Modeling-CNN-and-Transfer-Learning

August 8, 2022

1 Author Information

Student name: Milad Shirani

Student pace: self paced

Scheduled project review date/time: August, 3, 2022 at 12:30 PM Pacific Time.

Instructor name: Claude Fried

Github: https://github.com/miladshiraniUCB/Emotion-Detection-in-Speech.git

2 Introduction

In this section, we will use the spectrograms made in the EDA notebook to train CNN models and we will present different models including transfer learning.

3 Importing Libraries

[1]:

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

[4]:

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
import os


from sklearn.preprocessing import LabelEncoder
from sklearn.metrics import confusion_matrix, plot_confusion_matrix,␣
 ↪classification_report

import tensorflow as tf
from tensorflow.keras import layers
```

```python
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense,␣
 ↪Dropout
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.optimizers import RMSprop, Adam
from tensorflow.keras.models import Sequential

import warnings
warnings.filterwarnings('ignore')

print("GPU is", "available" if tf.config.list_physical_devices('GPU') else "NOT␣
 ↪AVAILABLE")
```

```
GPU is available
```

[5]:

```
!nvidia-smi
```

```
Mon Aug  8 22:03:37 2022
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 460.32.03    Driver Version: 460.32.03    CUDA Version: 11.2     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|                               |                      |               MIG M. |
|===============================+======================+======================|
|   0  Tesla T4            Off  | 00000000:00:04.0 Off |                    0 |
| N/A   53C    P8    10W /  70W |      3MiB / 15109MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                                  |
|  GPU   GI   CI        PID   Type   Process name                  GPU Memory |
|        ID   ID                                                   Usage      |
|=============================================================================|
|  No running processes found                                                 |
+-----------------------------------------------------------------------------+
```

4 Functions

The function `loss_acc` is defined to take the model and returns the loss and accuracy diagrams of the model.

[17]:

```python
def loss_acc(model):


    fig , ax = plt.subplots(1,2)
    test_acc = model.history.history["val_accuracy"]
    test_loss = model.history.history["val_loss"]
    train_acc = model.history.history["accuracy"]
    train_loss = model.history.history["loss"]

    epochs = range(1, len(test_acc) + 1)

    fig.set_size_inches(20,6)
    ax[0].plot(epochs , train_loss , label = 'Training Loss')
    ax[0].plot(epochs , test_loss , label = 'Testing Loss')
    ax[0].set_title('Training & Testing Loss')
    ax[0].legend()
    ax[0].set_xlabel("Epochs")

    ax[1].plot(epochs , train_acc , label = 'Training Accuracy')
    ax[1].plot(epochs , test_acc , label = 'Testing Accuracy')
    ax[1].set_title('Training & Testing Accuracy')
    ax[1].legend()
    ax[1].set_xlabel("Epochs")
    plt.show()
```

[55]:

```
def extract_features(directory, sample_amount, transfer, out_size, batch_size = 30, n_class = 7):
    features = np.zeros(shape=(sample_amount, out_size))
    labels = np.zeros(shape=(sample_amount, n_class))
    generator = datagen.flow_from_directory(
                                            directory,
                                            target_size = target_size,
                                            batch_size  = batch_size,
                                            class_mode  ='categorical')
    # i=0
    for i, (inputs_batch, labels_batch) in enumerate(generator):
        features_batch = transfer.predict(inputs_batch)
        features[i * batch_size : (i + 1) * batch_size] = features_batch
        labels[i * batch_size : (i + 1) * batch_size] = labels_batch
        i = i + 1
        if i * batch_size >= sample_amount:
            break
    return features, labels
```

[11]:

```
# def classification(model, data):

#   pred = model.predict_generator(data)
#   prediction = pred > 0.5

#   print(data.class_indices)
#   print(classification_report(data.classes, prediction))
```

5 Importing Train-Test data sets

[20]:

```
base_dir = "/content/drive/MyDrive/Emotion-Detection/mel_spectrogram"

train_dir = os.path.join(base_dir, 'train')
test_dir  = os.path.join(base_dir, 'test')

# Directory with training normal/abnormal spectrograms
train_angry    = os.path.join(train_dir, 'angry')
train_disgust  = os.path.join(train_dir, 'disgust')
train_fear     = os.path.join(train_dir, 'fear')
train_happy    = os.path.join(train_dir, 'happy')
train_neutral  = os.path.join(train_dir, 'neutral')
train_sad      = os.path.join(train_dir, 'sad')
train_surprise = os.path.join(train_dir, 'surprise')


# Directory with test normal/abnormal spectrograms

test_angry    = os.path.join(test_dir, 'angry')
test_disgust  = os.path.join(test_dir, 'disgust')
test_fear     = os.path.join(test_dir, 'fear')
test_happy    = os.path.join(test_dir, 'happy')
test_neutral  = os.path.join(test_dir, 'neutral')
test_sad      = os.path.join(test_dir, 'sad')
test_surprise = os.path.join(test_dir, 'surprise')
```

[21]:

```
# All images will be rescaled by 1./255.
train_datagen = ImageDataGenerator( rescale = 1.0/255, dtype= tf.float64)
test_datagen  = ImageDataGenerator( rescale = 1.0/255, dtype= tf.float64)

target_size = (150, 150)
train_generator = train_datagen.flow_from_directory(train_dir,
                                                    batch_size=30,
                                                    class_mode="categorical",
                                                    target_size=target_size)

validation_generator =  test_datagen.flow_from_directory(test_dir,
                                                         batch_size=20,
                                                         class_mode  =␣
  ↪"categorical",
                                                         target_size =␣
  ↪target_size)
```

```
Found 2240 images belonging to 7 classes.
Found 560 images belonging to 7 classes.
```

6 Modeling

In this section, we will present some CNN based models and we will use some transfer learning models to be used in this work.

6.1 Convolutional Neural Network

In this section, we will present the CNN based models. As we can see all of the models perform well with the test accuracy around 99%.

6.1.1 First Model

```
[22]: model_1 = tf.keras.models.Sequential([
          # Note the input shape is the desired size of the image 150x150 with 3␣
        ↪bytes color
          tf.keras.layers.Conv2D(16, (3,3), activation='relu',
                                 input_shape=(target_size[0], target_size[1], 3)),
          tf.keras.layers.MaxPooling2D(2,2),
          tf.keras.layers.Flatten(),
          tf.keras.layers.Dense(64, activation='relu'),
          tf.keras.layers.Dense(7, activation='softmax')
      ])
      model_1.summary()
```

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_1 (Conv2D)           (None, 148, 148, 16)      448

 max_pooling2d_1 (MaxPooling  (None, 74, 74, 16)       0
 2D)

 flatten_1 (Flatten)         (None, 87616)             0

 dense_2 (Dense)             (None, 64)                5607488

 dense_3 (Dense)             (None, 7)                 455

=================================================================
Total params: 5,608,391
Trainable params: 5,608,391
Non-trainable params: 0
_________________________________________________________________
```

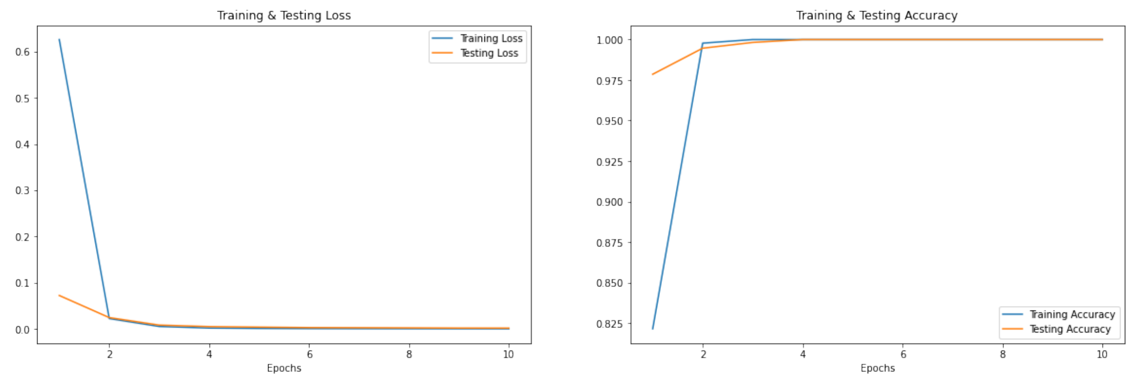
[23]:

```
model_1.compile(optimizer=Adam(learning_rate=0.001),
              loss="categorical_crossentropy",
              metrics = ['accuracy'])


model_1.fit_generator(
            train_generator,
            epochs=10,
            validation_data=validation_generator,
           )
print("Fitting Done")
```

```
Epoch 1/10
75/75 [==============================] - 94s 1s/step - loss: 0.6259 - accuracy:
0.8214 - val_loss: 0.0722 - val_accuracy: 0.9786
Epoch 2/10
75/75 [==============================] - 94s 1s/step - loss: 0.0222 - accuracy:
0.9978 - val_loss: 0.0243 - val_accuracy: 0.9946
Epoch 3/10
75/75 [==============================] - 94s 1s/step - loss: 0.0051 - accuracy:
1.0000 - val_loss: 0.0080 - val_accuracy: 0.9982
Epoch 4/10
75/75 [==============================] - 93s 1s/step - loss: 0.0017 - accuracy:
1.0000 - val_loss: 0.0048 - val_accuracy: 1.0000
Epoch 5/10
75/75 [==============================] - 93s 1s/step - loss: 8.6227e-04 -
accuracy: 1.0000 - val_loss: 0.0036 - val_accuracy: 1.0000
Epoch 6/10
75/75 [==============================] - 93s 1s/step - loss: 4.7626e-04 -
accuracy: 1.0000 - val_loss: 0.0026 - val_accuracy: 1.0000
Epoch 7/10
75/75 [==============================] - 92s 1s/step - loss: 3.0942e-04 -
accuracy: 1.0000 - val_loss: 0.0023 - val_accuracy: 1.0000
Epoch 8/10
75/75 [==============================] - 92s 1s/step - loss: 2.1078e-04 -
accuracy: 1.0000 - val_loss: 0.0020 - val_accuracy: 1.0000
Epoch 9/10
75/75 [==============================] - 92s 1s/step - loss: 1.5254e-04 -
accuracy: 1.0000 - val_loss: 0.0016 - val_accuracy: 1.0000
Epoch 10/10
75/75 [==============================] - 92s 1s/step - loss: 1.1298e-04 -
accuracy: 1.0000 - val_loss: 0.0015 - val_accuracy: 1.0000
Fitting Done
```

[24]:

```
loss_acc(model_1)
```



6.1.2 Second Model

```
[25]: model_2 = tf.keras.models.Sequential([
          tf.keras.layers.Conv2D(16, (3,3), activation='relu',
                                 input_shape=(target_size[0], target_size[1], 3)),
          tf.keras.layers.MaxPooling2D(2,2),
          tf.keras.layers.Conv2D(32, (3,3), activation='relu'),
          tf.keras.layers.MaxPooling2D(2,2),
          tf.keras.layers.Conv2D(64, (3,3), activation='relu'),
          tf.keras.layers.MaxPooling2D(2,2),
          tf.keras.layers.Flatten(),
          tf.keras.layers.Dense(512, activation='relu'),
          tf.keras.layers.Dense(7, activation='softmax')
      ])
      model_2.summary()
```

```
Model: "sequential_2"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_2 (Conv2D)           (None, 148, 148, 16)      448

 max_pooling2d_2 (MaxPooling  (None, 74, 74, 16)       0
 2D)

 conv2d_3 (Conv2D)           (None, 72, 72, 32)        4640

 max_pooling2d_3 (MaxPooling  (None, 36, 36, 32)       0
 2D)

 conv2d_4 (Conv2D)           (None, 34, 34, 64)        18496

 max_pooling2d_4 (MaxPooling  (None, 17, 17, 64)       0
 2D)
 flatten_2 (Flatten)         (None, 18496)             0

 dense_4 (Dense)             (None, 512)               9470464

 dense_5 (Dense)             (None, 7)                 3591

=================================================================
Total params: 9,497,639
Trainable params: 9,497,639
Non-trainable params: 0
_________________________________________________________________
```

[26]:

