In [ ]:

```
!unzip '/content/drive/MyDrive/Dataset.zip'
```

Image Preprocessing

Import ImageDataGenerator Library and Configure It

In [ ]:

```
from keras.preprocessing.image import ImageDataGenerator
train_datagen = ImageDataGenerator(rescale =1./255, shear_range=0.2, zoom_range=0.2, hori
zontal_flip=True)
test_datagen = ImageDataGenerator(rescale =1./255)
```

Apply ImageDataGenerator functionality To Train And Test

In [ ]:

```
x_train =train_datagen.flow_from_directory('Dataset/training_set' , target_size=(64,64),
batch_size=300,
                                           class_mode= 'categorical', color_mode="graysc
ale")
```

```
Found 100 images belonging to 1 classes.
```

In [ ]:

```
x_test = test_datagen.flow_from_directory('Dataset/test_set', target_size=(64,64), batch
_size=300,
                                          class_mode='categorical', color_mode="grayscal
e")
```

```
Found 100 images belonging to 1 classes.
```

Model Building

Import The Required Model Building Libraries

In [ ]:

```
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 [ ]:

```
model = Sequential()
```

Add The Convolution Layer

In [ ]:

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

Add The Pooling Layer

In [ ]:

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

Add The Flatten Layer

In [ ]:

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

Adding The Dense Layer

In [ ]:

```
model.add(Dense(units=512, activation='relu'))
model.add(Dense(units=9, activation='softmax'))
```

Compile The Model

In [ ]:

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

Fit And Save The Model

In [ ]:

```
model.fit_generator(x_train, steps_per_epoch=24, epochs=10, validation_data=x_test, validation_steps=40)
```

```
/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
 1/24 [>.............................] - ETA: 32s - loss: 19.8330 - accuracy: 0.0000e+00
```

```
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, 240 batches). You may need to use the repeat() function when building your dataset.
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 [==============================] - 2s 32ms/step - loss: 19.8330 - accuracy: 0.0000e+00 - val_loss: 20.3931 - val_accuracy: 0.0000e+00
```

Out[ ]:

```
<keras.callbacks.History at 0x7f80baefb590>
```

In [ ]:

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

Test The Model

Import The Packages And Load The Saved Model

In [ ]:

```
from keras.models import load_model
import numpy as np
import cv2
```

In [ ]:

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

Load The Test Image, Pre-Process It And Predict

In [ ]:

```
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)
    prediction = model.predict_classes(img)
    print(prediction)
```

In [ ]:

```
frame=cv2.imread(r"D:\\Nivetha\\Smart Bridge\\My_project\\conversation engine for deaf a
nd dumb\\Dataset\\test_set\\A\\2.png")
```