```
Start coding or generate with AI.
```

Name : Anurag Gupta

Email : anuragsgupta9211@gmail.com

Rollno: MPIITDCV53525

Lab -1

Question 1: Calculate the histogram of the following image. Histogram should have bins=50 and plt the histogram as well

Question 2: Threshold the provided pepper image into 5 classes.

Question 3: Clean the Noisy Lena image and then detect the edges of the image.

Question 4: Create a sequential network model using the following specification. Consider the input image(gray) as of size 200*200 and dataset contains 25 classes. Set the model accordingly.

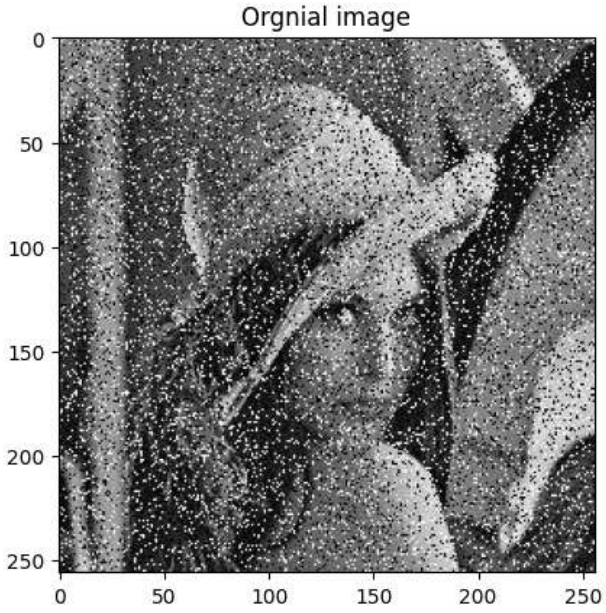
a) A 2d convolution layer of with 15 filters, 5*5 kernel, with padding and relu activation
b) A 2d convolution layer of with 15 filters, 5*5 kernel, with padding and relu activation
c) Max pool layer with stride=2
d) Repeat a&b by doubling the filters and add c
e) Followed by 2 dense layers with relu and softmax as activation function.

```
import numpy as np
import matplotlib.pyplot as plt
# %matplotlib inline
import PIL.Image as images
from skimage import io
import cv2 as cv
```

## Question 1

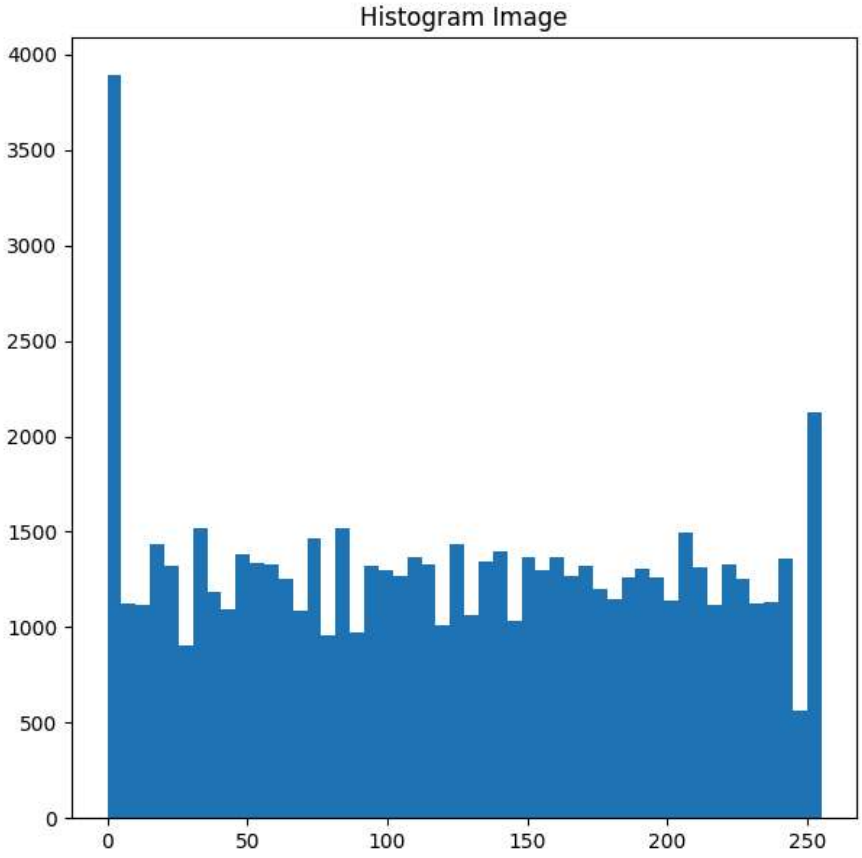
```
image_path = './noisy.jpg'
noisy_image = cv.imread(image_path)

noisy_image = cv.cvtColor(noisy_image, cv.COLOR_BGR2GRAY)


noisy_image = noisy_image.astype(np.uint8)


# fig.add_subplot(2,2,3)
# plt.title("Histro Org image")
# plt.hist(gray_lena_image.ravel(),bins=50,range=[0,255])


# fig.add_subplot(2,2,4)
# plt.hist(hist_equalis_img.ravel(),bins=50,range=[0,255])

plt.title("Orgnial image")
plt.imshow(noisy_image,cmap='gray')
```