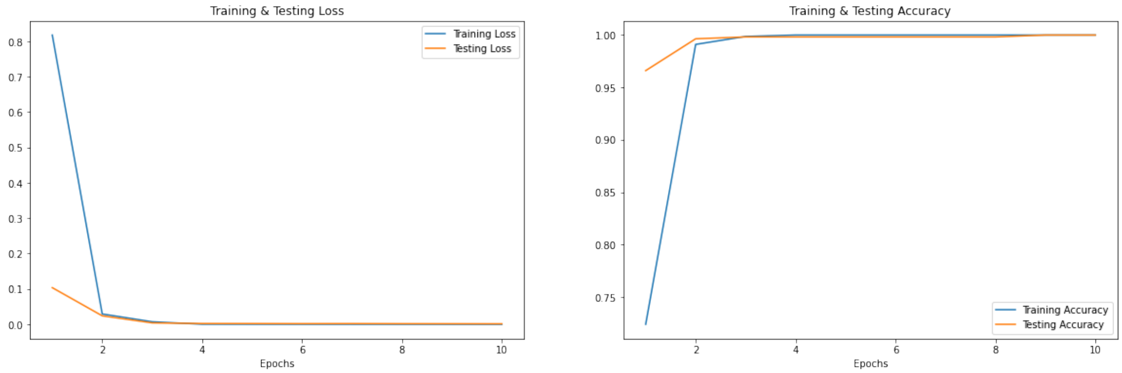
```python
model_2.compile(optimizer=Adam(learning_rate=0.001),
              loss="categorical_crossentropy",
              metrics = ['accuracy'])


model_2.fit_generator(
            train_generator,
            epochs=10,
            validation_data=validation_generator,
           )
print("Fitting Done")
```

```
Epoch 1/10
75/75 [==============================] - 94s 1s/step - loss: 0.8180 - accuracy:
0.7241 - val_loss: 0.1036 - val_accuracy: 0.9661
Epoch 2/10
75/75 [==============================] - 94s 1s/step - loss: 0.0291 - accuracy:
0.9911 - val_loss: 0.0239 - val_accuracy: 0.9964
Epoch 3/10
75/75 [==============================] - 95s 1s/step - loss: 0.0071 - accuracy:
0.9987 - val_loss: 0.0039 - val_accuracy: 0.9982
Epoch 4/10
75/75 [==============================] - 94s 1s/step - loss: 2.0548e-04 -
accuracy: 1.0000 - val_loss: 0.0022 - val_accuracy: 0.9982
Epoch 5/10
75/75 [==============================] - 94s 1s/step - loss: 1.1213e-04 -
accuracy: 1.0000 - val_loss: 0.0020 - val_accuracy: 0.9982
Epoch 6/10
75/75 [==============================] - 94s 1s/step - loss: 7.8411e-05 -
accuracy: 1.0000 - val_loss: 0.0018 - val_accuracy: 0.9982
Epoch 7/10
75/75 [==============================] - 92s 1s/step - loss: 5.8169e-05 -
accuracy: 1.0000 - val_loss: 0.0019 - val_accuracy: 0.9982
Epoch 8/10
75/75 [==============================] - 92s 1s/step - loss: 4.6619e-05 -
```

```
accuracy: 1.0000 - val_loss: 0.0017 - val_accuracy: 0.9982
Epoch 9/10
75/75 [==============================] - 93s 1s/step - loss: 3.7458e-05 -
accuracy: 1.0000 - val_loss: 0.0015 - val_accuracy: 1.0000
Epoch 10/10
75/75 [==============================] - 93s 1s/step - loss: 3.1001e-05 -
accuracy: 1.0000 - val_loss: 0.0014 - val_accuracy: 1.0000
Fitting Done
```

[27]:

```
loss_acc(model_2)
```



6.1.3 Third Model

[ ]:

```
model_3 = tf.keras.models.Sequential([
    # Note the input shape is the desired size of the image 150x150 with 3␣
  ↪bytes color
    tf.keras.layers.Conv2D(16, (3,3), activation='relu',
                           input_shape=(target_size[0], target_size[1], 3)),
    tf.keras.layers.MaxPooling2D(2,2),
    tf.keras.layers.Conv2D(32, (3,3), activation='relu'),
    tf.keras.layers.MaxPooling2D(2,2),
    tf.keras.layers.Conv2D(64, (3,3), activation='relu'),
    tf.keras.layers.MaxPooling2D(2,2),
    tf.keras.layers.Flatten(),
    tf.keras.layers.Dense(256, activation='relu'),
    tf.keras.layers.Dense(7, activation='softmax')
])
model_3.summary()
```

```
Model: "sequential_9"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_25 (Conv2D)          (None, 148, 148, 16)      448
 max_pooling2d_25 (MaxPoolin  (None, 74, 74, 16)       0
 g2D)

 conv2d_26 (Conv2D)          (None, 72, 72, 32)        4640

 max_pooling2d_26 (MaxPoolin  (None, 36, 36, 32)       0
 g2D)

 conv2d_27 (Conv2D)          (None, 34, 34, 64)        18496

 max_pooling2d_27 (MaxPoolin  (None, 17, 17, 64)       0
 g2D)

 flatten_9 (Flatten)         (None, 18496)             0

 dense_34 (Dense)            (None, 256)               4735232

 dense_35 (Dense)            (None, 7)                 1799

=================================================================
Total params: 4,760,615
Trainable params: 4,760,615
Non-trainable params: 0
_________________________________________________________________
```

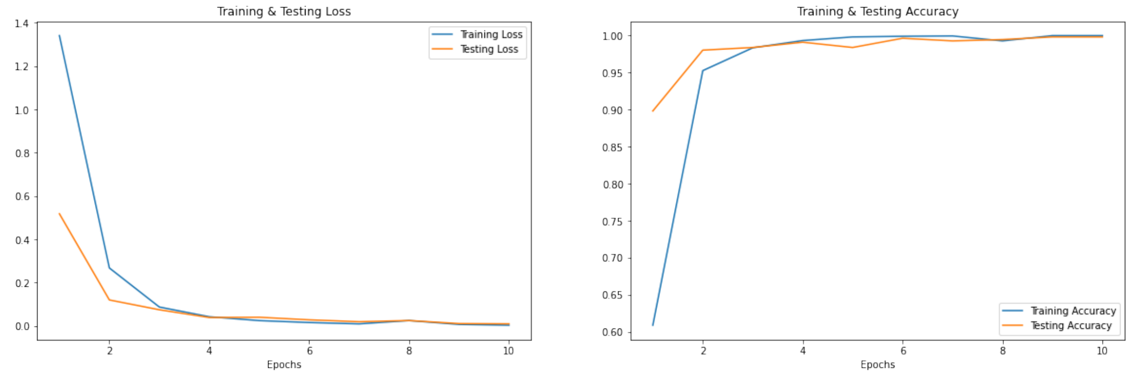
[ ]:

```
model_3.compile(optimizer=Adam(learning_rate=0.0001),
              loss="categorical_crossentropy",
              metrics = ['accuracy'])



model_3.fit_generator(
            train_generator,
            epochs=10,
            validation_data=validation_generator,
           )
print("Fitting Done")
```

```
Epoch 1/10
75/75 [==============================] - 98s 1s/step - loss: 1.3414 - accuracy:
0.6089 - val_loss: 0.5181 - val_accuracy: 0.8982
Epoch 2/10
75/75 [==============================] - 98s 1s/step - loss: 0.2679 - accuracy:
0.9527 - val_loss: 0.1200 - val_accuracy: 0.9804
Epoch 3/10
75/75 [==============================] - 99s 1s/step - loss: 0.0874 - accuracy:
0.9835 - val_loss: 0.0748 - val_accuracy: 0.9839
Epoch 4/10
```

```
75/75 [==============================] - 99s 1s/step - loss: 0.0425 - accuracy:
0.9933 - val_loss: 0.0389 - val_accuracy: 0.9911
Epoch 5/10
75/75 [==============================] - 98s 1s/step - loss: 0.0247 - accuracy:
0.9982 - val_loss: 0.0402 - val_accuracy: 0.9839
Epoch 6/10
75/75 [==============================] - 96s 1s/step - loss: 0.0158 - accuracy:
0.9991 - val_loss: 0.0282 - val_accuracy: 0.9964
Epoch 7/10
75/75 [==============================] - 96s 1s/step - loss: 0.0096 - accuracy:
0.9996 - val_loss: 0.0195 - val_accuracy: 0.9929
Epoch 8/10
75/75 [==============================] - 95s 1s/step - loss: 0.0249 - accuracy:
0.9929 - val_loss: 0.0249 - val_accuracy: 0.9946
Epoch 9/10
75/75 [==============================] - 95s 1s/step - loss: 0.0070 - accuracy:
1.0000 - val_loss: 0.0109 - val_accuracy: 0.9982
Epoch 10/10
75/75 [==============================] - 94s 1s/step - loss: 0.0034 - accuracy:
1.0000 - val_loss: 0.0098 - val_accuracy: 0.9982
Fitting Done
```

```
[ ]: loss_acc(model_3)
```



6.1.4 Forth Model

```
[28]: model_4 = tf.keras.models.Sequential([
          # Note the input shape is the desired size of the image 150x150 with 3␣
       ↪bytes color
          tf.keras.layers.Conv2D(16, (3,3), activation='relu',
                                 input_shape=(target_size[0], target_size[1], 3)),
          tf.keras.layers.MaxPooling2D(2,2),
          tf.keras.layers.Conv2D(32, (3,3), activation='relu'),
          tf.keras.layers.MaxPooling2D(2,2),
          tf.keras.layers.Conv2D(64, (3,3), activation='relu'),
          tf.keras.layers.MaxPooling2D(2,2),
          tf.keras.layers.Conv2D(128, (3,3), activation='relu'),
          tf.keras.layers.MaxPooling2D(2,2),
          tf.keras.layers.Flatten(),
          tf.keras.layers.Dense(256, activation='relu'),
          tf.keras.layers.Dense(7, activation='softmax')
      ])
      model_4.summary()
```

```
Model: "sequential_3"
-----------------------------------------------------------------
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_5 (Conv2D)           (None, 148, 148, 16)      448

 max_pooling2d_5 (MaxPooling  (None, 74, 74, 16)       0
 2D)

 conv2d_6 (Conv2D)           (None, 72, 72, 32)        4640

 max_pooling2d_6 (MaxPooling  (None, 36, 36, 32)       0
 2D)

 conv2d_7 (Conv2D)           (None, 34, 34, 64)        18496

 max_pooling2d_7 (MaxPooling  (None, 17, 17, 64)       0
 2D)

 conv2d_8 (Conv2D)           (None, 15, 15, 128)       73856

 max_pooling2d_8 (MaxPooling  (None, 7, 7, 128)        0
 2D)

 flatten_3 (Flatten)         (None, 6272)              0

 dense_6 (Dense)             (None, 256)               1605888

 dense_7 (Dense)             (None, 7)                 1799

=================================================================
Total params: 1,705,127
Trainable params: 1,705,127
Non-trainable params: 0
-----------------------------------------------------------------
```

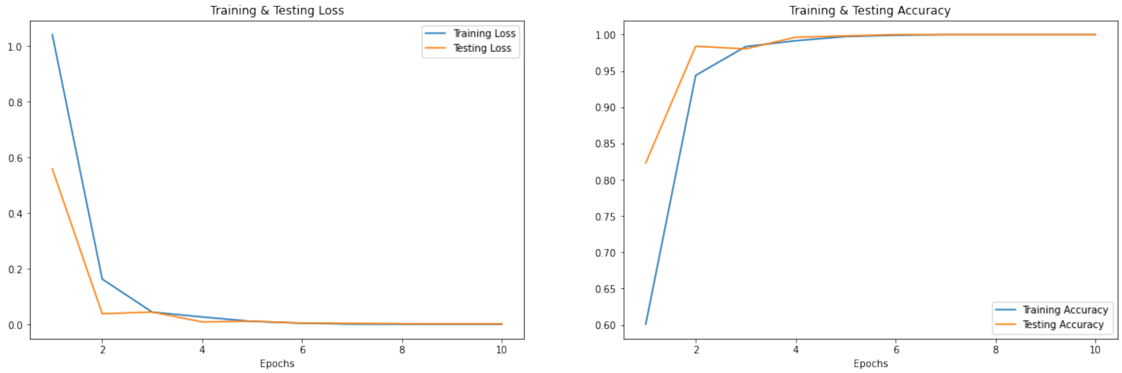
[29]:

```
model_4.compile(optimizer=Adam(learning_rate=0.001),
              loss="categorical_crossentropy",
              metrics = ['accuracy'])


model_4.fit_generator(
            train_generator,
            epochs=10,
            validation_data=validation_generator,
           )
print("Fitting Done")
```

```
Epoch 1/10
75/75 [==============================] - 98s 1s/step - loss: 1.0419 - accuracy:
0.6009 - val_loss: 0.5588 - val_accuracy: 0.8232
Epoch 2/10
75/75 [==============================] - 95s 1s/step - loss: 0.1623 - accuracy:
0.9438 - val_loss: 0.0377 - val_accuracy: 0.9839
Epoch 3/10
75/75 [==============================] - 94s 1s/step - loss: 0.0436 - accuracy:
0.9835 - val_loss: 0.0440 - val_accuracy: 0.9804
Epoch 4/10
75/75 [==============================] - 94s 1s/step - loss: 0.0261 - accuracy:
0.9915 - val_loss: 0.0082 - val_accuracy: 0.9964
Epoch 5/10
75/75 [==============================] - 94s 1s/step - loss: 0.0103 - accuracy:
0.9973 - val_loss: 0.0110 - val_accuracy: 0.9982
Epoch 6/10
75/75 [==============================] - 93s 1s/step - loss: 0.0037 - accuracy:
0.9991 - val_loss: 0.0041 - val_accuracy: 1.0000
Epoch 7/10
75/75 [==============================] - 93s 1s/step - loss: 3.0228e-04 -
accuracy: 1.0000 - val_loss: 0.0031 - val_accuracy: 1.0000
Epoch 8/10
75/75 [==============================] - 93s 1s/step - loss: 1.5405e-04 -
accuracy: 1.0000 - val_loss: 0.0019 - val_accuracy: 1.0000
Epoch 9/10
75/75 [==============================] - 95s 1s/step - loss: 9.8000e-05 -
accuracy: 1.0000 - val_loss: 0.0017 - val_accuracy: 1.0000
Epoch 10/10
75/75 [==============================] - 93s 1s/step - loss: 7.1014e-05 -
accuracy: 1.0000 - val_loss: 0.0017 - val_accuracy: 1.0000
Fitting Done
```

