“Computer Vision”

Shervin Halat

98131018

Homework 3

1.

All images were resized to 256x256 dimensions using cv2.resize with INTER_CUBIC interpolation.

Also, it should be noted that all images were normalized to the range of (0,1) by dividing them into 255 as the only preprocessing operation.

```
def load_images_from_folder(folder):
    images = []
    images_names = []
    for filename in os.listdir(folder):
        img = cv2.imread(os.path.join(folder,filename),0)

        ## resizing images:
        img = cv2.resize(img, (256,256), interpolation = cv2.INTER_CUBIC)
        images_names.append(filename.split('_')[0])

        ## preprocessing:
        if img is not None:
            images.append(img.astype(np.float32) / 255)
    return images,images_names
```

2.

A Gabor bank filter of size 72 (72 different kernels) was generated in order to reach a high accuracy.

Mentioned kernels were obtained using the following values:

```
kernels = []
for theta in range(8):
    theta = theta / 8. * np.pi
    for sigma in (1,3,5):
        for gamma in (1,0.5,0.25):
            kernel = cv2.getGaborKernel((21,21), sigma, theta, 10,\
                                        gamma, 0, cv2.CV_32F)
#             kernel /= math.sqrt((kernel * kernel).sum())
            kernels.append(kernel)
```

Parameter values are as follows as is shown in the picture above:

Ksize: Kernel size of (21x21) was considered.

Theta (θ): Ranging from 0 to π by (π/8) steps.

Gamma (ɣ): Values of 1, 0.5, and 0.25 were considered.

Sigma (σ): Different values of 1, 3, and 5 were considered.

Psi(Ψ): Constant value of zero was considered.

Lambda (λ): constant value of 10 was considered.

Parameters of getGaborKernel function:

a) First parameter (ksize) is the size of the Gabor kernel. If ksize = (a, b), we then have a Gabor kernel of size a x b pixels. Mostly, ksize should be odd and square.

b) Second parameter (sigma) controls the overall size of the Gabor envelope by controlling Standard deviation of the gaussian envelope. By increasing the sigma value, number of stripes in the Gabor filter increases.

c) Third parameter (theta) controls the orientation of the Gabor function. The zero degree theta corresponds to the vertical position of the Gabor filter.

