In [16]:

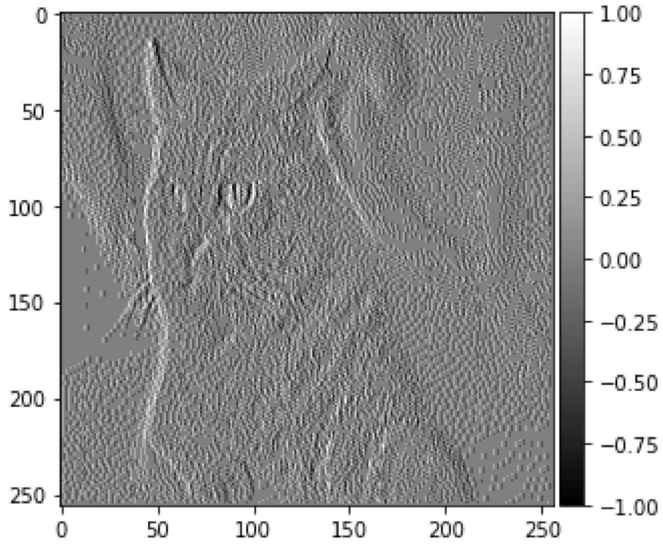
```
import cv2 as cv2
import numpy as np
import os
import PIL
from PIL import Image
import numpy as np
from keras.preprocessing.image import img_to_array
from skimage.io import imread, imshow
from skimage.filters import prewitt_h,prewitt_v
import matplotlib.pyplot as plt
%matplotlib inline
```

In [17]:

```
def binarize_images(x):
    """
    Convert images to range 0-1 and binarize them by making
    0 the values below 0.1 and 1 the values above 0.1.
    """
    x /= 255
    x[x > 0] = 1
    return x
```

In [18]:

```
files=os.listdir('test/')
count0=0
array=[]
for file in files:
    image = Image.open('test/'+file)
    imagefinal=image.resize((256,256))
    imagefinal = imagefinal.convert('1')
    count0+=1
    edges_prewitt_horizontal = prewitt_h(imagefinal)
    edges_prewitt_vertical = prewitt_v(imagefinal)
    imshow(edges_prewitt_vertical, cmap='gray')
    break
```



In [5]:

```
files=os.listdir('test_zero/')
count0=0
array=[]
for file in files:
    image = Image.open('test_zero/'+file)
    imagefinal=image.resize((256,256))
    imagefinal = imagefinal.convert('1')
    im2arr = img_to_array(imagefinal)
    im2arr=binarize_images(im2arr)
    edges_prewitt_horizontal = prewitt_h(imagefinal)
    edges_prewitt_vertical = prewitt_v(imagefinal)
    im2arr=im2arr.reshape(256,256)

    array.append(edges_prewitt_vertical)
    count0+=1
```

In [6]:

```
files=os.listdir('test_90/')
count90=0
array3=[]
for file in files:
    image = Image.open('test_90/'+file)
    imagefinal=image.resize((256,256))
    imagefinal = imagefinal.convert('1')
    imagefinal=imagefinal.rotate(90)
    im2arr = img_to_array(imagefinal)
    im2arr=binarize_images(im2arr)
    edges_prewitt_horizontal = prewitt_h(imagefinal)
    edges_prewitt_vertical = prewitt_v(imagefinal)
    im2arr=im2arr.reshape(256,256)

    array3.append(edges_prewitt_vertical)
    count90+=1
```

In [7]:

```
files=os.listdir('test_180/')
array1=[]
count180=0
for file in files:
    image = Image.open('test_180/'+file)
    imagefinal=image.resize((256,256))
    imagefinal = imagefinal.convert('1')
    imagefinal=imagefinal.rotate(180)
    im2arr = img_to_array(imagefinal)
    im2arr=binarize_images(im2arr)
    edges_prewitt_horizontal = prewitt_h(imagefinal)
    edges_prewitt_vertical = prewitt_v(imagefinal)
    im2arr=im2arr.reshape(256,256)

    count180+=1
    array1.append(edges_prewitt_vertical)
```

