TEAM ID:PNT2022TMID41505

```
from keras.preprocessing.image import
ImageDataGeneratortrain_datagen=ImageDataGenerator(rescale=1./255,s
hear_range=0.2,zoom_range=0.2,horizontal_flip=True)test_datagen=Ima
geDataGenerator(rescale=1./255)
```

In [3]:

```
x_train =
train_datagen.flow_from_directory(’/content/Dataset/training_set’,t
arget_size=(64,64),batch_size=300,class_mode=’categorical’,color_mo
de=″grayscale″)
```

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

In [4]:

```
x_test =
test_datagen.flow_from_directory(’/content/Dataset/test_set’,target
_size=(64,64),batch_size=300,class_mode=’categorical’,color_mode=″g
rayscale″)
```

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

In [5]:

```
from keras.models import Sequentialfrom keras.layers import
Densefrom keras.layers import Convolution2Dfrom keras.layers import
MaxPooling2Dfrom keras.layers import Dropoutfrom keras.layers
import Flatten
```

In [6]:

```
model = Sequential()
```

In [7]:

```
model.add(Convolution2D(32,(3,3),input_shape=(64,64,1),
activation=’relu’))#no. of feature detectors, size of feature
detector, image size, activation function
```

In [8]:

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

In [9]:

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

In [10]:

```
model.add(Dense(units=512, activation = ’relu’))
```

In [11]:

```
model.add(Dense(units=9,  activation = ’softmax’))
```

In [12]:

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

In [13]:

```
model.fit_generator(x_train,steps_per_epoch=24,epochs=10,validation
_data = x_test, validation_steps= 40)#steps_per_epoch = no. of
train images//batch size
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:1: UserW
arning: `Model.fit_generator` is deprecated and will be removed in a
future version. Please use `Model.fit`, which supports generators.
  """Entry point for launching an IPython kernel.
Epoch 1/10
24/24 [==============================] - ETA: 0s - loss: 1.2714 - acc
uracy: 0.6219
WARNING:tensorflow:Your input ran out of data; interrupting training.
 Make sure that your dataset or generator can generate at least `step
s_per_epoch * epochs` batches (in this case, 40 batches). You may nee
d to use the repeat() function when building your dataset.
24/24 [==============================] - 41s 2s/step - loss: 1.2714 -
 accuracy: 0.6219 - val_loss: 0.4031 - val_accuracy: 0.8982
Epoch 2/10
24/24 [==============================] - 33s 1s/step - loss: 0.2827 -
 accuracy: 0.9211
Epoch 3/10
24/24 [==============================] - 34s 1s/step - loss: 0.1448 -
 accuracy: 0.9615
Epoch 4/10
24/24 [==============================] - 32s 1s/step - loss: 0.0958 -
 accuracy: 0.9746
Epoch 5/10
24/24 [==============================] - 34s 1s/step - loss: 0.0679 -
 accuracy: 0.9826
Epoch 6/10
24/24 [==============================] - 32s 1s/step - loss: 0.0424 -
 accuracy: 0.9909
Epoch 7/10
```

```
24/24 [==============================] - 32s 1s/step - loss: 0.0373 -
 accuracy: 0.9908
Epoch 8/10
24/24 [==============================] - 33s 1s/step - loss: 0.0319 -
 accuracy: 0.9915
Epoch 9/10
24/24 [==============================] - 32s 1s/step - loss: 0.0235 -
 accuracy: 0.9940
Epoch 10/10
24/24 [==============================] - 32s 1s/step - loss: 0.0170 -
 accuracy: 0.9972
```

Out[13]:

In [14]:

```
model.save(’aslpng1.h5’)
```

In [17]:

```
from keras.models import load_modelimport numpy as npimport cv2
```

In [18]:

```
model=load_model(’aslpng1.h5’)
```

In [25]:

```
from skimage.transform import resizedef detect(frame):
  img = resize(frame,(64,64,1))
  img = np.expand_dims(img,axis=0)
  if(np.max(img)>1):
    img = img/255.0
  prediction = model.predict(img)
  print(prediction)
  prediction = np.argmax(prediction,axis=1)
  print(prediction)
```