<matplotlib.image.AxesImage at 0x796725883e20>



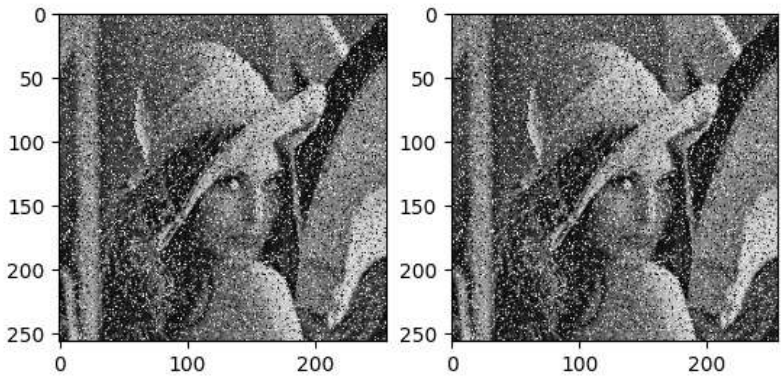

```
hist_equalis_img = cv.equalizeHist(noisy_image)

fig = plt.figure(figsize=(7,7))

# fig.add_subplot(1,2)
plt.title("Histogram Image")
plt.hist(hist_equalis_img.ravel(),bins=50,range=[0,255])
```

```
(array([3893., 1128., 1117., 1439., 1321.,  905., 1521., 1186., 1094.,
        1385., 1340., 1327., 1254., 1083., 1465.,  958., 1517.,  974.,
        1323., 1301., 1272., 1367., 1333., 1009., 1439., 1061., 1341.,
        1400., 1034., 1369., 1300., 1370., 1269., 1318., 1199., 1146.,
        1260., 1305., 1264., 1143., 1495., 1311., 1117., 1332., 1250.,
        1125., 1129., 1356.,  562., 2129.]),
 array([  0. ,   5.1,  10.2,  15.3,  20.4,  25.5,  30.6,  35.7,  40.8,
         45.9,  51. ,  56.1,  61.2,  66.3,  71.4,  76.5,  81.6,  86.7,
         91.8,  96.9, 102. , 107.1, 112.2, 117.3, 122.4, 127.5, 132.6,
        137.7, 142.8, 147.9, 153. , 158.1, 163.2, 168.3, 173.4, 178.5,
        183.6, 188.7, 193.8, 198.9, 204. , 209.1, 214.2, 219.3, 224.4,
        229.5, 234.6, 239.7, 244.8, 249.9, 255. ]),
 <BarContainer object of 50 artists>)
```




## Question 2

```
import cv2
import matplotlib.pyplot as plt

image_bw = cv2.imread('noisy.jpg', cv2.IMREAD_GRAYSCALE)

img = cv.imread('noisy.jpg')
dst = cv.fastNlMeansDenoisingColored(img,None,10,10,7,21)
plt.subplot(121),plt.imshow(img)
plt.subplot(122),plt.imshow(dst)
plt.show()

```



```
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense


model = Sequential()


input_shape = (200, 200, 1)


model.add(Conv2D(15, (5, 5), padding='same', activation='relu', input_shape=input_shape))


model.add(Conv2D(15, (5, 5), padding='same', activation='relu'))


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


model.add(Conv2D(30, (5, 5), padding='same', activation='relu'))
model.add(Conv2D(30, (5, 5), padding='same', activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2), strides=2))


model.add(Flatten())
model.add(Dense(128, activation='relu'))
model.add(Dense(25, activation='softmax'))

# Print the model summary
model.summary()

```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 200, 200, 15)      390

 conv2d_1 (Conv2D)           (None, 200, 200, 15)      5640

 max_pooling2d (MaxPooling2  (None, 100, 100, 15)      0
 D)

 conv2d_2 (Conv2D)           (None, 100, 100, 30)      11280

 conv2d_3 (Conv2D)           (None, 100, 100, 30)      22530

 max_pooling2d_1 (MaxPoolin  (None, 50, 50, 30)        0
 g2D)

 flatten (Flatten)           (None, 75000)             0

 dense (Dense)               (None, 128)               9600128

 dense_1 (Dense)             (None, 25)                3225
```

```
=================================================================
Total params: 9643193 (36.79 MB)
Trainable params: 9643193 (36.79 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

```python
import cv2
import numpy as np
import matplotlib.pyplot as plt

# Step 1: Read the image
image = cv2.imread('peppers.png')
image_rgb = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)

# Step 2: Reshape the image to a 2D array of pixels
pixel_values = image_rgb.reshape((-1, 3))
pixel_values = np.float32(pixel_values)

# Step 3: Apply K-means clustering
k = 5
criteria = (cv2.TERM_CRITERIA_EPS + cv2.TERM_CRITERIA_MAX_ITER, 100, 0.2)
_, labels, centers = cv2.kmeans(pixel_values, k, None, criteria, 10, cv2.KMEANS_RANDOM_CENTERS)


centers = np.uint8(centers)


segmented_image = centers[labels.flatten()]

segmented_image = segmented_image.reshape(image_rgb.shape)


plt.figure(figsize=(10, 5))


plt.subplot(1, 2, 1)
plt.imshow(image_rgb)
plt.title('Original Image')
plt.axis('off')


plt.subplot(1, 2, 2)
plt.imshow(segmented_image)
plt.title('Segmented Image into 5 Classes')
plt.axis('off')

plt.show()

```


```python
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense

```

```
model = Sequential()


input_shape = (200, 200, 1)


model.add(Conv2D(15, (5, 5), padding='same', activation='relu', input_shape=input_shape))


model.add(Conv2D(15, (5, 5), padding='same', activation='relu'))


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


model.add(Conv2D(30, (5, 5), padding='same', activation='relu'))
model.add(Conv2D(30, (5, 5), padding='same', activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2), strides=2))


model.add(Flatten())
model.add(Dense(128, activation='relu'))
model.add(Dense(25, activation='softmax'))

# Print the model summary
model.summary()

```

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
```