflow.keras.preprocessing.image import ImageDataGenerator
+ Code + Markdown | Run All | Clear All Outputs | Outline ... Python 3.10.7

[21] model.fit(x_train,epochs=1,validation_data=x_test,steps_per_epoch=len(x_train),validation_steps=len(x_test)) Python
... 525/525 [=====] - 87s 164ms/step - loss: 0.3325 - accuracy: 0.8909 - val_loss: 0.2924 - val_accuracy: 0.9444
... <keras.callbacks.History at 0x28e945d6350>

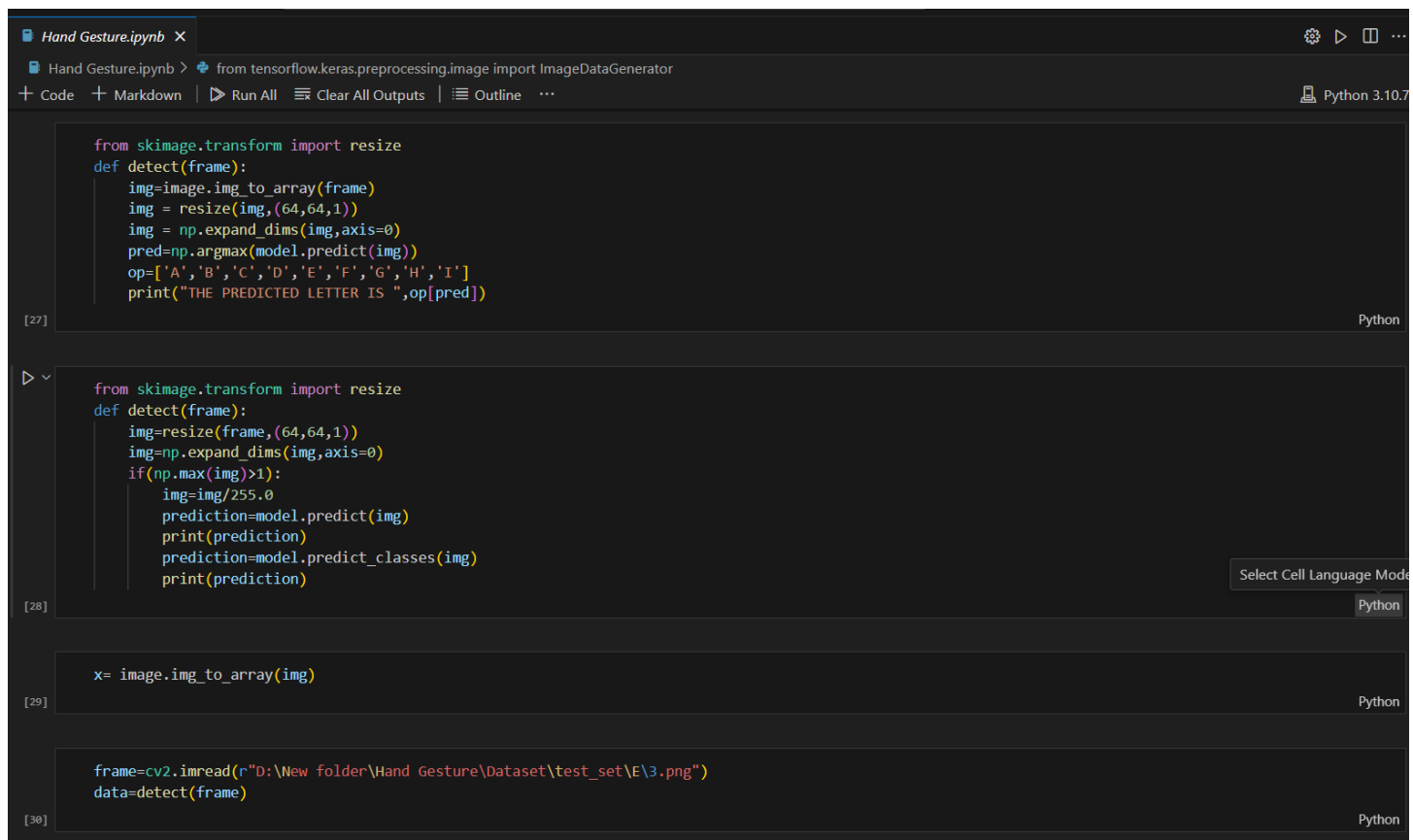
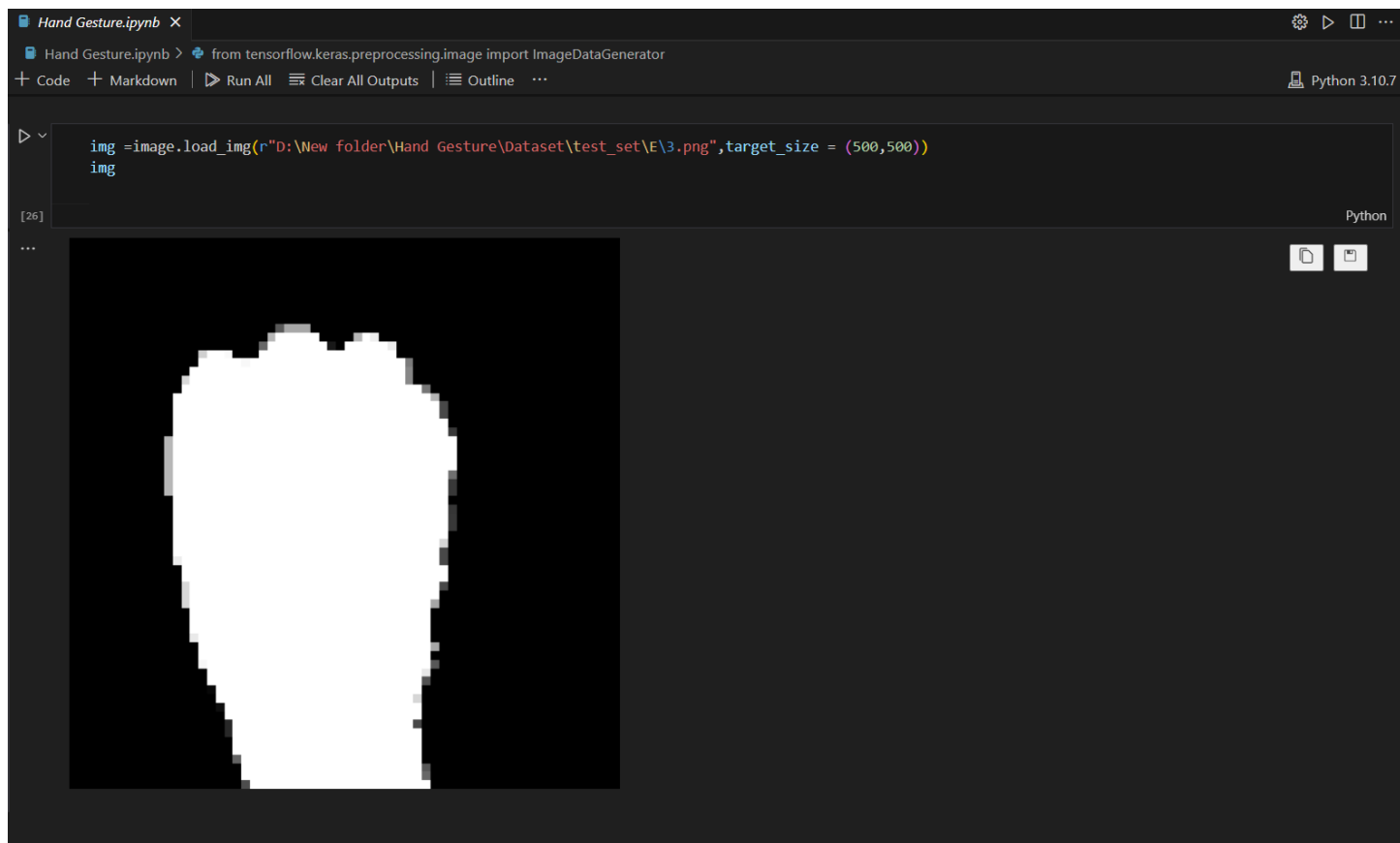
[22] model.save('Model.h5') Python

