```
#Importing libraries
import json
import numpy as np
import matplotlib.pyplot as plt
import tensorflow as tf
%matplotlib inline
import os
import pandas as pd
from keras.models import Sequential
from keras.layers import Dense, Flatten, Activation
from keras.layers import Dropout
from keras.layers.convolutional import Conv2D, MaxPooling2D
from keras.utils import np_utils
from keras.optimizers import SGD
import keras.callbacks
```

Printing packages versions

```
print(np.__version__)
print(tf.__version__)
print(pd.__version__)
print(keras.__version__)
```

```
1.18.5
2.2.0
1.0.5
2.3.1
```

Displaying the directories and files

```
os.listdir('/content/drive/My Drive/2020/ShipsData')
```

```
['Ship1.png',
 '0__20160710_182140_0c78__-122.33743457924228_37.75728971371875.png',
 'shipsnet.json',
 '1__20170716_180816_103a__-122.36098977286657_37.766640212768245.png',
 'shipsnet',
 'scenes']
```

Finding number of files present in folder

```
len(os.listdir('/content/drive/My Drive/2020/ShipsData/shipsnet/shipsnetfolder'))
```

```
4000
```

```
len(os.listdir('/content/drive/My Drive/2020/ShipsData/scenes/scenefolder'))
```

```
8
```

Creating multiple directories

```
basefolder="/content/drive/My Drive/2020/ShipsData"
shipsr=os.path.join(basefolder,'shipsnet')
scenesr=os.path.join(basefolder,'scenes')


shipsimg=os.path.join(shipsr,'shipsnetfolder')
scenesimg=os.path.join(scenesr,'scenefolder')
```

Selectimg random files

```
ships_images = os.listdir(shipsimg)
ships_images[:6]
```

```
['1__20170827_181130_1014__-122.33912795640656_37.73920986014904.png',
 '1__20170901_181520_0e14__-122.35948134600083_37.76776767609284.png',
 '1__20170901_181520_0e14__-122.36740002724083_37.80182586116483.png',
 '1__20170901_181520_0e14__-122.35176479998303_37.7815966425164.png',
 '1__20170901_181520_0e14__-122.35121081280293_37.74752401629368.png',
 '1__20170901_181520_0e14__-122.35466805228293_37.75722310404933.png']
```

```
plt.imshow(plt.imread(shipsimg+'/1__20170901_181520_0e14__-122.35466805228293_37.757223104049
```

