In [144]:
```
#importing useful module to read the file and send request to download the images

import pandas as pd
import numpy as np
import os
import csv
import requests
import urllib.request
```

In [145]:
```
os.chdir('C:\\Users\\rpshc\\Desktop\\Assingment') #change working dir
```

In [146]:
```
Data = pd.read_csv("vis10cat.txt", sep='\t',engine='python') #reading the .txt f
Data.columns = ['Graph Name', 'Links'] #assign the columns value
Data['Img_names'] = ''
```

In [148]:
```
c=Data.Links
cnt = 0
for i in c :
    Data['Img_names'][cnt] = i.rsplit('/', 1)[-1]
    cnt=cnt+1
```

In [155]:
```
Data
```

Out[155]:

	Graph Name	Links	Img_names
0	AreaGraph	http://1.bp.blogspot.com/_unlim-8jBw4/S75zvmJC...	Presentaci%25C3%25B3n21.jpg
1	AreaGraph	http://2.bp.blogspot.com/_-P5lluxODKA/S_r3hbO0...	GraficoBasico35.jpg
2	AreaGraph	http://2.bp.blogspot.com/_HmYxml0i5HM/SRNsfSIn...	grafico%2Bd%2Barea.jpg
3	AreaGraph	http://2.bp.blogspot.com/_TJb-wUgjb6E/SGSEvI7H...	areacolor06.jpg
4	AreaGraph	http://2.bp.blogspot.com/_q04f_XGZ4RM/S65vo-ke...	ZA100902863082%255B1%255D.gi
...	...	...	..
2217	VennDiagram	http://www.vincor.com/images/venn_diagram_logo...	venn_diagram_logo.jpg
2218	VennDiagram	http://www5a.biglobe.ne.jp/~tukasamt/image71.jpg	image71.jpg
2219	VennDiagram	http://www7.pioneer.co.jp/edu/contents/99.jpg	99.jpg
2220	VennDiagram	http://www97.intel.com/NR/rdonlyres/AE3E4F8F-1...	Venn.jpg
2221	VennDiagram	https://filebox.vt.edu/users/asimp08/Undergrad...	venn.png

2222 rows × 3 columns

In [169]:
```
#store the Data into thier respective class and make a sub set of Data.
Data_0 = Data.loc[Data['Graph Name'] == 'AreaGraph',['Graph Name','Links','Img_na
Data_1 = Data.loc[Data['Graph Name'] == 'BarGraph',['Graph Name','Links','Img_nar
Data_2 = Data.loc[Data['Graph Name'] == 'LineGraph',['Graph Name','Links','Img_na
Data_3 = Data.loc[Data['Graph Name'] == 'Map',['Graph Name','Links','Img_names']]
Data_4 = Data.loc[Data['Graph Name'] == 'ParetoChart',['Graph Name','Links','Img_
Data_5 = Data.loc[Data['Graph Name'] == 'PieChart',['Graph Name','Links','Img_nar
Data_6 = Data.loc[Data['Graph Name'] == 'RadarPlot',['Graph Name','Links','Img_na
Data_7 = Data.loc[Data['Graph Name'] == 'ScatterGraph',['Graph Name','Links','Img
Data_8 = Data.loc[Data['Graph Name'] == 'Table',['Graph Name','Links','Img_names'
Data_9 = Data.loc[Data['Graph Name'] == 'VennDiagram',['Graph Name','Links','Img_
Data_1
```

Out[169]:

	Graph Name	Links	Img_names
116	BarGraph	http://1.bp.blogspot.com/_KxK5xBFWvSs/TJUMrNMp...	hori.gif
117	BarGraph	http://2.bp.blogspot.com/_HyNWQHeGmKI/THHskrCQ...	HTMLHorizontalChart2.jpg
118	BarGraph	http://3.bp.blogspot.com/_0rVBe60IjJM/Sgf5SUhQ...	grafico%2Bde%2Bbarras.jpg
119	BarGraph	http://3d-stacked-horizontal-bar-graph-softwar...	3d-stacked-horizontal-bar-graph-software4946.jpg
120	BarGraph	http://4.bp.blogspot.com/_99tkcbyuK_8/TJ7E9Rjx...	image044.gif
...	...	...	...
442	BarGraph	http://www.webopedia.com/FIG/COLUMN.gif	COLUMN.gif
443	BarGraph	http://www.wmburgweb.com/Resources/Lesson/char...	charts3.gif
444	BarGraph	http://www.xn--grne-gifhorn-elb.de/Gr/gerechte...	gerechte_Verteilung7.jpg
445	BarGraph	http://www.zur-zeit.ch/berufseinstieg/bilder/s...	saeulen.jpg
446	BarGraph	https://www.bgw-online.de/quintas/generator/In...	pareto__Bild,property%3Dbild.gif

