Coding Tutorial

July 31, 2021

```
In [37]: import tensorflow as tf
         print(tf.__version__)
```

```
2.0.0
```

1 Saving and loading models

## Coding tutorials #### Section **??** #### Section **??** #### Section **??** #### Section **??** #### Section **??**

## Saving and loading model weights

Load and inspect CIFAR-10 dataset The CIFAR-10 dataset consists of, in total, 60000 color images, each with one of 10 labels: airplane, automobile, bird, cat, deer, dog, frog, horse, ship, truck. For an introduction and a download, see this link.

```
In [3]: # Import the CIFAR-10 dataset and rescale the pixel values

        (x_train, y_train), (x_test, y_test) = tf.keras.datasets.cifar10.load_data()
        x_train = x_train / 255.0
        x_test = x_test / 255.0

        # Use smaller subset -- speeds things up
        x_train = x_train[:10000]
        y_train = y_train[:10000]
        x_test = x_test[:1000]
        y_test = y_test[:1000]
```

```
Downloading data from https://www.cs.toronto.edu/~kriz/cifar-10-python.tar.gz
170500096/170498071 [==============================] - 4s 0us/step
```

```
In [4]: # Plot the first 10 CIFAR-10 images

        import matplotlib.pyplot as plt
        fig, ax = plt.subplots(1, 10, figsize=(10, 1))
        for i in range(10):
            ax[i].set_axis_off()
            ax[i].imshow(x_train[i])
```

Introduce two useful functions

```
In [4]: # Introduce function to test model accuracy

        def get_test_accuracy(model, x_test, y_test):
            test_loss, test_acc = model.evaluate(x=x_test, y=y_test, verbose=0)
            print('accuracy: {acc:0.3f}'.format(acc=test_acc))
```

```
In [5]: # Introduce function that creates a new instance of a simple CNN

        from tensorflow.keras.models import Sequential
        from tensorflow.keras.layers import Dense, Flatten, Conv2D, MaxPooling2D

        def get_new_model():
            model = Sequential([
                Conv2D(filters=16, input_shape=(32, 32, 3), kernel_size=(3, 3),
                       activation='relu', name='conv_1'),
                Conv2D(filters=8, kernel_size=(3, 3), activation='relu', name='conv_2'),
                MaxPooling2D(pool_size=(4, 4), name='pool_1'),
                Flatten(name='flatten'),
                Dense(units=32, activation='relu', name='dense_1'),
                Dense(units=10, activation='softmax', name='dense_2')
            ])
            model.compile(optimizer='adam',
                          loss='sparse_categorical_crossentropy',
                          metrics=['accuracy'])
            return model
```

Create simple convolutional neural network classifier

```
In [9]: # Create an instance of the model and show model summary

        model = get_new_model()
        model.summary()
```

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv_1 (Conv2D)              (None, 30, 30, 16)        448
_________________________________________________________________
conv_2 (Conv2D)              (None, 28, 28, 8)         1160
_________________________________________________________________
pool_1 (MaxPooling2D)        (None, 7, 7, 8)           0
_________________________________________________________________
flatten (Flatten)            (None, 392)               0
_________________________________________________________________
dense_1 (Dense)              (None, 32)                12576
_________________________________________________________________
dense_2 (Dense)              (None, 10)                330
=================================================================
Total params: 14,514
Trainable params: 14,514
Non-trainable params: 0
_________________________________________________________________
```

```
In [10]: # Test accuracy of the untrained model, around 10% (random)
         get_test_accuracy(model, x_test, y_test)
```

```
accuracy: 0.100
```

Train model with checkpoints

```
In [12]: from tensorflow.keras.callbacks import ModelCheckpoint
```

```
In [35]: # Create Tensorflow checkpoint object

         checkpoint_path  = 'model_checkpoints/checkpoint'
         checkpoint = ModelCheckpoint(filepath=checkpoint_path, frequency='epoch', save_weights
```

```
In [36]: # Fit model, with simple checkpoint which saves (and overwrites) model weights every

         model.fit(x=x_train,
                   y=y_train,
                   epochs=3,
                   callbacks=[checkpoint])
```

```
Train on 10000 samples
Epoch 1/3
 9984/10000 [============================>.] - ETA: 0s - loss: 1.6219 - accuracy: 0.4257
Epoch 00001: saving model to model_checkpoints/checkpoint
10000/10000 [==============================] - 46s 5ms/sample - loss: 1.6216 - accuracy: 0.425
```

```
Epoch 2/3
 9984/10000 [============================>.] - ETA: 0s - loss: 1.4801 - accuracy: 0.4712
Epoch 00002: saving model to model_checkpoints/checkpoint
10000/10000 [==============================] - 46s 5ms/sample - loss: 1.4798 - accuracy: 0.4713
Epoch 3/3
 9984/10000 [============================>.] - ETA: 0s - loss: 1.4169 - accuracy: 0.4973 ETA: 
Epoch 00003: saving model to model_checkpoints/checkpoint
10000/10000 [==============================] - 45s 5ms/sample - loss: 1.4168 - accuracy: 0.4975
```

```
Out[36]: <tensorflow.python.keras.callbacks.History at 0x7f330d698b00>
```

```
In [37]: # Have a look at what the checkpoint creates

         ! ls -lh model_checkpoints/
```

```
total 184K
-rw-r--r-- 1 jovyan users   77 Jul 30 19:01 checkpoint
-rw-r--r-- 1 jovyan users 174K Jul 30 19:01 checkpoint.data-00000-of-00001
-rw-r--r-- 1 jovyan users 2.0K Jul 30 19:01 checkpoint.index
```

```
In [38]: # Evaluate the performance of the trained model
         get_test_accuracy(model, x_test, y_test)
```

```
accuracy: 0.496
```

Create new model, load weights

```
In [39]: # Create a new instance of the (initialised) model, accuracy around 10% again
         model = get_new_model()
         get_test_accuracy(model, x_test, y_test)
```

```
accuracy: 0.102
```

```
In [40]: # Load weights -- accuracy is the same as the trained model
         model.load_weights(checkpoint_path)#'model_checkpoints/checkpoint/variables/variables
         get_test_accuracy(model, x_test, y_test)
```

```
accuracy: 0.496
```

Clear directory

```
In [33]: ! rm -r model_checkpoints
```

```
rm: cannot remove 'model_checkpoints': No such file or directory
```

## Model saving criteria

Create more customised checkpoint

```
In [42]: from tensorflow.keras.callbacks import ModelCheckpoint
```

```
In [43]: # Create Tensorflow checkpoint object with epoch and batch details
         checkpoint_5000_patch = \
             'model_checkpoints_5000/checkpoint_{epoch:02d}_{batch:04d}'

         checkpoint_5000 = ModelCheckpoint(filepath=checkpoint_5000_patch,
                                           save_weights_only=True,
                                           save_freq=5000,
                                           verbose=1
                                           )
```

```
In [46]: # Create and fit model with checkpoint

         model = get_new_model()
         model.fit(x=x_train,
                   y=y_train,
                   epochs=3,
                   validation_data=(x_test, y_test),
                   batch_size=10,
                   callbacks=[checkpoint_5000])
```