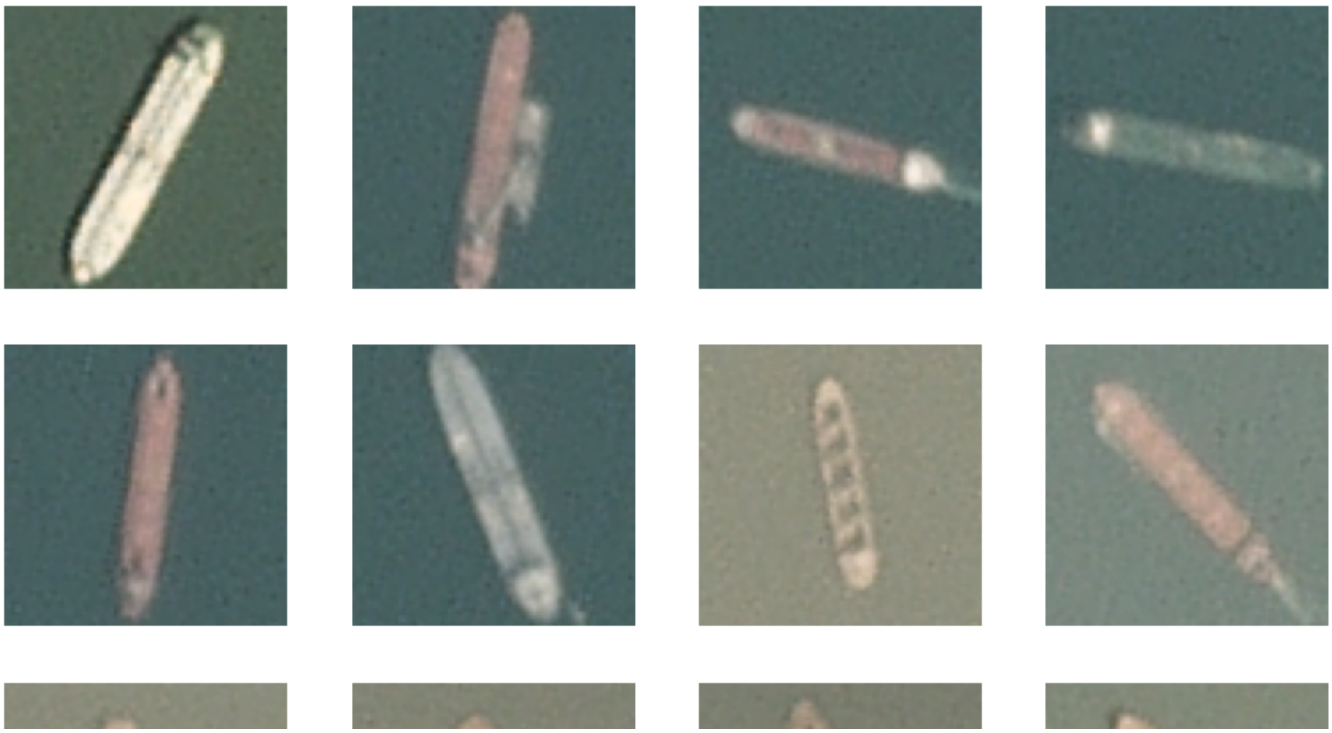
```
<matplotlib.image.AxesImage at 0x7f17be1ec2b0>
```

visualising some images from shipsnet

```
#Slicing
ships_images = os.listdir(shipsimg)
ships_images[:6]
```

```
['1__20170827_181130_1014__-122.33912795640656_37.73920986014904.png',
 '1__20170901_181520_0e14__-122.35948134600083_37.76776767609284.png',
 '1__20170901_181520_0e14__-122.36740002724083_37.80182586116483.png',
 '1__20170901_181520_0e14__-122.35176479998303_37.7815966425164.png',
 '1__20170901_181520_0e14__-122.35121081280293_37.74752401629368.png',
 '1__20170901_181520_0e14__-122.35466805228293_37.75722310404933.png']
```

```
plt.figure(figsize=(15,15))
j=1
for i in range(16):
  img=plt.imread(os.path.join(shipsimg,ships_images[i]))
  plt.subplot(4,4,j)
  plt.imshow(img)
  plt.title(img.shape)
  plt.axis('off')
  j+=1
```



visualising scenes images from scenes

```
scenes_images = os.listdir(scenesimg)
scenes_images[:6]
```

```
['lb_1.png', 'lb_2.png', 'sfbay_1.png', 'lb_3.png', 'lb_4.png', 'sfbay_3.png']
```

```
plt.figure(figsize=(15,15))
j=1
for i in range(8):
  img=plt.imread(os.path.join(scenesimg,scenes_images[i]))
  plt.subplot(4,4,j)
  plt.imshow(img)
  plt.title(img.shape)
  plt.axis('off')
  j+=1
```