In [8]:

```
files=os.listdir('test_270/')
count270=0
array2=[]
for file in files:
    image = Image.open('test_270/'+file)
    imagefinal=image.resize((256,256))
    imagefinal = imagefinal.convert('1')
    imagefinal=imagefinal.rotate(270)
    im2arr = img_to_array(imagefinal)
    im2arr=binarize_images(im2arr)
    edges_prewitt_horizontal = prewitt_h(imagefinal)
    edges_prewitt_vertical = prewitt_v(imagefinal)
    im2arr=im2arr.reshape(256,256)

    count270+=1
    array2.append(edges_prewitt_vertical)
```

In [9]:

```
array=np.array(array)
array1=np.array(array1)
array2=np.array(array2)
array3=np.array(array3)
```

In [10]:

```
finalarray=np.vstack((array,array1,array2,array3))
# TO SAVE RAM
del array
del array1
del array2
del array3
```

In [11]:

```
#we ave given class label of 0 for zero degrees
#we ave given class label of 90 for 90 degrees
#we ave given class label of 180 for 180 degrees
#we ave given class label of 270 for 270 degrees

finalclasslabels=[]
for i in range(count0):
    finalclasslabels.append(0)
for i in range(count90):
    finalclasslabels.append(1)
for i in range(count180):
    finalclasslabels.append(2)
for i in range(count270):
    finalclasslabels.append(3)
```

In [12]:

```
from sklearn.model_selection import train_test_split
xtrainfinal,xtestfinal,ytrainfinal,ytestfinal=train_test_split(finalarray,finalc
```

In [13]:

```
from keras.utils import to_categorical
ytrainfinal1=to_categorical(ytrainfinal,4)
ytestfinal1=to_categorical(ytestfinal,4)
```

In [14]:
```
import warnings
warnings.filterwarnings('ignore')
```

In [15]:
```


import keras
from keras.models import Sequential
from keras.layers import Dense, Dropout, Activation, Flatten
from keras.optimizers import Adam
from keras.layers.normalization import BatchNormalization
from keras.utils import np_utils
from keras.layers import Conv2D, MaxPooling2D, ZeroPadding2D, GlobalAveragePooli
from keras.layers.advanced_activations import LeakyReLU
```

In [0]:
```
ACTIVITIES = {

 3: '270 degree',
2: '180 degree',
1: '90 degree',
0: 'zero degree',


}
```

In [0]:
```
import warnings
warnings.filterwarnings('ignore')
```

In [0]:
```
def confusionmatrix(Y_true, Y_pred):
    Y_true = pd.Series([ACTIVITIES[y] for y in np.argmax(Y_true, axis=1)])
    Y_pred = pd.Series([ACTIVITIES[y] for y in np.argmax(Y_pred, axis=1)])

    return pd.crosstab(Y_true, Y_pred, rownames=['True'], colnames=['Pred'])
```