```
Train on 10000 samples, validate on 1000 samples
Epoch 1/3
 4990/10000 [=============>...] - ETA: 30s - loss: 2.0489 - accuracy: 0.2313- ETA: 42s - l - ET
Epoch 00001: saving model to model_checkpoints_5000/checkpoint_01_0499
 9980/10000 [============================>.] - ETA: 0s - loss: 1.8774 - accuracy: 0.3037
Epoch 00001: saving model to model_checkpoints_5000/checkpoint_01_0999
10000/10000 [==============================] - 62s 6ms/sample - loss: 1.8767 - accuracy: 0.3039
Epoch 2/3
 4990/10000 [=============>...] - ETA: 29s - loss: 1.5617 - accuracy: 0.4339
Epoch 00002: saving model to model_checkpoints_5000/checkpoint_02_0499
 9980/10000 [============================>.] - ETA: 0s - loss: 1.5402 - accuracy: 0.4440
Epoch 00002: saving model to model_checkpoints_5000/checkpoint_02_0999
10000/10000 [==============================] - 61s 6ms/sample - loss: 1.5403 - accuracy: 0.4437
Epoch 3/3
 4990/10000 [=============>...] - ETA: 28s - loss: 1.4145 - accuracy: 0.4810
Epoch 00003: saving model to model_checkpoints_5000/checkpoint_03_0499
 9990/10000 [============================>.] - ETA: 0s - loss: 1.4229 - accuracy: 0.4839
Epoch 00003: saving model to model_checkpoints_5000/checkpoint_03_0999
10000/10000 [==============================] - 59s 6ms/sample - loss: 1.4230 - accuracy: 0.4839
```

```
Out[46]: <tensorflow.python.keras.callbacks.History at 0x7f330c81d978>
```

```
In [47]: # Have a look at what the checkpoint creates

         ! ls -lh model_checkpoints_5000
```

```
total 1.1M
-rw-r--r-- 1 jovyan users   93 Jul 30 19:42 checkpoint
-rw-r--r-- 1 jovyan users 174K Jul 30 19:40 checkpoint_01_0499.data-00000-of-00001
-rw-r--r-- 1 jovyan users 2.0K Jul 30 19:40 checkpoint_01_0499.index
-rw-r--r-- 1 jovyan users 174K Jul 30 19:40 checkpoint_01_0999.data-00000-of-00001
-rw-r--r-- 1 jovyan users 2.0K Jul 30 19:40 checkpoint_01_0999.index
-rw-r--r-- 1 jovyan users 174K Jul 30 19:41 checkpoint_02_0499.data-00000-of-00001
-rw-r--r-- 1 jovyan users 2.0K Jul 30 19:41 checkpoint_02_0499.index
-rw-r--r-- 1 jovyan users 174K Jul 30 19:41 checkpoint_02_0999.data-00000-of-00001
-rw-r--r-- 1 jovyan users 2.0K Jul 30 19:41 checkpoint_02_0999.index
-rw-r--r-- 1 jovyan users 174K Jul 30 19:42 checkpoint_03_0499.data-00000-of-00001
-rw-r--r-- 1 jovyan users 2.0K Jul 30 19:42 checkpoint_03_0499.index
-rw-r--r-- 1 jovyan users 174K Jul 30 19:42 checkpoint_03_0999.data-00000-of-00001
-rw-r--r-- 1 jovyan users 2.0K Jul 30 19:42 checkpoint_03_0999.index
```

Work with model saving criteria

```
In [48]: # Use tiny training and test set -- will overfit!

         x_train = x_train[:100]
         y_train = y_train[:100]
         x_test = x_test[:100]
         y_test = y_test[:100]
```

```
In [57]: # Create a new instance of untrained model

         model = get_new_model()
```

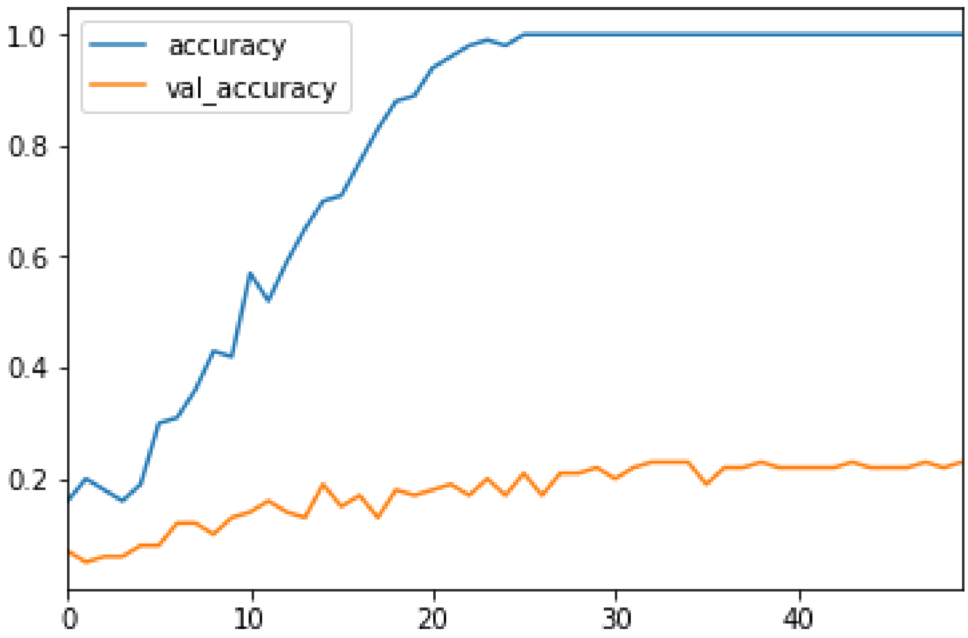
```
In [58]: # Create Tensorflow checkpoint object which monitors the validation accuracy
         checkpoint_best_path = 'model_checkpoint_best/checkpoint'
         checkpoint_best = ModelCheckpoint(filepath=checkpoint_best_path,
                                           save_weights_only=True,
                                           save_freq='epoch',
                                           monitor='val_accuracy',
                                           save_best_only=True,
                                           verbose=1)
```

```
In [59]: # Fit the model and save only the weights with the highest validation accuracy
         history = model.fit(x=x_train,
                             y=y_train,
                             epochs=50,
                             validation_data=(x_test, y_test),
                             batch_size=10,
                             callbacks=[checkpoint_best],
                             verbose=0)
```

```
WARNING:tensorflow:Unresolved object in checkpoint: (root).optimizer.iter
WARNING:tensorflow:Unresolved object in checkpoint: (root).optimizer.beta_1
```

```
WARNING:tensorflow:Unresolved object in checkpoint: (root).optimizer.beta_2
WARNING:tensorflow:Unresolved object in checkpoint: (root).optimizer.decay
WARNING:tensorflow:Unresolved object in checkpoint: (root).optimizer.learning_rate
WARNING:tensorflow:A checkpoint was restored (e.g. tf.train.Checkpoint.restore or tf.keras.Mode

Epoch 00001: val_accuracy improved from -inf to 0.07000, saving model to model_checkpoint_best/

Epoch 00002: val_accuracy did not improve from 0.07000

Epoch 00003: val_accuracy did not improve from 0.07000

Epoch 00004: val_accuracy did not improve from 0.07000

Epoch 00005: val_accuracy improved from 0.07000 to 0.08000, saving model to model_checkpoint_be

Epoch 00006: val_accuracy did not improve from 0.08000

Epoch 00007: val_accuracy improved from 0.08000 to 0.12000, saving model to model_checkpoint_be

Epoch 00008: val_accuracy did not improve from 0.12000

Epoch 00009: val_accuracy did not improve from 0.12000

Epoch 00010: val_accuracy improved from 0.12000 to 0.13000, saving model to model_checkpoint_be

Epoch 00011: val_accuracy improved from 0.13000 to 0.14000, saving model to model_checkpoint_be

Epoch 00012: val_accuracy improved from 0.14000 to 0.16000, saving model to model_checkpoint_be

Epoch 00013: val_accuracy did not improve from 0.16000

Epoch 00014: val_accuracy did not improve from 0.16000

Epoch 00015: val_accuracy improved from 0.16000 to 0.19000, saving model to model_checkpoint_be

Epoch 00016: val_accuracy did not improve from 0.19000

Epoch 00017: val_accuracy did not improve from 0.19000

Epoch 00018: val_accuracy did not improve from 0.19000

Epoch 00019: val_accuracy did not improve from 0.19000

Epoch 00020: val_accuracy did not improve from 0.19000

Epoch 00021: val_accuracy did not improve from 0.19000

Epoch 00022: val_accuracy did not improve from 0.19000
Epoch 00023: val_accuracy did not improve from 0.19000