[30]:

```
loss_acc(model_4)
```



6.2 Fifth Model

[31]:
```python
model_5 = tf.keras.models.Sequential([
    # Note the input shape is the desired size of the image 150x150 with 3 bytes color
    tf.keras.layers.Conv2D(16, (3,3), activation='relu',
                           input_shape=(target_size[0], target_size[1], 3)),
    tf.keras.layers.MaxPooling2D(2,2),
    tf.keras.layers.Conv2D(32, (3,3), activation='relu'),
    tf.keras.layers.MaxPooling2D(2,2),
    tf.keras.layers.Conv2D(64, (3,3), activation='relu'),
    tf.keras.layers.MaxPooling2D(2,2),
    tf.keras.layers.Conv2D(128, (3,3), activation='relu'),
    tf.keras.layers.MaxPooling2D(2,2),
    tf.keras.layers.Conv2D(256, (3,3), activation='relu'),
    tf.keras.layers.MaxPooling2D(2,2),
    tf.keras.layers.Flatten(),
    tf.keras.layers.Dense(256, activation='relu'),
    tf.keras.layers.Dense(7, activation='softmax')
])
model_5.summary()
```

```
Model: "sequential_4"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_9 (Conv2D)           (None, 148, 148, 16)      448

 max_pooling2d_9 (MaxPooling  (None, 74, 74, 16)       0
 2D)

 conv2d_10 (Conv2D)          (None, 72, 72, 32)        4640

 max_pooling2d_10 (MaxPoolin  (None, 36, 36, 32)       0
 g2D)

 conv2d_11 (Conv2D)          (None, 34, 34, 64)        18496

 max_pooling2d_11 (MaxPoolin  (None, 17, 17, 64)       0
 g2D)

 conv2d_12 (Conv2D)          (None, 15, 15, 128)       73856

 max_pooling2d_12 (MaxPoolin  (None, 7, 7, 128)        0
 g2D)

 conv2d_13 (Conv2D)          (None, 5, 5, 256)         295168

 max_pooling2d_13 (MaxPoolin  (None, 2, 2, 256)        0
 g2D)

 flatten_4 (Flatten)         (None, 1024)              0

 dense_8 (Dense)             (None, 256)               262400

 dense_9 (Dense)             (None, 7)                 1799

=================================================================
Total params: 656,807
Trainable params: 656,807
Non-trainable params: 0
_________________________________________________________________
```

[32]:

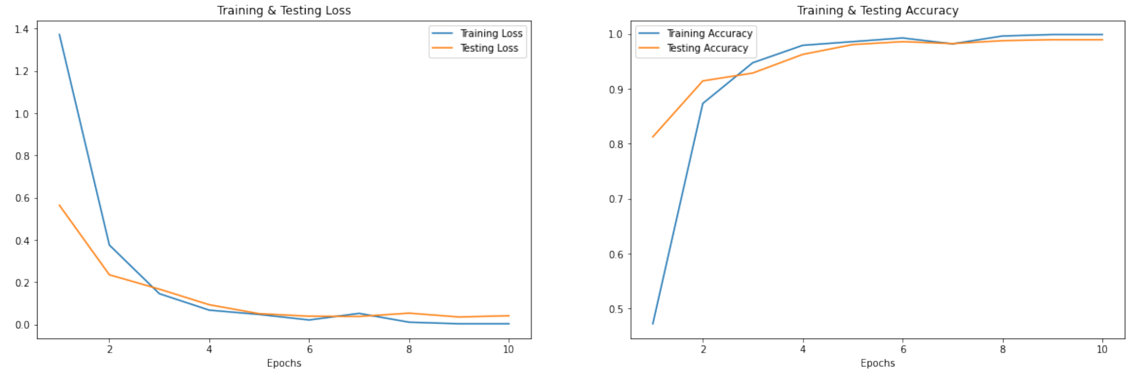
```
model_5.compile(optimizer=Adam(learning_rate=0.001),
              loss="categorical_crossentropy",
              metrics = ['accuracy'])


model_5.fit_generator(
            train_generator,
            epochs=10,
            validation_data=validation_generator,
           )
print("Fitting Done")
```

```
Epoch 1/10
75/75 [==============================] - 96s 1s/step - loss: 1.3712 - accuracy:
0.4723 - val_loss: 0.5640 - val_accuracy: 0.8125
Epoch 2/10
75/75 [==============================] - 93s 1s/step - loss: 0.3759 - accuracy:
0.8732 - val_loss: 0.2352 - val_accuracy: 0.9143
Epoch 3/10
```

```
75/75 [==============================] - 93s 1s/step - loss: 0.1460 - accuracy:
0.9473 - val_loss: 0.1675 - val_accuracy: 0.9286
Epoch 4/10
75/75 [==============================] - 94s 1s/step - loss: 0.0681 - accuracy:
0.9790 - val_loss: 0.0936 - val_accuracy: 0.9625
Epoch 5/10
75/75 [==============================] - 96s 1s/step - loss: 0.0477 - accuracy:
0.9857 - val_loss: 0.0513 - val_accuracy: 0.9804
Epoch 6/10
75/75 [==============================] - 96s 1s/step - loss: 0.0216 - accuracy:
0.9924 - val_loss: 0.0393 - val_accuracy: 0.9857
Epoch 7/10
75/75 [==============================] - 94s 1s/step - loss: 0.0527 - accuracy:
0.9817 - val_loss: 0.0382 - val_accuracy: 0.9821
Epoch 8/10
75/75 [==============================] - 93s 1s/step - loss: 0.0110 - accuracy:
0.9960 - val_loss: 0.0538 - val_accuracy: 0.9875
Epoch 9/10
75/75 [==============================] - 93s 1s/step - loss: 0.0036 - accuracy:
0.9987 - val_loss: 0.0358 - val_accuracy: 0.9893
Epoch 10/10
75/75 [==============================] - 93s 1s/step - loss: 0.0036 - accuracy:
0.9987 - val_loss: 0.0416 - val_accuracy: 0.9893
Fitting Done
```

[34]:

```
loss_acc(model_5)
```



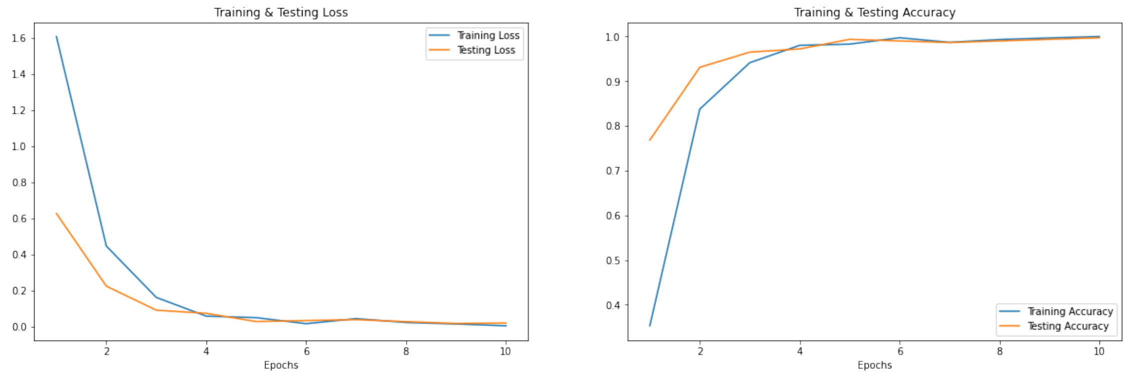
6.3 Sixth Model

[36]:

```
model_6 = tf.keras.models.Sequential([
    # Note the input shape is the desired size of the image 150x150 with 3␣
  ↪bytes color
    tf.keras.layers.Conv2D(16, (3,3), activation='relu',
                           input_shape=(target_size[0], target_size[1], 3)),
    tf.keras.layers.MaxPooling2D(2,2),
    tf.keras.layers.Conv2D(32, (3,3), activation='relu'),
    tf.keras.layers.MaxPooling2D(2,2),
    tf.keras.layers.Conv2D(64, (3,3), activation='relu'),
    tf.keras.layers.MaxPooling2D(2,2),
    tf.keras.layers.Conv2D(128, (3,3), activation='relu'),
    tf.keras.layers.MaxPooling2D(2,2),
    tf.keras.layers.Conv2D(256, (3,3), activation='relu'),
    tf.keras.layers.MaxPooling2D(2,2),
    tf.keras.layers.Flatten(),
    tf.keras.layers.Dense(256, activation='relu'),
    tf.keras.layers.Dense(128, activation='relu'),
    tf.keras.layers.Dense(7, activation='softmax')
])
model_6.summary()


model_6.compile(optimizer=Adam(learning_rate=0.001),
              loss="categorical_crossentropy",
              metrics = ['accuracy'])


model_6.fit_generator(
            train_generator,
            epochs=10,
            validation_data=validation_generator,
           )
print("Fitting Done")

loss_acc(model_6)
```

```
Model: "sequential_6"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_20 (Conv2D)          (None, 148, 148, 16)      448

 max_pooling2d_20 (MaxPoolin  (None, 74, 74, 16)       0
 g2D)

 conv2d_21 (Conv2D)          (None, 72, 72, 32)        4640

 max_pooling2d_21 (MaxPoolin  (None, 36, 36, 32)       0
 g2D)

 conv2d_22 (Conv2D)          (None, 34, 34, 64)        18496
 max_pooling2d_22 (MaxPoolin  (None, 17, 17, 64)       0
 g2D)

 conv2d_23 (Conv2D)          (None, 15, 15, 128)       73856

 max_pooling2d_23 (MaxPoolin  (None, 7, 7, 128)        0
 g2D)

 conv2d_24 (Conv2D)          (None, 5, 5, 256)         295168

 max_pooling2d_24 (MaxPoolin  (None, 2, 2, 256)        0
 g2D)

 flatten_6 (Flatten)         (None, 1024)              0

 dense_12 (Dense)            (None, 256)               262400

 dense_13 (Dense)            (None, 128)               32896

 dense_14 (Dense)            (None, 7)                 903

=================================================================
Total params: 688,807
Trainable params: 688,807
Non-trainable params: 0
_________________________________________________________________
Epoch 1/10
75/75 [==============================] - 97s 1s/step - loss: 1.6070 - accuracy:
0.3531 - val_loss: 0.6260 - val_accuracy: 0.7679
Epoch 2/10
75/75 [==============================] - 94s 1s/step - loss: 0.4452 - accuracy:
0.8371 - val_loss: 0.2231 - val_accuracy: 0.9304
Epoch 3/10
75/75 [==============================] - 93s 1s/step - loss: 0.1605 - accuracy:
0.9406 - val_loss: 0.0898 - val_accuracy: 0.9643
Epoch 4/10
75/75 [==============================] - 93s 1s/step - loss: 0.0568 - accuracy:
0.9795 - val_loss: 0.0725 - val_accuracy: 0.9714
Epoch 5/10
75/75 [==============================] - 93s 1s/step - loss: 0.0484 - accuracy:
0.9821 - val_loss: 0.0268 - val_accuracy: 0.9929
Epoch 6/10
75/75 [==============================] - 93s 1s/step - loss: 0.0150 - accuracy:
0.9964 - val_loss: 0.0323 - val_accuracy: 0.9893
Epoch 7/10
75/75 [==============================] - 92s 1s/step - loss: 0.0430 - accuracy:
0.9862 - val_loss: 0.0374 - val_accuracy: 0.9857
Epoch 8/10
```

```
75/75 [==============================] - 93s 1s/step - loss: 0.0216 - accuracy:
0.9924 - val_loss: 0.0261 - val_accuracy: 0.9893
Epoch 9/10
75/75 [==============================] - 92s 1s/step - loss: 0.0133 - accuracy:
0.9960 - val_loss: 0.0161 - val_accuracy: 0.9929
Epoch 10/10
75/75 [==============================] - 93s 1s/step - loss: 0.0034 - accuracy:
0.9991 - val_loss: 0.0183 - val_accuracy: 0.9964
Fitting Done
```



6.4 Transfer Learning

In this section, we will use transfer learning models `EfficientNetB3` and `EfficientNetB7` to train a neural network.