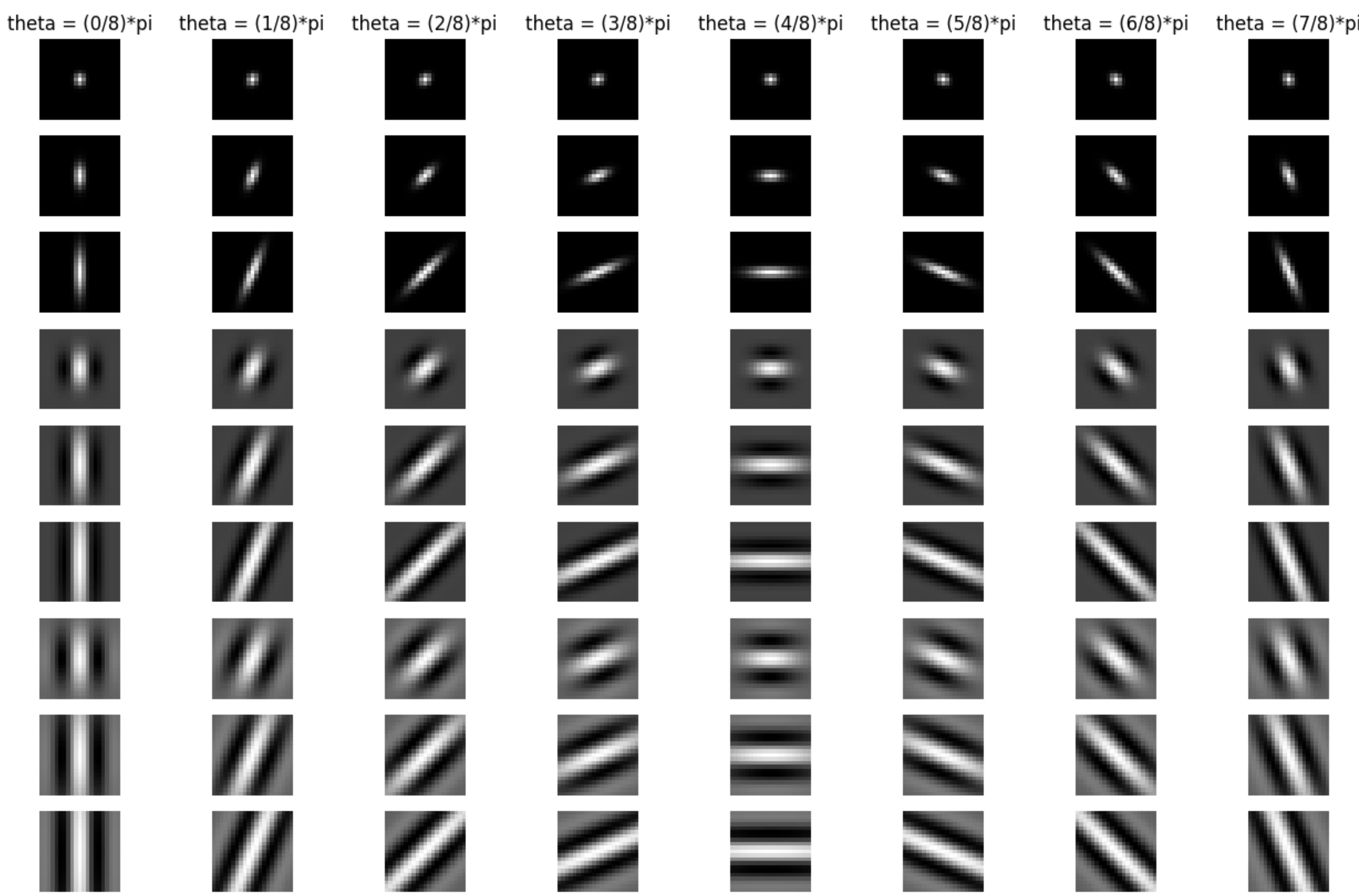
d) Fourth parameter (lambd) is Wavelength of the sinusoidal factor and it controls the width of the stripes of Gabor filter. Increasing this value produces thicker stripes and decreasing it produces thinner stripes.

e) Fifth parameter (gamma) is the filter spatial aspect ratio and it controls the height of the Gabor filter with regard to its width.

f) Sixth parameter (psi) is the phase offset.