In [26]:

```
frame=cv2.imread(’/content/Dataset/test_set/G/1.png’)data =
detect(frame)
```

```
1/1 [==============================] - 0s 25ms/step
[[2.9662006e-09 3.0511607e-09 5.7518361e-07 2.6636766e-09 7.6029876e-
09
  1.4324395e-08 9.9982303e-01 1.7639149e-04 1.6517550e-09]]
[6]
```

```
from keras.preprocessing.image import
ImageDataGeneratortrain_datagen=ImageDataGenerator(rescale=1./255,s
hear_range=0.2,zoom_range=0.2,horizontal_flip=True)test_datagen=Ima
geDataGenerator(rescale=1./255)
```

In [3]:

```
x_train =
train_datagen.flow_from_directory(’/content/Dataset/training_set’,t
arget_size=(64,64),batch_size=300,class_mode=’categorical’,color_mo
de=″grayscale″)
```

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

In [4]:

```
x_test =
test_datagen.flow_from_directory(’/content/Dataset/test_set’,target
_size=(64,64),batch_size=300,class_mode=’categorical’,color_mode=″g
rayscale″)
```

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

In [5]:

```
from keras.models import Sequentialfrom keras.layers import
Densefrom keras.layers import Convolution2Dfrom keras.layers import
MaxPooling2Dfrom keras.layers import Dropoutfrom keras.layers
import Flatten
```

In [6]:

```
model = Sequential()
```

In [7]:

```
model.add(Convolution2D(32,(3,3),input_shape=(64,64,1),
activation=’relu’))#no. of feature detectors, size of feature
detector, image size, activation function
```

In [8]:

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

In [9]:

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

In [10]:

```
model.add(Dense(units=512, activation = ’relu’))
```

In [11]:

```
model.add(Dense(units=9,  activation = ’softmax’))
```

In [12]:

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

In [13]:

```
model.fit_generator(x_train,steps_per_epoch=24,epochs=10,validation
_data = x_test, validation_steps= 40)#steps_per_epoch = no. of
train images//batch size
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:1: UserW
arning: `Model.fit_generator` is deprecated and will be removed in a
future version. Please use `Model.fit`, which supports generators.
  """Entry point for launching an IPython kernel.
Epoch 1/10
24/24 [==============================] - ETA: 0s - loss: 1.2714 - acc
uracy: 0.6219
WARNING:tensorflow:Your input ran out of data; interrupting training.
 Make sure that your dataset or generator can generate at least `step
s_per_epoch * epochs` batches (in this case, 40 batches). You may nee
d to use the repeat() function when building your dataset.
24/24 [==============================] - 41s 2s/step - loss: 1.2714 -
 accuracy: 0.6219 - val_loss: 0.4031 - val_accuracy: 0.8982
Epoch 2/10
24/24 [==============================] - 33s 1s/step - loss: 0.2827 -
 accuracy: 0.9211
Epoch 3/10
24/24 [==============================] - 34s 1s/step - loss: 0.1448 -
 accuracy: 0.9615
Epoch 4/10
24/24 [==============================] - 32s 1s/step - loss: 0.0958 -
 accuracy: 0.9746
Epoch 5/10
24/24 [==============================] - 34s 1s/step - loss: 0.0679 -
 accuracy: 0.9826
Epoch 6/10
24/24 [==============================] - 32s 1s/step - loss: 0.0424 -
 accuracy: 0.9909
Epoch 7/10
24/24 [==============================] - 32s 1s/step - loss: 0.0373 -
 accuracy: 0.9908
Epoch 8/10
```

```
24/24 [==============================] - 33s 1s/step - loss: 0.0319 -
 accuracy: 0.9915
Epoch 9/10
24/24 [==============================] - 32s 1s/step - loss: 0.0235 -
 accuracy: 0.9940
Epoch 10/10
24/24 [==============================] - 32s 1s/step - loss: 0.0170 -
 accuracy: 0.9972
```

Out[13]:

In [14]:

```
model.save(’aslpng1.h5’)
```

In [17]:

```
from keras.models import load_modelimport numpy as npimport cv2
```

In [18]:

```
model=load_model(’aslpng1.h5’)
```

In [25]:

```
from skimage.transform import resizedef detect(frame):
  img = resize(frame,(64,64,1))
  img = np.expand_dims(img,axis=0)
  if(np.max(img)>1):
    img = img/255.0
  prediction = model.predict(img)
  print(prediction)
  prediction = np.argmax(prediction,axis=1)
  print(prediction)
```