In [69]:

```
import warnings
warnings.filterwarnings('ignore')
classifier = Sequential()
classifier.add(Conv2D(96, (2, 2), input_shape = (256,256, 1), activation = 'relu
classifier.add(MaxPooling2D(pool_size = (2, 2)))
classifier.add(Conv2D(256, (2, 2), activation = 'relu'))
classifier.add(MaxPooling2D(pool_size = (1, 1)))
classifier.add(Conv2D(512, (2, 2), input_shape = (256,256, 1), activation = 'rel
classifier.add(MaxPooling2D(pool_size = (2, 2)))
classifier.add(Conv2D(1024, (2, 2), activation = 'relu'))
classifier.add(MaxPooling2D(pool_size = (1, 1)))
classifier.add(Dropout(0.2))
classifier.add(Flatten())
classifier.add(Dense(units = 3072, activation = 'relu'))
classifier.add(Dropout(0.3))
classifier.add(Dense(units = 4096, activation = 'relu'))
classifier.add(Dropout(0.3))
classifier.add(Dense(units = 4, activation = 'softmax'))
classifier.compile(optimizer = 'adam', loss = 'categorical_crossentropy', metric
history=classifier.fit(xtrainfinal,ytrainfinal1, epochs =40, validation_data = (
scores = classifier.evaluate(xtestfinal,ytestfinal1, verbose=1)
print("Accuracy: %.2f%%" % (scores[1]*100))
model_3_test = scores[1]
model_3_train = max(history.history['acc'])
```

```
Train on 7424 samples, validate on 1856 samples
Epoch 1/40
7424/7424 [==============================] - 13s 2ms/step - loss: 1.2442 - acc:
0.3627 - val_loss: 0.7684 - val_acc: 0.5032
Epoch 2/40
7424/7424 [==============================] - 11s 1ms/step - loss: 0.7818 - acc:
0.4958 - val_loss: 0.7622 - val_acc: 0.5140
Epoch 3/40
7424/7424 [==============================] - 11s 1ms/step - loss: 0.7669 - acc:
0.5082 - val_loss: 0.7381 - val_acc: 0.5194
Epoch 4/40
7424/7424 [==============================] - 11s 1ms/step - loss: 0.7318 - acc:
0.5178 - val_loss: 0.7081 - val_acc: 0.5453
Epoch 5/40
7424/7424 [==============================] - 11s 1ms/step - loss: 0.7293 - acc:
0.5353 - val_loss: 0.6971 - val_acc: 0.5927
Epoch 6/40
7424/7424 [==============================] - 11s 1ms/step - loss: 0.6954 - acc:
0.5787 - val_loss: 0.6558 - val_acc: 0.6255
Epoch 7/40
7424/7424 [==============================] - 11s 1ms/step - loss: 0.6715 - acc:
0.6149 - val_loss: 0.6188 - val_acc: 0.6627
Epoch 8/40
7424/7424 [==============================] - 11s 1ms/step - loss: 0.6511 - acc:
0.6303 - val_loss: 0.6068 - val_acc: 0.6681
Epoch 9/40
7424/7424 [==============================] - 11s 1ms/step - loss: 0.6683 - acc:
0.6312 - val_loss: 0.6094 - val_acc: 0.6724
Epoch 10/40
7424/7424 [==============================] - 11s 1ms/step - loss: 0.6405 - acc:
0.6478 - val_loss: 0.5728 - val_acc: 0.6988
```

```
Epoch 11/40
7424/7424 [==============================] - 11s 1ms/step - loss: 0.6062 - acc:
0.6740 - val_loss: 0.5689 - val_acc: 0.7139
Epoch 12/40
7424/7424 [==============================] - 11s 1ms/step - loss: 0.5727 - acc:
0.6956 - val_loss: 0.6067 - val_acc: 0.6827
Epoch 13/40
7424/7424 [==============================] - 11s 1ms/step - loss: 0.5737 - acc:
0.7054 - val_loss: 0.5043 - val_acc: 0.7522
Epoch 14/40
7424/7424 [==============================] - 11s 1ms/step - loss: 0.5336 - acc:
0.7311 - val_loss: 0.5253 - val_acc: 0.7252
Epoch 15/40
7424/7424 [==============================] - 11s 1ms/step - loss: 0.4963 - acc:
0.7507 - val_loss: 0.4794 - val_acc: 0.7980
Epoch 16/40
7424/7424 [==============================] - 11s 1ms/step - loss: 0.4691 - acc:
