Week 3: Transfer Learning

Welcome to this assignment! This week, you are going to use a technique called `Transfer Learning` in which you utilize an already trained network to help you solve a similar problem to the one it was originally trained to solve.

Let's get started!

NOTE: *To prevent errors from the autograder, please avoid editing or deleting non-graded cells in this notebook . Please only put your solutions in between the `### START CODE HERE` and `### END CODE HERE` code comments, and refrain from adding any new cells.*

```
# grader-required-cell

import os
import zipfile
import matplotlib.pyplot as plt
import tensorflow as tf
from tensorflow.keras import layers
from tensorflow.keras import Model
from tensorflow.keras.optimizers import RMSprop
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.utils import img_to_array, load_img
```

Dataset

For this assignment, you will use the `Horse or Human dataset`, which contains images of horses and humans.

Download the `training` and `validation` sets by running the cell below:

```
# Get the Horse or Human training dataset
!wget -q -P /content/ https://storage.googleapis.com/tensorflow-1-public/course2/week3/horse-or-human.zip

# Get the Horse or Human validation dataset
!wget -q -P /content/ https://storage.googleapis.com/tensorflow-1-public/course2/week3/validation-horse-or-human.zip

test_local_zip = './horse-or-human.zip'
zip_ref = zipfile.ZipFile(test_local_zip, 'r')
zip_ref.extractall('/tmp/training')

val_local_zip = './validation-horse-or-human.zip'
zip_ref = zipfile.ZipFile(val_local_zip, 'r')
zip_ref.extractall('/tmp/validation')

zip_ref.close()
```

This dataset already has an structure that is compatible with Keras' `flow_from_directory` so you don't need to move the images into subdirectories as you did in the previous assignments. However, it is still a good idea to save the paths of the images so you can use them later on:

```
# grader-required-cell

# Define the training and validation base directories
train_dir = '/tmp/training'
validation_dir = '/tmp/validation'

# Directory with training horse pictures
train_horses_dir = os.path.join(train_dir, 'horses')
# Directory with training humans pictures
train_humans_dir = os.path.join(train_dir, 'humans')
# Directory with validation horse pictures
validation_horses_dir = os.path.join(validation_dir, 'horses')
# Directory with validation human pictures
validation_humans_dir = os.path.join(validation_dir, 'humans')

# Check the number of images for each class and set
print(f"There are {len(os.listdir(train_horses_dir))} images of horses for training.\n")
print(f"There are {len(os.listdir(train_humans_dir))} images of humans for training.\n")
print(f"There are {len(os.listdir(validation_horses_dir))} images of horses for validation.\n")
print(f"There are {len(os.listdir(validation_humans_dir))} images of humans for validation.\n")
```

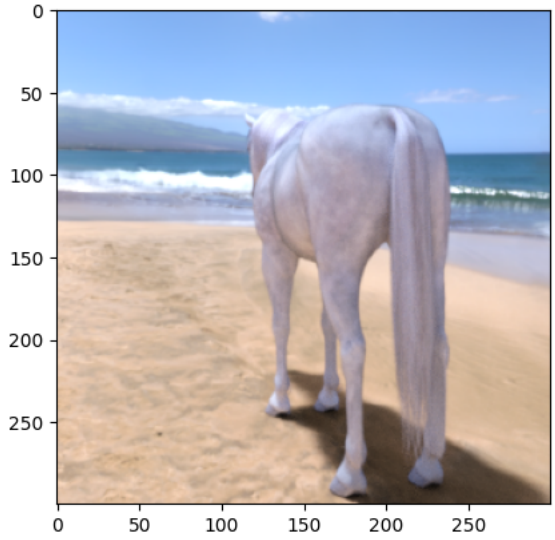
```
There are 500 images of horses for training.

There are 527 images of humans for training.
```

```
There are 128 images of horses for validation.

There are 128 images of humans for validation.
```