In [26]:

```
frame=cv2.imread(’/content/Dataset/test_set/G/1.png’)data =
detect(frame)
```

```
1/1 [==============================] - 0s 25ms/step
[[2.9662006e-09 3.0511607e-09 5.7518361e-07 2.6636766e-09 7.6029876e-
09
  1.4324395e-08 9.9982303e-01 1.7639149e-04 1.6517550e-09]]
[6]
```

```
from keras.preprocessing.image import
ImageDataGeneratortrain_datagen=ImageDataGenerator(rescale=1./255,s
hear_range=0.2,zoom_range=0.2,horizontal_flip=True)test_datagen=Ima
geDataGenerator(rescale=1./255)
```

In [3]:

```
x_train =
train_datagen.flow_from_directory('/content/Dataset/training_set',t
arget_size=(64,64),batch_size=300,class_mode='categorical',color_mo
de="grayscale")
```

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

In [4]:

```
x_test =
test_datagen.flow_from_directory('/content/Dataset/test_set',target
_size=(64,64),batch_size=300,class_mode='categorical',color_mode="g
rayscale")
```

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

In [5]:

```
from keras.models import Sequentialfrom keras.layers import
Densefrom keras.layers import Convolution2Dfrom keras.layers import
MaxPooling2Dfrom keras.layers import Dropoutfrom keras.layers
import Flatten
```

In [6]:

```
model = Sequential()
```

In [7]:

```
model.add(Convolution2D(32,(3,3),input_shape=(64,64,1),
activation='relu'))#no. of feature detectors, size of feature
detector, image size, activation function
```

In [8]:

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

In [9]:

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

In [10]:

```
model.add(Dense(units=512, activation = 'relu'))
```

In [11]:

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

In [12]:

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

In [13]:

```
model.fit_generator(x_train,steps_per_epoch=24,epochs=10,validation_data = x_test, validation_steps= 40)#steps_per_epoch = no. of train images//batch size
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:1: UserWarning: `Model.fit_generator` is deprecated and will be removed in a future version. Please use `Model.fit`, which supports generators.
  """Entry point for launching an IPython kernel.
Epoch 1/10
24/24 [==============================] - ETA: 0s - loss: 1.2714 - accuracy: 0.6219
WARNING:tensorflow:Your input ran out of data; interrupting training. Make sure that your dataset or generator can generate at least `steps_per_epoch * epochs` batches (in this case, 40 batches). You may need to use the repeat() function when building your dataset.
24/24 [==============================] - 41s 2s/step - loss: 1.2714 - accuracy: 0.6219 - val_loss: 0.4031 - val_accuracy: 0.8982
Epoch 2/10
24/24 [==============================] - 33s 1s/step - loss: 0.2827 - accuracy: 0.9211
Epoch 3/10
24/24 [==============================] - 34s 1s/step - loss: 0.1448 - accuracy: 0.9615
Epoch 4/10
24/24 [==============================] - 32s 1s/step - loss: 0.0958 - accuracy: 0.9746
Epoch 5/10
24/24 [==============================] - 34s 1s/step - loss: 0.0679 - accuracy: 0.9826
Epoch 6/10
24/24 [==============================] - 32s 1s/step - loss: 0.0424 - accuracy: 0.9909
Epoch 7/10
24/24 [==============================] - 32s 1s/step - loss: 0.0373 - accuracy: 0.9908
Epoch 8/10
24/24 [==============================] - 33s 1s/step - loss: 0.0319 - accuracy: 0.9915
Epoch 9/10
```

```
24/24 [==============================] - 32s 1s/step - loss: 0.0235 -
 accuracy: 0.9940
Epoch 10/10
24/24 [==============================] - 32s 1s/step - loss: 0.0170 -
 accuracy: 0.9972
```

Out[13]:

In [14]:

```
model.save('aslpng1.h5')
```

In [17]:

```
from keras.models import load_modelimport numpy as npimport cv2
```

In [18]:

```
model=load_model('aslpng1.h5')
```

In [25]:

```
from skimage.transform import resizedef detect(frame):
  img = resize(frame,(64,64,1))
  img = np.expand_dims(img,axis=0)
  if(np.max(img)>1):
    img = img/255.0
  prediction = model.predict(img)
  print(prediction)
  prediction = np.argmax(prediction,axis=1)
  print(prediction)
```

In [26]:

```
frame=cv2.imread('/content/Dataset/test_set/G/1.png')data =
detect(frame)
```

```
1/1 [==============================] - 0s 25ms/step
[[2.9662006e-09 3.0511607e-09 5.7518361e-07 2.6636766e-09 7.6029876e-
09
  1.4324395e-08 9.9982303e-01 1.7639149e-04 1.6517550e-09]]
[6]
```