331 rows × 3 columns

In [161]:
```
#Run the respect class file.
u = Data_0['Links'].tolist() #covert the Links colunms into list.
cnt = 0
for i in u:
    os.chdir('C:\\Users\\rpshc\\Desktop\\Assingment\\Images\\AreaGraph') #change
    name =   str(cnt)+ ' ' + Data['Img_names'][cnt]   #save the file correnpondi
    cnt = cnt +1;
    try:                                    #try  the url if courrepted pass and
        urllib.request.urlretrieve(i, name) #use the urllib to send request and
    except:
        pass
```

In [179]:
```
u = Data_1['Links'].tolist()

cnt = 0
for i in u:
    os.chdir('C:\\Users\\rpshc\\Desktop\\Assingment\\Images\\BarGraph')
    name =   str(cnt) + ' ' + Data_1.reset_index()['Img_names'][cnt]
    cnt = cnt +1;
    try:
        urllib.request.urlretrieve(i, name)
    except:
        pass
```

In [180]:
```
u = Data_2['Links'].tolist()

cnt = 0
for i in u:
    os.chdir('C:\\Users\\rpshc\\Desktop\\Assingment\\Images\\LineGraph')
    name = str(cnt) + ' ' + Data_2.reset_index()['Img_names'][cnt]
    cnt = cnt +1;
    try:
        urllib.request.urlretrieve(i, name)
    except:
        pass
```

In [181]:
```
u = Data_3['Links'].tolist()

cnt = 0
for i in u:
    os.chdir('C:\\Users\\rpshc\\Desktop\\Assingment\\Images\\Map')
    name = str(cnt) + ' ' + Data_3.reset_index()['Img_names'][cnt]
    cnt = cnt +1;
    try:
        urllib.request.urlretrieve(i, name)
    except:
        pass
```

In [182]:
```
u = Data_4['Links'].tolist()

cnt = 0
for i in u:
    os.chdir('C:\\Users\\rpshc\\Desktop\\Assingment\\Images\\ParetoChart')
    name = str(cnt) + ' ' + Data_4.reset_index()['Img_names'][cnt]
    cnt = cnt +1;
    try:
        urllib.request.urlretrieve(i, name)
    except:
        pass
```

In [183]:
```
u = Data_5['Links'].tolist()

cnt = 0
for i in u:
    os.chdir('C:\\Users\\rpshc\\Desktop\\Assingment\\Images\\PieChart')
    name =str(cnt) + ' ' + Data_5.reset_index()['Img_names'][cnt]
    cnt = cnt +1;
    try:
        urllib.request.urlretrieve(i, name)
    except:
        pass
```

In [184]:
```
u = Data_6['Links'].tolist()

cnt = 0
for i in u:
    os.chdir('C:\\Users\\rpshc\\Desktop\\Assingment\\Images\\RadarPlot')
    name =str(cnt) + ' ' + Data_6.reset_index()['Img_names'][cnt]
    cnt = cnt +1;
    try:
        urllib.request.urlretrieve(i, name)
    except:
        pass
```

In [185]:
```
u = Data_7['Links'].tolist()

cnt = 0
for i in u:
    os.chdir('C:\\Users\\rpshc\\Desktop\\Assingment\\Images\\ScatterGraph')
    name = str(cnt) + ' ' + Data_7.reset_index()['Img_names'][cnt]
    cnt = cnt +1;
    try:
        urllib.request.urlretrieve(i, name)
    except:
        pass
```

In [186]:
```
u = Data_8['Links'].tolist()

cnt = 0
for i in u:
    os.chdir('C:\\Users\\rpshc\\Desktop\\Assingment\\Images\\Table')
    name = str(cnt) + ' ' + Data_8.reset_index()['Img_names'][cnt]
    cnt = cnt +1;
    try:
        urllib.request.urlretrieve(i, name)
    except:
        pass
```

In [187]:
```
u = Data_9['Links'].tolist()
cnt = 0
for i in u:
    os.chdir('C:\\Users\\rpshc\\Desktop\\Assingment\\Images\\VennDiagram')
    name = str(cnt) + ' ' + Data_9.reset_index()['Img_names'][cnt]
    cnt = cnt +1;
    try:
        urllib.request.urlretrieve(i, name)
    except:
        pass
```