0.7742 - val_loss: 0.4041 - val_acc: 0.8130
Epoch 17/40
7424/7424 [==============================] - 11s 1ms/step - loss: 0.4219 - acc:
0.8028 - val_loss: 0.3884 - val_acc: 0.8254
Epoch 18/40
7424/7424 [==============================] - 11s 1ms/step - loss: 0.3793 - acc:
0.8270 - val_loss: 0.3840 - val_acc: 0.8357
Epoch 19/40
7424/7424 [==============================] - 11s 1ms/step - loss: 0.3316 - acc:
0.8513 - val_loss: 0.3275 - val_acc: 0.8696
Epoch 20/40
7424/7424 [==============================] - 11s 1ms/step - loss: 0.3113 - acc:
0.8649 - val_loss: 0.3317 - val_acc: 0.8680
Epoch 21/40
7424/7424 [==============================] - 11s 1ms/step - loss: 0.2570 - acc:
0.8905 - val_loss: 0.2797 - val_acc: 0.9025
Epoch 22/40
7424/7424 [==============================] - 11s 1ms/step - loss: 0.2241 - acc:
0.9023 - val_loss: 0.2972 - val_acc: 0.8992
Epoch 23/40
7424/7424 [==============================] - 11s 1ms/step - loss: 0.2054 - acc:
0.9138 - val_loss: 0.4614 - val_acc: 0.8400
Epoch 24/40
7424/7424 [==============================] - 11s 1ms/step - loss: 0.2698 - acc:
0.8976 - val_loss: 0.2828 - val_acc: 0.9176
Epoch 25/40
7424/7424 [==============================] - 11s 1ms/step - loss: 0.1601 - acc:
0.9341 - val_loss: 0.2668 - val_acc: 0.9203
Epoch 26/40
7424/7424 [==============================] - 11s 1ms/step - loss: 0.1316 - acc:
0.9481 - val_loss: 0.2885 - val_acc: 0.9208
Epoch 27/40
7424/7424 [==============================] - 11s 1ms/step - loss: 0.1266 - acc:
0.9500 - val_loss: 0.2664 - val_acc: 0.9364
Epoch 28/40
7424/7424 [==============================] - 11s 1ms/step - loss: 0.1001 - acc:
0.9619 - val_loss: 0.2613 - val_acc: 0.9359
Epoch 29/40
7424/7424 [==============================] - 11s 1ms/step - loss: 0.1041 - acc:
0.9605 - val_loss: 0.2436 - val_acc: 0.9359
```

```
Epoch 30/40
7424/7424 [==============================] - 11s 1ms/step - loss: 0.1106 - acc:
0.9582 - val_loss: 0.2884 - val_acc: 0.9316
Epoch 31/40
7424/7424 [==============================] - 11s 1ms/step - loss: 0.0814 - acc:
0.9704 - val_loss: 0.2736 - val_acc: 0.9370
Epoch 32/40
7424/7424 [==============================] - 11s 1ms/step - loss: 0.0649 - acc:
0.9754 - val_loss: 0.2710 - val_acc: 0.9440
Epoch 33/40
7424/7424 [==============================] - 11s 1ms/step - loss: 0.0545 - acc:
0.9790 - val_loss: 0.2887 - val_acc: 0.9386
Epoch 34/40
7424/7424 [==============================] - 11s 1ms/step - loss: 0.0559 - acc:
0.9802 - val_loss: 0.3066 - val_acc: 0.9359
Epoch 35/40
7424/7424 [==============================] - 11s 1ms/step - loss: 0.0657 - acc:
0.9763 - val_loss: 0.3061 - val_acc: 0.9402
Epoch 36/40
7424/7424 [==============================] - 11s 1ms/step - loss: 0.0663 - acc:
0.9774 - val_loss: 0.2543 - val_acc: 0.9445
Epoch 37/40
7424/7424 [==============================] - 11s 1ms/step - loss: 0.0480 - acc:
0.9832 - val_loss: 0.2869 - val_acc: 0.9445
Epoch 38/40
7424/7424 [==============================] - 11s 1ms/step - loss: 0.0615 - acc:
0.9795 - val_loss: 0.2705 - val_acc: 0.9440
Epoch 39/40
7424/7424 [==============================] - 11s 1ms/step - loss: 0.0730 - acc:
0.9744 - val_loss: 0.2909 - val_acc: 0.9418
Epoch 40/40
7424/7424 [==============================] - 11s 1ms/step - loss: 0.0349 - acc:
0.9875 - val_loss: 0.3139 - val_acc: 0.9456
1856/1856 [==============================] - 1s 429us/step
Accuracy: 94.56%
```