6.5 `EfficientNetB1`

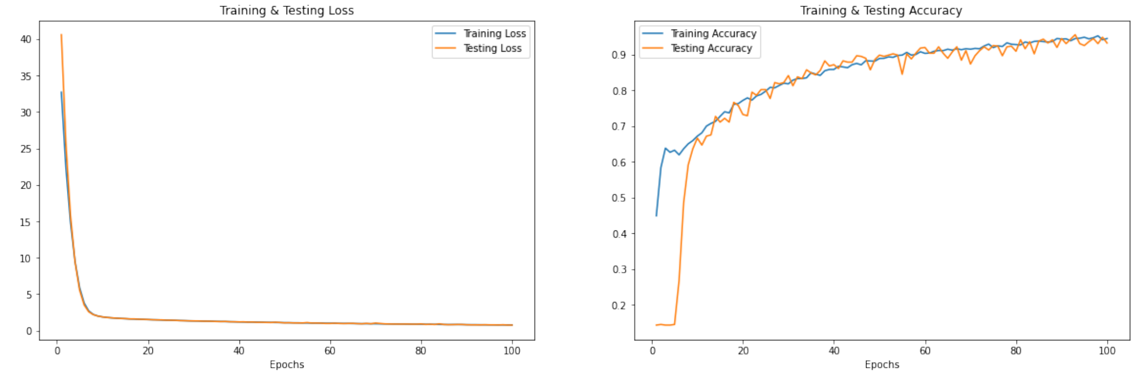
[52]:
```
# All images will be rescaled by 1./255.
datagen = ImageDataGenerator( rescale = 1.0/255, dtype= tf.float64)
batch_size = 30
target_size = (150, 150)
input_shape = (target_size[0], target_size[1], 3)



model_name='EfficientNetB1'

efficient_1=tf.keras.applications.EfficientNetB1(
                                     include_top=False,
                                     weights="imagenet",
                                     input_shape=input_shape,
                                     classes=7,
                                     classifier_activation='softmax',
                                     pooling='max')
```

```
# efficient_1.summary()
```

[58]:
```
# You should be able to divide sample_amount by batch_size
train_features_1, train_labels_1 = extract_features(train_dir, 2240, out_size =␣
 ↪1280, transfer = efficient_1)
test_features_1, test_labels_1 = extract_features(test_dir, 560, out_size =␣
 ↪1280, transfer = efficient_1)
print("DONE")

eff_model_1 = Sequential()
eff_model_1.add(tf.keras.layers.BatchNormalization(axis=-1, momentum=0.99,␣
 ↪epsilon=0.001))
eff_model_1.add(layers.Dense(512,
                       kernel_regularizer   = tf.keras.regularizers.l2(l = 0.
 ↪05),
                       activity_regularizer = tf.keras.regularizers.l1(0.05),
                       bias_regularizer     = tf.keras.regularizers.l1(0.05),
                       activation='relu',
                       input_dim=1280))
# eff_model_1.add(Dropout(0.45))
# eff_model_1.add(tf.keras.layers.BatchNormalization(axis=-1, momentum=0.99,␣
 ↪epsilon=0.001))
# eff_model_1.add(layers.Dense(256, activation='relu'))
# eff_model_1.add(Dropout(0.45))
# eff_model_1.add(layers.Dense(128, activation='relu'))
# eff_model_1.add(Dropout(0.45))

eff_model_1.add(layers.Dense(7, activation='softmax'))

eff_model_1.compile(optimizer=tf.keras.optimizers.RMSprop(lr=1e-4),
              loss='categorical_crossentropy',
              metrics=['accuracy'])

eff_model_1.fit(train_features_1, train_labels_1,
                    epochs=100,
                    batch_size=30,
                    verbose = 0,
                    validation_data=(test_features_1, test_labels_1))
print("Fitting Done")
```

```
Found 2240 images belonging to 7 classes.
Found 560 images belonging to 7 classes.
DONE
Fitting Done
```

[59]:
```
loss_acc(eff_model_1)
```



6.6 EfficientNetB2

[61]:

```
# All images will be rescaled by 1./255.
datagen = ImageDataGenerator( rescale = 1.0/255, dtype= tf.float64)
batch_size = 30
target_size = (150, 150)
input_shape = (target_size[0], target_size[1], 3)



model_name='EfficientNetB2'

efficient_2=tf.keras.applications.EfficientNetB2(
                                      include_top=False,
                                      weights="imagenet",
                                      input_shape=input_shape,
                                      classes=7,
                                      classifier_activation='softmax',
                                      pooling='max')

# efficient_2.summary()
```

[63]:

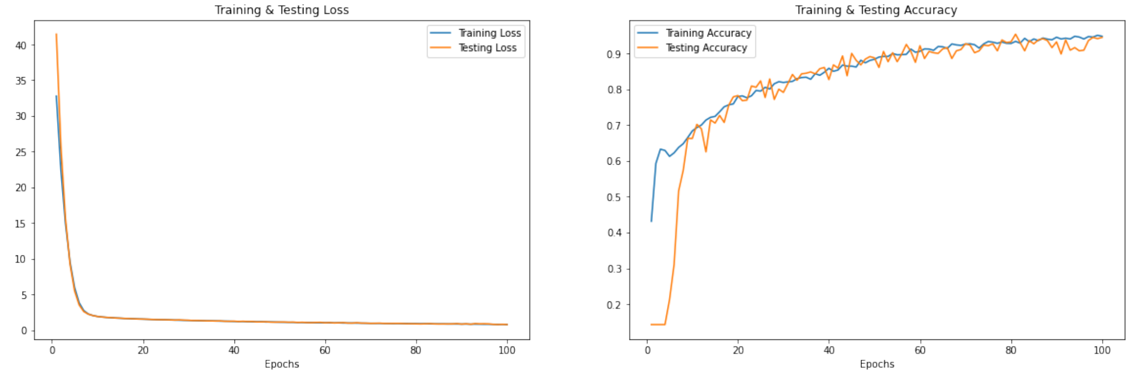
```
# You should be able to divide sample_amount by batch_size
train_features_2, train_labels_2 = extract_features(train_dir, 2240,
                                                    out_size = 1408,
                                                    transfer = efficient_2)
test_features_2, test_labels_2 = extract_features(test_dir, 560,
                                                  out_size = 1408,
                                                  transfer = efficient_2)
print("DONE")

eff_model_2 = Sequential()
```

```
eff_model_2.add(tf.keras.layers.BatchNormalization(axis=-1, momentum=0.99,␣
 ↪epsilon=0.001))
eff_model_2.add(layers.Dense(512,
                       kernel_regularizer   = tf.keras.regularizers.l2(l = 0.
 ↪05),
                       activity_regularizer = tf.keras.regularizers.l1(0.05),
                       bias_regularizer     = tf.keras.regularizers.l1(0.05),
                       activation='relu',
                       input_dim=1408))
# eff_model_2.add(Dropout(0.45))
# eff_model_2.add(tf.keras.layers.BatchNormalization(axis=-1, momentum=0.99,␣
 ↪epsilon=0.001))
# eff_model_2.add(layers.Dense(256, activation='relu'))
# eff_model_2.add(Dropout(0.45))
# eff_model_2.add(layers.Dense(128, activation='relu'))
# eff_model_2.add(Dropout(0.45))

eff_model_2.add(layers.Dense(7, activation='softmax'))

eff_model_2.compile(optimizer=tf.keras.optimizers.RMSprop(lr=1e-4),
              loss='categorical_crossentropy',
              metrics=['accuracy'])

eff_model_2.fit(train_features_1, train_labels_1,
                    epochs=100,
                    batch_size=30,
                    verbose = 0,
                    validation_data=(test_features_1, test_labels_1))
print("Fitting Done")
```

```
Found 2240 images belonging to 7 classes.
Found 560 images belonging to 7 classes.
DONE
Fitting Done
```

[64]:

```
loss_acc(eff_model_2)
```



6.6.1 EfficientNetB3

[ ]:

```
# All images will be rescaled by 1./255.
train_datagen = ImageDataGenerator( rescale = 1.0/255, dtype= tf.float64)
test_datagen  = ImageDataGenerator( rescale = 1.0/255, dtype= tf.float64)

target_size = (150, 150)
train_generator = train_datagen.flow_from_directory(train_dir,
                                                    batch_size=50,
                                                    class_mode="categorical",
                                                    target_size=target_size)
# --------------------
# Flow validation images in batches of 20 using test_datagen generator
# --------------------
validation_generator =  test_datagen.flow_from_directory(test_dir,
                                                         batch_size=25,
                                                         class_mode  =␣
 ↪"categorical",
                                                         target_size =␣
 ↪target_size)
```

```
Found 2240 images belonging to 7 classes.
Found 560 images belonging to 7 classes.
```

[ ]:

```
model_name='EfficientNetB3'
efficient_model=tf.keras.applications.EfficientNetB3(
                                      include_top=False,
                                      weights="imagenet",
                                      input_shape=(target_size[0],␣
 ↪target_size[1], 3),
                                      pooling='max')

x=efficient_model.output

x=tf.keras.layers.BatchNormalization(axis=-1,
                                  momentum=0.99,
                                  epsilon=0.001 )(x)

x = Dense(256,
          kernel_regularizer   = tf.keras.regularizers.l2(l = 0.05),
          activity_regularizer = tf.keras.regularizers.l1(0.05),
          bias_regularizer     = tf.keras.regularizers.l1(0.05),
          activation='relu')(x)
```

```
x=tf.keras.layers.Dropout(rate=.45, seed=123)(x)

output=Dense(7, activation='softmax')(x)

eff_model=tf.keras.models.Model(inputs=efficient_model.input, outputs=output)

eff_model.compile(tf.keras.optimizers.Adamax(learning_rate=.001),
              loss='categorical_crossentropy',
              metrics=['accuracy'])

# eff_model.summary()
```

[ ]:

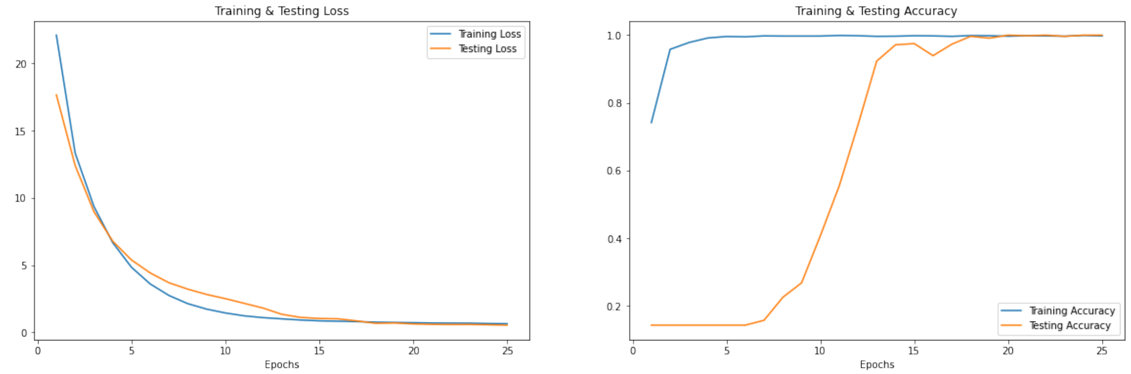
```
eff_model.fit(
            train_generator,
            epochs=25,
            validation_data=validation_generator,
           )
print("Fitting Done")
```

```
Epoch 1/25
45/45 [==============================] - 112s 2s/step - loss: 22.1103 -
accuracy: 0.7415 - val_loss: 17.6656 - val_accuracy: 0.1429
Epoch 2/25
45/45 [==============================] - 94s 2s/step - loss: 13.3407 - accuracy:
0.9580 - val_loss: 12.4169 - val_accuracy: 0.1429
Epoch 3/25
45/45 [==============================] - 94s 2s/step - loss: 9.3593 - accuracy:
0.9781 - val_loss: 8.9780 - val_accuracy: 0.1429
Epoch 4/25
45/45 [==============================] - 97s 2s/step - loss: 6.6729 - accuracy:
0.9915 - val_loss: 6.7828 - val_accuracy: 0.1429
Epoch 5/25
45/45 [==============================] - 104s 2s/step - loss: 4.8382 - accuracy:
0.9960 - val_loss: 5.3822 - val_accuracy: 0.1429
Epoch 6/25
45/45 [==============================] - 102s 2s/step - loss: 3.5955 - accuracy:
0.9951 - val_loss: 4.4163 - val_accuracy: 0.1429
Epoch 7/25
45/45 [==============================] - 98s 2s/step - loss: 2.7382 - accuracy:
0.9978 - val_loss: 3.6858 - val_accuracy: 0.1571
Epoch 8/25
45/45 [==============================] - 96s 2s/step - loss: 2.1322 - accuracy:
0.9973 - val_loss: 3.2119 - val_accuracy: 0.2250
Epoch 9/25
45/45 [==============================] - 95s 2s/step - loss: 1.7276 - accuracy:
0.9973 - val_loss: 2.8182 - val_accuracy: 0.2679
Epoch 10/25
```

```
45/45 [==============================] - 96s 2s/step - loss: 1.4420 - accuracy:
0.9973 - val_loss: 2.5049 - val_accuracy: 0.4071
Epoch 11/25
45/45 [==============================] - 97s 2s/step - loss: 1.2283 - accuracy:
0.9991 - val_loss: 2.1579 - val_accuracy: 0.5536
Epoch 12/25
45/45 [==============================] - 98s 2s/step - loss: 1.0947 - accuracy:
0.9982 - val_loss: 1.8073 - val_accuracy: 0.7339
Epoch 13/25
45/45 [==============================] - 100s 2s/step - loss: 1.0040 - accuracy:
0.9964 - val_loss: 1.3436 - val_accuracy: 0.9232
Epoch 14/25
45/45 [==============================] - 100s 2s/step - loss: 0.9166 - accuracy:
0.9969 - val_loss: 1.1130 - val_accuracy: 0.9714
Epoch 15/25
45/45 [==============================] - 100s 2s/step - loss: 0.8564 - accuracy:
0.9982 - val_loss: 1.0300 - val_accuracy: 0.9750
Epoch 16/25
45/45 [==============================] - 100s 2s/step - loss: 0.8295 - accuracy:
0.9978 - val_loss: 1.0069 - val_accuracy: 0.9393
Epoch 17/25
45/45 [==============================] - 100s 2s/step - loss: 0.7996 - accuracy:
0.9964 - val_loss: 0.8502 - val_accuracy: 0.9732
Epoch 18/25
45/45 [==============================] - 98s 2s/step - loss: 0.7569 - accuracy:
0.9987 - val_loss: 0.6719 - val_accuracy: 0.9964
Epoch 19/25
45/45 [==============================] - 99s 2s/step - loss: 0.7363 - accuracy:
0.9982 - val_loss: 0.6954 - val_accuracy: 0.9911
Epoch 20/25
45/45 [==============================] - 98s 2s/step - loss: 0.7172 - accuracy:
0.9969 - val_loss: 0.6294 - val_accuracy: 1.0000
Epoch 21/25
45/45 [==============================] - 95s 2s/step - loss: 0.6949 - accuracy:
0.9982 - val_loss: 0.5916 - val_accuracy: 0.9982
Epoch 22/25
45/45 [==============================] - 96s 2s/step - loss: 0.6884 - accuracy:
0.9978 - val_loss: 0.5821 - val_accuracy: 1.0000
Epoch 23/25
45/45 [==============================] - 97s 2s/step - loss: 0.6833 - accuracy:
0.9969 - val_loss: 0.5843 - val_accuracy: 0.9964
Epoch 24/25
45/45 [==============================] - 96s 2s/step - loss: 0.6533 - accuracy:
0.9991 - val_loss: 0.5655 - val_accuracy: 1.0000
Epoch 25/25
45/45 [==============================] - 95s 2s/step - loss: 0.6424 - accuracy:
0.9978 - val_loss: 0.5359 - val_accuracy: 1.0000
Fitting Done
```