In [608]:
```
#Building the CNN Model
```

In [609]:
```
# Importing the Keras libraries and packages
from keras.models import Sequential
from keras.layers import Convolution2D
from keras.layers import MaxPooling2D
from keras.layers import Flatten
from keras.layers import Dense
from keras.models import model_from_json

batch_size = 32
```

In [610]:
```
from tensorflow.keras.preprocessing.image import ImageDataGenerator

# All images will be rescaled by 1./255
train_datagen = ImageDataGenerator(rescale=1/255)

# Flow training images in batches of 32 using train_datagen generator
train_generator = train_datagen.flow_from_directory(
        'C:/Users/rpshc/Desktop/Assingment/Base_Folder/train/',  # This is the so
        target_size=(200, 200),  # All images will be resized to 200 x 200
        batch_size=batch_size,
        # Specify the classes explicitly
        classes = ['AreaGraph', 'BarGraph', 'LineGraph', 'Map', 'ParetoChart',
               'PieChart','RadarPlot','ScatterGraph','Table','VennDiagram'],
        # Since we use categorical_crossentropy loss, we need categorical labels
        class_mode='categorical')
```

```
Found 439 images belonging to 10 classes.
```

In [611]:
```
import tensorflow as tf

model = tf.keras.models.Sequential([
    # Note the input shape is the desired size of the image 200x 200 with 3 bytes
    # The first convolution
    tf.keras.layers.Conv2D(16, (3,3), activation='relu', input_shape=(200, 200, 3
    tf.keras.layers.MaxPooling2D(2, 2),
    # The second convolution
    tf.keras.layers.Conv2D(32, (3,3), activation='relu'),
    tf.keras.layers.MaxPooling2D(2,2),
    # The third convolution
    tf.keras.layers.Conv2D(64, (3,3), activation='relu'),
    tf.keras.layers.MaxPooling2D(2,2),
    # The fourth convolution
    tf.keras.layers.Conv2D(64, (3,3), activation='relu'),
    tf.keras.layers.MaxPooling2D(2,2),
    # The fifth convolution
    tf.keras.layers.Conv2D(64, (3,3), activation='relu'),
    tf.keras.layers.MaxPooling2D(2,2),
    # Flatten the results to feed into a dense layer
    tf.keras.layers.Flatten(),
    # 128 neuron in the fully-connected layer
    tf.keras.layers.Dense(128, activation='relu'),
    # 10 output neurons for 10 classes with the softmax activation
    tf.keras.layers.Dense(10, activation='softmax')
])
```

In [612]:
```
model.summary()
```

```
Model: "sequential_21"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_64 (Conv2D)           (None, 198, 198, 16)      448
_________________________________________________________________
max_pooling2d_64 (MaxPooling (None, 99, 99, 16)        0
_________________________________________________________________
conv2d_65 (Conv2D)           (None, 97, 97, 32)        4640
_________________________________________________________________
max_pooling2d_65 (MaxPooling (None, 48, 48, 32)        0
_________________________________________________________________
conv2d_66 (Conv2D)           (None, 46, 46, 64)        18496
_________________________________________________________________
max_pooling2d_66 (MaxPooling (None, 23, 23, 64)        0
_________________________________________________________________
conv2d_67 (Conv2D)           (None, 21, 21, 64)        36928
_________________________________________________________________
max_pooling2d_67 (MaxPooling (None, 10, 10, 64)        0
_________________________________________________________________
conv2d_68 (Conv2D)           (None, 8, 8, 64)          36928
_________________________________________________________________
max_pooling2d_68 (MaxPooling (None, 4, 4, 64)          0
_________________________________________________________________
flatten_21 (Flatten)         (None, 1024)              0
_________________________________________________________________
dense_42 (Dense)             (None, 128)               131200
_________________________________________________________________
dense_43 (Dense)             (None, 10)                1290
=================================================================
Total params: 229,930
Trainable params: 229,930
Non-trainable params: 0
_________________________________________________________________
```

In [613]:
```
from tensorflow.keras.optimizers import RMSprop

model.compile(loss='categorical_crossentropy',
              optimizer=RMSprop(lr=0.001),
              metrics=['acc'])

total_sample=train_generator.n

n_epochs = 25

history = model.fit_generator(
        train_generator,
        steps_per_epoch=int(total_sample/batch_size),
        epochs=n_epochs,
        verbose=1)

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