Epoch 00024: val_accuracy improved from 0.19000 to 0.20000, saving model to model_checkpoint_be

Epoch 00025: val_accuracy did not improve from 0.20000

Epoch 00026: val_accuracy improved from 0.20000 to 0.21000, saving model to model_checkpoint_be

Epoch 00027: val_accuracy did not improve from 0.21000

Epoch 00028: val_accuracy did not improve from 0.21000

Epoch 00029: val_accuracy did not improve from 0.21000

Epoch 00030: val_accuracy improved from 0.21000 to 0.22000, saving model to model_checkpoint_be

Epoch 00031: val_accuracy did not improve from 0.22000

Epoch 00032: val_accuracy did not improve from 0.22000

Epoch 00033: val_accuracy improved from 0.22000 to 0.23000, saving model to model_checkpoint_be

Epoch 00034: val_accuracy did not improve from 0.23000

Epoch 00035: val_accuracy did not improve from 0.23000

Epoch 00036: val_accuracy did not improve from 0.23000

Epoch 00037: val_accuracy did not improve from 0.23000

Epoch 00038: val_accuracy did not improve from 0.23000

Epoch 00039: val_accuracy did not improve from 0.23000

Epoch 00040: val_accuracy did not improve from 0.23000

Epoch 00041: val_accuracy did not improve from 0.23000

Epoch 00042: val_accuracy did not improve from 0.23000

Epoch 00043: val_accuracy did not improve from 0.23000

Epoch 00044: val_accuracy did not improve from 0.23000

Epoch 00045: val_accuracy did not improve from 0.23000

Epoch 00046: val_accuracy did not improve from 0.23000
Epoch 00047: val_accuracy did not improve from 0.23000

Epoch 00048: val_accuracy did not improve from 0.23000

Epoch 00049: val_accuracy did not improve from 0.23000

Epoch 00050: val_accuracy did not improve from 0.23000
```

```
In [60]: # Plot training and testing curves

         import pandas as pd

         df = pd.DataFrame(history.history)
         df.plot(y=['accuracy', 'val_accuracy'])

Out[60]: <matplotlib.axes._subplots.AxesSubplot at 0x7f330d225400>
```



```
In [61]: # Inspect the checkpoint directory

         ! ls -lh model_checkpoint_best/

total 184K
-rw-r--r-- 1 jovyan users   77 Jul 30 19:50 checkpoint
-rw-r--r-- 1 jovyan users 174K Jul 30 19:50 checkpoint.data-00000-of-00001
```

```
-rw-r--r-- 1 jovyan users 2.0K Jul 30 19:50 checkpoint.index
```

```
In [65]: # Create a new model with the saved weights
         new_model = get_new_model()
         new_model.load_weights(checkpoint_best_path)
         get_test_accuracy(new_model, x_test, y_test)
```

```
accuracy: 0.230
```

Clear directory

```
In [67]: ! rm -r model_checkpoints_5000 model_checkpoint_best
```

```
rm: cannot remove 'model_checkpoints_5000': No such file or directory
```

## Saving the entire model

Create checkpoint that saves whole model, not just weights

```
In [1]: from tensorflow.keras.callbacks import ModelCheckpoint
```

```
In [6]: # Create Tensorflow checkpoint object

        checkpoint_path = 'model_checkpoints'
        checkpoints = ModelCheckpoint(filepath=checkpoint_path,
                                      save_weights_only=False,
                                      frequency='epoch',
                                      verbose=1)
```

```
In [9]: # Create and fit model with checkpoint

        model = get_new_model()
        model.fit(x=x_train,
                  y=y_train,
                  epochs=3,
                  callbacks=[checkpoints])
```

```
Train on 10000 samples
Epoch 1/3
 9984/10000 [============================>.] - ETA: 0s - loss: 1.9769 - accuracy: 0.2631
Epoch 00001: saving model to model_checkpoints
WARNING:tensorflow:From /opt/conda/lib/python3.7/site-packages/tensorflow_core/python/ops/reso
Instructions for updating:
If using Keras pass *_constraint arguments to layers.
```

```
INFO:tensorflow:Assets written to: model_checkpoints/assets
10000/10000 [==============================] - 49s 5ms/sample - loss: 1.9765 - accuracy: 0.2634
Epoch 2/3
 9984/10000 [============================>.] - ETA: 0s - loss: 1.6491 - accuracy: 0.3943
Epoch 00002: saving model to model_checkpoints
INFO:tensorflow:Assets written to: model_checkpoints/assets
10000/10000 [==============================] - 45s 5ms/sample - loss: 1.6497 - accuracy: 0.3939
Epoch 3/3
 9984/10000 [============================>.] - ETA: 0s - loss: 1.5379 - accuracy: 0.4366
Epoch 00003: saving model to model_checkpoints
INFO:tensorflow:Assets written to: model_checkpoints/assets
10000/10000 [==============================] - 45s 4ms/sample - loss: 1.5385 - accuracy: 0.4366
```

```
Out[9]: <tensorflow.python.keras.callbacks.History at 0x7f94ac0ad940>
```

Inspect what the checkpoint has created

```
In [10]: # Have a look at what the checkpoint creates

         !ls -lh model_checkpoints/
```

```
total 312K
drwxr-xr-x 2 jovyan users 6.0K Jul 30 22:37 assets
-rw-r--r-- 1 jovyan users   77 Jul 30 19:01 checkpoint
-rw-r--r-- 1 jovyan users 174K Jul 30 19:01 checkpoint.data-00000-of-00001
-rw-r--r-- 1 jovyan users 2.0K Jul 30 19:01 checkpoint.index
-rw-r--r-- 1 jovyan users 119K Jul 30 22:38 saved_model.pb
drwxr-xr-x 2 jovyan users 6.0K Jul 30 22:38 variables
```

```
In [11]: # Enter variables directory

         !ls -lh model_checkpoints/variables/
```

```
total 184K
-rw-r--r-- 1 jovyan users 177K Jul 30 22:38 variables.data-00000-of-00001
-rw-r--r-- 1 jovyan users 2.1K Jul 30 22:38 variables.index
```

```
In [12]: # Get the model's test accuracy

         get_test_accuracy(model, x_test, y_test)
```

```
accuracy: 0.464
```

Create new model from scratch

```
In [14]: # Delete model

         model
```

```
        ---------------------------------------------------------------------------

        NameError                                 Traceback (most recent call last)

        <ipython-input-14-4ad52f4df476> in <module>
          1 # Delete model
          2
    ----> 3 model


        NameError: name 'model' is not defined
```

```
In [15]: from tensorflow.keras.models import load_model
```

```
In [18]: # Reload model from scratch

         model = load_model(checkpoint_path)
         get_test_accuracy(model, x_test, y_test)
```

```
accuracy: 0.464
```

Use the .h5 format to save model

```
In [19]: # Save the model in .h5 format

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

```
In [20]: # Inspect .h5 file

         !ls -lh my_model.h5
```

```
-rw-r--r-- 1 jovyan users 77K Jul 30 22:46 my_model.h5
```

```
In [22]: # Delete model

         model
```

```
        ---------------------------------------------------------------------------
```

```
    NameError                                 Traceback (most recent call last)

    <ipython-input-22-4ad52f4df476> in <module>
      1 # Delete model
      2
----> 3 model


    NameError: name 'model' is not defined
```

```
In [24]: # Reload model from scratch

         model = load_model('my_model.h5')
         get_test_accuracy(model, x_test, y_test)
```

```
accuracy: 0.464
```

Clear directory

```
In [25]: ! rm -r model_checkpoints
         ! rm my_model.h5
```

## Loading pre-trained Keras models

Import and build Keras ResNet50 model Today we'll be using the ResNet50 model designed by a team at Microsoft Research, available through Keras applications. Please see the description on the Keras applications page for details. If you continue using it, please cite it properly! The paper it comes from is:

Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun. "Deep Residual Learning for Image Recognition", 2015.