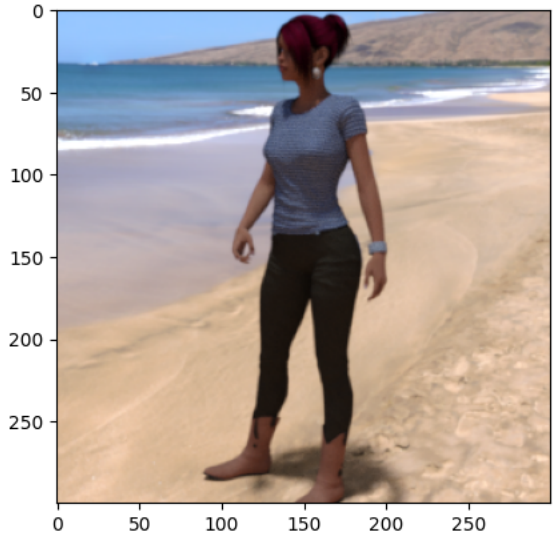
Now take a look at a sample image of each one of the classes:

```
# grader-required-cell

print("Sample horse image:")
plt.imshow(load_img(f"{os.path.join(train_horses_dir, os.listdir(train_horses_dir)[0])}"))
plt.show()

print("\nSample human image:")
plt.imshow(load_img(f"{os.path.join(train_humans_dir, os.listdir(train_humans_dir)[0])}"))
plt.show()
```

```
Sample horse image:
```




```
Sample human image:
```



`matplotlib` makes it easy to see that these images have a resolution of 300x300 and are colored, but you can double check this by using the code below:

```
# grader-required-cell

# Load the first example of a horse
sample_image  = load_img(f"{os.path.join(train_horses_dir, os.listdir(train_horses_dir)[0])}")

# Convert the image into its numpy array representation
sample_array = img_to_array(sample_image)

print(f"Each image has shape: {sample_array.shape}")
```

```
Each image has shape: (300, 300, 3)
```

As expected, the sample image has a resolution of 300x300 and the last dimension is used for each one of the RGB channels to represent color.

Training and Validation Generators

Now that you know the images you are dealing with, it is time for you to code the generators that will fed these images to your Network. For this, complete the `train_val_generators` function below:

Important Note: The images have a resolution of 300x300 but the `flow_from_directory` method you will use allows you to set a target resolution. In this case, **set a `target_size` of (150, 150)**. This will heavily lower the number of trainable parameters in your final network, yielding much quicker training times without compromising the accuracy!

```
# grader-required-cell

# GRADED FUNCTION: train_val_generators
def train_val_generators(TRAINING_DIR, VALIDATION_DIR):
  ### START CODE HERE
  # Instantiate the ImageDataGenerator class (don't forget to set the arguments to augment the images)
  train_datagen = ImageDataGenerator(rescale=1.0/255,
                                     rotation_range=40,
                                     width_shift_range=0.2,
                                     height_shift_range=0.2,
                                     shear_range=0.2,
                                     zoom_range=0.2,
                                     horizontal_flip=True,
                                     fill_mode='nearest')

  # Pass in the appropriate arguments to the flow_from_directory method
  train_generator = train_datagen.flow_from_directory(directory=TRAINING_DIR,
                                                      batch_size=64,
                                                      class_mode='binary',
                                                      target_size=(150, 150))

  # Instantiate the ImageDataGenerator class (don't forget to set the rescale argument)
  validation_datagen =  ImageDataGenerator(rescale=1.0/255,
                                     rotation_range=40,
                                     width_shift_range=0.2,
                                     height_shift_range=0.2,
                                     shear_range=0.2,
                                     zoom_range=0.2,
                                     horizontal_flip=True,
                                     fill_mode='nearest')

  # Pass in the appropriate arguments to the flow_from_directory method
  validation_generator = validation_datagen.flow_from_directory(directory=VALIDATION_DIR,
                                                                batch_size=64,
                                                                class_mode='binary',
                                                                target_size=(150, 150))
  ### END CODE HERE
  return train_generator, validation_generator


# grader-required-cell

# Test your generators
train_generator, validation_generator = train_val_generators(train_dir, validation_dir)
```

```
Found 1027 images belonging to 2 classes.
Found 256 images belonging to 2 classes.
```

Expected Output:

```
Found 1027 images belonging to 2 classes.
Found 256 images belonging to 2 classes.
```