```
Epoch 1/30
13/13 [==============================] - 17s 1s/step - loss: 2.2890 - acc: 0.16
95
Epoch 2/30
13/13 [==============================] - 17s 1s/step - loss: 2.3546 - acc: 0.20
15
Epoch 3/30
13/13 [==============================] - 16s 1s/step - loss: 2.1384 - acc: 0.22
12
Epoch 4/30
13/13 [==============================] - 16s 1s/step - loss: 2.0949 - acc: 0.25
06
Epoch 5/30
13/13 [==============================] - 16s 1s/step - loss: 2.0105 - acc: 0.28
50
Epoch 6/30
13/13 [==============================] - 15s 1s/step - loss: 1.8336 - acc: 0.32
68
Epoch 7/30
13/13 [==============================] - 16s 1s/step - loss: 1.8477 - acc: 0.34
40
Epoch 8/30
13/13 [==============================] - 16s 1s/step - loss: 1.9011 - acc: 0.32
92
Epoch 9/30
13/13 [==============================] - 19s 1s/step - loss: 1.6835 - acc: 0.42
01
Epoch 10/30
13/13 [==============================] - 16s 1s/step - loss: 1.5749 - acc: 0.45
70
Epoch 11/30
13/13 [==============================] - 15s 1s/step - loss: 1.6151 - acc: 0.43
24
Epoch 12/30
13/13 [==============================] - 16s 1s/step - loss: 1.4358 - acc: 0.52
58
Epoch 13/30
13/13 [==============================] - 17s 1s/step - loss: 1.4813 - acc: 0.51
84
```

```
Epoch 14/30
13/13 [==============================] - 15s 1s/step - loss: 1.2526 - acc: 0.60
20
Epoch 15/30
13/13 [==============================] - 17s 1s/step - loss: 1.1673 - acc: 0.60
20
Epoch 16/30
13/13 [==============================] - 16s 1s/step - loss: 1.0531 - acc: 0.65
60
Epoch 17/30
13/13 [==============================] - 16s 1s/step - loss: 1.0693 - acc: 0.62
90
Epoch 18/30
13/13 [==============================] - 15s 1s/step - loss: 0.8703 - acc: 0.67
81
Epoch 19/30
13/13 [==============================] - 15s 1s/step - loss: 0.6910 - acc: 0.73
96
Epoch 20/30
13/13 [==============================] - 16s 1s/step - loss: 0.7334 - acc: 0.75
92
Epoch 21/30
13/13 [==============================] - 15s 1s/step - loss: 0.6391 - acc: 0.76
90
Epoch 22/30
13/13 [==============================] - 15s 1s/step - loss: 0.5101 - acc: 0.81
57
Epoch 23/30
13/13 [==============================] - 15s 1s/step - loss: 0.5427 - acc: 0.79
36
Epoch 24/30
13/13 [==============================] - 15s 1s/step - loss: 0.4506 - acc: 0.85
26
Epoch 25/30
13/13 [==============================] - 15s 1s/step - loss: 0.4212 - acc: 0.87
96
Epoch 26/30
13/13 [==============================] - 13s 975ms/step - loss: 0.3185 - acc:
0.8821
Epoch 27/30
13/13 [==============================] - 15s 1s/step - loss: 0.2664 - acc: 0.92
14
Epoch 28/30
13/13 [==============================] - 15s 1s/step - loss: 0.3959 - acc: 0.88
21
Epoch 29/30
13/13 [==============================] - 15s 1s/step - loss: 0.2104 - acc: 0.93
12
Epoch 30/30
13/13 [==============================] - 15s 1s/step - loss: 0.2660 - acc: 0.91
89
```

In [635]:
```
from tensorflow.keras.preprocessing.image import ImageDataGenerator

# All images will be rescaled by 1./255
val_generator = ImageDataGenerator(rescale=1/255)

# Flow training images in batches of 32 using val_generator generator
val_generator = train_datagen.flow_from_directory(
        'C:/Users/rpshc/Desktop/Assingment/Base_Folder/val/',  # This is the sour
        target_size=(200, 200),  # All images will be resized to 200 x 200
        batch_size=batch_size,
        # Specify the classes explicitly
        classes = ['AreaGraph', 'BarGraph', 'LineGraph', 'Map', 'ParetoChart',
               'PieChart','RadarPlot','ScatterGraph','Table','VennDiagram'],
        # Since we use categorical_crossentropy loss, we need categorical labels
        class_mode='categorical')
```