[23] from keras.models import load_model
import numpy as np
import h5py
import cv2 Python

[24] from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
import numpy as np Python

[25] model = load_model('Model.h5') Python

img = image.load_img(r"D:\New folder\Hand Gesture\Dataset\test_set\E\3.png",target_size = (500,500))
```



```
Hand Gesture.ipynb X
Hand Gesture.ipynb > from tensorflow.keras.preprocessing.image import ImageDataGenerator
+ Code + Markdown + Run All Clear All Outputs Outline ... Python 3.10.7

cv2.imshow("frame",frame)
cv2.waitKey(0)
cv2.destroyAllWindows()

[31] Python

import cv2
import numpy as np
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
model=load_model('Model.h5')
video=cv2.VideoCapture(0)
index=['A','B','C','D','E','F','G','H','I']
while 1:
    succes,frame=video.read()
    cv2.imwrite('image.jpg',frame)
    img=image.load_img('image.jpg',target_size=(64,64))
    x=image.img_to_array(img)
    x=np.expand_dims(x,axis=0)
    pred=np.argmax(model.predict(x),axis=1)
    y=pred[0]
    copy = frame.copy()
    cv2.rectangle(copy, (320, 100), (620,400), (255,0,0), 5)
    cv2.putText(frame,'The Predicted Alphabet is: '+str(index[y]),(100,100),cv2.FONT_HERSHEY_SIMPLEX,1,(0,0,0),4)
    cv2.imshow('image',frame)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break
video.release()
cv2.destroyAllWindows()

[1] Python

... 1/1 [=====] - 0s 238ms/step
1/1 [=====] - 0s 127ms/step
1/1 [=====] - 0s 52ms/step
```