Transfer learning - Create the pre-trained model

Download the `inception V3` weights into the `/tmp/` directory:

```
# Download the inception v3 weights
!wget --no-check-certificate \
    https://storage.googleapis.com/mledu-datasets/inception_v3_weights_tf_dim_ordering_tf_kernels_notop.h5 \
    -O /tmp/inception_v3_weights_tf_dim_ordering_tf_kernels_notop.h5
```

```
--2024-08-10 22:41:01--  https://storage.googleapis.com/mledu-datasets/inception_v3_weights_tf_dim_ordering_tf_kernels_n
Resolving storage.googleapis.com (storage.googleapis.com)... 74.125.130.207, 74.125.68.207, 64.233.170.207, ...
Connecting to storage.googleapis.com (storage.googleapis.com)|74.125.130.207|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 87910968 (84M) [application/x-hdf]
Saving to: '/tmp/inception_v3_weights_tf_dim_ordering_tf_kernels_notop.h5'

/tmp/inception_v3_w 100%[===================>]  83.84M  18.7MB/s    in 5.5s

2024-08-10 22:41:07 (15.2 MB/s) - '/tmp/inception_v3_weights_tf_dim_ordering_tf_kernels_notop.h5' saved [87910968/879109
```

Now load the `InceptionV3` model and save the path to the weights you just downloaded:

```
# grader-required-cell

# Import the inception model
from tensorflow.keras.applications.inception_v3 import InceptionV3

# Create an instance of the inception model from the local pre-trained weights
local_weights_file = '/tmp/inception_v3_weights_tf_dim_ordering_tf_kernels_notop.h5'
```

Complete the `create_pre_trained_model` function below. You should specify the correct `input_shape` for the model (remember that you set a new resolution for the images instead of the native 300x300) and make all of the layers non-trainable:

```
# grader-required-cell

# GRADED FUNCTION: create_pre_trained_model
def create_pre_trained_model(local_weights_file,include_top = False):
  """
  Initializes an InceptionV3 model.

  Args:
    local_weights_file (string): path pointing to a pretrained weights H5 file

  Returns:
    pre_trained_model: the initialized InceptionV3 model
  """
  ### START CODE HERE
  pre_trained_model = InceptionV3(input_shape = (150, 150, 3),
                                  include_top = False,
                                  weights = None)

  pre_trained_model.load_weights(local_weights_file)

  # Make all the layers in the pre-trained model non-trainable
  for layer in pre_trained_model.layers:
    layer.trainable = False

  ### END CODE HERE

  return pre_trained_model
```

Check that everything went well by comparing the last few rows of the model summary to the expected output:

```
# grader-required-cell

pre_trained_model = create_pre_trained_model(local_weights_file)

# Print the model summary
pre_trained_model.summary()
```

Model: "inception_v3"