[ ]:

```
loss_acc(eff_model)
```



6.7 EfficientNetB4

[67]:

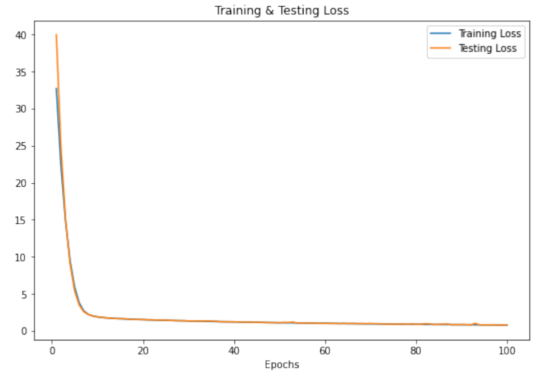
```
# All images will be rescaled by 1./255.
datagen = ImageDataGenerator( rescale = 1.0/255, dtype= tf.float64)
batch_size = 30
target_size = (150, 150)
input_shape = (target_size[0], target_size[1], 3)



model_name='EfficientNetB4'

efficient_4=tf.keras.applications.EfficientNetB4(
                                      include_top=False,
                                      weights="imagenet",
                                      input_shape=input_shape,
                                      classes=7,
                                      classifier_activation='softmax',
                                      pooling='max')

# efficient_4.summary()
```

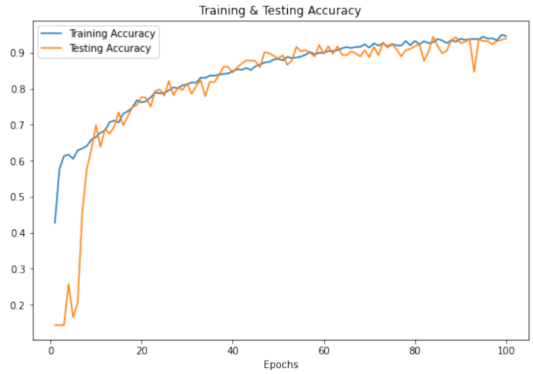
[66]:

```
out_size = 1792
train_features_4, train_labels_4 = extract_features(train_dir, 2240,
                                                    out_size = out_size,
                                                    transfer = efficient_4)
test_features_4, test_labels_4 = extract_features(test_dir, 560,
                                                  out_size = out_size,
                                                  transfer = efficient_4)
print("DONE")
```

```
eff_model_4 = Sequential()
eff_model_4.add(tf.keras.layers.BatchNormalization(axis=-1, momentum=0.99,␣
 ↪epsilon=0.001))
eff_model_4.add(layers.Dense(512,
                       kernel_regularizer   = tf.keras.regularizers.l2(l = 0.
 ↪05),
                       activity_regularizer = tf.keras.regularizers.l1(0.05),
                       bias_regularizer     = tf.keras.regularizers.l1(0.05),
                       activation='relu',
                       input_dim = out_size))
# eff_model_4.add(Dropout(0.45))
# eff_model_4.add(tf.keras.layers.BatchNormalization(axis=-1, momentum=0.99,␣
 ↪epsilon=0.001))
# eff_model_4.add(layers.Dense(256, activation='relu'))
# eff_model_4.add(Dropout(0.45))
# eff_model_4.add(layers.Dense(128, activation='relu'))
# eff_model_4.add(Dropout(0.45))

eff_model_4.add(layers.Dense(7, activation='softmax'))

eff_model_4.compile(optimizer=tf.keras.optimizers.RMSprop(lr=1e-4),
              loss='categorical_crossentropy',
              metrics=['accuracy'])

eff_model_4.fit(train_features_4, train_labels_4,
                    epochs=100,
                    batch_size=30,
                    verbose = 0,
                    validation_data=(test_features_4, test_labels_4))
print("Fitting Done")

loss_acc(eff_model_4)
```

```
Found 2240 images belonging to 7 classes.
Found 560 images belonging to 7 classes.
DONE
Fitting Done
```





6.8 EfficientNetB5

[70]:

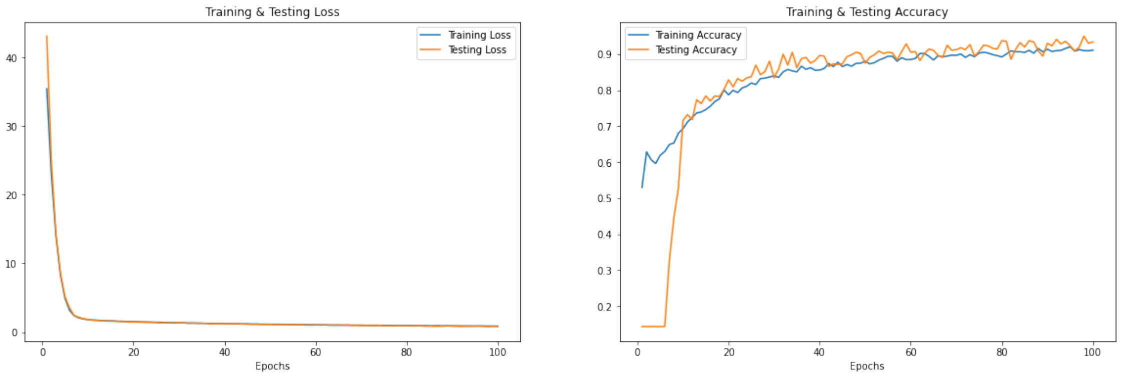
```
# All images will be rescaled by 1./255.
datagen = ImageDataGenerator( rescale = 1.0/255, dtype= tf.float64)
batch_size = 30
target_size = (150, 150)
input_shape = (target_size[0], target_size[1], 3)



model_name='EfficientNetB4'

efficient_5=tf.keras.applications.EfficientNetB5(
                                            include_top=False,
                                            weights="imagenet",
                                            input_shape=input_shape,
                                            classes=7,
                                            classifier_activation='softmax',
                                            pooling='max')

# efficient_5.summary()
```

[69]:

```
out_size = 2048
train_features_5, train_labels_5 = extract_features(train_dir, 2240,
                                                    out_size = out_size,
                                                    transfer = efficient_5)
test_features_5, test_labels_5 = extract_features(test_dir, 560,
                                                  out_size = out_size,
                                                  transfer = efficient_5)
print("DONE")

eff_model_5 = Sequential()
```

```
eff_model_5.add(tf.keras.layers.BatchNormalization(axis=-1, momentum=0.99,␣
 ↪epsilon=0.001))
eff_model_5.add(layers.Dense(512,
                       kernel_regularizer   = tf.keras.regularizers.l2(l = 0.
 ↪05),
                       activity_regularizer = tf.keras.regularizers.l1(0.05),
                       bias_regularizer     = tf.keras.regularizers.l1(0.05),
                       activation='relu',
                       input_dim = out_size))
# eff_model_5.add(Dropout(0.45))
# eff_model_5.add(tf.keras.layers.BatchNormalization(axis=-1, momentum=0.99,␣
 ↪epsilon=0.001))
# eff_model_5.add(layers.Dense(256, activation='relu'))
# eff_model_5.add(Dropout(0.45))
# eff_model_5.add(layers.Dense(128, activation='relu'))
# eff_model_5.add(Dropout(0.45))

eff_model_5.add(layers.Dense(7, activation='softmax'))

eff_model_5.compile(optimizer=tf.keras.optimizers.RMSprop(lr=1e-4),
              loss='categorical_crossentropy',
              metrics=['accuracy'])

eff_model_5.fit(train_features_5, train_labels_5,
                    epochs=100,
                    batch_size=30,
                    verbose = 0,
                    validation_data=(test_features_5, test_labels_5))
print("Fitting Done")

loss_acc(eff_model_5)
```

```
Found 2240 images belonging to 7 classes.
Found 560 images belonging to 7 classes.
DONE
Fitting Done
```



