Real-Time Communication System Powered By AI For Specially Abled

TEAM ID - 2022TMID21808

Image Preprocessing

Import ImageDataGenerator Library And Configure It

In [1]:
```
from tensorflow.keras.preprocessing.image import ImageDataGenerator
```

In [2]:
```
train_datagen=ImageDataGenerator(rescale=1./255,horizontal_flip=True,vertical_flip=True,zoom_range=0.2)
```

In [3]:
```
test_datagen=ImageDataGenerator(rescale=1./255)
```

Apply ImageDataGenerator Functionality To Train And Test Set

In [4]:
```
x_train=train_datagen.flow_from_directory(r"C:\Users\rajes\Desktop\Dataset\training_set",target_size=(64,64
                                          class_mode="categorical",batch_size=30)
```

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

In [5]:
```
x_test=test_datagen.flow_from_directory(r"C:\Users\rajes\Desktop\Dataset\test_set",target_size=(64,64),
                                        class_mode="categorical",batch_size=30)
```

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

Model Building

Import The Required Model Building Libraries

In [6]:
```
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
```

Initialize The Model

In [7]:
```
model=Sequential()
```

Add The Convolution Layer

In [10]:
```
model.add(Convolution2D(32,(3,3),activation="relu",input_shape=(64,64,3)))
#No of feature detectors, size of feature detector, image size, activation function
```

Add The Pooling Layer

In [11]:
```
model.add(MaxPooling2D(pool_size=(2,2)))
```

Add The Flatten Layer

In [12]:
```
model.add(Flatten())
```

Adding The Dense Layers

In [13]:
```
model.add(Dense(200,activation='relu'))
```

In [15]:
```
model.add(Dense(9,activation="softmax"))
```

Compile The Model

In [16]:
```
model.compile(loss="categorical_crossentropy",metrics=["accuracy"],optimizer='adam')
```

In [17]:
```
len(x_train)
```

Out[17]: 525

In [18]:
```
len(x_test)
```

Out[18]: 75

Fit And Save The Model

In [19]:
```
model.fit(x_train,epochs=9,validation_data=x_test,steps_per_epoch=len(x_train),validation_steps=len(x_test))
```

```
Epoch 1/9
525/525 [==============================] - 260s 471ms/step - loss: 0.2521 - accuracy: 0.9121 - val_loss:
0.1845 - val_accuracy: 0.9738
Epoch 2/9
525/525 [==============================] - 202s 384ms/step - loss: 0.0586 - accuracy: 0.9825 - val_loss:
0.1392 - val_accuracy: 0.9809
Epoch 3/9
525/525 [==============================] - 198s 378ms/step - loss: 0.0341 - accuracy: 0.9890 - val_loss:
0.2347 - val_accuracy: 0.9778
Epoch 4/9
525/525 [==============================] - 185s 352ms/step - loss: 0.0328 - accuracy: 0.9897 - val_loss:
0.1672 - val_accuracy: 0.9813
Epoch 5/9
525/525 [==============================] - 169s 322ms/step - loss: 0.0213 - accuracy: 0.9940 - val_loss:
0.2407 - val_accuracy: 0.9782
Epoch 6/9
525/525 [==============================] - 169s 321ms/step - loss: 0.0184 - accuracy: 0.9950 - val_loss:
0.2907 - val_accuracy: 0.9787
Epoch 7/9
525/525 [==============================] - 214s 408ms/step - loss: 0.0134 - accuracy: 0.9962 - val_loss:
0.1543 - val_accuracy: 0.9831
Epoch 8/9
525/525 [==============================] - 186s 353ms/step - loss: 0.0144 - accuracy: 0.9952 - val_loss:
0.1681 - val_accuracy: 0.9782
Epoch 9/9
525/525 [==============================] - 240s 457ms/step - loss: 0.0163 - accuracy: 0.9950 - val_loss:
0.2694 - val_accuracy: 0.9804
```

Out[19]:
```
<keras.callbacks.History at 0x1ac5a5bc6d0>
```

In [22]:
```
model.save("signlanguage-new.h5")
```