This model takes a long time to download on the Coursera platform, so it is pre-downloaded in your workspace and saved in Keras HDF5 format. If you want to import it on your personal machine, use the following code:

```
from tensorflow.keras.applications import ResNet50
model = ResNet50(weights='imagenet')
```

In this coding tutorial, you will instead load the model directly from disk.

```
In [27]: from tensorflow.keras.models import load_model
```

```
In [28]: # Build Keras ResNet50 model

         model = load_model('models/Keras_ResNet50.h5')
         model.summary()
```

```
WARNING:tensorflow:No training configuration found in save file: the model was *not* compiled.
Model: "resnet50"
_______________________________________________________________________________________________
Layer (type)                    Output Shape         Param #     Connected to
===============================================================================================
input_1 (InputLayer)            [(None, 224, 224, 3) 0
_______________________________________________________________________________________________
conv1_pad (ZeroPadding2D)       (None, 230, 230, 3)  0           input_1[0][0]
_______________________________________________________________________________________________
conv1_conv (Conv2D)             (None, 112, 112, 64) 9472        conv1_pad[0][0]
_______________________________________________________________________________________________
conv1_bn (BatchNormalization)   (None, 112, 112, 64) 256         conv1_conv[0][0]
_______________________________________________________________________________________________
conv1_relu (Activation)         (None, 112, 112, 64) 0           conv1_bn[0][0]
_______________________________________________________________________________________________
pool1_pad (ZeroPadding2D)       (None, 114, 114, 64) 0           conv1_relu[0][0]
_______________________________________________________________________________________________
pool1_pool (MaxPooling2D)       (None, 56, 56, 64)   0           pool1_pad[0][0]
_______________________________________________________________________________________________
conv2_block1_1_conv (Conv2D)    (None, 56, 56, 64)   4160        pool1_pool[0][0]
_______________________________________________________________________________________________
conv2_block1_1_bn (BatchNormali (None, 56, 56, 64)   256         conv2_block1_1_conv[0][0]
_______________________________________________________________________________________________
conv2_block1_1_relu (Activation (None, 56, 56, 64)   0           conv2_block1_1_bn[0][0]
_______________________________________________________________________________________________
conv2_block1_2_conv (Conv2D)    (None, 56, 56, 64)   36928       conv2_block1_1_relu[0][0]
_______________________________________________________________________________________________
conv2_block1_2_bn (BatchNormali (None, 56, 56, 64)   256         conv2_block1_2_conv[0][0]
_______________________________________________________________________________________________
conv2_block1_2_relu (Activation (None, 56, 56, 64)   0           conv2_block1_2_bn[0][0]
_______________________________________________________________________________________________
conv2_block1_0_conv (Conv2D)    (None, 56, 56, 256)  16640       pool1_pool[0][0]
_______________________________________________________________________________________________
conv2_block1_3_conv (Conv2D)    (None, 56, 56, 256)  16640       conv2_block1_2_relu[0][0]
_______________________________________________________________________________________________
conv2_block1_0_bn (BatchNormali (None, 56, 56, 256)  1024        conv2_block1_0_conv[0][0]
_______________________________________________________________________________________________
conv2_block1_3_bn (BatchNormali (None, 56, 56, 256)  1024        conv2_block1_3_conv[0][0]
_______________________________________________________________________________________________
conv2_block1_add (Add)          (None, 56, 56, 256)  0           conv2_block1_0_bn[0][0]
                                                                 conv2_block1_3_bn[0][0]
_______________________________________________________________________________________________
conv2_block1_out (Activation)   (None, 56, 56, 256)  0           conv2_block1_add[0][0]
_______________________________________________________________________________________________
conv2_block2_1_conv (Conv2D)    (None, 56, 56, 64)   16448       conv2_block1_out[0][0]
_______________________________________________________________________________________________
conv2_block2_1_bn (BatchNormali (None, 56, 56, 64)   256         conv2_block2_1_conv[0][0]
_______________________________________________________________________________________________
```

```
conv2_block2_1_relu (Activation (None, 56, 56, 64)   0           conv2_block2_1_bn[0][0]
-----------------------------------------------------------------------------------------------
conv2_block2_2_conv (Conv2D)    (None, 56, 56, 64)   36928       conv2_block2_1_relu[0][0]
-----------------------------------------------------------------------------------------------
conv2_block2_2_bn (BatchNormali (None, 56, 56, 64)   256         conv2_block2_2_conv[0][0]
-----------------------------------------------------------------------------------------------
conv2_block2_2_relu (Activation (None, 56, 56, 64)   0           conv2_block2_2_bn[0][0]
-----------------------------------------------------------------------------------------------
conv2_block2_3_conv (Conv2D)    (None, 56, 56, 256)  16640       conv2_block2_2_relu[0][0]
-----------------------------------------------------------------------------------------------
conv2_block2_3_bn (BatchNormali (None, 56, 56, 256)  1024        conv2_block2_3_conv[0][0]
-----------------------------------------------------------------------------------------------
conv2_block2_add (Add)          (None, 56, 56, 256)  0           conv2_block1_out[0][0]
                                                                 conv2_block2_3_bn[0][0]
-----------------------------------------------------------------------------------------------
conv2_block2_out (Activation)   (None, 56, 56, 256)  0           conv2_block2_add[0][0]
-----------------------------------------------------------------------------------------------
conv2_block3_1_conv (Conv2D)    (None, 56, 56, 64)   16448       conv2_block2_out[0][0]
-----------------------------------------------------------------------------------------------
conv2_block3_1_bn (BatchNormali (None, 56, 56, 64)   256         conv2_block3_1_conv[0][0]
-----------------------------------------------------------------------------------------------
conv2_block3_1_relu (Activation (None, 56, 56, 64)   0           conv2_block3_1_bn[0][0]
-----------------------------------------------------------------------------------------------
conv2_block3_2_conv (Conv2D)    (None, 56, 56, 64)   36928       conv2_block3_1_relu[0][0]
-----------------------------------------------------------------------------------------------
conv2_block3_2_bn (BatchNormali (None, 56, 56, 64)   256         conv2_block3_2_conv[0][0]
-----------------------------------------------------------------------------------------------
conv2_block3_2_relu (Activation (None, 56, 56, 64)   0           conv2_block3_2_bn[0][0]
-----------------------------------------------------------------------------------------------
conv2_block3_3_conv (Conv2D)    (None, 56, 56, 256)  16640       conv2_block3_2_relu[0][0]
-----------------------------------------------------------------------------------------------
conv2_block3_3_bn (BatchNormali (None, 56, 56, 256)  1024        conv2_block3_3_conv[0][0]
-----------------------------------------------------------------------------------------------
conv2_block3_add (Add)          (None, 56, 56, 256)  0           conv2_block2_out[0][0]
                                                                 conv2_block3_3_bn[0][0]
-----------------------------------------------------------------------------------------------
conv2_block3_out (Activation)   (None, 56, 56, 256)  0           conv2_block3_add[0][0]
-----------------------------------------------------------------------------------------------
conv3_block1_1_conv (Conv2D)    (None, 28, 28, 128)  32896       conv2_block3_out[0][0]
-----------------------------------------------------------------------------------------------
conv3_block1_1_bn (BatchNormali (None, 28, 28, 128)  512         conv3_block1_1_conv[0][0]
-----------------------------------------------------------------------------------------------
conv3_block1_1_relu (Activation (None, 28, 28, 128)  0           conv3_block1_1_bn[0][0]
-----------------------------------------------------------------------------------------------
conv3_block1_2_conv (Conv2D)    (None, 28, 28, 128)  147584      conv3_block1_1_relu[0][0]
-----------------------------------------------------------------------------------------------
conv3_block1_2_bn (BatchNormali (None, 28, 28, 128)  512         conv3_block1_2_conv[0][0]
-----------------------------------------------------------------------------------------------
```

```
conv3_block1_2_relu (Activation (None, 28, 28, 128)  0           conv3_block1_2_bn[0][0]
-----------------------------------------------------------------------------------------------
conv3_block1_0_conv (Conv2D)    (None, 28, 28, 512)  131584      conv2_block3_out[0][0]
-----------------------------------------------------------------------------------------------
conv3_block1_3_conv (Conv2D)    (None, 28, 28, 512)  66048       conv3_block1_2_relu[0][0]
-----------------------------------------------------------------------------------------------
conv3_block1_0_bn (BatchNormali (None, 28, 28, 512)  2048        conv3_block1_0_conv[0][0]
-----------------------------------------------------------------------------------------------
conv3_block1_3_bn (BatchNormali (None, 28, 28, 512)  2048        conv3_block1_3_conv[0][0]
-----------------------------------------------------------------------------------------------
conv3_block1_add (Add)          (None, 28, 28, 512)  0           conv3_block1_0_bn[0][0]
                                                                 conv3_block1_3_bn[0][0]
-----------------------------------------------------------------------------------------------
conv3_block1_out (Activation)   (None, 28, 28, 512)  0           conv3_block1_add[0][0]
-----------------------------------------------------------------------------------------------
conv3_block2_1_conv (Conv2D)    (None, 28, 28, 128)  65664       conv3_block1_out[0][0]
-----------------------------------------------------------------------------------------------
conv3_block2_1_bn (BatchNormali (None, 28, 28, 128)  512         conv3_block2_1_conv[0][0]
-----------------------------------------------------------------------------------------------
conv3_block2_1_relu (Activation (None, 28, 28, 128)  0           conv3_block2_1_bn[0][0]
-----------------------------------------------------------------------------------------------
conv3_block2_2_conv (Conv2D)    (None, 28, 28, 128)  147584      conv3_block2_1_relu[0][0]
-----------------------------------------------------------------------------------------------
conv3_block2_2_bn (BatchNormali (None, 28, 28, 128)  512         conv3_block2_2_conv[0][0]
-----------------------------------------------------------------------------------------------
conv3_block2_2_relu (Activation (None, 28, 28, 128)  0           conv3_block2_2_bn[0][0]
-----------------------------------------------------------------------------------------------