6.8.1 EfficientNetB6

[71]:
```
# All images will be rescaled by 1./255.
datagen = ImageDataGenerator( rescale = 1.0/255, dtype= tf.float64)
batch_size = 30
target_size = (150, 150)
input_shape = (target_size[0], target_size[1], 3)



model_name='EfficientNetB6'

efficient_6=tf.keras.applications.EfficientNetB6(
                                      include_top=False,
                                      weights="imagenet",
                                      input_shape=input_shape,
                                      classes=7,
                                      classifier_activation='softmax',
                                      pooling='max')

efficient_6.summary()
```

```
Downloading data from https://storage.googleapis.com/keras-
applications/efficientnetb6_notop.h5
165240832/165234480 [==============================] - 1s 0us/step
165249024/165234480 [==============================] - 1s 0us/step
Model: "efficientnetb6"

________________________________________________________________________________
__________________
 Layer (type)                   Output Shape         Param #     Connected to
================================================================================
==================
 input_10 (InputLayer)          [(None, 150, 150, 3  0           []
                                )]

 rescaling_9 (Rescaling)        (None, 150, 150, 3)  0
['input_10[0][0]']

 normalization_9 (Normalization  (None, 150, 150, 3)  7
['rescaling_9[0][0]']
 )

 stem_conv_pad (ZeroPadding2D)  (None, 151, 151, 3)  0
['normalization_9[0][0]']
```

```
 stem_conv (Conv2D)             (None, 75, 75, 56)   1512
['stem_conv_pad[0][0]']

 stem_bn (BatchNormalization)   (None, 75, 75, 56)   224
['stem_conv[0][0]']

 stem_activation (Activation)   (None, 75, 75, 56)   0
['stem_bn[0][0]']

 block1a_dwconv (DepthwiseConv2  (None, 75, 75, 56)  504
['stem_activation[0][0]']
 D)

 block1a_bn (BatchNormalization  (None, 75, 75, 56)  224
['block1a_dwconv[0][0]']
 )

 block1a_activation (Activation  (None, 75, 75, 56)  0
['block1a_bn[0][0]']
 )

 block1a_se_squeeze (GlobalAver  (None, 56)          0
['block1a_activation[0][0]']
 agePooling2D)

 block1a_se_reshape (Reshape)   (None, 1, 1, 56)     0
['block1a_se_squeeze[0][0]']

 block1a_se_reduce (Conv2D)     (None, 1, 1, 14)     798
['block1a_se_reshape[0][0]']

 block1a_se_expand (Conv2D)     (None, 1, 1, 56)     840
['block1a_se_reduce[0][0]']

 block1a_se_excite (Multiply)   (None, 75, 75, 56)   0
['block1a_activation[0][0]',
'block1a_se_expand[0][0]']

 block1a_project_conv (Conv2D)  (None, 75, 75, 32)   1792
['block1a_se_excite[0][0]']

 block1a_project_bn (BatchNorma  (None, 75, 75, 32)  128
['block1a_project_conv[0][0]']
 lization)

 block1b_dwconv (DepthwiseConv2  (None, 75, 75, 32)  288
['block1a_project_bn[0][0]']
 D)
 block1b_bn (BatchNormalization  (None, 75, 75, 32)  128
['block1b_dwconv[0][0]']
 )

 block1b_activation (Activation  (None, 75, 75, 32)  0
['block1b_bn[0][0]']
 )

 block1b_se_squeeze (GlobalAver  (None, 32)          0
['block1b_activation[0][0]']
 agePooling2D)

 block1b_se_reshape (Reshape)   (None, 1, 1, 32)     0
['block1b_se_squeeze[0][0]']

 block1b_se_reduce (Conv2D)     (None, 1, 1, 8)      264
['block1b_se_reshape[0][0]']

 block1b_se_expand (Conv2D)     (None, 1, 1, 32)     288
['block1b_se_reduce[0][0]']

 block1b_se_excite (Multiply)   (None, 75, 75, 32)   0
['block1b_activation[0][0]',
'block1b_se_expand[0][0]']

 block1b_project_conv (Conv2D)  (None, 75, 75, 32)   1024
['block1b_se_excite[0][0]']

 block1b_project_bn (BatchNorma  (None, 75, 75, 32)  128
['block1b_project_conv[0][0]']
 lization)

 block1b_drop (Dropout)         (None, 75, 75, 32)   0
['block1b_project_bn[0][0]']

 block1b_add (Add)              (None, 75, 75, 32)   0
['block1b_drop[0][0]',
'block1a_project_bn[0][0]']

 block1c_dwconv (DepthwiseConv2  (None, 75, 75, 32)  288
['block1b_add[0][0]']
 D)

 block1c_bn (BatchNormalization  (None, 75, 75, 32)  128
['block1c_dwconv[0][0]']
 )
 block1c_activation (Activation  (None, 75, 75, 32)  0
['block1c_bn[0][0]']
 )

 block1c_se_squeeze (GlobalAver  (None, 32)          0
['block1c_activation[0][0]']
 agePooling2D)

 block1c_se_reshape (Reshape)   (None, 1, 1, 32)     0
['block1c_se_squeeze[0][0]']

 block1c_se_reduce (Conv2D)     (None, 1, 1, 8)      264
['block1c_se_reshape[0][0]']

 block1c_se_expand (Conv2D)     (None, 1, 1, 32)     288
['block1c_se_reduce[0][0]']

 block1c_se_excite (Multiply)   (None, 75, 75, 32)   0
['block1c_activation[0][0]',
'block1c_se_expand[0][0]']

 block1c_project_conv (Conv2D)  (None, 75, 75, 32)   1024
['block1c_se_excite[0][0]']

 block1c_project_bn (BatchNorma  (None, 75, 75, 32)  128
['block1c_project_conv[0][0]']
 lization)

 block1c_drop (Dropout)         (None, 75, 75, 32)   0
['block1c_project_bn[0][0]']

 block1c_add (Add)              (None, 75, 75, 32)   0
['block1c_drop[0][0]',
'block1b_add[0][0]']

 block2a_expand_conv (Conv2D)   (None, 75, 75, 192)  6144
['block1c_add[0][0]']

 block2a_expand_bn (BatchNormal  (None, 75, 75, 192)  768
['block2a_expand_conv[0][0]']
 ization)

 block2a_expand_activation (Act  (None, 75, 75, 192)  0
['block2a_expand_bn[0][0]']
 ivation)

 block2a_dwconv_pad (ZeroPaddin  (None, 77, 77, 192)  0
['block2a_expand_activation[0][0]
```

```
 g2D)                                                            ']

 block2a_dwconv (DepthwiseConv2  (None, 38, 38, 192)  1728
['block2a_dwconv_pad[0][0]']
 D)

 block2a_bn (BatchNormalization  (None, 38, 38, 192)  768
['block2a_dwconv[0][0]']
 )

 block2a_activation (Activation  (None, 38, 38, 192)  0
['block2a_bn[0][0]']
 )

 block2a_se_squeeze (GlobalAver  (None, 192)         0
['block2a_activation[0][0]']
 agePooling2D)

 block2a_se_reshape (Reshape)   (None, 1, 1, 192)    0
['block2a_se_squeeze[0][0]']

 block2a_se_reduce (Conv2D)     (None, 1, 1, 8)      1544
['block2a_se_reshape[0][0]']

 block2a_se_expand (Conv2D)     (None, 1, 1, 192)    1728
['block2a_se_reduce[0][0]']

 block2a_se_excite (Multiply)   (None, 38, 38, 192)  0
['block2a_activation[0][0]',
'block2a_se_expand[0][0]']

 block2a_project_conv (Conv2D)  (None, 38, 38, 40)   7680
['block2a_se_excite[0][0]']

 block2a_project_bn (BatchNorma  (None, 38, 38, 40)  160
['block2a_project_conv[0][0]']
 lization)

 block2b_expand_conv (Conv2D)   (None, 38, 38, 240)  9600
['block2a_project_bn[0][0]']

 block2b_expand_bn (BatchNormal  (None, 38, 38, 240)  960
['block2b_expand_conv[0][0]']
 ization)

 block2b_expand_activation (Act  (None, 38, 38, 240)  0
['block2b_expand_bn[0][0]']
 ivation)
 block2b_dwconv (DepthwiseConv2  (None, 38, 38, 240)  2160
['block2b_expand_activation[0][0]
 D)                                                               ']

 block2b_bn (BatchNormalization  (None, 38, 38, 240)  960
['block2b_dwconv[0][0]']
 )

 block2b_activation (Activation  (None, 38, 38, 240)  0
['block2b_bn[0][0]']
 )

 block2b_se_squeeze (GlobalAver  (None, 240)          0
['block2b_activation[0][0]']
 agePooling2D)

 block2b_se_reshape (Reshape)   (None, 1, 1, 240)    0
['block2b_se_squeeze[0][0]']

 block2b_se_reduce (Conv2D)     (None, 1, 1, 10)     2410
['block2b_se_reshape[0][0]']

 block2b_se_expand (Conv2D)     (None, 1, 1, 240)    2640
['block2b_se_reduce[0][0]']

 block2b_se_excite (Multiply)   (None, 38, 38, 240)  0
['block2b_activation[0][0]',
'block2b_se_expand[0][0]']

 block2b_project_conv (Conv2D)  (None, 38, 38, 40)   9600
['block2b_se_excite[0][0]']

 block2b_project_bn (BatchNorma  (None, 38, 38, 40)  160
['block2b_project_conv[0][0]']
 lization)

 block2b_drop (Dropout)         (None, 38, 38, 40)   0
['block2b_project_bn[0][0]']

 block2b_add (Add)              (None, 38, 38, 40)   0
['block2b_drop[0][0]',
'block2a_project_bn[0][0]']

 block2c_expand_conv (Conv2D)   (None, 38, 38, 240)  9600
['block2b_add[0][0]']

 block2c_expand_bn (BatchNormal  (None, 38, 38, 240)  960
['block2c_expand_conv[0][0]']
 ization)

 block2c_expand_activation (Act  (None, 38, 38, 240)  0
['block2c_expand_bn[0][0]']
 ivation)

 block2c_dwconv (DepthwiseConv2  (None, 38, 38, 240)  2160
['block2c_expand_activation[0][0]
 D)                                                               ']

 block2c_bn (BatchNormalization  (None, 38, 38, 240)  960
['block2c_dwconv[0][0]']
 )

 block2c_activation (Activation  (None, 38, 38, 240)  0
['block2c_bn[0][0]']
 )

 block2c_se_squeeze (GlobalAver  (None, 240)         0
['block2c_activation[0][0]']
 agePooling2D)

 block2c_se_reshape (Reshape)   (None, 1, 1, 240)    0
['block2c_se_squeeze[0][0]']

 block2c_se_reduce (Conv2D)     (None, 1, 1, 10)     2410
['block2c_se_reshape[0][0]']

 block2c_se_expand (Conv2D)     (None, 1, 1, 240)    2640
['block2c_se_reduce[0][0]']

 block2c_se_excite (Multiply)   (None, 38, 38, 240)  0
['block2c_activation[0][0]',
'block2c_se_expand[0][0]']

 block2c_project_conv (Conv2D)  (None, 38, 38, 40)   9600
['block2c_se_excite[0][0]']

 block2c_project_bn (BatchNorma  (None, 38, 38, 40)  160
['block2c_project_conv[0][0]']
 lization)

 block2c_drop (Dropout)         (None, 38, 38, 40)   0
['block2c_project_bn[0][0]']

 block2c_add (Add)              (None, 38, 38, 40)   0
['block2c_drop[0][0]',
'block2b_add[0][0]']

 block2d_expand_conv (Conv2D)   (None, 38, 38, 240)  9600
['block2c_add[0][0]']

 block2d_expand_bn (BatchNormal  (None, 38, 38, 240)  960
['block2d_expand_conv[0][0]']
 ization)

 block2d_expand_activation (Act  (None, 38, 38, 240)  0
['block2d_expand_bn[0][0]']
 ivation)

 block2d_dwconv (DepthwiseConv2  (None, 38, 38, 240)  2160
['block2d_expand_activation[0][0]
 D)                                                              ']

 block2d_bn (BatchNormalization  (None, 38, 38, 240)  960
['block2d_dwconv[0][0]']
 )

 block2d_activation (Activation  (None, 38, 38, 240)  0
['block2d_bn[0][0]']
 )

 block2d_se_squeeze (GlobalAver  (None, 240)         0
['block2d_activation[0][0]']
 agePooling2D)

 block2d_se_reshape (Reshape)   (None, 1, 1, 240)    0
['block2d_se_squeeze[0][0]']

 block2d_se_reduce (Conv2D)     (None, 1, 1, 10)     2410
['block2d_se_reshape[0][0]']

 block2d_se_expand (Conv2D)     (None, 1, 1, 240)    2640
['block2d_se_reduce[0][0]']

 block2d_se_excite (Multiply)   (None, 38, 38, 240)  0
['block2d_activation[0][0]',
'block2d_se_expand[0][0]']

 block2d_project_conv (Conv2D)  (None, 38, 38, 40)   9600
['block2d_se_excite[0][0]']

 block2d_project_bn (BatchNorma  (None, 38, 38, 40)  160
['block2d_project_conv[0][0]']
 lization)
 block2d_drop (Dropout)         (None, 38, 38, 40)   0
['block2d_project_bn[0][0]']

 block2d_add (Add)              (None, 38, 38, 40)   0
['block2d_drop[0][0]',
'block2c_add[0][0]']

 block2e_expand_conv (Conv2D)   (None, 38, 38, 240)  9600
['block2d_add[0][0]']

 block2e_expand_bn (BatchNormal  (None, 38, 38, 240)  960
['block2e_expand_conv[0][0]']
 ization)

 block2e_expand_activation (Act  (None, 38, 38, 240)  0
['block2e_expand_bn[0][0]']
 ivation)

 block2e_dwconv (DepthwiseConv2  (None, 38, 38, 240)  2160
['block2e_expand_activation[0][0]