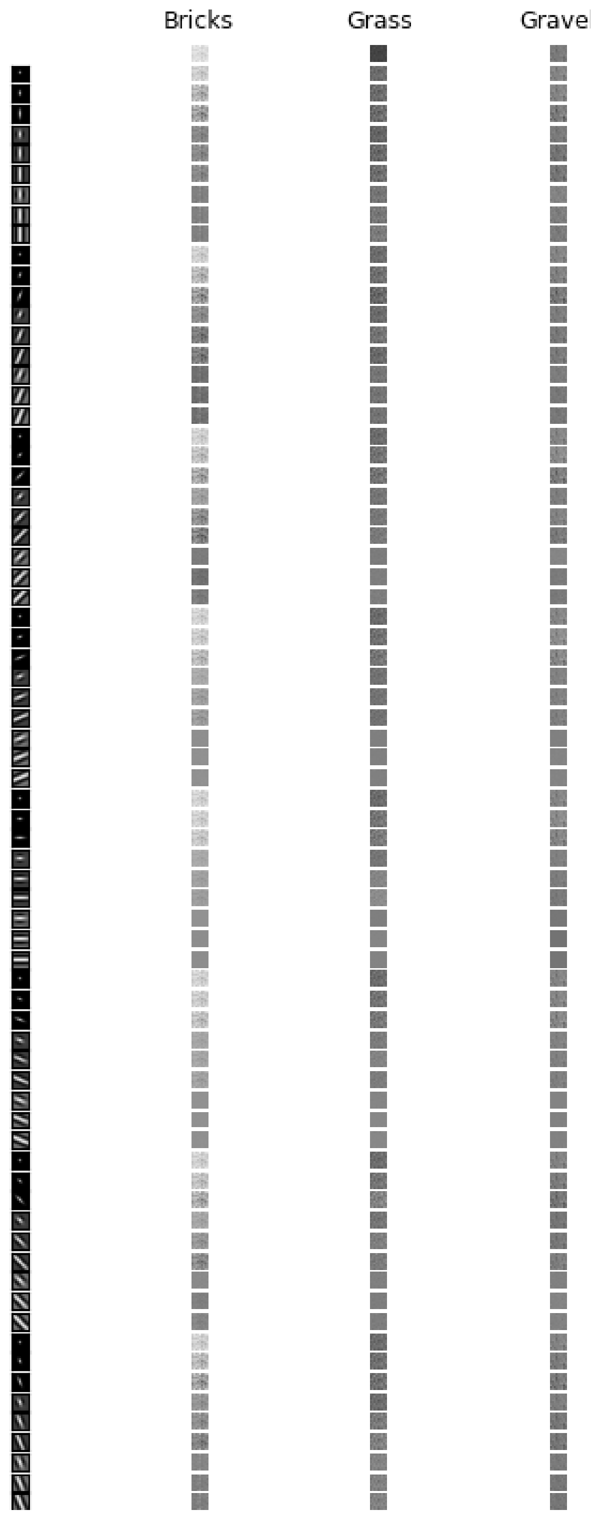
g) Seventh parameter (ktype) indicates the type and range of values that each pixel in the Gabor kernel can hold.

"generated Gobor bank filter"