```
Found 190 images belonging to 10 classes.
```

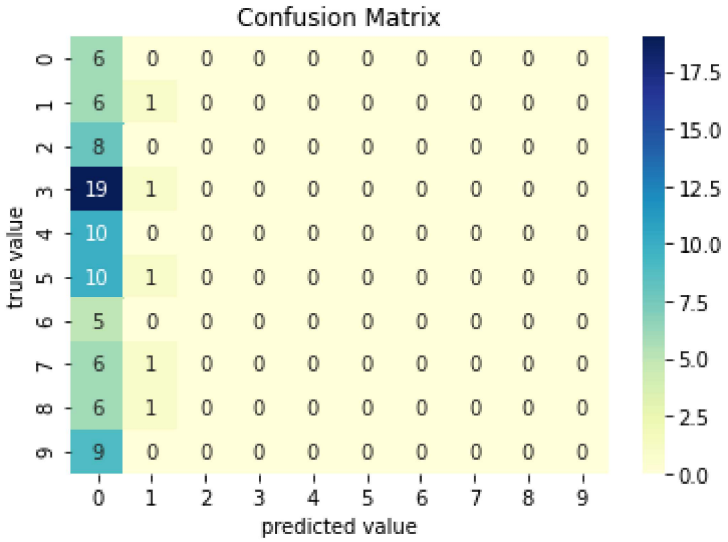
In [636]:
```
from tensorflow.keras.preprocessing.image import ImageDataGenerator

# All images will be rescaled by 1./255
test_datagen = ImageDataGenerator(rescale=1/255)

# Flow training images in batches of 32 using test_generator generator
test_generator = train_datagen.flow_from_directory(
        'C:/Users/rpshc/Desktop/Assingment/Base_Folder/test/',  # This is the sou
        target_size=(200, 200),  # All images will be resized to 200 x 200
        batch_size=batch_size,
        # Specify the classes explicitly
        classes = ['AreaGraph', 'BarGraph', 'LineGraph', 'Map', 'ParetoChart',
               'PieChart','RadarPlot','ScatterGraph','Table','VennDiagram'],
        # Since we use categorical_crossentropy loss, we need categorical labels
        class_mode='categorical')
```

```
Found 90 images belonging to 10 classes.
```

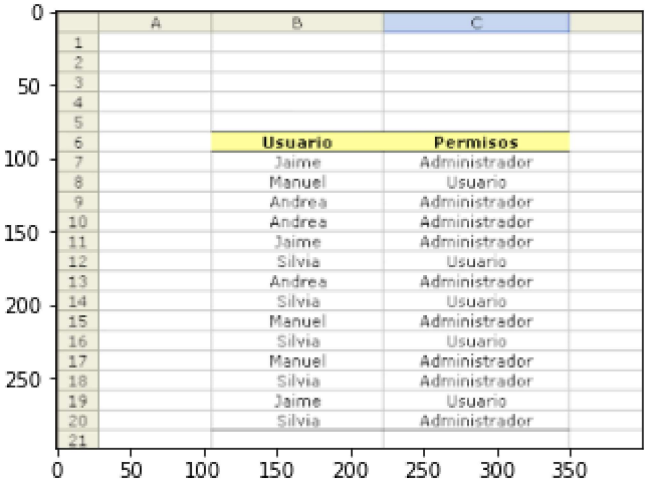
In [683]:
```
import pandas as pd
test_generator.reset
ytesthat = model.predict_generator(test_generator)
df = pd.DataFrame({
    'filename':test_set.filenames,
    'predict':ytesthat[:,0],
    'y':test_generator.classes
})
```

In [684]:
```
pd.set_option('display.float_format', lambda x: '%.5f' % x)
df['y_pred'] = df['predict'] >0.5
df.y_pred = df.y_pred.astype(int)
```

In [685]:
```
#Prediction of test set
from sklearn.metrics import confusion_matrix
import matplotlib.pyplot as plt
import seaborn as sns

conf_matrix = confusion_matrix(df.y,df.y_pred)
sns.heatmap(conf_matrix,cmap="YlGnBu",annot=True,fmt='g');
plt.xlabel('predicted value')
plt.ylabel('true value');
plt.title('Confusion Matrix');
```



In [620]:
```
#Input Image for Layer visualization
img1 = image.load_img('C:/Users/rpshc/Desktop/Assingment/Images/Table/14 tabla-de
plt.imshow(img1);
#preprocess image
img1 = image.load_img('C:/Users/rpshc/Desktop/Assingment/Images/Table/14 tabla-de
img = image.img_to_array(img1)
img = img/255
img = np.expand_dims(img, axis=0)
```



In [621]:
```
model_layers = [ layer.name for layer in classifier.layers]
print('layer name : ',model_layers)
```