 D)                                                               ']

 block2e_bn (BatchNormalization  (None, 38, 38, 240)  960
['block2e_dwconv[0][0]']
 )

 block2e_activation (Activation  (None, 38, 38, 240)  0
['block2e_bn[0][0]']
 )

 block2e_se_squeeze (GlobalAver  (None, 240)         0
['block2e_activation[0][0]']
 agePooling2D)

 block2e_se_reshape (Reshape)   (None, 1, 1, 240)    0
['block2e_se_squeeze[0][0]']

 block2e_se_reduce (Conv2D)     (None, 1, 1, 10)     2410
['block2e_se_reshape[0][0]']

 block2e_se_expand (Conv2D)     (None, 1, 1, 240)    2640
['block2e_se_reduce[0][0]']

 block2e_se_excite (Multiply)   (None, 38, 38, 240)  0
['block2e_activation[0][0]',
'block2e_se_expand[0][0]']
```

```
 block2e_project_conv (Conv2D)  (None, 38, 38, 40)   9600
['block2e_se_excite[0][0]']

 block2e_project_bn (BatchNorma  (None, 38, 38, 40)  160
['block2e_project_conv[0][0]']
 lization)

 block2e_drop (Dropout)         (None, 38, 38, 40)   0
['block2e_project_bn[0][0]']

 block2e_add (Add)              (None, 38, 38, 40)   0
['block2e_drop[0][0]',
'block2d_add[0][0]']

 block2f_expand_conv (Conv2D)   (None, 38, 38, 240)  9600
['block2e_add[0][0]']

 block2f_expand_bn (BatchNormal  (None, 38, 38, 240)  960
['block2f_expand_conv[0][0]']
 ization)

 block2f_expand_activation (Act  (None, 38, 38, 240)  0
['block2f_expand_bn[0][0]']
 ivation)

 block2f_dwconv (DepthwiseConv2  (None, 38, 38, 240)  2160
['block2f_expand_activation[0][0]
 D)                                                              ']

 block2f_bn (BatchNormalization  (None, 38, 38, 240)  960
['block2f_dwconv[0][0]']
 )

 block2f_activation (Activation  (None, 38, 38, 240)  0
['block2f_bn[0][0]']
 )

 block2f_se_squeeze (GlobalAver  (None, 240)         0
['block2f_activation[0][0]']
 agePooling2D)

 block2f_se_reshape (Reshape)   (None, 1, 1, 240)    0
['block2f_se_squeeze[0][0]']

 block2f_se_reduce (Conv2D)     (None, 1, 1, 10)     2410
['block2f_se_reshape[0][0]']

 block2f_se_expand (Conv2D)     (None, 1, 1, 240)    2640
```

```
['block2f_se_reduce[0][0]']

 block2f_se_excite (Multiply)   (None, 38, 38, 240)  0
['block2f_activation[0][0]',
'block2f_se_expand[0][0]']

 block2f_project_conv (Conv2D)  (None, 38, 38, 40)   9600
['block2f_se_excite[0][0]']

 block2f_project_bn (BatchNorma  (None, 38, 38, 40)  160
['block2f_project_conv[0][0]']
 lization)

 block2f_drop (Dropout)         (None, 38, 38, 40)   0
['block2f_project_bn[0][0]']

 block2f_add (Add)              (None, 38, 38, 40)   0
['block2f_drop[0][0]',
'block2e_add[0][0]']

 block3a_expand_conv (Conv2D)   (None, 38, 38, 240)  9600
['block2f_add[0][0]']

 block3a_expand_bn (BatchNormal  (None, 38, 38, 240)  960
['block3a_expand_conv[0][0]']
 ization)

 block3a_expand_activation (Act  (None, 38, 38, 240)  0
['block3a_expand_bn[0][0]']
 ivation)

 block3a_dwconv_pad (ZeroPaddin  (None, 41, 41, 240)  0
['block3a_expand_activation[0][0]
 g2D)                                                            ']

 block3a_dwconv (DepthwiseConv2  (None, 19, 19, 240)  6000
['block3a_dwconv_pad[0][0]']
 D)

 block3a_bn (BatchNormalization  (None, 19, 19, 240)  960
['block3a_dwconv[0][0]']
 )

 block3a_activation (Activation  (None, 19, 19, 240)  0
['block3a_bn[0][0]']
 )

 block3a_se_squeeze (GlobalAver  (None, 240)         0
['block3a_activation[0][0]']
 agePooling2D)

 block3a_se_reshape (Reshape)   (None, 1, 1, 240)    0
['block3a_se_squeeze[0][0]']

 block3a_se_reduce (Conv2D)     (None, 1, 1, 10)     2410
['block3a_se_reshape[0][0]']

 block3a_se_expand (Conv2D)     (None, 1, 1, 240)    2640
['block3a_se_reduce[0][0]']

 block3a_se_excite (Multiply)   (None, 19, 19, 240)  0
['block3a_activation[0][0]',
'block3a_se_expand[0][0]']

 block3a_project_conv (Conv2D)  (None, 19, 19, 72)   17280
['block3a_se_excite[0][0]']

 block3a_project_bn (BatchNorma  (None, 19, 19, 72)  288
['block3a_project_conv[0][0]']
 lization)

 block3b_expand_conv (Conv2D)   (None, 19, 19, 432)  31104
['block3a_project_bn[0][0]']

 block3b_expand_bn (BatchNormal  (None, 19, 19, 432)  1728
['block3b_expand_conv[0][0]']
 ization)

 block3b_expand_activation (Act  (None, 19, 19, 432)  0
['block3b_expand_bn[0][0]']
 ivation)

 block3b_dwconv (DepthwiseConv2  (None, 19, 19, 432)  10800
['block3b_expand_activation[0][0]
 D)                                                               ']

 block3b_bn (BatchNormalization  (None, 19, 19, 432)  1728
['block3b_dwconv[0][0]']
 )

 block3b_activation (Activation  (None, 19, 19, 432)  0
['block3b_bn[0][0]']
 )

 block3b_se_squeeze (GlobalAver  (None, 432)         0
['block3b_activation[0][0]']
```

```
 agePooling2D)

 block3b_se_reshape (Reshape)   (None, 1, 1, 432)    0
['block3b_se_squeeze[0][0]']

 block3b_se_reduce (Conv2D)     (None, 1, 1, 18)     7794
['block3b_se_reshape[0][0]']

 block3b_se_expand (Conv2D)     (None, 1, 1, 432)    8208
['block3b_se_reduce[0][0]']

 block3b_se_excite (Multiply)   (None, 19, 19, 432)  0
['block3b_activation[0][0]',
'block3b_se_expand[0][0]']

 block3b_project_conv (Conv2D)  (None, 19, 19, 72)   31104
['block3b_se_excite[0][0]']

 block3b_project_bn (BatchNorma  (None, 19, 19, 72)  288
['block3b_project_conv[0][0]']
 lization)

 block3b_drop (Dropout)         (None, 19, 19, 72)   0
['block3b_project_bn[0][0]']

 block3b_add (Add)              (None, 19, 19, 72)   0
['block3b_drop[0][0]',
'block3a_project_bn[0][0]']

 block3c_expand_conv (Conv2D)   (None, 19, 19, 432)  31104
['block3b_add[0][0]']

 block3c_expand_bn (BatchNormal  (None, 19, 19, 432)  1728
['block3c_expand_conv[0][0]']
 ization)

 block3c_expand_activation (Act  (None, 19, 19, 432)  0
['block3c_expand_bn[0][0]']
 ivation)

 block3c_dwconv (DepthwiseConv2  (None, 19, 19, 432)  10800
['block3c_expand_activation[0][0]
 D)                                                              ']

 block3c_bn (BatchNormalization  (None, 19, 19, 432)  1728
['block3c_dwconv[0][0]']
 )
 block3c_activation (Activation  (None, 19, 19, 432)  0
['block3c_bn[0][0]']
 )

 block3c_se_squeeze (GlobalAver  (None, 432)         0
['block3c_activation[0][0]']
 agePooling2D)

 block3c_se_reshape (Reshape)   (None, 1, 1, 432)    0
['block3c_se_squeeze[0][0]']

 block3c_se_reduce (Conv2D)     (None, 1, 1, 18)     7794
['block3c_se_reshape[0][0]']

 block3c_se_expand (Conv2D)     (None, 1, 1, 432)    8208
['block3c_se_reduce[0][0]']

 block3c_se_excite (Multiply)   (None, 19, 19, 432)  0
['block3c_activation[0][0]',
'block3c_se_expand[0][0]']

 block3c_project_conv (Conv2D)  (None, 19, 19, 72)   31104
['block3c_se_excite[0][0]']

 block3c_project_bn (BatchNorma  (None, 19, 19, 72)  288
['block3c_project_conv[0][0]']
 lization)

 block3c_drop (Dropout)         (None, 19, 19, 72)   0
['block3c_project_bn[0][0]']

 block3c_add (Add)              (None, 19, 19, 72)   0
['block3c_drop[0][0]',
'block3b_add[0][0]']

 block3d_expand_conv (Conv2D)   (None, 19, 19, 432)  31104
['block3c_add[0][0]']

 block3d_expand_bn (BatchNormal  (None, 19, 19, 432)  1728
['block3d_expand_conv[0][0]']
 ization)

 block3d_expand_activation (Act  (None, 19, 19, 432)  0
['block3d_expand_bn[0][0]']
 ivation)

 block3d_dwconv (DepthwiseConv2  (None, 19, 19, 432)  10800
['block3d_expand_activation[0][0]
```

```
 D)                                                              ']

 block3d_bn (BatchNormalization  (None, 19, 19, 432)  1728
['block3d_dwconv[0][0]']
 )

 block3d_activation (Activation  (None, 19, 19, 432)  0
['block3d_bn[0][0]']
 )

 block3d_se_squeeze (GlobalAver  (None, 432)         0
['block3d_activation[0][0]']
 agePooling2D)

 block3d_se_reshape (Reshape)   (None, 1, 1, 432)    0
['block3d_se_squeeze[0][0]']

 block3d_se_reduce (Conv2D)     (None, 1, 1, 18)     7794
['block3d_se_reshape[0][0]']

 block3d_se_expand (Conv2D)     (None, 1, 1, 432)    8208
['block3d_se_reduce[0][0]']

 block3d_se_excite (Multiply)   (None, 19, 19, 432)  0
['block3d_activation[0][0]',
'block3d_se_expand[0][0]']

 block3d_project_conv (Conv2D)  (None, 19, 19, 72)   31104
['block3d_se_excite[0][0]']

 block3d_project_bn (BatchNorma  (None, 19, 19, 72)  288
['block3d_project_conv[0][0]']
 lization)

 block3d_drop (Dropout)         (None, 19, 19, 72)   0
['block3d_project_bn[0][0]']

 block3d_add (Add)              (None, 19, 19, 72)   0
['block3d_drop[0][0]',
'block3c_add[0][0]']

 block3e_expand_conv (Conv2D)   (None, 19, 19, 432)  31104
['block3d_add[0][0]']

 block3e_expand_bn (BatchNormal  (None, 19, 19, 432)  1728
['block3e_expand_conv[0][0]']
 ization)
 block3e_expand_activation (Act  (None, 19, 19, 432)  0
['block3e_expand_bn[0][0]']
 ivation)

 block3e_dwconv (DepthwiseConv2  (None, 19, 19, 432)  10800
['block3e_expand_activation[0][0]
 D)                                                              ']

 block3e_bn (BatchNormalization  (None, 19, 19, 432)  1728
['block3e_dwconv[0][0]']
 )

 block3e_activation (Activation  (None, 19, 19, 432)  0
['block3e_bn[0][0]']
 )

 block3e_se_squeeze (GlobalAver  (None, 432)         0
['block3e_activation[0][0]']
 agePooling2D)

 block3e_se_reshape (Reshape)   (None, 1, 1, 432)    0
['block3e_se_squeeze[0][0]']

 block3e_se_reduce (Conv2D)     (None, 1, 1, 18)     7794
['block3e_se_reshape[0][0]']

 block3e_se_expand (Conv2D)     (None, 1, 1, 432)    8208
['block3e_se_reduce[0][0]']

 block3e_se_excite (Multiply)   (None, 19, 19, 432)  0
['block3e_activation[0][0]',
'block3e_se_expand[0][0]']

 block3e_project_conv (Conv2D)  (None, 19, 19, 72)   31104
['block3e_se_excite[0][0]']

 block3e_project_bn (BatchNorma  (None, 19, 19, 72)  288
['block3e_project_conv[0][0]']
 lization)

 block3e_drop (Dropout)         (None, 19, 19, 72)   0
['block3e_project_bn[0][0]']

 block3e_add (Add)              (None, 19, 19, 72)   0
['block3e_drop[0][0]',
'block3d_add[0][0]']

 block3f_expand_conv (Conv2D)   (None, 19, 19, 432)  31104
```

```
['block3e_add[0][0]']

 block3f_expand_bn (BatchNormal  (None, 19, 19, 432)  1728
['block3f_expand_conv[0][0]']
 ization)

 block3f_expand_activation (Act  (None, 19, 19, 432)  0
['block3f_expand_bn[0][0]']
 ivation)

 block3f_dwconv (DepthwiseConv2  (None, 19, 19, 432)  10800
['block3f_expand_activation[0][0]
 D)                                                              ']

 block3f_bn (BatchNormalization  (None, 19, 19, 432)  1728
['block3f_dwconv[0][0]']
 )

 block3f_activation (Activation  (None, 19, 19, 432)  0
['block3f_bn[0][0]']
 )

 block3f_se_squeeze (GlobalAver  (None, 432)         0
['block3f_activation[0][0]']
 agePooling2D)

 block3f_se_reshape (Reshape)   (None, 1, 1, 432)    0
['block3f_se_squeeze[0][0]']

 block3f_se_reduce (Conv2D)     (None, 1, 1, 18)     7794
['block3f_se_reshape[0][0]']

 block3f_se_expand (Conv2D)     (None, 1, 1, 432)    8208
['block3f_se_reduce[0][0]']

 block3f_se_excite (Multiply)   (None, 19, 19, 432)  0
['block3f_activation[0][0]',
'block3f_se_expand[0][0]']

 block3f_project_conv (Conv2D)  (None, 19, 19, 72)   31104
['block3f_se_excite[0][0]']

 block3f_project_bn (BatchNorma  (None, 19, 19, 72)  288
['block3f_project_conv[0][0]']