Layer (type)	Output Shape	Param #	Connected to
input_layer (InputLayer)	(None, 150, 150, 3)	0	-
conv2d (Conv2D)	(None, 74, 74, 32)	864	input_layer[0][0]
batch_normalization (BatchNormalization)	(None, 74, 74, 32)	96	conv2d[0][0]
activation (Activation)	(None, 74, 74, 32)	0	batch_normalization[0…
conv2d_1 (Conv2D)	(None, 72, 72, 32)	9,216	activation[0][0]
batch_normalization_1 (BatchNormalization)	(None, 72, 72, 32)	96	conv2d_1[0][0]
activation_1 (Activation)	(None, 72, 72, 32)	0	batch_normalization_1…
conv2d_2 (Conv2D)	(None, 72, 72, 64)	18,432	activation_1[0][0]
batch_normalization_2 (BatchNormalization)	(None, 72, 72, 64)	192	conv2d_2[0][0]
activation_2 (Activation)	(None, 72, 72, 64)	0	batch_normalization_2…
max_pooling2d (MaxPooling2D)	(None, 35, 35, 64)	0	activation_2[0][0]
conv2d_3 (Conv2D)	(None, 35, 35, 80)	5,120	max_pooling2d[0][0]
batch_normalization_3 (BatchNormalization)	(None, 35, 35, 80)	240	conv2d_3[0][0]
activation_3 (Activation)	(None, 35, 35, 80)	0	batch_normalization_3…
conv2d_4 (Conv2D)	(None, 33, 33, 192)	138,240	activation_3[0][0]
batch_normalization_4 (BatchNormalization)	(None, 33, 33, 192)	576	conv2d_4[0][0]
activation_4 (Activation)	(None, 33, 33, 192)	0	batch_normalization_4…
max_pooling2d_1 (MaxPooling2D)	(None, 16, 16, 192)	0	activation_4[0][0]
conv2d_8 (Conv2D)	(None, 16, 16, 64)	12,288	max_pooling2d_1[0][0]
batch_normalization_8 (BatchNormalization)	(None, 16, 16, 64)	192	conv2d_8[0][0]
activation_8 (Activation)	(None, 16, 16, 64)	0	batch_normalization_8…
conv2d_6 (Conv2D)	(None, 16, 16, 48)	9,216	max_pooling2d_1[0][0]
conv2d_9 (Conv2D)	(None, 16, 16, 96)	55,296	activation_8[0][0]
batch_normalization_6 (BatchNormalization)	(None, 16, 16, 48)	144	conv2d_6[0][0]
batch_normalization_9 (BatchNormalization)	(None, 16, 16, 96)	288	conv2d_9[0][0]
activation_6 (Activation)	(None, 16, 16, 48)	0	batch_normalization_6…
activation_9 (Activation)	(None, 16, 16, 96)	0	batch_normalization_9…
average_pooling2d (AveragePooling2D)	(None, 16, 16, 192)	0	max_pooling2d_1[0][0]
conv2d_5 (Conv2D)	(None, 16, 16, 64)	12,288	max_pooling2d_1[0][0]
conv2d_7 (Conv2D)	(None, 16, 16, 64)	76,800	activation_6[0][0]
conv2d_10 (Conv2D)	(None, 16, 16, 96)	82,944	activation_9[0][0]
conv2d_11 (Conv2D)	(None, 16, 16, 32)	6,144	average_pooling2d[0][…
batch_normalization_5 (BatchNormalization)	(None, 16, 16, 64)	192	conv2d_5[0][0]
batch_normalization_7 (BatchNormalization)	(None, 16, 16, 64)	192	conv2d_7[0][0]
batch_normalization_10 (BatchNormalization)	(None, 16, 16, 96)	288	conv2d_10[0][0]
batch_normalization_11 (BatchNormalization)	(None, 16, 16, 32)	96	conv2d_11[0][0]
activation_5 (Activation)	(None, 16, 16, 64)	0	batch_normalization_5…
activation_7 (Activation)	(None, 16, 16, 64)	0	batch_normalization_7…
activation_10 (Activation)	(None, 16, 16, 96)	0	batch_normalization_1…

activation_11 (Activation)	(None, 16, 16, 32)	0	batch_normalization_1…
mixed0 (Concatenate)	(None, 16, 16, 256)	0	activation_5[0][0], activation_7[0][0], activation_10[0][0], activation_11[0][0]
conv2d_15 (Conv2D)	(None, 16, 16, 64)	16,384	mixed0[0][0]
batch_normalization_15 (BatchNormalization)	(None, 16, 16, 64)	192	conv2d_15[0][0]
activation_15 (Activation)	(None, 16, 16, 64)	0	batch_normalization_1…
conv2d_13 (Conv2D)	(None, 16, 16, 48)	12,288	mixed0[0][0]
conv2d_16 (Conv2D)	(None, 16, 16, 96)	55,296	activation_15[0][0]
batch_normalization_13 (BatchNormalization)	(None, 16, 16, 48)	144	conv2d_13[0][0]
batch_normalization_16 (BatchNormalization)	(None, 16, 16, 96)	288	conv2d_16[0][0]
activation_13 (Activation)	(None, 16, 16, 48)	0	batch_normalization_1…
activation_16 (Activation)	(None, 16, 16, 96)	0	batch_normalization_1…
average_pooling2d_1 (AveragePooling2D)	(None, 16, 16, 256)	0	mixed0[0][0]
conv2d_12 (Conv2D)	(None, 16, 16, 64)	16,384	mixed0[0][0]
conv2d_14 (Conv2D)	(None, 16, 16, 64)	76,800	activation_13[0][0]
conv2d_17 (Conv2D)	(None, 16, 16, 96)	82,944	activation_16[0][0]
conv2d_18 (Conv2D)	(None, 16, 16, 64)	16,384	average_pooling2d_1[0…
batch_normalization_12 (BatchNormalization)	(None, 16, 16, 64)	192	conv2d_12[0][0]
batch_normalization_14 (BatchNormalization)	(None, 16, 16, 64)	192	conv2d_14[0][0]
batch_normalization_17 (BatchNormalization)	(None, 16, 16, 96)	288	conv2d_17[0][0]
batch_normalization_18 (BatchNormalization)	(None, 16, 16, 64)	192	conv2d_18[0][0]
activation_12 (Activation)	(None, 16, 16, 64)	0	batch_normalization_1…
activation_14 (Activation)	(None, 16, 16, 64)	0	batch_normalization_1…
activation_17 (Activation)	(None, 16, 16, 96)	0	batch_normalization_1…
activation_18 (Activation)	(None, 16, 16, 64)	0	batch_normalization_1…
mixed1 (Concatenate)	(None, 16, 16, 288)	0	activation_12[0][0], activation_14[0][0], activation_17[0][0], activation_18[0][0]
conv2d_22 (Conv2D)	(None, 16, 16, 64)	18,432	mixed1[0][0]
batch_normalization_22 (BatchNormalization)	(None, 16, 16, 64)	192	conv2d_22[0][0]
activation_22 (Activation)	(None, 16, 16, 64)	0	batch_normalization_2…
conv2d_20 (Conv2D)	(None, 16, 16, 48)	13,824	mixed1[0][0]
conv2d_23 (Conv2D)	(None, 16, 16, 96)	55,296	activation_22[0][0]
batch_normalization_20 (BatchNormalization)	(None, 16, 16, 48)	144	conv2d_20[0][0]
batch_normalization_23 (BatchNormalization)	(None, 16, 16, 96)	288	conv2d_23[0][0]
activation_20 (Activation)	(None, 16, 16, 48)	0	batch_normalization_2…
activation_23 (Activation)	(None, 16, 16, 96)	0	batch_normalization_2…
average_pooling2d_2 (AveragePooling2D)	(None, 16, 16, 288)	0	mixed1[0][0]

