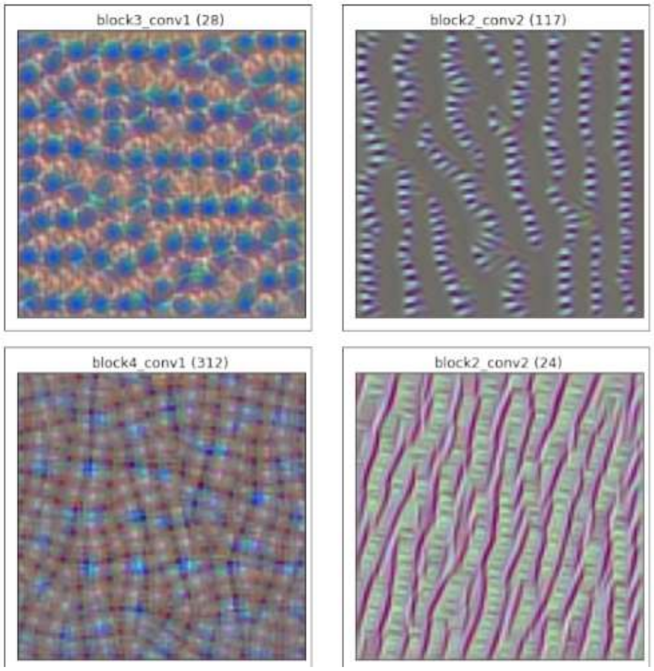
Visualizing Filters of a Convolutional Neural Network



Task 2: Downloading the Model

```
import tensorflow as tf
import random
import matplotlib.pyplot as plt

print('Tensorflow version',tf.__version__)
```

```
Tensorflow version 2.9.2
```

```
model = tf.keras.applications.vgg16.VGG16(
    include_top = False, weights = 'imagenet',
    input_shape = (96, 96, 3)
)
model.summary()
```

```
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/vgg16/vgg16_weights_tf_dim_ordering_tf_kernels_notop.
58889256/58889256 [==============================] - 4s 0us/step
Model: "vgg16"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_1 (InputLayer)        [(None, 96, 96, 3)]       0

 block1_conv1 (Conv2D)       (None, 96, 96, 64)        1792

 block1_conv2 (Conv2D)       (None, 96, 96, 64)        36928

 block1_pool (MaxPooling2D)  (None, 48, 48, 64)        0

 block2_conv1 (Conv2D)       (None, 48, 48, 128)       73856

 block2_conv2 (Conv2D)       (None, 48, 48, 128)       147584

 block2_pool (MaxPooling2D)  (None, 24, 24, 128)       0

 block3_conv1 (Conv2D)       (None, 24, 24, 256)       295168

 block3_conv2 (Conv2D)       (None, 24, 24, 256)       590080

 block3_conv3 (Conv2D)       (None, 24, 24, 256)       590080

 block3_pool (MaxPooling2D)  (None, 12, 12, 256)       0
 block4_conv1 (Conv2D)       (None, 12, 12, 512)       1180160   
                                                                 
 block4_conv2 (Conv2D)       (None, 12, 12, 512)       2359808   
                                                                 
 block4_conv3 (Conv2D)       (None, 12, 12, 512)       2359808   
                                                                 
 block4_pool (MaxPooling2D)  (None, 6, 6, 512)         0         
                                                                 
 block5_conv1 (Conv2D)       (None, 6, 6, 512)         2359808   
                                                                 
 block5_conv2 (Conv2D)       (None, 6, 6, 512)         2359808   
                                                                 
 block5_conv3 (Conv2D)       (None, 6, 6, 512)         2359808   
                                                                 
 block5_pool (MaxPooling2D)  (None, 3, 3, 512)         0         
                                                                 
=================================================================
Total params: 14,714,688
Trainable params: 14,714,688
Non-trainable params: 0
_________________________________________________________________
```

Task 3: Get Layer Output

```
def get_submodel(layer_name):
  return tf.keras.models.Model(
      model.input,model.get_layer(layer_name).output
      )
get_submodel('block1_conv2').summary()
```

```
Model: "model"
_________________________________________________________________
 Layer (type)                Output Shape              Param #   
=================================================================
 input_1 (InputLayer)        [(None, 96, 96, 3)]       0         
                                                                 
 block1_conv1 (Conv2D)       (None, 96, 96, 64)        1792      
                                                                 
 block1_conv2 (Conv2D)       (None, 96, 96, 64)        36928     
                                                                 
=================================================================
Total params: 38,720
Trainable params: 38,720
Non-trainable params: 0
_________________________________________________________________
```

Task 4: Image Visualization

```
def create_image():
   return tf.random.uniform((96, 96, 3), minval=0.5, maxval=0.5)

def plot_image(image, title='random'):
  image = image - tf.math.reduce_min(image)
  image = image / tf.math.reduce_max(image)
  plt.imshow(image)
  plt.xticks([])
  plt.yticks([])
  plt.title(title)
  plt.show()


image = create_image()
plot_image(image)
```

random

[redacted]

Task 5: Training Loop

[redacted]

```
def visualize_filter(layer_name, f_index = None, iters = 50):
  submodel = get_submodel(layer_name)
  num_filters = submodel.output.shape[-1]

  if f_index is None:
    f_index = random.randint(0, num_filters - 1)
  assert num_filters > f_index, 'f_index is out of bounds'

  image = create_image()
  verbose_step = int(iters/10)

  for i in range(0, iters):
    with tf.GradientTape() as tape:
      tape.watch(image)
      out = submodel(tf.expand_dims(image, axis=0))[:,:,:,f_index]
      loss = tf.math.reduce_mean(out)
    grads = tape.gradient(loss, image)
    grads = tf.math.l2_normalize(grads)
    image+= grads *10

    if (i+1)%verbose_step ==0:
      print(f'Iteration:{i+1},Loss:{loss.numpy():.4f}')
  plot_image(image, f'{layer_name},{f_index}')
```

Task 6: Final Results

```
print([layer.name for layer in model.layers if 'conv' in layer.name])
```

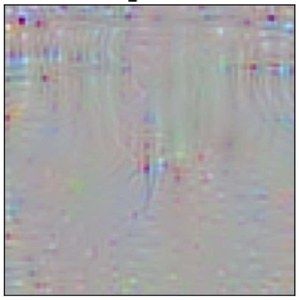
```
['block1_conv1', 'block1_conv2', 'block2_conv1', 'block2_conv2', 'block3_conv1', 'block3_conv2', 'block3_conv3', 'block4_conv1', 'block4
```

```
layer_name = 'block5_conv3' #@param [
visualize_filter(layer_name, iters=10
```

layer_name: block5_conv3

```
Iteration:10,Loss:5.1704
Iteration:20,Loss:11.9184
Iteration:30,Loss:18.7169
Iteration:40,Loss:26.0385
Iteration:50,Loss:33.7981
Iteration:60,Loss:42.6144
Iteration:70,Loss:51.5873
Iteration:80,Loss:61.1006
Iteration:90,Loss:71.2144
Iteration:100,Loss:82.2850
```

block5_conv3,148



✓ 3s completed at 6:19 PM ● ×