Import The Packages And Load The Saved Model

Import The Required Model Building Libraries

```
#import imagedatagenerator
from keras.preprocessing.image import ImageDataGenerator
```

```
#training datagen
train_datagen=ImageDataGenerator(rescale=1./255,shear_range=0.2,zoom_range=0.2,horizontal_
```

```
#testing datagen
test_datagen=ImageDataGenerator(rescale=1./255)
```

Import tensorflow

```
import tensorflow as tf
import os
```

Initialize The Model

```
#create model
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
from tensorflow.keras.preprocessing.image import ImageDataGenerator
```

```
import numpy as np
import matplotlib.pyplot as plt #to view graph in colab itself
import IPython.display as display
from PIL import Image
import pathlib
```

Unzipping the dataset

```
!unzip '/content/conversation engine for deaf and dumb (1).zip'
```

```
      extracting: Dataset/training_set/G/1225.png
      extracting: Dataset/training_set/G/1226.png
      extracting: Dataset/training_set/G/1227.png
      extracting: Dataset/training_set/G/1228.png
```

```
   extracting: Dataset/training_set/G/1229.png  
    inflating: Dataset/training_set/G/123.png  
   extracting: Dataset/training_set/G/1230.png  
   extracting: Dataset/training_set/G/1231.png  
   extracting: Dataset/training_set/G/1232.png  
    inflating: Dataset/training_set/G/1233.png  
    inflating: Dataset/training_set/G/1234.png  
    inflating: Dataset/training_set/G/1235.png  
    inflating: Dataset/training_set/G/1236.png  
    inflating: Dataset/training_set/G/1237.png  
    inflating: Dataset/training_set/G/1238.png  
    inflating: Dataset/training_set/G/1239.png  
    inflating: Dataset/training_set/G/124.png  
    inflating: Dataset/training_set/G/1240.png  
    inflating: Dataset/training_set/G/1241.png  
    inflating: Dataset/training_set/G/1242.png  
    inflating: Dataset/training_set/G/1243.png  
    inflating: Dataset/training_set/G/1244.png  
    inflating: Dataset/training_set/G/1245.png  
   extracting: Dataset/training_set/G/1246.png  
    inflating: Dataset/training_set/G/1247.png  
    inflating: Dataset/training_set/G/1248.png  
    inflating: Dataset/training_set/G/1249.png  
    inflating: Dataset/training_set/G/125.png  
    inflating: Dataset/training_set/G/1250.png  
    inflating: Dataset/training_set/G/1251.png  
    inflating: Dataset/training_set/G/1252.png  
    inflating: Dataset/training_set/G/1253.png  
    inflating: Dataset/training_set/G/1254.png  
    inflating: Dataset/training_set/G/1255.png  
    inflating: Dataset/training_set/G/1256.png  
    inflating: Dataset/training_set/G/1257.png  
    inflating: Dataset/training_set/G/1258.png  
    inflating: Dataset/training_set/G/1259.png  
    inflating: Dataset/training_set/G/126.png  
    inflating: Dataset/training_set/G/1260.png  
    inflating: Dataset/training_set/G/1261.png  
   extracting: Dataset/training_set/G/1262.png  
    inflating: Dataset/training_set/G/1263.png  
    inflating: Dataset/training_set/G/1264.png  
    inflating: Dataset/training_set/G/1265.png  
    inflating: Dataset/training_set/G/1266.png  
    inflating: Dataset/training_set/G/1267.png  
   extracting: Dataset/training_set/G/1268.png  
    inflating: Dataset/training_set/G/1269.png  
    inflating: Dataset/training_set/G/127.png  
    inflating: Dataset/training_set/G/1270.png  
    inflating: Dataset/training_set/G/1271.png  
    inflating: Dataset/training_set/G/1272.png  
    inflating: Dataset/training_set/G/1273.png  
    inflating: Dataset/training_set/G/1274.png  
    inflating: Dataset/training_set/G/1275.png  
    inflating: Dataset/training_set/G/1276.png  
    inflating: Dataset/training_set/G/1277
```