In [70]:

```
import pandas as pd
# Confusion Matrix

confusionmatrix(ytestfinall2,model.predict(xtestfinall2))
```

Out[70]:

Pred	180 degree	270 degree	90 degree	zero degree
True				
180 degree	430	38	0	2
270 degree	27	440	0	0
90 degree	1	0	451	25
zero degree	1	2	28	411

In [80]:

```
import warnings
warnings.filterwarnings('ignore')
model = keras.Sequential()
from keras import layers

model.add(layers.Conv2D(filters=6, kernel_size=(3, 3), activation='relu', input_
model.add(layers.AveragePooling2D())

model.add(layers.Conv2D(filters=16, kernel_size=(3, 3), activation='relu'))
model.add(layers.AveragePooling2D())


model.add(layers.Flatten())

model.add(layers.Dense(units=120, activation='relu'))

model.add(layers.Dense(units=84, activation='relu'))

model.add(layers.Dense(units=10, activation = 'softmax'))
model.add(Dense(units = 4, activation = 'softmax'))
model.compile(optimizer = 'adam', loss = 'categorical_crossentropy', metrics = [

history=model.fit(xtrainfinal,ytrainfinal1, epochs =100, validation_data = (xtes
score = model.evaluate(xtestfinal,ytestfinal1, verbose=1)
print("Accuracy: %.2f%%" % (scores[1]*100))
model_3_test = score[1]
model_3_train = max(history.history['acc'])
```

```
c: 0.9147 - val_loss: 0.3310 - val_acc: 0.8852
Epoch 58/100
7424/7424 [==============================] - 7s 1ms/step - loss: 0.2143 - ac
c: 0.9137 - val_loss: 0.3186 - val_acc: 0.8939
Epoch 59/100
7424/7424 [==============================] - 8s 1ms/step - loss: 0.1884 - ac
c: 0.9282 - val_loss: 0.3127 - val_acc: 0.8971
Epoch 60/100
7424/7424 [==============================] - 8s 1ms/step - loss: 0.1885 - ac
c: 0.9286 - val_loss: 0.3575 - val_acc: 0.8772
Epoch 61/100
7424/7424 [==============================] - 8s 1ms/step - loss: 0.1815 - ac
c: 0.9345 - val_loss: 0.3220 - val_acc: 0.8960
Epoch 62/100
7424/7424 [==============================] - 8s 1ms/step - loss: 0.1618 - ac
c: 0.9436 - val_loss: 0.3307 - val_acc: 0.8885
Epoch 63/100
7424/7424 [==============================] - 8s 1ms/step - loss: 0.1672 - ac
c: 0.9386 - val_loss: 0.3247 - val_acc: 0.9057
Epoch 64/100
```

In [81]:

```
import pandas as pd
# Confusion Matrix

confusionmatrix(ytestfinall2,model.predict(xtestfinall2))
```

Out[81]:

Pred	180 degree	270 degree	90 degree	zero degree
True				
180 degree	452	15	2	1
270 degree	26	440	1	0
90 degree	1	0	436	40
zero degree	1	0	19	422

