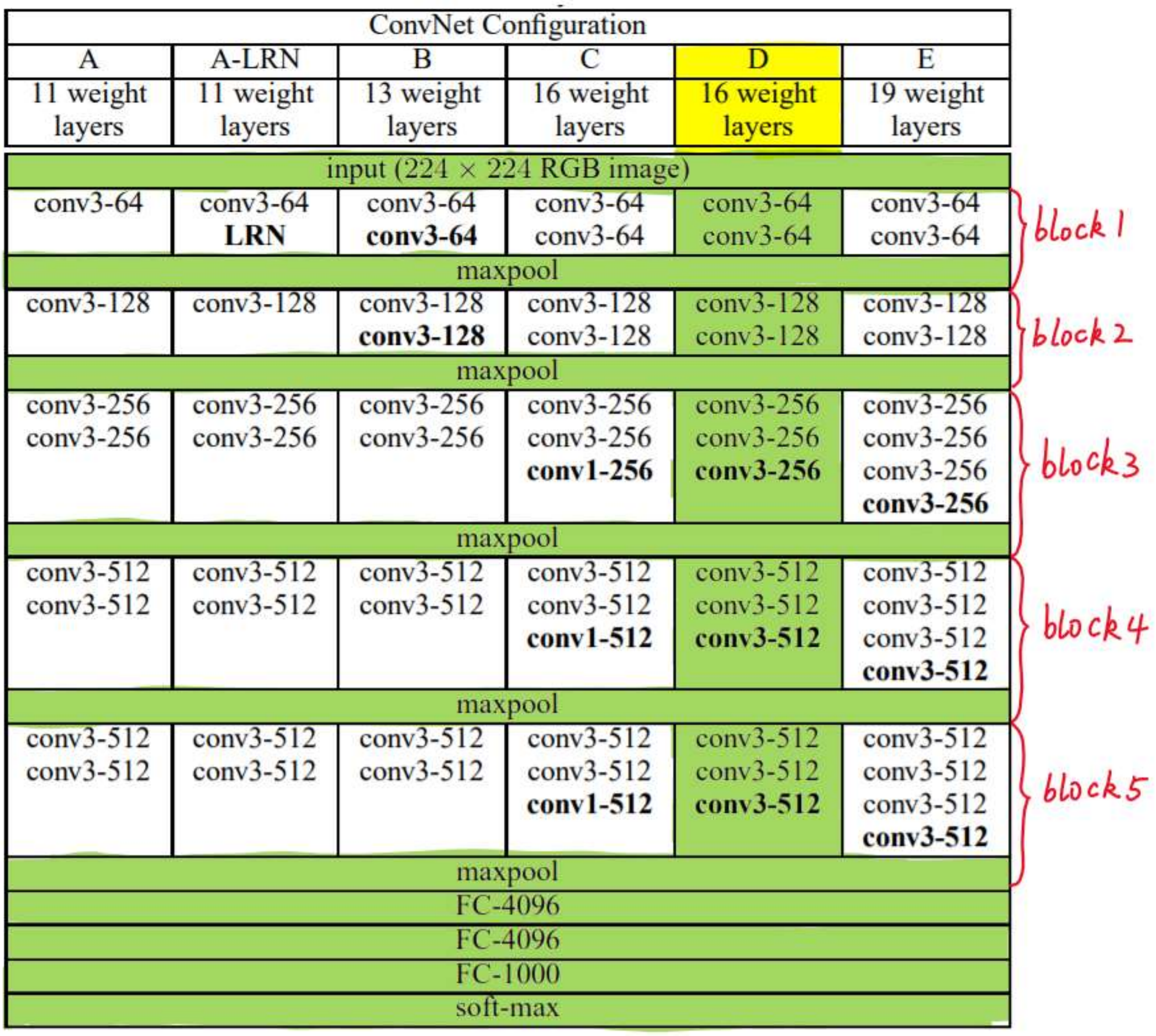
VGG

參考文獻：《Very Deep Convolutional Networks for Large Scale Image Recognrition》

架構:



ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

VGG16

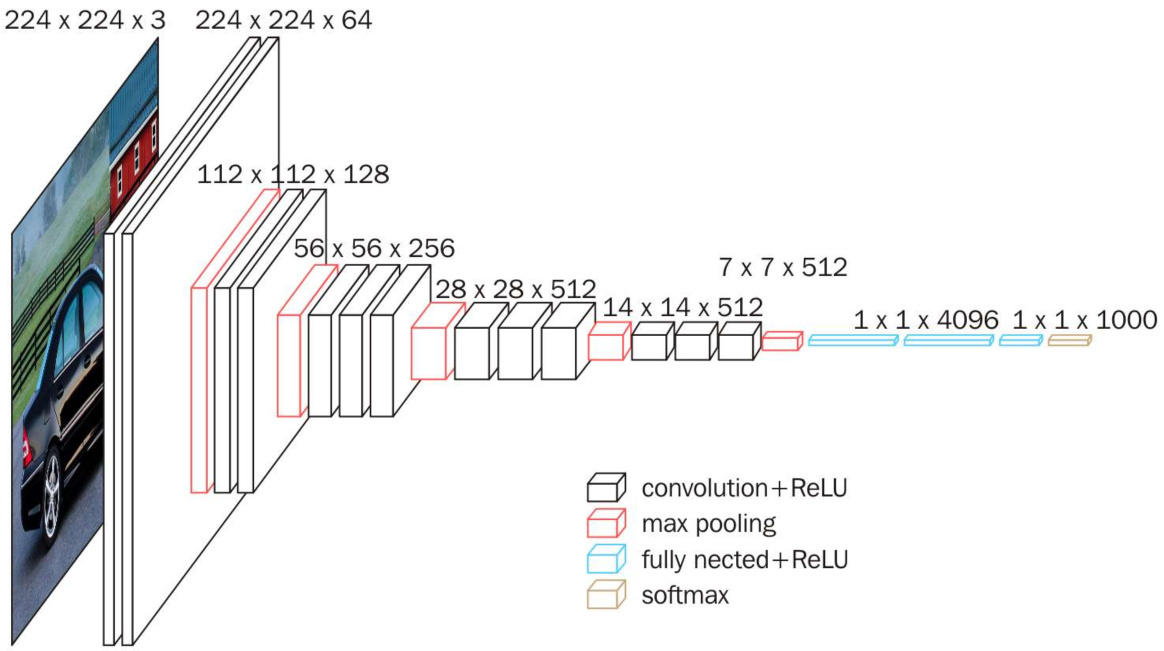
- 13個Convolutional Layer，以conv3-XXX表示
- 3個Fully Connected Layer，以FC-XXXX表示
- 5個Pool layer以maxpool表示

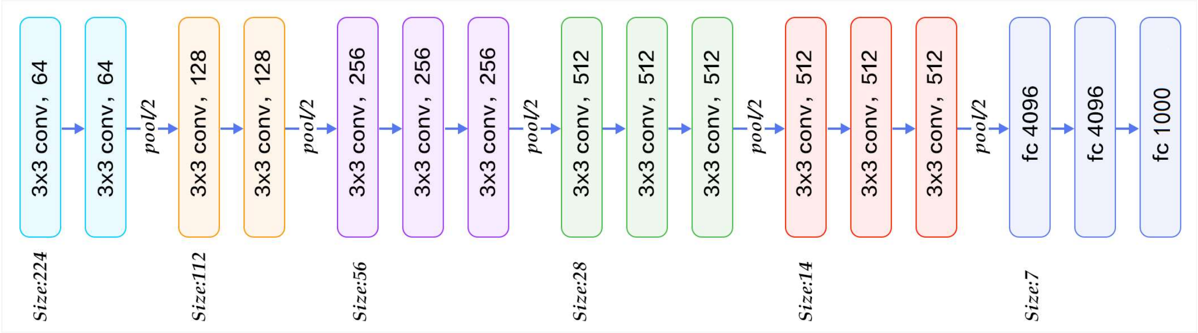
- 權重層:13(conv3)+3(FC) =16

VGG16特性

- Convolutional Layer均採用相同的卷積核參數：
  ** conv3-XXX: kernel size = 3 => 長寬皆為3 =>3x3; XXX => channel number, 其他參數:stride=1，padding=same

- Pool layer均採用相同的池化核參數:
  ** 參數:2×2, stride=2, => 每一個Pool layer的寬高是前一層1/2





- Channel數乘倍，64->128->256->512後，不再翻倍
- 寬高減半 224->112->56->28->14->7

In [ ]:

In [2]:

```
import matplotlib.pyplot as plt
from tensorflow.keras.applications.vgg16 import VGG16
from tensorflow.keras.preprocessing import image
from tensorflow.keras.applications.vgg16 import preprocess_input, decode_predictions
import numpy as np
```

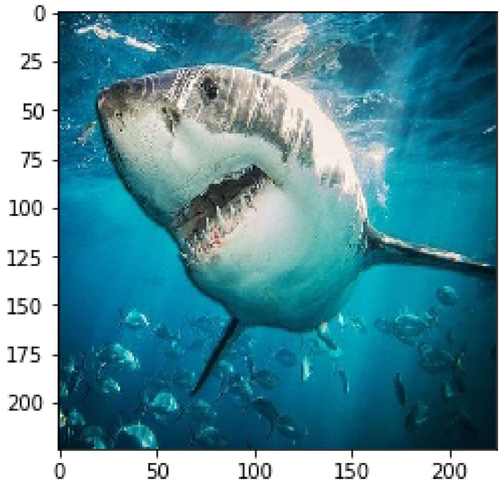
In [3]:

```
model = VGG16(weights='imagenet',include_top=True)
```

In [4]:

```
imgPath = 'data/img/shark.jpg'
img = image.load_img(imgPath, target_size=(224, 224))

plt.imshow(img)
plt.show()
```



In [5]:

```
x = image.img_to_array(img)
x = np.expand_dims(x, axis=0)#轉化為tensor size(1, 224, 224, 3)
x = preprocess_input(x)
```