conv3_block2_3_conv (Conv2D)    (None, 28, 28, 512)  66048       conv3_block2_2_relu[0][0]
-----------------------------------------------------------------------------------------------
conv3_block2_3_bn (BatchNormali (None, 28, 28, 512)  2048        conv3_block2_3_conv[0][0]
-----------------------------------------------------------------------------------------------
conv3_block2_add (Add)          (None, 28, 28, 512)  0           conv3_block1_out[0][0]
                                                                 conv3_block2_3_bn[0][0]
-----------------------------------------------------------------------------------------------
conv3_block2_out (Activation)   (None, 28, 28, 512)  0           conv3_block2_add[0][0]
-----------------------------------------------------------------------------------------------
conv3_block3_1_conv (Conv2D)    (None, 28, 28, 128)  65664       conv3_block2_out[0][0]
-----------------------------------------------------------------------------------------------
conv3_block3_1_bn (BatchNormali (None, 28, 28, 128)  512         conv3_block3_1_conv[0][0]
-----------------------------------------------------------------------------------------------
conv3_block3_1_relu (Activation (None, 28, 28, 128)  0           conv3_block3_1_bn[0][0]
-----------------------------------------------------------------------------------------------
conv3_block3_2_conv (Conv2D)    (None, 28, 28, 128)  147584      conv3_block3_1_relu[0][0]
-----------------------------------------------------------------------------------------------
conv3_block3_2_bn (BatchNormali (None, 28, 28, 128)  512         conv3_block3_2_conv[0][0]
-----------------------------------------------------------------------------------------------
conv3_block3_2_relu (Activation (None, 28, 28, 128)  0           conv3_block3_2_bn[0][0]
-----------------------------------------------------------------------------------------------
```

```
conv3_block3_3_conv (Conv2D)    (None, 28, 28, 512)  66048       conv3_block3_2_relu[0][0]
-----------------------------------------------------------------------------------------------
conv3_block3_3_bn (BatchNormali (None, 28, 28, 512)  2048        conv3_block3_3_conv[0][0]
-----------------------------------------------------------------------------------------------
conv3_block3_add (Add)          (None, 28, 28, 512)  0           conv3_block2_out[0][0]
                                                                 conv3_block3_3_bn[0][0]
-----------------------------------------------------------------------------------------------
conv3_block3_out (Activation)   (None, 28, 28, 512)  0           conv3_block3_add[0][0]
-----------------------------------------------------------------------------------------------
conv3_block4_1_conv (Conv2D)    (None, 28, 28, 128)  65664       conv3_block3_out[0][0]
-----------------------------------------------------------------------------------------------
conv3_block4_1_bn (BatchNormali (None, 28, 28, 128)  512         conv3_block4_1_conv[0][0]
-----------------------------------------------------------------------------------------------
conv3_block4_1_relu (Activation (None, 28, 28, 128)  0           conv3_block4_1_bn[0][0]
-----------------------------------------------------------------------------------------------
conv3_block4_2_conv (Conv2D)    (None, 28, 28, 128)  147584      conv3_block4_1_relu[0][0]
-----------------------------------------------------------------------------------------------
conv3_block4_2_bn (BatchNormali (None, 28, 28, 128)  512         conv3_block4_2_conv[0][0]
-----------------------------------------------------------------------------------------------
conv3_block4_2_relu (Activation (None, 28, 28, 128)  0           conv3_block4_2_bn[0][0]
-----------------------------------------------------------------------------------------------
conv3_block4_3_conv (Conv2D)    (None, 28, 28, 512)  66048       conv3_block4_2_relu[0][0]
-----------------------------------------------------------------------------------------------
conv3_block4_3_bn (BatchNormali (None, 28, 28, 512)  2048        conv3_block4_3_conv[0][0]
-----------------------------------------------------------------------------------------------
conv3_block4_add (Add)          (None, 28, 28, 512)  0           conv3_block3_out[0][0]
                                                                 conv3_block4_3_bn[0][0]
-----------------------------------------------------------------------------------------------
conv3_block4_out (Activation)   (None, 28, 28, 512)  0           conv3_block4_add[0][0]
-----------------------------------------------------------------------------------------------
conv4_block1_1_conv (Conv2D)    (None, 14, 14, 256)  131328      conv3_block4_out[0][0]
-----------------------------------------------------------------------------------------------
conv4_block1_1_bn (BatchNormali (None, 14, 14, 256)  1024        conv4_block1_1_conv[0][0]
-----------------------------------------------------------------------------------------------
conv4_block1_1_relu (Activation (None, 14, 14, 256)  0           conv4_block1_1_bn[0][0]
-----------------------------------------------------------------------------------------------
conv4_block1_2_conv (Conv2D)    (None, 14, 14, 256)  590080      conv4_block1_1_relu[0][0]
-----------------------------------------------------------------------------------------------
conv4_block1_2_bn (BatchNormali (None, 14, 14, 256)  1024        conv4_block1_2_conv[0][0]
-----------------------------------------------------------------------------------------------
conv4_block1_2_relu (Activation (None, 14, 14, 256)  0           conv4_block1_2_bn[0][0]
-----------------------------------------------------------------------------------------------
conv4_block1_0_conv (Conv2D)    (None, 14, 14, 1024) 525312      conv3_block4_out[0][0]
-----------------------------------------------------------------------------------------------
conv4_block1_3_conv (Conv2D)    (None, 14, 14, 1024) 263168      conv4_block1_2_relu[0][0]
-----------------------------------------------------------------------------------------------
conv4_block1_0_bn (BatchNormali (None, 14, 14, 1024) 4096        conv4_block1_0_conv[0][0]
-----------------------------------------------------------------------------------------------
```

```
conv4_block1_3_bn (BatchNormali (None, 14, 14, 1024) 4096        conv4_block1_3_conv[0][0]
----------------------------------------------------------------------------------------------
conv4_block1_add (Add)          (None, 14, 14, 1024) 0           conv4_block1_0_bn[0][0]
                                                                 conv4_block1_3_bn[0][0]
----------------------------------------------------------------------------------------------
conv4_block1_out (Activation)   (None, 14, 14, 1024) 0           conv4_block1_add[0][0]
----------------------------------------------------------------------------------------------
conv4_block2_1_conv (Conv2D)    (None, 14, 14, 256)  262400      conv4_block1_out[0][0]
----------------------------------------------------------------------------------------------
conv4_block2_1_bn (BatchNormali (None, 14, 14, 256)  1024        conv4_block2_1_conv[0][0]
----------------------------------------------------------------------------------------------
conv4_block2_1_relu (Activation (None, 14, 14, 256)  0           conv4_block2_1_bn[0][0]
----------------------------------------------------------------------------------------------
conv4_block2_2_conv (Conv2D)    (None, 14, 14, 256)  590080      conv4_block2_1_relu[0][0]
----------------------------------------------------------------------------------------------
conv4_block2_2_bn (BatchNormali (None, 14, 14, 256)  1024        conv4_block2_2_conv[0][0]
----------------------------------------------------------------------------------------------
conv4_block2_2_relu (Activation (None, 14, 14, 256)  0           conv4_block2_2_bn[0][0]
----------------------------------------------------------------------------------------------
conv4_block2_3_conv (Conv2D)    (None, 14, 14, 1024) 263168      conv4_block2_2_relu[0][0]
----------------------------------------------------------------------------------------------
conv4_block2_3_bn (BatchNormali (None, 14, 14, 1024) 4096        conv4_block2_3_conv[0][0]
----------------------------------------------------------------------------------------------
conv4_block2_add (Add)          (None, 14, 14, 1024) 0           conv4_block1_out[0][0]
                                                                 conv4_block2_3_bn[0][0]
----------------------------------------------------------------------------------------------
conv4_block2_out (Activation)   (None, 14, 14, 1024) 0           conv4_block2_add[0][0]
----------------------------------------------------------------------------------------------
conv4_block3_1_conv (Conv2D)    (None, 14, 14, 256)  262400      conv4_block2_out[0][0]
----------------------------------------------------------------------------------------------
conv4_block3_1_bn (BatchNormali (None, 14, 14, 256)  1024        conv4_block3_1_conv[0][0]
----------------------------------------------------------------------------------------------
conv4_block3_1_relu (Activation (None, 14, 14, 256)  0           conv4_block3_1_bn[0][0]
----------------------------------------------------------------------------------------------
conv4_block3_2_conv (Conv2D)    (None, 14, 14, 256)  590080      conv4_block3_1_relu[0][0]
----------------------------------------------------------------------------------------------
conv4_block3_2_bn (BatchNormali (None, 14, 14, 256)  1024        conv4_block3_2_conv[0][0]
----------------------------------------------------------------------------------------------
conv4_block3_2_relu (Activation (None, 14, 14, 256)  0           conv4_block3_2_bn[0][0]
----------------------------------------------------------------------------------------------
conv4_block3_3_conv (Conv2D)    (None, 14, 14, 1024) 263168      conv4_block3_2_relu[0][0]
----------------------------------------------------------------------------------------------
conv4_block3_3_bn (BatchNormali (None, 14, 14, 1024) 4096        conv4_block3_3_conv[0][0]
----------------------------------------------------------------------------------------------
conv4_block3_add (Add)          (None, 14, 14, 1024) 0           conv4_block2_out[0][0]
                                                                 conv4_block3_3_bn[0][0]
----------------------------------------------------------------------------------------------
conv4_block3_out (Activation)   (None, 14, 14, 1024) 0           conv4_block3_add[0][0]
```

```
______________________________________________________________________________________________
conv4_block4_1_conv (Conv2D)    (None, 14, 14, 256)  262400      conv4_block3_out[0][0]
______________________________________________________________________________________________
conv4_block4_1_bn (BatchNormali (None, 14, 14, 256)  1024        conv4_block4_1_conv[0][0]
______________________________________________________________________________________________
conv4_block4_1_relu (Activation (None, 14, 14, 256)  0           conv4_block4_1_bn[0][0]
______________________________________________________________________________________________
conv4_block4_2_conv (Conv2D)    (None, 14, 14, 256)  590080      conv4_block4_1_relu[0][0]
______________________________________________________________________________________________
conv4_block4_2_bn (BatchNormali (None, 14, 14, 256)  1024        conv4_block4_2_conv[0][0]