scaling the model

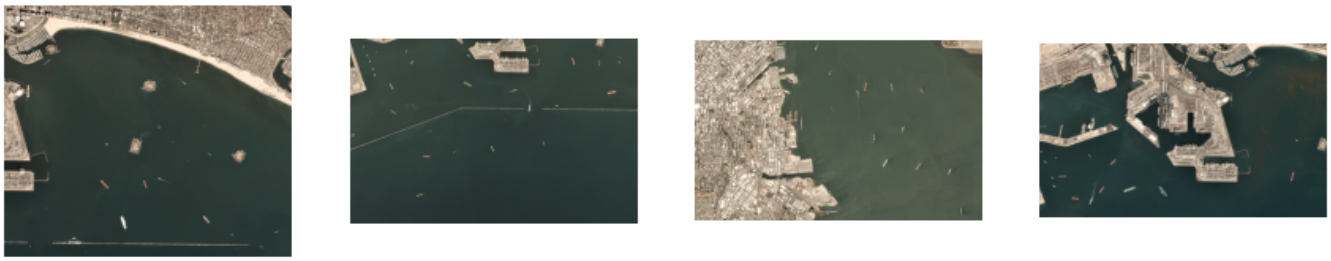
```
x_data = x / 255
```

Building the data

```
model = Sequential()
#add a convolutional layer followed by maxpooling
model.add(Conv2D(12,3, padding='same', activation='relu' , input_shape=(80,80,3)))
model.add(MaxPooling2D(pool_size=(2, 2)))
#add a convolutional layer followed by maxpooling
model.add(Conv2D(24,3,padding='same', activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
#add a convolutional layer followed by maxpooling
model.add(Conv2D(48,3, padding='same', activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Conv2D(96,3, padding='same', activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Flatten())
model.add(Dense(512, activation='relu'))
#Final output layer
model.add(Dense(1, activation='sigmoid'))
```

Model Summary

```
model.summary()
```



Opening .json file

```
with open('/content/drive/My Drive/2020/ShipsData/shipsnet.json') as file_name:
  z=json.load(file_name)
  ships=pd.DataFrame(z)

ships.head()
```

	data	labels	locations	scene_ids
0	[82, 89, 91, 87, 89, 87, 86, 86, 86, 86, 84, 8...	1	[-118.2254694333423, 33.73803725920789]	20180708_180909_0f47
1	[76, 75, 67, 62, 68, 72, 73, 73, 68, 69, 69, 6...	1	[-122.33222866289329, 37.7491755586813]	20170705_180816_103e
2	[125, 127, 129, 130, 126, 125, 129, 133, 132, ...	1	[-118.14283073363218, 33.736016066914175]	20180712_211331_0f06

```
x=np.array(z['data']).astype('int64')
y=np.array(z['labels']).astype('int64')


x.shape
```

```
(4000, 19200)
```

```
y
```

```
array([1, 1, 1, ..., 0, 0, 0])
```

```
x = x.reshape([-1,3,80,80]).transpose([0,2,3,1])
x.shape
```

```
(4000, 80, 80, 3)
```

```
Model: "sequential_4"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_13 (Conv2D)           (None, 80, 80, 12)        336
_________________________________________________________________
max_pooling2d_13 (MaxPooling (None, 40, 40, 12)        0
_________________________________________________________________
conv2d_14 (Conv2D)           (None, 40, 40, 24)        2616
_________________________________________________________________
max_pooling2d_14 (MaxPooling (None, 20, 20, 24)        0
_________________________________________________________________
conv2d_15 (Conv2D)           (None, 20, 20, 48)        10416
_________________________________________________________________
max_pooling2d_15 (MaxPooling (None, 10, 10, 48)        0
```

compiling the model

```
model.compile(optimizer='adam',loss=tf.keras.losses.BinaryCrossentropy(),metrics=['accuracy']
```

```
dense_7 (Dense)              (None, 512)               1229312
```