conv2d_19 (Conv2D)	(None, 16, 16, 64)	18,432	mixed1[0][0]
conv2d_21 (Conv2D)	(None, 16, 16, 64)	76,800	activation_20[0][0]
conv2d_24 (Conv2D)	(None, 16, 16, 96)	82,944	activation_23[0][0]
conv2d_25 (Conv2D)	(None, 16, 16, 64)	18,432	average_pooling2d_2[0…
batch_normalization_19 (BatchNormalization)	(None, 16, 16, 64)	192	conv2d_19[0][0]
batch_normalization_21 (BatchNormalization)	(None, 16, 16, 64)	192	conv2d_21[0][0]
batch_normalization_24 (BatchNormalization)	(None, 16, 16, 96)	288	conv2d_24[0][0]
batch_normalization_25 (BatchNormalization)	(None, 16, 16, 64)	192	conv2d_25[0][0]
activation_19 (Activation)	(None, 16, 16, 64)	0	batch_normalization_1…
activation_21 (Activation)	(None, 16, 16, 64)	0	batch_normalization_2…
activation_24 (Activation)	(None, 16, 16, 96)	0	batch_normalization_2…
activation_25 (Activation)	(None, 16, 16, 64)	0	batch_normalization_2…
mixed2 (Concatenate)	(None, 16, 16, 288)	0	activation_19[0][0], activation_21[0][0], activation_24[0][0], activation_25[0][0]
conv2d_27 (Conv2D)	(None, 16, 16, 64)	18,432	mixed2[0][0]
batch_normalization_27 (BatchNormalization)	(None, 16, 16, 64)	192	conv2d_27[0][0]
activation_27 (Activation)	(None, 16, 16, 64)	0	batch_normalization_2…
conv2d_28 (Conv2D)	(None, 16, 16, 96)	55,296	activation_27[0][0]
batch_normalization_28 (BatchNormalization)	(None, 16, 16, 96)	288	conv2d_28[0][0]
activation_28 (Activation)	(None, 16, 16, 96)	0	batch_normalization_2…
conv2d_26 (Conv2D)	(None, 7, 7, 384)	995,328	mixed2[0][0]
conv2d_29 (Conv2D)	(None, 7, 7, 96)	82,944	activation_28[0][0]
batch_normalization_26 (BatchNormalization)	(None, 7, 7, 384)	1,152	conv2d_26[0][0]
batch_normalization_29 (BatchNormalization)	(None, 7, 7, 96)	288	conv2d_29[0][0]
activation_26 (Activation)	(None, 7, 7, 384)	0	batch_normalization_2…
activation_29 (Activation)	(None, 7, 7, 96)	0	batch_normalization_2…
max_pooling2d_2 (MaxPooling2D)	(None, 7, 7, 288)	0	mixed2[0][0]
mixed3 (Concatenate)	(None, 7, 7, 768)	0	activation_26[0][0], activation_29[0][0], max_pooling2d_2[0][0]
conv2d_34 (Conv2D)	(None, 7, 7, 128)	98,304	mixed3[0][0]
batch_normalization_34 (BatchNormalization)	(None, 7, 7, 128)	384	conv2d_34[0][0]
activation_34 (Activation)	(None, 7, 7, 128)	0	batch_normalization_3…
conv2d_35 (Conv2D)	(None, 7, 7, 128)	114,688	activation_34[0][0]
batch_normalization_35 (BatchNormalization)	(None, 7, 7, 128)	384	conv2d_35[0][0]
activation_35 (Activation)	(None, 7, 7, 128)	0	batch_normalization_3…
conv2d_31 (Conv2D)	(None, 7, 7, 128)	98,304	mixed3[0][0]
conv2d_36 (Conv2D)	(None, 7, 7, 128)	114,688	activation_35[0][0]
batch_normalization_31	(None, 7, 7, 128)	384	conv2d_31[0][0]