______________________________________________________________________________________________
conv4_block4_2_relu (Activation (None, 14, 14, 256)  0           conv4_block4_2_bn[0][0]
______________________________________________________________________________________________
conv4_block4_3_conv (Conv2D)    (None, 14, 14, 1024) 263168      conv4_block4_2_relu[0][0]
______________________________________________________________________________________________
conv4_block4_3_bn (BatchNormali (None, 14, 14, 1024) 4096        conv4_block4_3_conv[0][0]
______________________________________________________________________________________________
conv4_block4_add (Add)          (None, 14, 14, 1024) 0           conv4_block3_out[0][0]
                                                                 conv4_block4_3_bn[0][0]
______________________________________________________________________________________________
conv4_block4_out (Activation)   (None, 14, 14, 1024) 0           conv4_block4_add[0][0]
______________________________________________________________________________________________
conv4_block5_1_conv (Conv2D)    (None, 14, 14, 256)  262400      conv4_block4_out[0][0]
______________________________________________________________________________________________
conv4_block5_1_bn (BatchNormali (None, 14, 14, 256)  1024        conv4_block5_1_conv[0][0]
______________________________________________________________________________________________
conv4_block5_1_relu (Activation (None, 14, 14, 256)  0           conv4_block5_1_bn[0][0]
______________________________________________________________________________________________
conv4_block5_2_conv (Conv2D)    (None, 14, 14, 256)  590080      conv4_block5_1_relu[0][0]
______________________________________________________________________________________________
conv4_block5_2_bn (BatchNormali (None, 14, 14, 256)  1024        conv4_block5_2_conv[0][0]
______________________________________________________________________________________________
conv4_block5_2_relu (Activation (None, 14, 14, 256)  0           conv4_block5_2_bn[0][0]
______________________________________________________________________________________________
conv4_block5_3_conv (Conv2D)    (None, 14, 14, 1024) 263168      conv4_block5_2_relu[0][0]
______________________________________________________________________________________________
conv4_block5_3_bn (BatchNormali (None, 14, 14, 1024) 4096        conv4_block5_3_conv[0][0]
______________________________________________________________________________________________
conv4_block5_add (Add)          (None, 14, 14, 1024) 0           conv4_block4_out[0][0]
                                                                 conv4_block5_3_bn[0][0]
______________________________________________________________________________________________
conv4_block5_out (Activation)   (None, 14, 14, 1024) 0           conv4_block5_add[0][0]
______________________________________________________________________________________________
conv4_block6_1_conv (Conv2D)    (None, 14, 14, 256)  262400      conv4_block5_out[0][0]
______________________________________________________________________________________________
conv4_block6_1_bn (BatchNormali (None, 14, 14, 256)  1024        conv4_block6_1_conv[0][0]
______________________________________________________________________________________________
conv4_block6_1_relu (Activation (None, 14, 14, 256)  0           conv4_block6_1_bn[0][0]
_______________________________________________________________________________________________
conv4_block6_2_conv (Conv2D)    (None, 14, 14, 256)  590080      conv4_block6_1_relu[0][0]

_______________________________________________________________________________________________
conv4_block6_2_bn (BatchNormali (None, 14, 14, 256)  1024        conv4_block6_2_conv[0][0]

_______________________________________________________________________________________________
conv4_block6_2_relu (Activation (None, 14, 14, 256)  0           conv4_block6_2_bn[0][0]

_______________________________________________________________________________________________
conv4_block6_3_conv (Conv2D)    (None, 14, 14, 1024) 263168      conv4_block6_2_relu[0][0]

_______________________________________________________________________________________________
conv4_block6_3_bn (BatchNormali (None, 14, 14, 1024) 4096        conv4_block6_3_conv[0][0]

_______________________________________________________________________________________________
conv4_block6_add (Add)          (None, 14, 14, 1024) 0           conv4_block5_out[0][0]
                                                                 conv4_block6_3_bn[0][0]

_______________________________________________________________________________________________
conv4_block6_out (Activation)   (None, 14, 14, 1024) 0           conv4_block6_add[0][0]

_______________________________________________________________________________________________
conv5_block1_1_conv (Conv2D)    (None, 7, 7, 512)    524800      conv4_block6_out[0][0]

_______________________________________________________________________________________________
conv5_block1_1_bn (BatchNormali (None, 7, 7, 512)    2048        conv5_block1_1_conv[0][0]

_______________________________________________________________________________________________
conv5_block1_1_relu (Activation (None, 7, 7, 512)    0           conv5_block1_1_bn[0][0]

_______________________________________________________________________________________________
conv5_block1_2_conv (Conv2D)    (None, 7, 7, 512)    2359808     conv5_block1_1_relu[0][0]

_______________________________________________________________________________________________
conv5_block1_2_bn (BatchNormali (None, 7, 7, 512)    2048        conv5_block1_2_conv[0][0]

_______________________________________________________________________________________________
conv5_block1_2_relu (Activation (None, 7, 7, 512)    0           conv5_block1_2_bn[0][0]

_______________________________________________________________________________________________
conv5_block1_0_conv (Conv2D)    (None, 7, 7, 2048)   2099200     conv4_block6_out[0][0]

_______________________________________________________________________________________________
conv5_block1_3_conv (Conv2D)    (None, 7, 7, 2048)   1050624     conv5_block1_2_relu[0][0]

_______________________________________________________________________________________________
conv5_block1_0_bn (BatchNormali (None, 7, 7, 2048)   8192        conv5_block1_0_conv[0][0]

_______________________________________________________________________________________________
conv5_block1_3_bn (BatchNormali (None, 7, 7, 2048)   8192        conv5_block1_3_conv[0][0]

_______________________________________________________________________________________________
conv5_block1_add (Add)          (None, 7, 7, 2048)   0           conv5_block1_0_bn[0][0]
                                                                 conv5_block1_3_bn[0][0]

_______________________________________________________________________________________________
conv5_block1_out (Activation)   (None, 7, 7, 2048)   0           conv5_block1_add[0][0]

_______________________________________________________________________________________________
conv5_block2_1_conv (Conv2D)    (None, 7, 7, 512)    1049088     conv5_block1_out[0][0]

_______________________________________________________________________________________________
conv5_block2_1_bn (BatchNormali (None, 7, 7, 512)    2048        conv5_block2_1_conv[0][0]

_______________________________________________________________________________________________
conv5_block2_1_relu (Activation (None, 7, 7, 512)    0           conv5_block2_1_bn[0][0]

_______________________________________________________________________________________________
conv5_block2_2_conv (Conv2D)    (None, 7, 7, 512)    2359808     conv5_block2_1_relu[0][0]
```

```
__________________________________________________________________________________________________
conv5_block2_2_bn (BatchNormali (None, 7, 7, 512)    2048        conv5_block2_2_conv[0][0]
__________________________________________________________________________________________________
conv5_block2_2_relu (Activation (None, 7, 7, 512)    0           conv5_block2_2_bn[0][0]
__________________________________________________________________________________________________
conv5_block2_3_conv (Conv2D)    (None, 7, 7, 2048)   1050624     conv5_block2_2_relu[0][0]
__________________________________________________________________________________________________
conv5_block2_3_bn (BatchNormali (None, 7, 7, 2048)   8192        conv5_block2_3_conv[0][0]
__________________________________________________________________________________________________
conv5_block2_add (Add)          (None, 7, 7, 2048)   0           conv5_block1_out[0][0]
                                                                 conv5_block2_3_bn[0][0]
__________________________________________________________________________________________________
conv5_block2_out (Activation)   (None, 7, 7, 2048)   0           conv5_block2_add[0][0]
__________________________________________________________________________________________________
conv5_block3_1_conv (Conv2D)    (None, 7, 7, 512)    1049088     conv5_block2_out[0][0]
__________________________________________________________________________________________________
conv5_block3_1_bn (BatchNormali (None, 7, 7, 512)    2048        conv5_block3_1_conv[0][0]
__________________________________________________________________________________________________
conv5_block3_1_relu (Activation (None, 7, 7, 512)    0           conv5_block3_1_bn[0][0]
__________________________________________________________________________________________________
conv5_block3_2_conv (Conv2D)    (None, 7, 7, 512)    2359808     conv5_block3_1_relu[0][0]
__________________________________________________________________________________________________
conv5_block3_2_bn (BatchNormali (None, 7, 7, 512)    2048        conv5_block3_2_conv[0][0]
__________________________________________________________________________________________________
conv5_block3_2_relu (Activation (None, 7, 7, 512)    0           conv5_block3_2_bn[0][0]
__________________________________________________________________________________________________
conv5_block3_3_conv (Conv2D)    (None, 7, 7, 2048)   1050624     conv5_block3_2_relu[0][0]
__________________________________________________________________________________________________
conv5_block3_3_bn (BatchNormali (None, 7, 7, 2048)   8192        conv5_block3_3_conv[0][0]
__________________________________________________________________________________________________
conv5_block3_add (Add)          (None, 7, 7, 2048)   0           conv5_block2_out[0][0]
                                                                 conv5_block3_3_bn[0][0]
__________________________________________________________________________________________________
conv5_block3_out (Activation)   (None, 7, 7, 2048)   0           conv5_block3_add[0][0]
__________________________________________________________________________________________________
avg_pool (GlobalAveragePooling2 (None, 2048)         0           conv5_block3_out[0][0]
__________________________________________________________________________________________________
probs (Dense)                   (None, 1000)         2049000     avg_pool[0][0]
==================================================================================================
Total params: 25,636,712
Trainable params: 25,583,592
Non-trainable params: 53,120
__________________________________________________________________________________________________
```