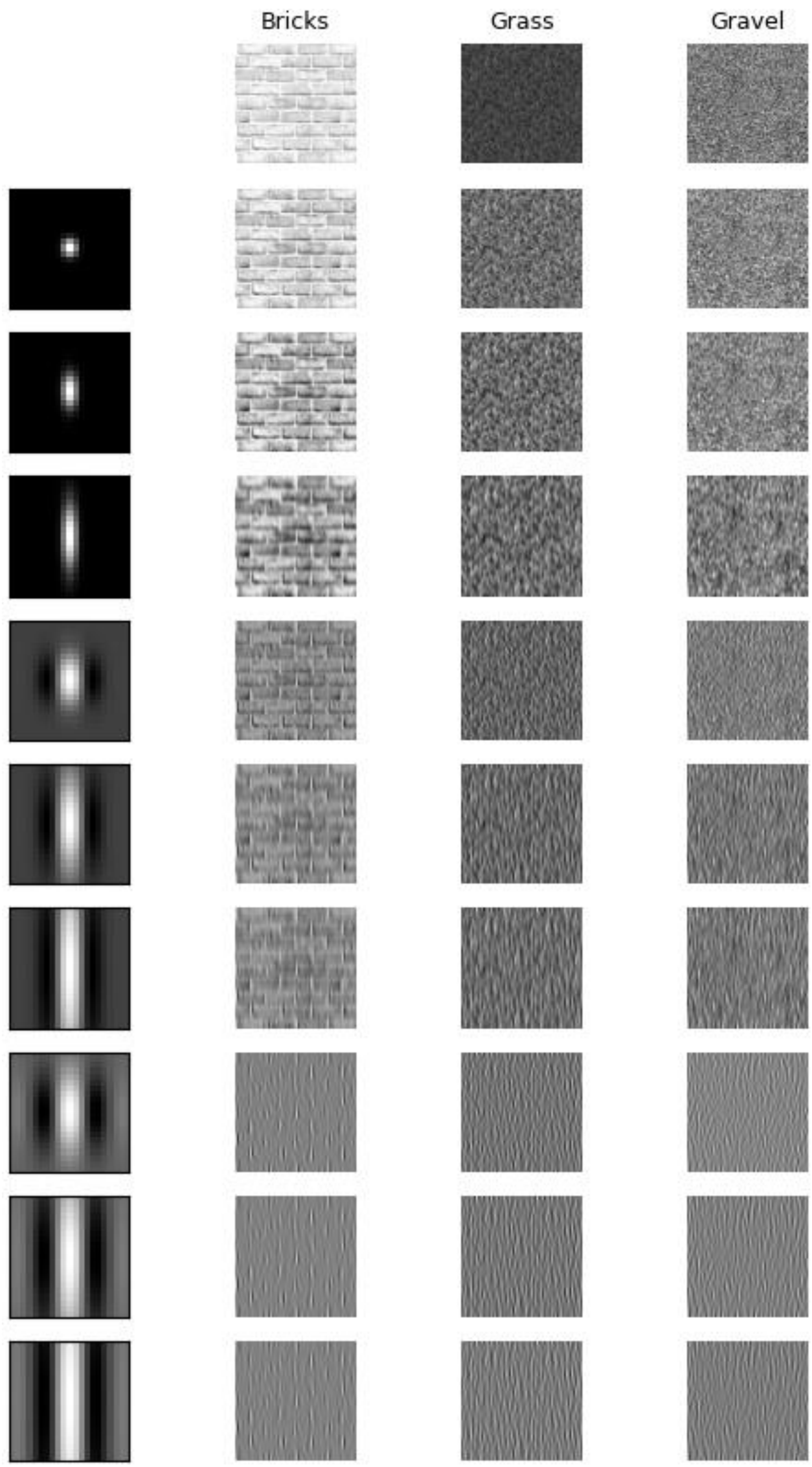


3.

Generated features by convolution:



In the following, an example of first nine kernels are shown below as a better visualization:



4.

The explained process has been implemented by functions of 'mse', 'classify', 'features', and 'features2'.