(BatchNormalization)			
batch_normalization_36 (BatchNormalization)	(None, 7, 7, 128)	384	conv2d_36[0][0]
activation_31 (Activation)	(None, 7, 7, 128)	0	batch_normalization_3…
activation_36 (Activation)	(None, 7, 7, 128)	0	batch_normalization_3…
conv2d_32 (Conv2D)	(None, 7, 7, 128)	114,688	activation_31[0][0]
conv2d_37 (Conv2D)	(None, 7, 7, 128)	114,688	activation_36[0][0]
batch_normalization_32 (BatchNormalization)	(None, 7, 7, 128)	384	conv2d_32[0][0]
batch_normalization_37 (BatchNormalization)	(None, 7, 7, 128)	384	conv2d_37[0][0]
activation_32 (Activation)	(None, 7, 7, 128)	0	batch_normalization_3…
activation_37 (Activation)	(None, 7, 7, 128)	0	batch_normalization_3…
average_pooling2d_3 (AveragePooling2D)	(None, 7, 7, 768)	0	mixed3[0][0]
conv2d_30 (Conv2D)	(None, 7, 7, 192)	147,456	mixed3[0][0]
conv2d_33 (Conv2D)	(None, 7, 7, 192)	172,032	activation_32[0][0]
conv2d_38 (Conv2D)	(None, 7, 7, 192)	172,032	activation_37[0][0]
conv2d_39 (Conv2D)	(None, 7, 7, 192)	147,456	average_pooling2d_3[0…
batch_normalization_30 (BatchNormalization)	(None, 7, 7, 192)	576	conv2d_30[0][0]
batch_normalization_33 (BatchNormalization)	(None, 7, 7, 192)	576	conv2d_33[0][0]
batch_normalization_38 (BatchNormalization)	(None, 7, 7, 192)	576	conv2d_38[0][0]
batch_normalization_39 (BatchNormalization)	(None, 7, 7, 192)	576	conv2d_39[0][0]
activation_30 (Activation)	(None, 7, 7, 192)	0	batch_normalization_3…
activation_33 (Activation)	(None, 7, 7, 192)	0	batch_normalization_3…
activation_38 (Activation)	(None, 7, 7, 192)	0	batch_normalization_3…
activation_39 (Activation)	(None, 7, 7, 192)	0	batch_normalization_3…
mixed4 (Concatenate)	(None, 7, 7, 768)	0	activation_30[0][0], activation_33[0][0], activation_38[0][0], activation_39[0][0]
conv2d_44 (Conv2D)	(None, 7, 7, 160)	122,880	mixed4[0][0]
batch_normalization_44 (BatchNormalization)	(None, 7, 7, 160)	480	conv2d_44[0][0]
activation_44 (Activation)	(None, 7, 7, 160)	0	batch_normalization_4…
conv2d_45 (Conv2D)	(None, 7, 7, 160)	179,200	activation_44[0][0]
batch_normalization_45 (BatchNormalization)	(None, 7, 7, 160)	480	conv2d_45[0][0]
activation_45 (Activation)	(None, 7, 7, 160)	0	batch_normalization_4…
conv2d_41 (Conv2D)	(None, 7, 7, 160)	122,880	mixed4[0][0]
conv2d_46 (Conv2D)	(None, 7, 7, 160)	179,200	activation_45[0][0]
batch_normalization_41 (BatchNormalization)	(None, 7, 7, 160)	480	conv2d_41[0][0]
batch_normalization_46 (BatchNormalization)	(None, 7, 7, 160)	480	conv2d_46[0][0]
activation_41 (Activation)	(None, 7, 7, 160)	0	batch_normalization_4…
activation_46 (Activation)	(None, 7, 7, 160)	0	batch_normalization_4…