In [72]:

```
#####################################MY-OWN-NET#################################
import warnings
warnings.filterwarnings('ignore')
classifier = Sequential()
classifier.add(Conv2D(64, (2, 2), input_shape = (256,256,1), activation = 'relu'
classifier.add(MaxPooling2D(pool_size = (2, 2)))
classifier.add(Conv2D(64, (2, 2), activation = 'relu'))
classifier.add(MaxPooling2D(pool_size = (1, 1)))
classifier.add(Dropout(0.2))
classifier.add(Flatten())
classifier.add(Dense(units = 128, activation = 'relu'))
classifier.add(Dropout(0.3))
classifier.add(Dense(units = 4, activation = 'softmax'))
classifier.compile(optimizer = 'adam', loss = 'categorical_crossentropy', metric

history=classifier.fit(xtrainfinal,ytrainfinal1, epochs =30,batch_size=128)
scores = classifier.evaluate(xtrainfinal,ytestfinall2, verbose=1)
print("Accuracy: %.2f%%" % (scores[1]*100))
model_3_test = scores[1]
model_3_train = max(history.history['acc'])
```

```
Epoch 1/30
7424/7424 [==============================] - 2s 321us/step - loss: 1.1425 - ac
c: 0.4263
Epoch 2/30
7424/7424 [==============================] - 1s 122us/step - loss: 0.7631 - ac
c: 0.5524
Epoch 3/30
7424/7424 [==============================] - 1s 122us/step - loss: 0.7060 - ac
c: 0.6002
Epoch 4/30
7424/7424 [==============================] - 1s 121us/step - loss: 0.6571 - ac
c: 0.6381
Epoch 5/30
7424/7424 [==============================] - 1s 122us/step - loss: 0.6164 - ac
c: 0.6747
Epoch 6/30
7424/7424 [==============================] - 1s 121us/step - loss: 0.5876 - ac
c: 0.6949
Epoch 7/30
7424/7424 [==============================] - 1s 123us/step - loss: 0.5461 - ac
c: 0.7267
Epoch 8/30
7424/7424 [==============================] - 1s 122us/step - loss: 0.5167 - ac
c: 0.7457
Epoch 9/30
7424/7424 [==============================] - 1s 120us/step - loss: 0.4807 - ac
c: 0.7761
Epoch 10/30
7424/7424 [==============================] - 1s 121us/step - loss: 0.4460 - ac
c: 0.7989
Epoch 11/30
7424/7424 [==============================] - 1s 122us/step - loss: 0.4045 - ac
c: 0.8187
Epoch 12/30
7424/7424 [==============================] - 1s 122us/step - loss: 0.3892 - ac
```

```
c: 0.8296
Epoch 13/30
7424/7424 [==============================] - 1s 121us/step - loss: 0.3624 - ac
c: 0.8415
Epoch 14/30
7424/7424 [==============================] - 1s 121us/step - loss: 0.3210 - ac
c: 0.8622
Epoch 15/30
7424/7424 [==============================] - 1s 120us/step - loss: 0.3223 - ac
c: 0.8638
Epoch 16/30
7424/7424 [==============================] - 1s 122us/step - loss: 0.2865 - ac
c: 0.8811
Epoch 17/30
7424/7424 [==============================] - 1s 121us/step - loss: 0.2494 - ac
c: 0.8984
Epoch 18/30
7424/7424 [==============================] - 1s 121us/step - loss: 0.2301 - ac
c: 0.9067
Epoch 19/30
7424/7424 [==============================] - 1s 122us/step - loss: 0.2097 - ac
c: 0.9137
Epoch 20/30
7424/7424 [==============================] - 1s 121us/step - loss: 0.1936 - ac
c: 0.9254
Epoch 21/30
7424/7424 [==============================] - 1s 121us/step - loss: 0.1731 - ac
c: 0.9337
Epoch 22/30
7424/7424 [==============================] - 1s 123us/step - loss: 0.1640 - ac
c: 0.9391
Epoch 23/30
7424/7424 [==============================] - 1s 121us/step - loss: 0.1600 - ac
c: 0.9367
Epoch 24/30
7424/7424 [==============================] - 1s 121us/step - loss: 0.1421 - ac
c: 0.9457
Epoch 25/30
7424/7424 [==============================] - 1s 121us/step - loss: 0.1389 - ac
c: 0.9449
Epoch 26/30
7424/7424 [==============================] - 1s 122us/step - loss: 0.1355 - ac
c: 0.9483
Epoch 27/30
7424/7424 [==============================] - 1s 121us/step - loss: 0.1250 - ac
c: 0.9518
Epoch 28/30
7424/7424 [==============================] - 1s 122us/step - loss: 0.1095 - ac
c: 0.9596
Epoch 29/30
7424/7424 [==============================] - 1s 121us/step - loss: 0.1075 - ac
c: 0.9592
Epoch 30/30
7424/7424 [==============================] - 1s 122us/step - loss: 0.1028 - ac
c: 0.9600
1856/1856 [==============================] - 1s 317us/step
Accuracy: 95.85%
```

In [73]:

```
import pandas as pd
# Confusion Matrix

confusionmatrix(ytestfinall2,classifier.predict(xtestfinall2))
```

Out[73]:

Pred	180 degree	270 degree	90 degree	zero degree
True				
180 degree	456	13	1	0
270 degree	13	453	1	0
90 degree	1	0	456	20
zero degree	0	1	27	414

In [0]:

```
references:
https://www.analyticsvidhya.com/blog/2019/08/3-techniques-extract-features-from-
```