Applying ImageDataGenerator to training set

```
x_train=train_datagen.flow_from_directory('/content/Dataset/training_set',target_size=(64,
                                          class_mode='categorical',color_mode="grayscale")
```

```
    Found 15750 images belonging to 9 classes.
```

Applying ImageDataGenerator to test set

```
x_test=test_datagen.flow_from_directory('/content/Dataset/test_set',target_size=(64,64),ba
                                          class_mode='categorical',color_mode="grayscale")
```

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

```
a=len(x_train)
b=len(x_test)
```

Length of training set

```
print(a)
```

```
    79
```

Length of test set

```
print(b)
```

```
    12
```

Add Layers

```
#create model
model=Sequential()
```

Add The Convolution Layer

```
model.add(Convolution2D(32,(3,3),input_shape=(64,64,1),activation='relu'))
```

Add Pooling Layer

```
model.add(MaxPooling2D(pool_size=(2,2)))
```

Add The Flatten Layer

```
model.add(Flatten())
```

Adding The Dense Layers

```
#1st hidden layer
model.add(Dense(units=512,activation='relu'))
#2nd hidden layer
model.add(Dense(units=261,activation='relu'))
```

```
#output layer
model.add(Dense(units=9,activation='softmax'))
```

Compile The Model

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

Fit The Model

```
model.fit_generator(x_train,steps_per_epoch=len(x_train),epochs=10,validation_data=x_test,
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:1: UserWarning: `Model.f
  """Entry point for launching an IPython kernel.
Epoch 1/10
79/79 [==============================] - 90s 1s/step - loss: 0.3965 - accuracy: 0.874
Epoch 2/10
79/79 [==============================] - 86s 1s/step - loss: 0.0419 - accuracy: 0.988
Epoch 3/10
79/79 [==============================] - 84s 1s/step - loss: 0.0195 - accuracy: 0.994
Epoch 4/10
79/79 [==============================] - 87s 1s/step - loss: 0.0083 - accuracy: 0.998
Epoch 5/10
79/79 [==============================] - 83s 1s/step - loss: 0.0066 - accuracy: 0.998
Epoch 6/10
79/79 [==============================] - 88s 1s/step - loss: 0.0072 - accuracy: 0.997
Epoch 7/10
79/79 [==============================] - 86s 1s/step - loss: 0.0059 - accuracy: 0.998
Epoch 8/10
79/79 [==============================] - 86s 1s/step - loss: 0.0027 - accuracy: 0.999
Epoch 9/10
79/79 [==============================] - 84s 1s/step - loss: 0.0073 - accuracy: 0.998
Epoch 10/10
79/79 [==============================] - 85s 1s/step - loss: 0.0048 - accuracy: 0.998
<keras.callbacks.History at 0x7f445adcd7d0>
```

Save The Model

```
model.save('aslpng2.h5')
```

Import The Packages And Load The Saved Model

```
from tensorflow.keras.models import load_model
import numpy as np
import cv2
from tensorflow.keras.preprocessing import image
```

```
#load the model
model=load_model('aslpng2.h5')
```

```
img=image.load_img('/content/Dataset/test_set/A/10.png',target_size=(400,500))
img
```

Load The Test Image, Pre-Process It And Predict

```
from skimage.transform import resize
def detect(frame):
  img=resize(frame,(64,64,1))
  img=np.expand_dims(img,axis=0)
  if(np.max(img)>1):
    prediction=model.predict(img)
    print(prediction)
    prediction=model.predict_classes(img)
    print(prediction)
```

```
arr= image.img_to_array(img)
```

```
frame=cv2.imread('/content/Dataset/test_set/A/10.png')
data=detect(frame)
from google.colab.patches import cv2_imshow
```

```
cv2_imshow(frame)
cv2.waitKey(0)
cv2.destroyAllWindows()
```