conv2d_42 (Conv2D)	(None, 7, 7, 160)	179,200	activation_41[0][0]
conv2d_47 (Conv2D)	(None, 7, 7, 160)	179,200	activation_46[0][0]
batch_normalization_42 (BatchNormalization)	(None, 7, 7, 160)	480	conv2d_42[0][0]
batch_normalization_47 (BatchNormalization)	(None, 7, 7, 160)	480	conv2d_47[0][0]
activation_42 (Activation)	(None, 7, 7, 160)	0	batch_normalization_4…
activation_47 (Activation)	(None, 7, 7, 160)	0	batch_normalization_4…
average_pooling2d_4 (AveragePooling2D)	(None, 7, 7, 768)	0	mixed4[0][0]
conv2d_40 (Conv2D)	(None, 7, 7, 192)	147,456	mixed4[0][0]
conv2d_43 (Conv2D)	(None, 7, 7, 192)	215,040	activation_42[0][0]
conv2d_48 (Conv2D)	(None, 7, 7, 192)	215,040	activation_47[0][0]
conv2d_49 (Conv2D)	(None, 7, 7, 192)	147,456	average_pooling2d_4[0…
batch_normalization_40 (BatchNormalization)	(None, 7, 7, 192)	576	conv2d_40[0][0]
batch_normalization_43 (BatchNormalization)	(None, 7, 7, 192)	576	conv2d_43[0][0]
batch_normalization_48 (BatchNormalization)	(None, 7, 7, 192)	576	conv2d_48[0][0]
batch_normalization_49 (BatchNormalization)	(None, 7, 7, 192)	576	conv2d_49[0][0]
activation_40 (Activation)	(None, 7, 7, 192)	0	batch_normalization_4…
activation_43 (Activation)	(None, 7, 7, 192)	0	batch_normalization_4…
activation_48 (Activation)	(None, 7, 7, 192)	0	batch_normalization_4…
activation_49 (Activation)	(None, 7, 7, 192)	0	batch_normalization_4…
mixed5 (Concatenate)	(None, 7, 7, 768)	0	activation_40[0][0], activation_43[0][0], activation_48[0][0], activation_49[0][0]
conv2d_54 (Conv2D)	(None, 7, 7, 160)	122,880	mixed5[0][0]
batch_normalization_54 (BatchNormalization)	(None, 7, 7, 160)	480	conv2d_54[0][0]
activation_54 (Activation)	(None, 7, 7, 160)	0	batch_normalization_5…
conv2d_55 (Conv2D)	(None, 7, 7, 160)	179,200	activation_54[0][0]
batch_normalization_55 (BatchNormalization)	(None, 7, 7, 160)	480	conv2d_55[0][0]
activation_55 (Activation)	(None, 7, 7, 160)	0	batch_normalization_5…
conv2d_51 (Conv2D)	(None, 7, 7, 160)	122,880	mixed5[0][0]
conv2d_56 (Conv2D)	(None, 7, 7, 160)	179,200	activation_55[0][0]
batch_normalization_51 (BatchNormalization)	(None, 7, 7, 160)	480	conv2d_51[0][0]
batch_normalization_56 (BatchNormalization)	(None, 7, 7, 160)	480	conv2d_56[0][0]
activation_51 (Activation)	(None, 7, 7, 160)	0	batch_normalization_5…
activation_56 (Activation)	(None, 7, 7, 160)	0	batch_normalization_5…
conv2d_52 (Conv2D)	(None, 7, 7, 160)	179,200	activation_51[0][0]
conv2d_57 (Conv2D)	(None, 7, 7, 160)	179,200	activation_56[0][0]
batch_normalization_52 (BatchNormalization)	(None, 7, 7, 160)	480	conv2d_52[0][0]
batch_normalization_57 (BatchNormalization)	(None, 7, 7, 160)	480	conv2d_57[0][0]