```
layer name :  ['conv2d_42', 'max_pooling2d_42', 'conv2d_43', 'max_pooling2d_4
3', 'flatten_16', 'dense_32', 'dense_33']
```

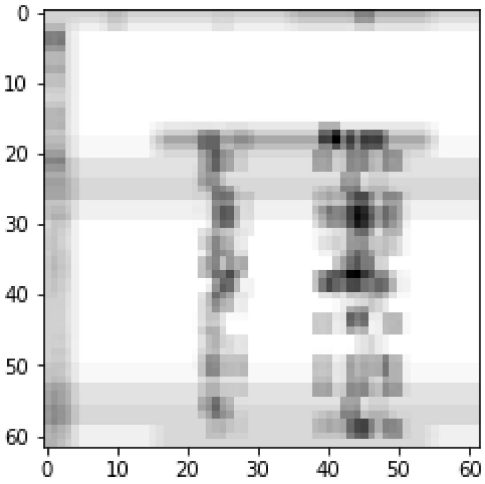
In [622]:
```
from tensorflow.keras.models import Model
conv2d_6_output = Model(inputs=classifier.input, outputs=classifier.get_layer('co
conv2d_7_output = Model(inputs=classifier.input,outputs=classifier.get_layer('con
```

In [623]:
```
conv2d_6_features = conv2d_6_output.predict(img)
conv2d_7_features = conv2d_7_output.predict(img)
print('First conv layer feature output shape : ',conv2d_6_features.shape)
print('First conv layer feature output shape : ',conv2d_7_features.shape)
```

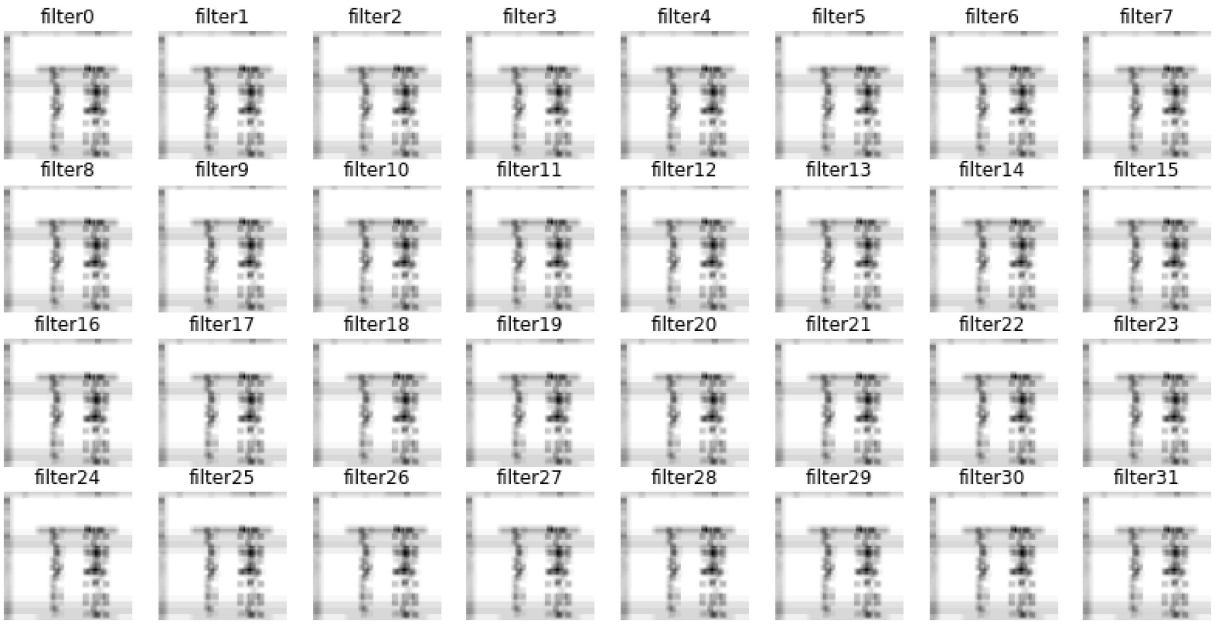
```
First conv layer feature output shape :  (1, 62, 62, 32)
First conv layer feature output shape :  (1, 29, 29, 32)
```

In [624]:
```
plt.imshow(conv2d_6_features[0, :, :, 4], cmap='gray')
```

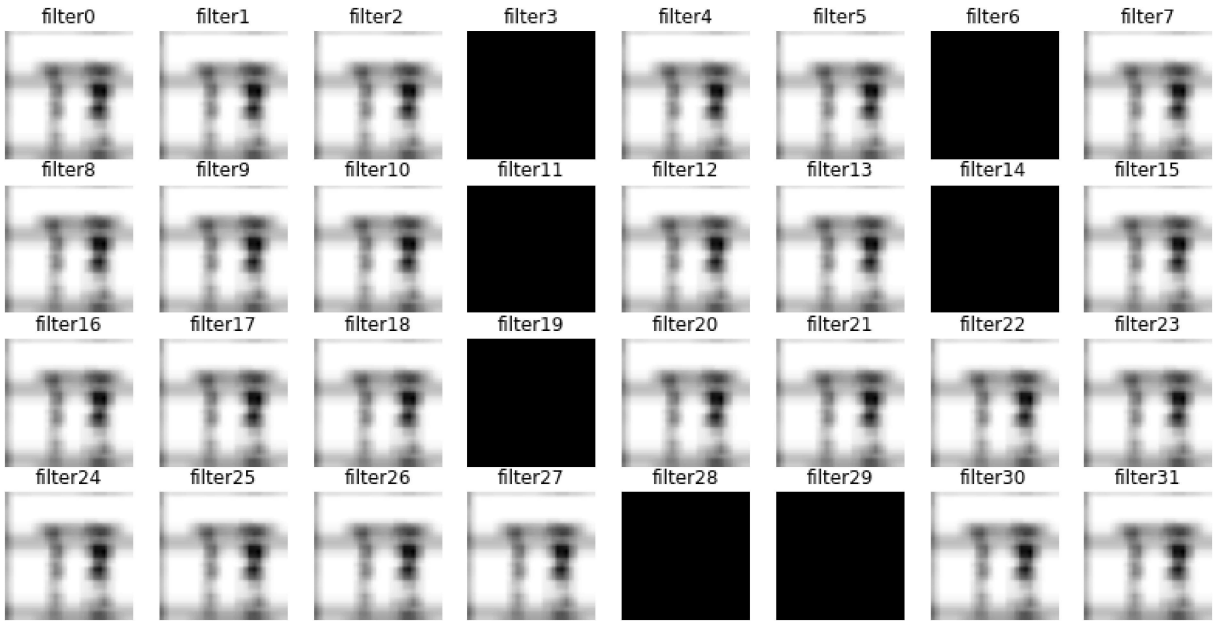
Out[624]:
```
<matplotlib.image.AxesImage at 0x1a88a150250>
```



In [625]:
```
import matplotlib.image as mpimg