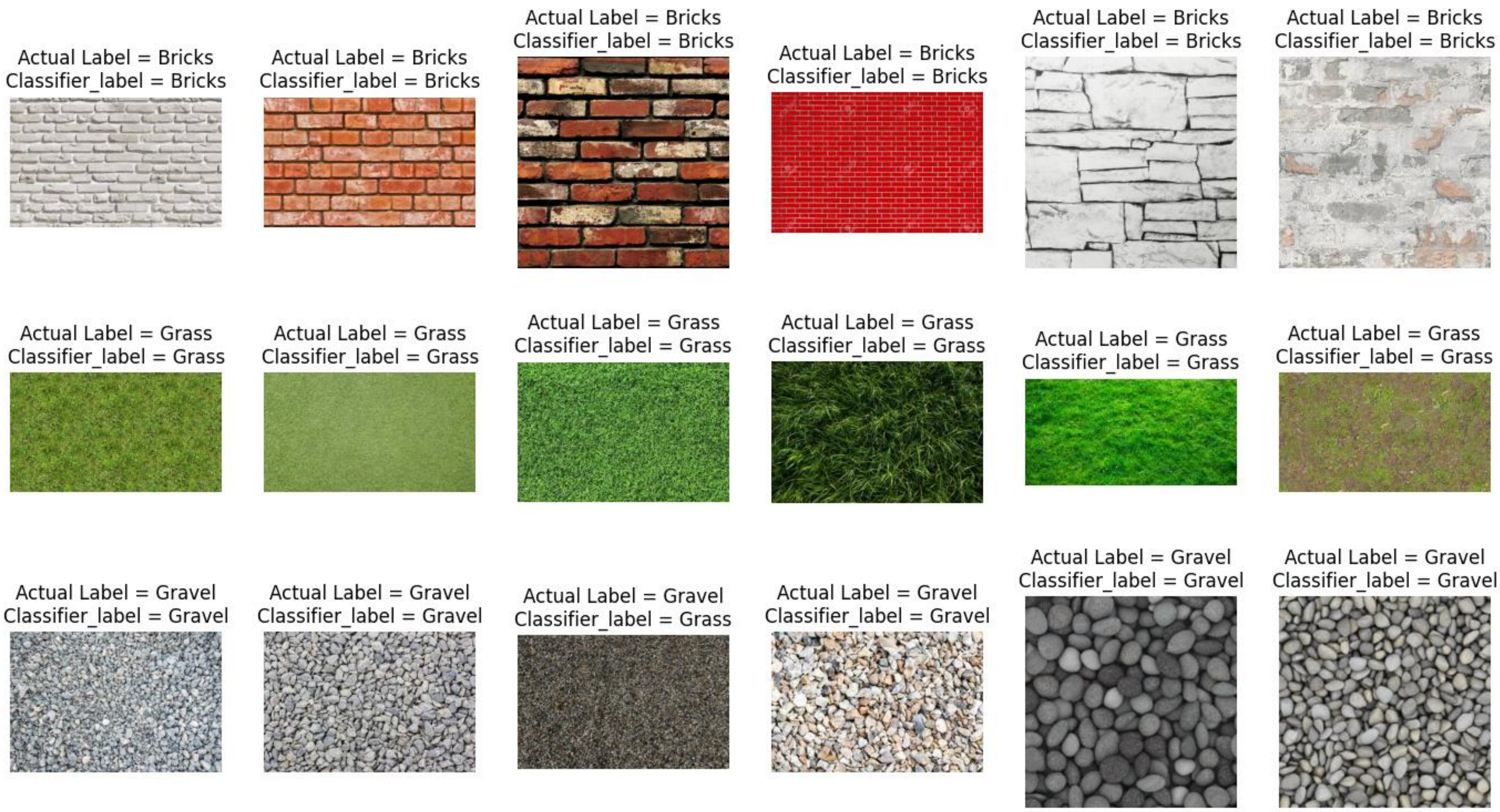
It should be noted that the function 'features2' substitutes mean and variance of each feature as feature! This way the accuracy of the classifier significantly improves. Function 'features2' has been used for the next two problems as it outperforms 'features1' for classification.

```
def features(image,kernel_bank):
    features = []
    for ker in kernel_bank:
        features.append(cv2.filter2D(image, -1, ker))
    return np.array(features)
```

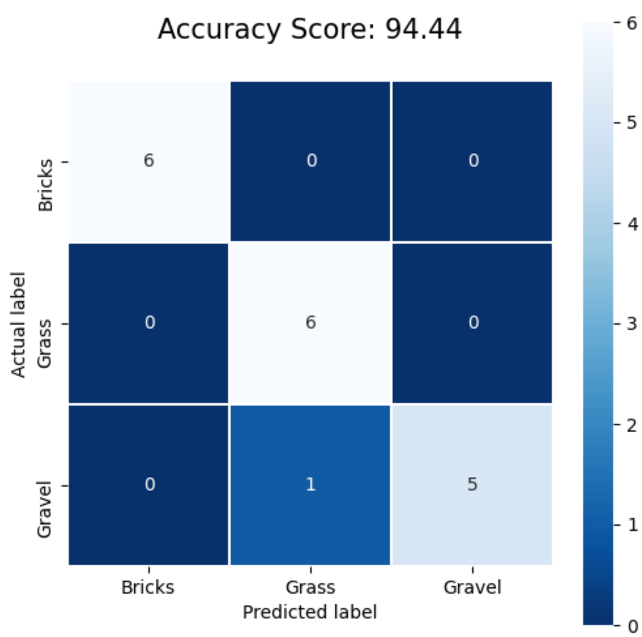
```
def features2(image,kernel_bank):
    features = []
    for ker in kernel_bank:
        features.append([np.mean(cv2.filter2D(image, -1, ker))\
                          ,np.var(cv2.filter2D(image, -1, ker))])
    return np.array(features)
```

5.

Actual and Classifier Labels:



Confusion Matrix:



6.

Generally, the design of a proper Gabor filter bank is usually a crucial step in texture classification. Therefore, an evaluation on how changing number and values of various parameters affect the classifier accuracy has been performed. The results are as follow:

First, effect of 'theta' parameter will be evaluated:

Theta (θ):

For the second experiment, a Gabor bank filter of size 27 (27 different kernels) was generated. This time 3 values for theta were considered (as opposed to first experiment in which 8 values were considered for theta parameter).

Mentioned kernels were obtained using the following values:

```
kernels = []
for theta in range(3):
    theta = theta / 3. * np.pi
    for sigma in (1,3,5):
        for gamma in (1,0.5,0.25):
            kernel = cv2.getGaborKernel((21,21), sigma, theta, 10,\
                                            gamma, 0, cv2.CV_32F)
#             kernel /= math.sqrt((kernel * kernel).sum())
            kernels.append(kernel)
```

Parameter values are as follows as is shown in the picture above:

Ksize: Kernel size of (21x21) was considered.

Theta (θ): Ranging from 0 to π by (π/3) steps.

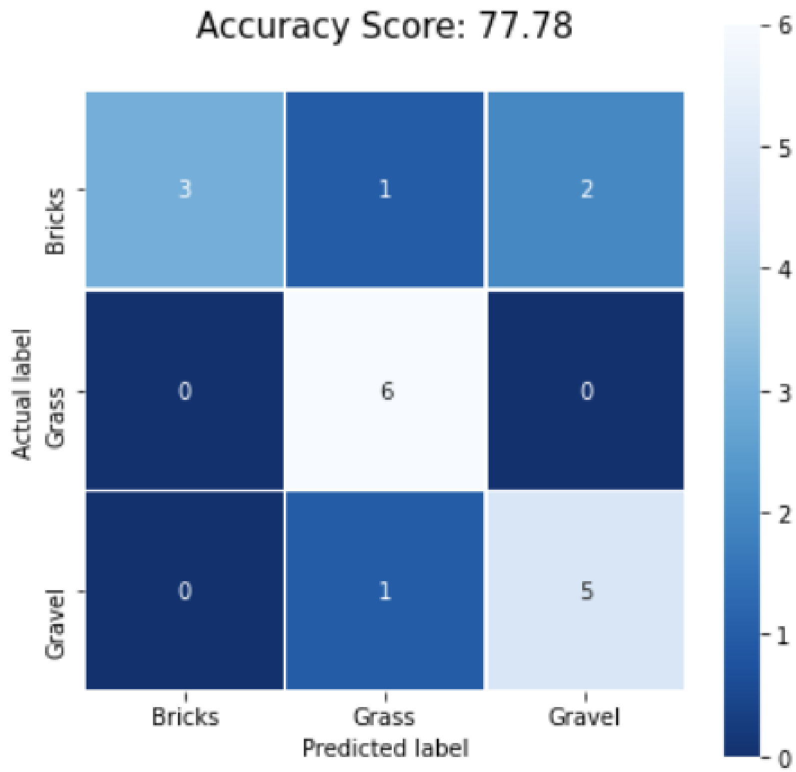
Gamma (ɣ): Values of 1, 0.5, and 0.25 were considered.

Sigma (σ): Different values of 1, 3, and 5 were considered.

Psi(Ψ): Constant value of zero was considered.

Lambda (λ): constant value of 10 was considered.

