IMAGE PREPROCESSING

In [1]:
```
from keras.preprocessing.image import ImageDataGenerator
train_datagen = ImageDataGenerator(rescale = 1./255, shear_range=0.2, zoom_range=0.2, horizontal_flip=True)
test_datagen = ImageDataGenerator(rescale = 1./255)
```

In [2]:
```
x_train = train_datagen.flow_from_directory('C:/Users/kaviy/OneDrive/Desktop/Dataset/training_set',
                                            target_size=(64,64), batch_size=300, class_mode='categorical', color_mode = "grayscal
```

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

In [3]:
```
x_test = test_datagen.flow_from_directory('C:/Users/kaviy/OneDrive/Desktop/Dataset/test_set', target_size=(64,64), batch_size=300
```

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

BUILDING THE MODEL

```
In [1]: 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

In [5]: model = Sequential()

In [6]: model.add(Convolution2D(32, (3,3), input_shape=(64,64,1), activation = 'relu'))

In [7]: model.add(MaxPooling2D(pool_size=(2,2)))

In [8]: model.add(Flatten())

In [9]: model.add(Dense(units=512, activation='relu'))
        model.add(Dense(units=9, activation='softmax'))

In [10]: model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])

In [11]: model.fit(x_train, steps_per_epoch=24, epochs=10, validation_data=x_test, validation_steps=40)
Epoch 1/10
24/24 [==============================] - ETA: 0s - loss: 0.7514 - accuracy: 0.7431WARNING:tensorflow:Your input ran out of dat
a; 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 [==============================] - 48s 2s/step - loss: 0.7514 - accuracy: 0.7431 - val_loss: 0.3166 - val_accuracy: 0.910
2
Epoch 2/10
24/24 [==============================] - 34s 1s/step - loss: 0.1895 - accuracy: 0.9458
Epoch 3/10
24/24 [==============================] - 34s 1s/step - loss: 0.0959 - accuracy: 0.9767
Epoch 4/10
24/24 [==============================] - 34s 1s/step - loss: 0.0508 - accuracy: 0.9885
Epoch 5/10
24/24 [==============================] - 33s 1s/step - loss: 0.0365 - accuracy: 0.9912
Epoch 6/10
24/24 [==============================] - 33s 1s/step - loss: 0.0268 - accuracy: 0.9940
Epoch 7/10
24/24 [==============================] - 33s 1s/step - loss: 0.0209 - accuracy: 0.9946
Epoch 8/10
24/24 [==============================] - 32s 1s/step - loss: 0.0218 - accuracy: 0.9932
Epoch 9/10
24/24 [==============================] - 32s 1s/step - loss: 0.0138 - accuracy: 0.9972
Epoch 10/10
24/24 [==============================] - 32s 1s/step - loss: 0.0119 - accuracy: 0.9967
Out[11]: <keras.callbacks.History at 0x265805b61c0>

In [12]: model.save('aslpng1.h5')
```

TEST THE MODEL

In [13]:
```
from keras.models import load_model
import numpy as np
import cv2
```

In [14]:
```
model=load_model('aslpng1.h5')
```

In [15]:
```
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):
        img = img/255.0
    prediction = model.predict(img)
    print(prediction)
    predictions = np.argmax(prediction,axis=1)
    print(predictions)
```

In [16]:
```
frame=cv2.imread(r"C:/Users/kaviy/OneDrive/Desktop/Dataset/test_set/G/1.png")
data = detect(frame)
```

```
1/1 [==============================] - 0s 283ms/step
[[7.5161104e-09 6.3304383e-14 3.4441828e-07 3.6681456e-11 2.6866505e-13
  3.3903615e-09 9.9999774e-01 1.9178990e-06 5.2218429e-12]]
[6]
```