fig=plt.figure(figsize=(14,7))
columns = 8
rows = 4
for i in range(columns*rows):
    #img = mpimg.imread()
    fig.add_subplot(rows, columns, i+1)
    plt.axis('off')
    plt.title('filter'+str(i))
    plt.imshow(conv2d_6_features[0, :, :, i], cmap='gray')
plt.show()
```



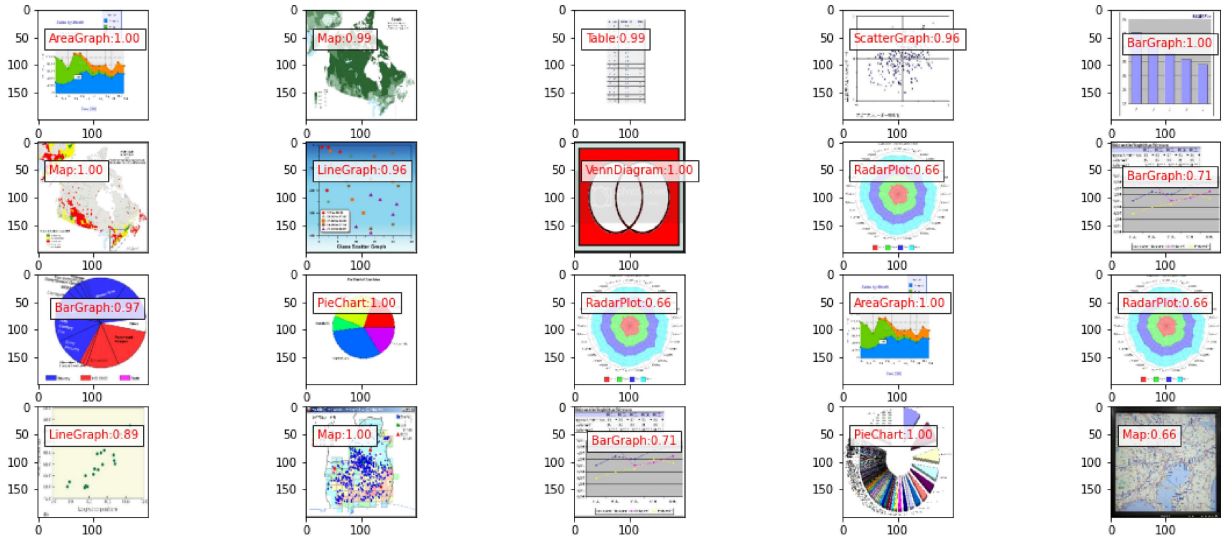
In [626]:
```
fig=plt.figure(figsize=(14,7))
columns = 8
rows = 4
for i in range(columns*rows):
    #img = mpimg.imread()
    fig.add_subplot(rows, columns, i+1)
    plt.axis('off')
    plt.title('filter'+str(i))
    plt.imshow(conv2d_7_features[0, :, :, i], cmap='gray')
plt.show()
```



In [631]:
```
import numpy as np
from keras.preprocessing import image
test_image = image.load_img('C:/Users/rpshc/Desktop/Assingment/Images/Table/14 ta
#test_image = image.img_to_array(test_image)
test_image = np.expand_dims(test_image, axis=0)
result = model.predict(test_image)