The confusion matrix and accuracy of second experiment are as follows:



As it can be figured out from comparing accuracy and confusion matrix above with the first experiment's results, decrease in number of 'theta' parameter results in decrease in accuracy of the classifier (94.44 decreased to 77.89).

Ksize:

For the third experiment, a Gabor bank filter of size 72 (72 different kernels) was generated. This time kernel size of (7x7) was considered (as opposed to first experiment in which (21x21) was considered for kernel size).

Mentioned kernels were obtained using the following values:

```
kernels = []
for theta in range(8):
    theta = theta / 8. * np.pi
    for sigma in (1,3,5):
        for gamma in (1,0.5,0.25):
            kernel = cv2.getGaborKernel((7,7), sigma, theta, 10,\
                                        gamma, 0, cv2.CV_32F)
#              kernel /= math.sqrt((kernel * kernel).sum())
            kernels.append(kernel)
```

Parameter values are as follows as is shown in the picture above:

Ksize: Kernel size of (7,7) was considered.

Theta (θ): Ranging from 0 to π by (π/3) steps.

Gamma (ɣ): Values of 1, 0.5, and 0.25 were considered.

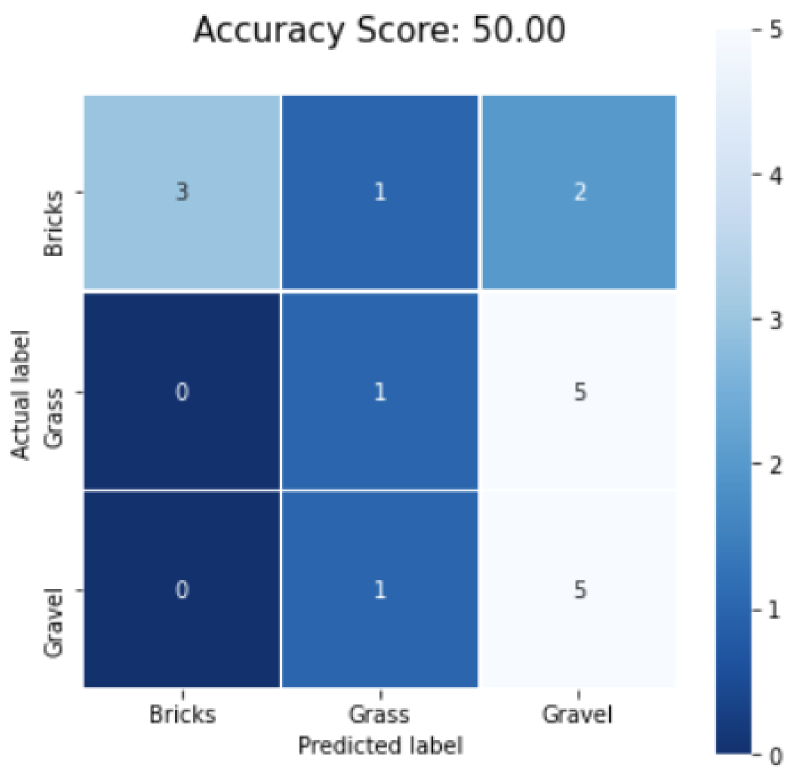
Sigma (σ): Different values of 1, 3, and 5 were considered.

Psi(Ψ): Constant value of zero was considered.

Lambda (λ): constant value of 10 was considered.

Therefore, the only difference in this experiment is that the kernel size has decreased from (21x21) to (7x7).

The confusion matrix and accuracy of third experiment are as follows:



As it can be figured out from comparing accuracy and confusion matrix above with the first experiment’s results, decrease in kernel size results in significant decrease in accuracy of the classifier (94.44 decreased to 50).

Conclusion:

Generally, decrease in number of values of parameters and kernel size in Gabor filter texture classification method results in decrease in classifier performance and accuracy. This is mainly because of the fact that each kernel distinguishes specific and unique features of the given image and therefore, more valuable features will be extracted with more number and various kernels.