fitting the data

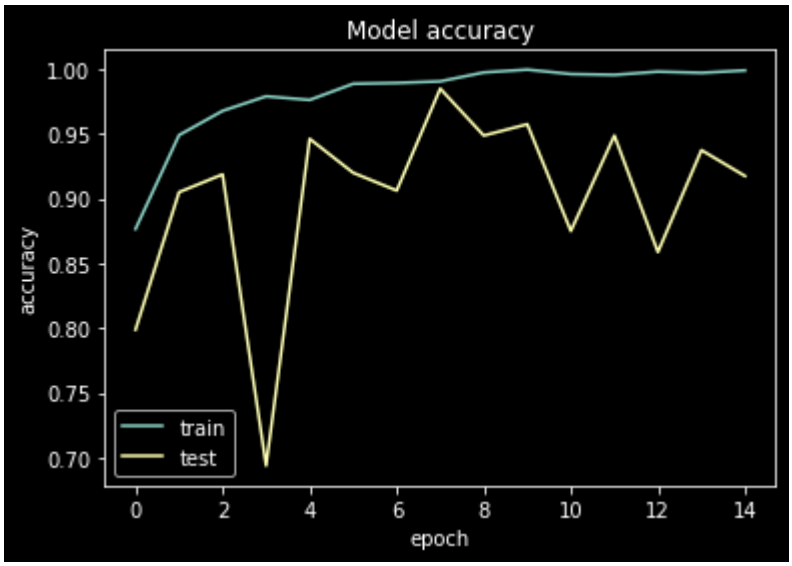
```
Trainable params: 1.284.761
```

```
history = model.fit(x_data,y,epochs=15,batch_size=32,validation_split=0.2)
```

```
Train on 3200 samples, validate on 800 samples
Epoch 1/15
3200/3200 [==============================] - 22s 7ms/step - loss: 0.2739 - accuracy: 0.8
Epoch 2/15
3200/3200 [==============================] - 22s 7ms/step - loss: 0.1332 - accuracy: 0.9
Epoch 3/15
3200/3200 [==============================] - 22s 7ms/step - loss: 0.0904 - accuracy: 0.9
```

Model Accuracy graph

```
from matplotlib import style
style.use('dark_background')
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('Model accuracy')
plt.ylabel('accuracy')
plt.xlabel('epoch')
plt.legend(['train', 'test'])
plt.show()
```



Value Loass graph

```
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('Model loss')
plt.ylabel('accuracy')
plt.xlabel('epoch')
plt.legend(['train', 'test'])
plt.show()
```

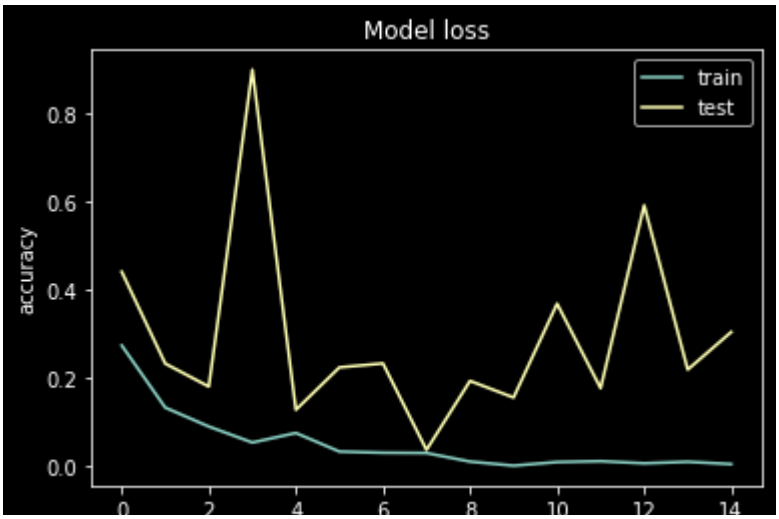


Image Prediction

```
from tensorflow.keras.preprocessing import image
img=image.load_img('/content/drive/My Drive/2020/ShipsData/1__20170716_180816_103a__-122.3609
type(img)
```

```
PIL.PngImagePlugin.PngImageFile
```

```
img=tf.keras.preprocessing.image.img_to_array(img)
print(img.shape)
print(type(img))
```

```
(80, 80, 3)
<class 'numpy.ndarray'>
```

```
img=tf.image.resize(img,(80,80))
img=img/255
print(img.shape)
```

```
(80, 80, 3)
```

```
img=np.expand_dims(img,axis=0)
print(img.shape)
```

```
(1, 80, 80, 3)
```

```
model.predict(img)
```

```
array([[1.]], dtype=float32)
```