result
```

Out[631]:
```
array([[0., 0., 0., 0., 0., 0., 0., 0., 1., 0.]], dtype=float32)
```

In [689]:
```
# for generator image set u can use
# ypred = classifier.predict_generator(test_set)

fig=plt.figure(figsize=(20, 8))
columns = 5
rows = 4
for i in range(columns*rows):
    fig.add_subplot(rows, columns, i+1)
    img1 = image.load_img('C:\\Users\\rpshc\\Desktop\\Assingment\\Base_Folder\\te
    img = image.img_to_array(img1)
    img = img/255
    img = np.expand_dims(img, axis=0)
    result = model.predict(img, batch_size=None,steps=1) #gives all class prob.
    if(result[0][0]>0.5):
        value = 'AreaGraph:%1.2f'%result[0][0]
        plt.text(20, 58,value,color='red',fontsize=10,bbox=dict(facecolor='white'
    elif (result[0][1]>0.5):
        value = 'BarGraph:%1.2f'%result[0][1]
        plt.text(30, 68,value,color='red',fontsize=10,bbox=dict(facecolor='white'
    elif (result[0][2]>0.5):
        value ='LineGraph:%1.2f'%result[0][2]
        plt.text(20, 58,value,color='red',fontsize=10,bbox=dict(facecolor='white'
    elif (result[0][3]>0.5):
        value ='Map:%1.2f'%result[0][3]
        plt.text(20, 58,value,color='red',fontsize=10,bbox=dict(facecolor='white'
    elif (result[0][4]>0.5):
        value ='ParetoChart:%1.2f'%result[0][4]
        plt.text(20, 58,value,color='red',fontsize=10,bbox=dict(facecolor='white'

    elif (result[0][5]>0.5):
        value ='PieChart:%1.2f'%result[0][5]
        plt.text(20, 58,value,color='red',fontsize=10,bbox=dict(facecolor='white'

    elif (result[0][6]>0.5):
        value ='RadarPlot:%1.2f'%result[0][6]
        plt.text(20, 58,value,color='red',fontsize=10,bbox=dict(facecolor='white'

    elif (result[0][7]>0.5):
        value ='ScatterGraph:%1.2f'%result[0][7]
        plt.text(20, 58,value,color='red',fontsize=10,bbox=dict(facecolor='white'

    elif (result[0][8]>0.5):
        value ='Table:%1.2f'%result[0][8]
        plt.text(20, 58,value,color='red',fontsize=10,bbox=dict(facecolor='white'

    elif (result[0][9]>0.5):
        value ='VennDiagram:%1.2f'%result[0][9]
        plt.text(20, 58,value,color='red',fontsize=10,bbox=dict(facecolor='white'
    plt.imshow(img1)
```



In [675]:
```
%%capture
# Model Accuracy
x1 = classifier.evaluate_generator(train_set)
x2 = classifier.evaluate_generator(val_set)
```

In [677]:
```
print('Training Accuracy  : %1.2f%%     Training loss  : %1.4f'%(x1[1]*100,x1[0]))
print('Validation Accuracy: %1.2f%%     Validation loss: %1.4f'%(x2[1]*100,x2[0]))
```

```
Training Accuracy  : 12.07%     Training loss  : -61784207360.0000
Validation Accuracy: 11.58%     Validation loss: -62473428992.0000
```

In [ ]: