

## Model Building

In [1]:

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

In [2]:

```
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 [25]:

```
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 [35]:

```
!!unzip '/content/conversation engine for deaf and dumb (9) (2) (1).zip'
```

Streaming output truncated to the last 5000 lines.

```
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
```

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]



[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]



[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]



[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]



[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]



[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]



[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]



[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]



[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

inflating: Dataset/training\_set/I/940.png  
inflating: Dataset/training\_set/I/941.png  
inflating: Dataset/training\_set/I/942.png  
inflating: Dataset/training\_set/I/943.png  
inflating: Dataset/training\_set/I/944.png  
inflating: Dataset/training\_set/I/945.png  
inflating: Dataset/training\_set/I/946.png  
inflating: Dataset/training\_set/I/947.png  
inflating: Dataset/training\_set/I/948.png  
inflating: Dataset/training\_set/I/949.png  
inflating: Dataset/training\_set/I/95.png  
inflating: Dataset/training\_set/I/950.png  
inflating: Dataset/training\_set/I/951.png  
inflating: Dataset/training\_set/I/952.png  
inflating: Dataset/training\_set/I/953.png  
inflating: Dataset/training\_set/I/954.png  
inflating: Dataset/training\_set/I/955.png  
inflating: Dataset/training\_set/I/956.png  
inflating: Dataset/training\_set/I/957.png  
inflating: Dataset/training\_set/I/958.png  
inflating: Dataset/training\_set/I/959.png  
inflating: Dataset/training\_set/I/96.png  
inflating: Dataset/training\_set/I/960.png  
inflating: Dataset/training\_set/I/961.png  
inflating: Dataset/training\_set/I/962.png  
inflating: Dataset/training\_set/I/963.png  
inflating: Dataset/training\_set/I/964.png  
inflating: Dataset/training\_set/I/965.png  
inflating: Dataset/training\_set/I/966.png  
inflating: Dataset/training\_set/I/967.png  
inflating: Dataset/training\_set/I/968.png  
inflating: Dataset/training\_set/I/969.png  
inflating: Dataset/training\_set/I/97.png  
inflating: Dataset/training\_set/I/970.png  
inflating: Dataset/training\_set/I/971.png  
inflating: Dataset/training\_set/I/972.png  
extracting: Dataset/training\_set/I/973.png  
inflating: Dataset/training\_set/I/974.png  
inflating: Dataset/training\_set/I/975.png  
inflating: Dataset/training\_set/I/976.png  
inflating: Dataset/training\_set/I/977.png  
inflating: Dataset/training\_set/I/978.png  
inflating: Dataset/training\_set/I/979.png  
inflating: Dataset/training\_set/I/98.png  
inflating: Dataset/training\_set/I/980.png  
inflating: Dataset/training\_set/I/981.png  
inflating: Dataset/training\_set/I/982.png  
extracting: Dataset/training\_set/I/983.png  
inflating: Dataset/training\_set/I/984.png  
inflating: Dataset/training\_set/I/985.png  
inflating: Dataset/training\_set/I/986.png  
inflating: Dataset/training\_set/I/987.png  
inflating: Dataset/training\_set/I/988.png  
inflating: Dataset/training\_set/I/989.png  
inflating: Dataset/training\_set/I/99.png  
inflating: Dataset/training\_set/I/990.png  
inflating: Dataset/training\_set/I/991.png  
inflating: Dataset/training\_set/I/992.png  
extracting: Dataset/training\_set/I/993.png  
inflating: Dataset/training\_set/I/994.png  
inflating: Dataset/training\_set/I/995.png  
extracting: Dataset/training\_set/I/996.png  
inflating: Dataset/training\_set/I/997.png  
inflating: Dataset/training\_set/I/998.png  
inflating: Dataset/training\_set/I/999.png

In [38]:

```
!unzip '/content/conversation engine for deaf and dumb (9) (2) (1).zip'
```

Archive: /content/conversation engine for deaf and dumb (9) (2) (1).zip  
replace Dataset/test set/A/1.png? [y]es, [n]o, [A]ll, [N]one, [r]ename:

In [39]:

```
from keras.utils.generic_utils import class_and_config_for_serialized_keras_object
x_train = train_datagen.flow_from_directory('Dataset/training_set', target_size=(64, 64),
batch_size=300, class_mode= 'categorical', color_mode="grayscale")
x_test = test_datagen.flow_from_directory('Dataset/test_set', target_size=(64, 64), batch
_size=300, class_mode= 'categorical', color_mode="grayscale")
```

Found 15750 images belonging to 9 classes.

Found 2250 images belonging to 9 classes.

In [3]:

```
model=Sequential()
```

In [4]:

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

In [5]:

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

In [6]:

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

In [7]:

```
model.add(Dense(200, activation='relu'))
model.add(Dense(9, activation="softmax"))
```

In [8]:

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

In [40]:

```
len(x_train)
```

Out[40]:

53

In [41]:

```
len(x_test)
```

Out[41]:

8

In [58]:

```
model.save("/content/aslpng.h5")
```

## Testing the model

In [59]:

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

In [60]:

```
model=load_model('/content/aslpng.h5')
img=image.load_img(r'/content/1.png',
                    target_size=(64,64))
```

In [61]:

```
model=load_model("aslpng.h5")
img = image.load_img("/content/1.png",target_size=(64,64))
img
```

Out[61]:



In [62]:

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

Out[62]:

```
array([[ [0., 0., 0.],
         [0., 0., 0.],
         [0., 0., 0.],
         ...,
         [0., 0., 0.],
         [0., 0., 0.],
         [0., 0., 0.]],

       [ [0., 0., 0.],
         [0., 0., 0.],
         [0., 0., 0.],
         ...,
         [0., 0., 0.],
         [0., 0., 0.],
         [0., 0., 0.]],

       [ [0., 0., 0.],
         [0., 0., 0.],
         [0., 0., 0.],
         ...,
         [0., 0., 0.],
         [0., 0., 0.],
         [0., 0., 0.]],

       ...,

       [ [0., 0., 0.],
         [0., 0., 0.],
         [0., 0., 0.],
         ...,
         [0., 0., 0.],
         [0., 0., 0.],
         [0., 0., 0.]],

       [ [0., 0., 0.],
         [0., 0., 0.],
         [0., 0., 0.],
         ...,
         [0., 0., 0.],
         [0., 0., 0.],
         [0., 0., 0.]],

       [ [0., 0., 0.],
         [0., 0., 0.],
         [0., 0., 0.],
         ...,
         [0., 0., 0.],
         [0., 0., 0.],
         [0., 0., 0.]]], dtype=float32)
```



In [63]:

```
x.shape
```

Out[63]:

```
(64, 64, 3)
```

In [64]:

```
x = np.expand_dims(x,axis=0)
x.shape
```

Out[64]:

```
(1, 64, 64, 3)
```

In [65]:

```
pred = model.predict(x)
pred
class_name=["A","B","C","D","E","F","G","H","I"]
pred_id = pred.argmax(axis=1)[0]
pred_id
print("the alphabet is ",str(class_name[pred_id]))
```

```
1/1 [=====] - 0s 254ms/step
the alphabet is  B
```