Import and preprocess 3 sample images

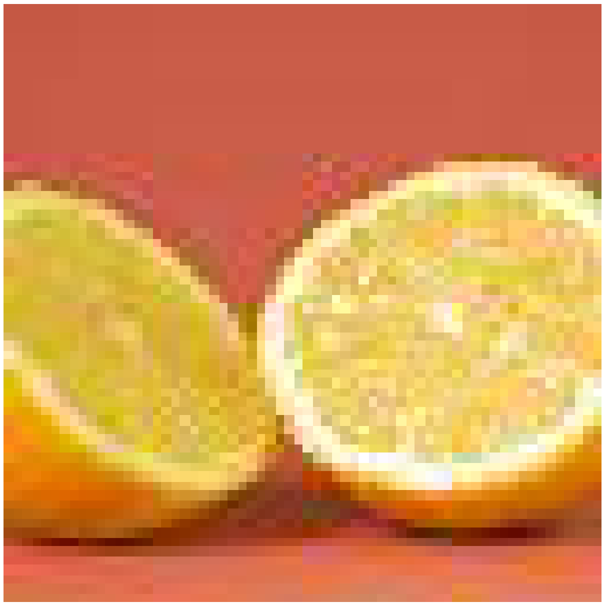
```
In [29]: # Import 3 sample ImageNet images

         from tensorflow.keras.preprocessing.image import load_img

         lemon_img = load_img('data/lemon.jpg', target_size=(224, 224))
         viaduct_img = load_img('data/viaduct.jpg', target_size=(224, 224))
         water_tower_img = load_img('data/water_tower.jpg', target_size=(224, 224))
```