In [6]:

```
# 進行預測，取得features，維度為 (1,1000)
features = model.predict(x)
```

In [7]:

```
# 取得前五個最可能類別跟機率
pred=decode_predictions(features, top=5)[0]
```

In [10]:

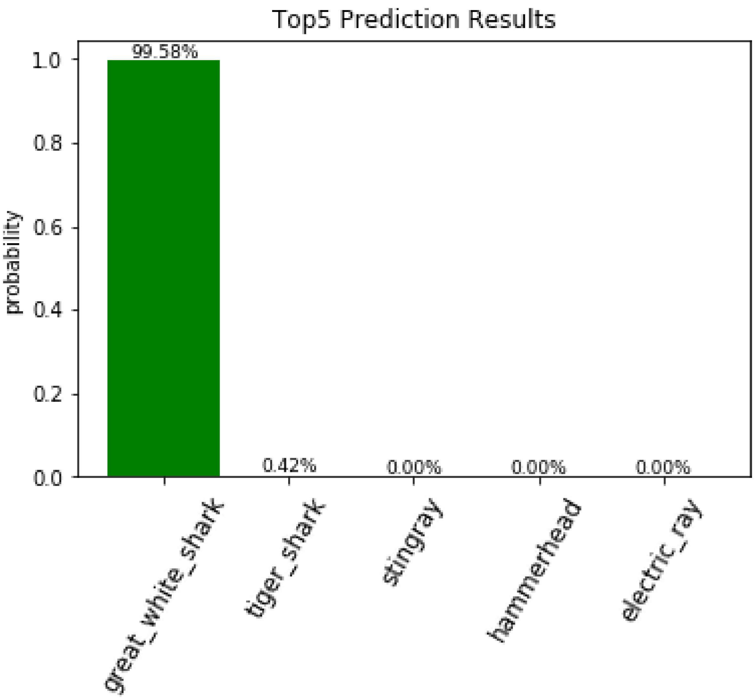
```
def percent(value):
    return '%.2f%%' % (value * 100)

#整理預測結果及數值
values = []
label = []
for elem in pred:
    values.append(elem[2])
    label.append(elem[1])


fig=plt.figure(u"Top5 Prediction Results")
ax = fig.add_subplot(111)
ax.bar(range(len(values)), values, tick_label=label, width=0.9, fc='g')
ax.set_ylabel(u'probability')
ax.set_title(u'Top5 Prediction Results')
for a,b in zip(range(len(values)), values):
    ax.text(a, b, percent(b), ha='center', va = 'bottom', fontsize=9)

plt.xticks(fontsize=12,rotation=60)

fig = plt.gcf()
plt.show()
```



In [9]:

```
model.summary()
```

```
Model: "vgg16"
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
input_1 (InputLayer)         [(None, 224, 224, 3)]     0         
_________________________________________________________________
block1_conv1 (Conv2D)        (None, 224, 224, 64)      1792      
_________________________________________________________________
block1_conv2 (Conv2D)        (None, 224, 224, 64)      36928     
_________________________________________________________________
block1_pool (MaxPooling2D)   (None, 112, 112, 64)      0         
_________________________________________________________________
block2_conv1 (Conv2D)        (None, 112, 112, 128)     73856     
_________________________________________________________________
block2_conv2 (Conv2D)        (None, 112, 112, 128)     147584    
_________________________________________________________________
block2_pool (MaxPooling2D)   (None, 56, 56, 128)       0         
_________________________________________________________________
block3_conv1 (Conv2D)        (None, 56, 56, 256)       295168    
_________________________________________________________________
block3_conv2 (Conv2D)        (None, 56, 56, 256)       590080    
_________________________________________________________________
block3_conv3 (Conv2D)        (None, 56, 56, 256)       590080    
_________________________________________________________________
block3_pool (MaxPooling2D)   (None, 28, 28, 256)       0         
_________________________________________________________________
block4_conv1 (Conv2D)        (None, 28, 28, 512)       1180160   
_________________________________________________________________
block4_conv2 (Conv2D)        (None, 28, 28, 512)       2359808   
_________________________________________________________________
block4_conv3 (Conv2D)        (None, 28, 28, 512)       2359808   
_________________________________________________________________
block4_pool (MaxPooling2D)   (None, 14, 14, 512)       0         
_________________________________________________________________
block5_conv1 (Conv2D)        (None, 14, 14, 512)       2359808   
_________________________________________________________________
block5_conv2 (Conv2D)        (None, 14, 14, 512)       2359808   
_________________________________________________________________
block5_conv3 (Conv2D)        (None, 14, 14, 512)       2359808   
_________________________________________________________________
block5_pool (MaxPooling2D)   (None, 7, 7, 512)         0         
```

```
_________________________________________________________________
flatten (Flatten)            (None, 25088)             0         
_________________________________________________________________
fc1 (Dense)                  (None, 4096)              102764544 
_________________________________________________________________
fc2 (Dense)                  (None, 4096)              16781312  
_________________________________________________________________
predictions (Dense)          (None, 1000)              4097000   
=================================================================
Total params: 138,357,544
Trainable params: 138,357,544
Non-trainable params: 0
_________________________________________________________________
```

```
In [ ]:
```

```
In [ ]:
```