activation_52 (Activation)	(None, 7, 7, 160)	0	batch_normalization_5…
activation_57 (Activation)	(None, 7, 7, 160)	0	batch_normalization_5…
average_pooling2d_5 (AveragePooling2D)	(None, 7, 7, 768)	0	mixed5[0][0]
conv2d_50 (Conv2D)	(None, 7, 7, 192)	147,456	mixed5[0][0]
conv2d_53 (Conv2D)	(None, 7, 7, 192)	215,040	activation_52[0][0]
conv2d_58 (Conv2D)	(None, 7, 7, 192)	215,040	activation_57[0][0]
conv2d_59 (Conv2D)	(None, 7, 7, 192)	147,456	average_pooling2d_5[0…
batch_normalization_50 (BatchNormalization)	(None, 7, 7, 192)	576	conv2d_50[0][0]
batch_normalization_53 (BatchNormalization)	(None, 7, 7, 192)	576	conv2d_53[0][0]
batch_normalization_58 (BatchNormalization)	(None, 7, 7, 192)	576	conv2d_58[0][0]
batch_normalization_59 (BatchNormalization)	(None, 7, 7, 192)	576	conv2d_59[0][0]
activation_50 (Activation)	(None, 7, 7, 192)	0	batch_normalization_5…
activation_53 (Activation)	(None, 7, 7, 192)	0	batch_normalization_5…
activation_58 (Activation)	(None, 7, 7, 192)	0	batch_normalization_5…
activation_59 (Activation)	(None, 7, 7, 192)	0	batch_normalization_5…
mixed6 (Concatenate)	(None, 7, 7, 768)	0	activation_50[0][0], activation_53[0][0], activation_58[0][0], activation_59[0][0]
conv2d_64 (Conv2D)	(None, 7, 7, 192)	147,456	mixed6[0][0]
batch_normalization_64 (BatchNormalization)	(None, 7, 7, 192)	576	conv2d_64[0][0]
activation_64 (Activation)	(None, 7, 7, 192)	0	batch_normalization_6…
conv2d_65 (Conv2D)	(None, 7, 7, 192)	258,048	activation_64[0][0]
batch_normalization_65 (BatchNormalization)	(None, 7, 7, 192)	576	conv2d_65[0][0]
activation_65 (Activation)	(None, 7, 7, 192)	0	batch_normalization_6…
conv2d_61 (Conv2D)	(None, 7, 7, 192)	147,456	mixed6[0][0]
conv2d_66 (Conv2D)	(None, 7, 7, 192)	258,048	activation_65[0][0]
batch_normalization_61 (BatchNormalization)	(None, 7, 7, 192)	576	conv2d_61[0][0]
batch_normalization_66 (BatchNormalization)	(None, 7, 7, 192)	576	conv2d_66[0][0]
activation_61 (Activation)	(None, 7, 7, 192)	0	batch_normalization_6…
activation_66 (Activation)	(None, 7, 7, 192)	0	batch_normalization_6…
conv2d_62 (Conv2D)	(None, 7, 7, 192)	258,048	activation_61[0][0]
conv2d_67 (Conv2D)	(None, 7, 7, 192)	258,048	activation_66[0][0]
batch_normalization_62 (BatchNormalization)	(None, 7, 7, 192)	576	conv2d_62[0][0]
batch_normalization_67 (BatchNormalization)	(None, 7, 7, 192)	576	conv2d_67[0][0]
activation_62 (Activation)	(None, 7, 7, 192)	0	batch_normalization_6…
activation_67 (Activation)	(None, 7, 7, 192)	0	batch_normalization_6…
average_pooling2d_6 (AveragePooling2D)	(None, 7, 7, 768)	0	mixed6[0][0]
conv2d_60 (Conv2D)	(None, 7, 7, 192)	147,456	mixed6[0][0]