Use ResNet50 model to classify images

```
In [30]: # Useful function: presents top 5 predictions and probabilities

         from tensorflow.keras.preprocessing.image import img_to_array
         from tensorflow.keras.applications.resnet50 import preprocess_input, decode_predictio
         import numpy as np
         import pandas as pd

         def get_top_5_predictions(img):
             x = img_to_array(img)[np.newaxis, ...]
             x = preprocess_input(x)
             preds = decode_predictions(model.predict(x), top=5)
             top_preds = pd.DataFrame(columns=['prediction', 'probability'],
                                      index=np.arange(5)+1)
             for i in range(5):
                 top_preds.loc[i+1, 'prediction'] = preds[0][i][1]
                 top_preds.loc[i+1, 'probability'] = preds[0][i][2]
             return top_preds
```

Image 1: lemon

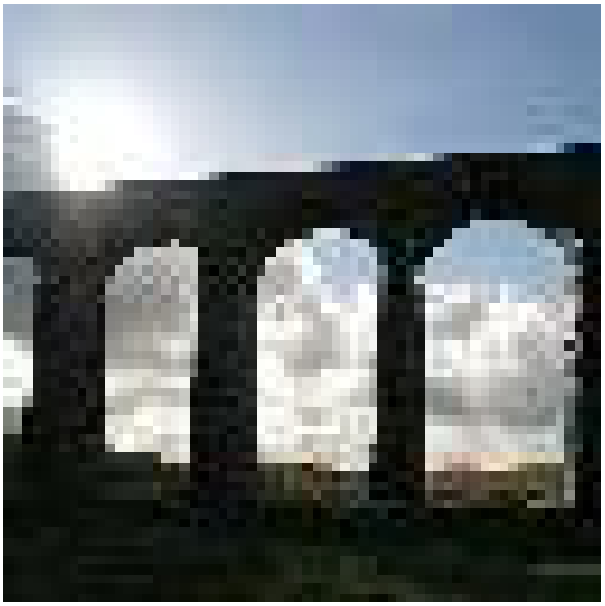
```
In [31]: # Display image
         lemon_img

Out[31]:
```



```
In [32]: # Display top 5 predictions
         get_top_5_predictions(lemon_img)

Downloading data from https://storage.googleapis.com/download.tensorflow.org/data/imagenet_clas
40960/35363 [==================================] - 0s 0us/step


Out[32]:   prediction probability
         1   consomme    0.227801
         2      lemon    0.221758
         3     eggnog    0.151335
         4      ladle   0.0400883
         5  spotlight   0.0291972
```

Image 2: viaduct

```
In [33]: # Display image
         viaduct_img

Out[33]:
```



```
In [34]: # Display top 5 predictions
         get_top_5_predictions(viaduct_img)

Out[34]:       prediction probability
         1          vault    0.367951
         2         prison    0.111742
         3        viaduct    0.110738
         4  window_screen   0.0882307
         5    fire_screen   0.0206752
```

Image 3: water tower

```
In [35]: # Display image

         water_tower_img

Out[35]:
```

```
In [36]: # Display top 5 predictions

         get_top_5_predictions(water_tower_img)

Out[36]:       prediction probability
         1     solar_dish    0.384681
         2          ladle    0.196179
         3    planetarium    0.116682
         4       strainer     0.04999
         5  jigsaw_puzzle   0.0219473
```

## Tensorflow Hub modules

Import and build Tensorflow Hub MobileNet v1 model Today we'll be using Google's MobileNet v1 model, available on Tensorflow Hub. Please see the description on the Tensorflow Hub page for details on it's architecture, how it's trained, and the reference. If you continue using it, please cite it properly! The paper it comes from is:

Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, Hartwig Adam: "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications", 2017.

This model takes a long time to download on the Coursera platform, so it is pre-downloaded in your workspace and saved in Tensorflow SavedModel format. If you want to import it on your personal machine, use the following code:

```
module_url = "https://tfhub.dev/google/imagenet/mobilenet_v1_050_160/classification/4"
model = Sequential([hub.KerasLayer(module_url)])
model.build(input_shape=[None, 160, 160, 3])
```

In this coding tutorial, you will instead load the model directly from disk.

```
In [40]: import tensorflow_hub as hub
         from tensorflow.keras.models import load_model
```

```
In [42]: # Build Google's Mobilenet v1 model

         module = load_model('models/Tensorflow_MobileNet_v1/')
         model = Sequential(hub.KerasLayer(module))
         model.build(input_shape=[None, 160, 160, 3])
         model.summary()
```

```
Model: "sequential_3"

_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
keras_layer (KerasLayer)     multiple                  1343049   
=================================================================
Total params: 1,343,049
Trainable params: 0
Non-trainable params: 1,343,049

_________________________________________________________________
```

Use MobileNet model to classify images

```
In [43]: # Import and preprocess 3 sample ImageNet images

         from tensorflow.keras.preprocessing.image import load_img

         lemon_img = load_img("data/lemon.jpg", target_size=(160, 160))
         viaduct_img = load_img("data/viaduct.jpg", target_size=(160, 160))
         water_tower_img = load_img("data/water_tower.jpg", target_size=(160, 160))
```

```
In [44]: # Read in categories text file

         with open('data/imagenet_categories.txt') as txt_file:
             categories = txt_file.read().splitlines()
```

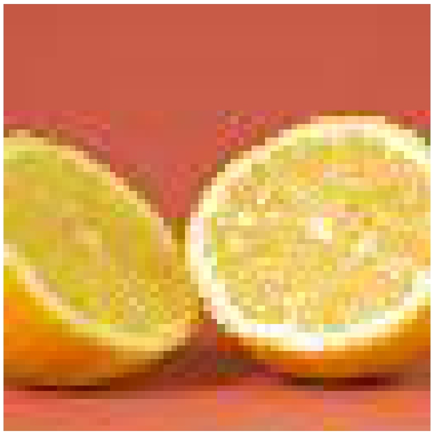
```
In [45]: # Useful function: presents top 5 predictions

         import pandas as pd

         def get_top_5_predictions(img):
             x = img_to_array(img)[np.newaxis, ...] / 255.0
             preds = model.predict(x)
             top_preds = pd.DataFrame(columns=['prediction'],
                                      index=np.arange(5)+1)
             sorted_index = np.argsort(-preds[0])
```

```
    for i in range(5):
        ith_pred = categories[sorted_index[i]]
        top_preds.loc[i+1, 'prediction'] = ith_pred

    return top_preds
```

Image 1: lemon

```
In [46]: lemon_img
```

```
Out[46]:
```



```
In [47]: get_top_5_predictions(lemon_img)
```

```
Out[47]:          prediction
         1          shower cap
         2                tray
         3              candle
         4           brassiere
         5   African chameleon
```

Image 2: viaduct

```
In [48]: viaduct_img
```

```
Out[48]:
```

```
In [49]: get_top_5_predictions(viaduct_img)

Out[49]:    prediction
         1     viaduct
         2        pier
         3         dam
         4      prison
         5  solar dish
```

Image 3: water tower

```
In [50]: water_tower_img

Out[50]:
```

```
In [51]: get_top_5_predictions(water_tower_img)
```

```
Out[51]:          prediction
        1        solar dish
        2       water tower
        3  aircraft carrier
        4     jigsaw puzzle
        5       oxygen mask
```

```
In [ ]:
```