 lization)

 block3f_drop (Dropout)         (None, 19, 19, 72)   0
['block3f_project_bn[0][0]']
```

```
 block3f_add (Add)              (None, 19, 19, 72)   0
['block3f_drop[0][0]',
'block3e_add[0][0]']

 block4a_expand_conv (Conv2D)   (None, 19, 19, 432)  31104
['block3f_add[0][0]']

 block4a_expand_bn (BatchNormal  (None, 19, 19, 432)  1728
['block4a_expand_conv[0][0]']
 ization)

 block4a_expand_activation (Act  (None, 19, 19, 432)  0
['block4a_expand_bn[0][0]']
 ivation)

 block4a_dwconv_pad (ZeroPaddin  (None, 21, 21, 432)  0
['block4a_expand_activation[0][0]
 g2D)                                                            ']

 block4a_dwconv (DepthwiseConv2  (None, 10, 10, 432)  3888
['block4a_dwconv_pad[0][0]']
 D)

 block4a_bn (BatchNormalization  (None, 10, 10, 432)  1728
['block4a_dwconv[0][0]']
 )

 block4a_activation (Activation  (None, 10, 10, 432)  0
['block4a_bn[0][0]']
 )

 block4a_se_squeeze (GlobalAver  (None, 432)         0
['block4a_activation[0][0]']
 agePooling2D)

 block4a_se_reshape (Reshape)   (None, 1, 1, 432)    0
['block4a_se_squeeze[0][0]']

 block4a_se_reduce (Conv2D)     (None, 1, 1, 18)     7794
['block4a_se_reshape[0][0]']

 block4a_se_expand (Conv2D)     (None, 1, 1, 432)    8208
['block4a_se_reduce[0][0]']

 block4a_se_excite (Multiply)   (None, 10, 10, 432)  0
['block4a_activation[0][0]',
'block4a_se_expand[0][0]']
```

```
 block4a_project_conv (Conv2D)  (None, 10, 10, 144)  62208
['block4a_se_excite[0][0]']

 block4a_project_bn (BatchNorma  (None, 10, 10, 144)  576
['block4a_project_conv[0][0]']
 lization)

 block4b_expand_conv (Conv2D)   (None, 10, 10, 864)  124416
['block4a_project_bn[0][0]']

 block4b_expand_bn (BatchNormal  (None, 10, 10, 864)  3456
['block4b_expand_conv[0][0]']
 ization)

 block4b_expand_activation (Act  (None, 10, 10, 864)  0
['block4b_expand_bn[0][0]']
 ivation)

 block4b_dwconv (DepthwiseConv2  (None, 10, 10, 864)  7776
['block4b_expand_activation[0][0]
 D)                                                              ']

 block4b_bn (BatchNormalization  (None, 10, 10, 864)  3456
['block4b_dwconv[0][0]']
 )

 block4b_activation (Activation  (None, 10, 10, 864)  0
['block4b_bn[0][0]']
 )

 block4b_se_squeeze (GlobalAver  (None, 864)         0
['block4b_activation[0][0]']
 agePooling2D)

 block4b_se_reshape (Reshape)   (None, 1, 1, 864)    0
['block4b_se_squeeze[0][0]']

 block4b_se_reduce (Conv2D)     (None, 1, 1, 36)     31140
['block4b_se_reshape[0][0]']

 block4b_se_expand (Conv2D)     (None, 1, 1, 864)    31968
['block4b_se_reduce[0][0]']

 block4b_se_excite (Multiply)   (None, 10, 10, 864)  0
['block4b_activation[0][0]',
'block4b_se_expand[0][0]']
```

```
 block4b_project_conv (Conv2D)  (None, 10, 10, 144)  124416
['block4b_se_excite[0][0]']

 block4b_project_bn (BatchNorma  (None, 10, 10, 144)  576
['block4b_project_conv[0][0]']
 lization)

 block4b_drop (Dropout)         (None, 10, 10, 144)  0
['block4b_project_bn[0][0]']

 block4b_add (Add)              (None, 10, 10, 144)  0
['block4b_drop[0][0]',
'block4a_project_bn[0][0]']

 block4c_expand_conv (Conv2D)   (None, 10, 10, 864)  124416
['block4b_add[0][0]']

 block4c_expand_bn (BatchNormal  (None, 10, 10, 864)  3456
['block4c_expand_conv[0][0]']
 ization)

 block4c_expand_activation (Act  (None, 10, 10, 864)  0
['block4c_expand_bn[0][0]']
 ivation)

 block4c_dwconv (DepthwiseConv2  (None, 10, 10, 864)  7776
['block4c_expand_activation[0][0]
 D)                                                              ']

 block4c_bn (BatchNormalization  (None, 10, 10, 864)  3456
['block4c_dwconv[0][0]']
 )

 block4c_activation (Activation  (None, 10, 10, 864)  0
['block4c_bn[0][0]']
 )

 block4c_se_squeeze (GlobalAver  (None, 864)         0
['block4c_activation[0][0]']
 agePooling2D)

 block4c_se_reshape (Reshape)   (None, 1, 1, 864)    0
['block4c_se_squeeze[0][0]']

 block4c_se_reduce (Conv2D)     (None, 1, 1, 36)     31140
['block4c_se_reshape[0][0]']

 block4c_se_expand (Conv2D)     (None, 1, 1, 864)    31968
```

```
['block4c_se_reduce[0][0]']

 block4c_se_excite (Multiply)   (None, 10, 10, 864)  0
['block4c_activation[0][0]',
'block4c_se_expand[0][0]']

 block4c_project_conv (Conv2D)  (None, 10, 10, 144)  124416
['block4c_se_excite[0][0]']

 block4c_project_bn (BatchNorma  (None, 10, 10, 144)  576
['block4c_project_conv[0][0]']
 lization)

 block4c_drop (Dropout)         (None, 10, 10, 144)  0
['block4c_project_bn[0][0]']

 block4c_add (Add)              (None, 10, 10, 144)  0
['block4c_drop[0][0]',
'block4b_add[0][0]']

 block4d_expand_conv (Conv2D)   (None, 10, 10, 864)  124416
['block4c_add[0][0]']

 block4d_expand_bn (BatchNormal  (None, 10, 10, 864)  3456
['block4d_expand_conv[0][0]']
 ization)

 block4d_expand_activation (Act  (None, 10, 10, 864)  0
['block4d_expand_bn[0][0]']
 ivation)

 block4d_dwconv (DepthwiseConv2  (None, 10, 10, 864)  7776
['block4d_expand_activation[0][0]
 D)                                                              ']

 block4d_bn (BatchNormalization  (None, 10, 10, 864)  3456
['block4d_dwconv[0][0]']
 )

 block4d_activation (Activation  (None, 10, 10, 864)  0
['block4d_bn[0][0]']
 )

 block4d_se_squeeze (GlobalAver  (None, 864)         0
['block4d_activation[0][0]']
 agePooling2D)

 block4d_se_reshape (Reshape)   (None, 1, 1, 864)    0
```

```
['block4d_se_squeeze[0][0]']

 block4d_se_reduce (Conv2D)     (None, 1, 1, 36)     31140
['block4d_se_reshape[0][0]']

 block4d_se_expand (Conv2D)     (None, 1, 1, 864)    31968
['block4d_se_reduce[0][0]']

 block4d_se_excite (Multiply)   (None, 10, 10, 864)  0
['block4d_activation[0][0]',
'block4d_se_expand[0][0]']

 block4d_project_conv (Conv2D)  (None, 10, 10, 144)  124416
['block4d_se_excite[0][0]']

 block4d_project_bn (BatchNorma  (None, 10, 10, 144)  576
['block4d_project_conv[0][0]']
 lization)

 block4d_drop (Dropout)         (None, 10, 10, 144)  0
['block4d_project_bn[0][0]']

 block4d_add (Add)              (None, 10, 10, 144)  0
['block4d_drop[0][0]',
'block4c_add[0][0]']

 block4e_expand_conv (Conv2D)   (None, 10, 10, 864)  124416
['block4d_add[0][0]']

 block4e_expand_bn (BatchNormal  (None, 10, 10, 864)  3456
['block4e_expand_conv[0][0]']
 ization)

 block4e_expand_activation (Act  (None, 10, 10, 864)  0
['block4e_expand_bn[0][0]']
 ivation)

 block4e_dwconv (DepthwiseConv2  (None, 10, 10, 864)  7776
['block4e_expand_activation[0][0]
 D)                                                               ']

 block4e_bn (BatchNormalization  (None, 10, 10, 864)  3456
['block4e_dwconv[0][0]']
 )

 block4e_activation (Activation  (None, 10, 10, 864)  0
['block4e_bn[0][0]']
 )
 block4e_se_squeeze (GlobalAver  (None, 864)         0
['block4e_activation[0][0]']
 agePooling2D)

 block4e_se_reshape (Reshape)   (None, 1, 1, 864)    0
['block4e_se_squeeze[0][0]']

 block4e_se_reduce (Conv2D)     (None, 1, 1, 36)     31140
['block4e_se_reshape[0][0]']

 block4e_se_expand (Conv2D)     (None, 1, 1, 864)    31968
['block4e_se_reduce[0][0]']

 block4e_se_excite (Multiply)   (None, 10, 10, 864)  0
['block4e_activation[0][0]',
'block4e_se_expand[0][0]']

 block4e_project_conv (Conv2D)  (None, 10, 10, 144)  124416
['block4e_se_excite[0][0]']

 block4e_project_bn (BatchNorma  (None, 10, 10, 144)  576
['block4e_project_conv[0][0]']
 lization)

 block4e_drop (Dropout)         (None, 10, 10, 144)  0
['block4e_project_bn[0][0]']

 block4e_add (Add)              (None, 10, 10, 144)  0
['block4e_drop[0][0]',
'block4d_add[0][0]']

 block4f_expand_conv (Conv2D)   (None, 10, 10, 864)  124416
['block4e_add[0][0]']

 block4f_expand_bn (BatchNormal  (None, 10, 10, 864)  3456
['block4f_expand_conv[0][0]']
 ization)

 block4f_expand_activation (Act  (None, 10, 10, 864)  0
['block4f_expand_bn[0][0]']
 ivation)

 block4f_dwconv (DepthwiseConv2  (None, 10, 10, 864)  7776
['block4f_expand_activation[0][0]
 D)                                                              ']

 block4f_bn (BatchNormalization  (None, 10, 10, 864)  3456
['block4f_dwconv[0][0]']
 )

 block4f_activation (Activation  (None, 10, 10, 864)  0
['block4f_bn[0][0]']
 )

 block4f_se_squeeze (GlobalAver  (None, 864)         0
['block4f_activation[0][0]']
 agePooling2D)

 block4f_se_reshape (Reshape)   (None, 1, 1, 864)    0
['block4f_se_squeeze[0][0]']

 block4f_se_reduce (Conv2D)     (None, 1, 1, 36)     31140
['block4f_se_reshape[0][0]']

 block4f_se_expand (Conv2D)     (None, 1, 1, 864)    31968
['block4f_se_reduce[0][0]']

 block4f_se_excite (Multiply)   (None, 10, 10, 864)  0
['block4f_activation[0][0]',
'block4f_se_expand[0][0]']

 block4f_project_conv (Conv2D)  (None, 10, 10, 144)  124416
['block4f_se_excite[0][0]']

 block4f_project_bn (BatchNorma  (None, 10, 10, 144)  576
['block4f_project_conv[0][0]']
 lization)

 block4f_drop (Dropout)         (None, 10, 10, 144)  0
['block4f_project_bn[0][0]']

 block4f_add (Add)              (None, 10, 10, 144)  0
['block4f_drop[0][0]',
'block4e_add[0][0]']

 block4g_expand_conv (Conv2D)   (None, 10, 10, 864)  124416
['block4f_add[0][0]']

 block4g_expand_bn (BatchNormal  (None, 10, 10, 864)  3456
['block4g_expand_conv[0][0]']
 ization)

 block4g_expand_activation (Act  (None, 10, 10, 864)  0
['block4g_expand_bn[0][0]']
 ivation)
 block4g_dwconv (DepthwiseConv2  (None, 10, 10, 864)  7776
['block4g_expand_activation[0][0]
 D)                                                              ']

 block4g_bn (BatchNormalization  (None, 10, 10, 864)  3456
['block4g_dwconv[0][0]']
 )

 block4g_activation (Activation  (None, 10, 10, 864)  0
['block4g_bn[0][0]']
 )

 block4g_se_squeeze (GlobalAver  (None, 864)         0
['block4g_activation[0][0]']
 agePooling2D)

 block4g_se_reshape (Reshape)   (None, 1, 1, 864)    0
['block4g_se_squeeze[0][0]']

 block4g_se_reduce (Conv2D)     (None, 1, 1, 36)     31140
['block4g_se_reshape[0][0]']

 block4g_se_expand (Conv2D)     (None, 1, 1, 864)    31968
['block4g_se_reduce[0][0]']

 block4g_se_excite (Multiply)   (None, 10, 10, 864)  0
['block4g_activation[0][0]',
'block4g_se_expand[0][0]']

 block4g_project_conv (Conv2D)  (None, 10, 10, 144)  124416
['block4g_se_excite[0][0]']

 block4g_project_bn (BatchNorma  (None, 10, 10, 144)  576
['block4g_project_conv[0][0]']
 lization)

 block4g_drop (Dropout)         (None, 10, 10, 144)  0
['block4g_project_bn[0][0]']

 block4g_add (Add)              (None, 10, 10, 144)  0
['block4g_drop[0][0]',
'block4f_add[0][0]']

 block4h_expand_conv (Conv2D)   (None, 10, 10, 864)  124416
['block4g_add[0][0]']

 block4h_expand_bn (BatchNormal  (None, 10, 10, 864)  3456
['block4h_expand_conv[0][0]']
 ization)

 block4h_expand_activation (Act  (None, 10, 10, 864)  0
['block4h_expand_bn[0][0]']
 ivation)

 block4h_dwconv (DepthwiseConv2  (None, 10, 10, 864)  7776
['block4h_expand_activation[0][0]
 D)                                                               ']

 block4h_bn (BatchNormalization  (None, 10, 10, 864)  3456
['block4h_dwconv[0][0]']
 )

 block4h_activation (Activation  (None, 10, 10, 864)  0
['block4h_bn[0][0]']
 )

 block4h_se_squeeze (GlobalAver  (None, 864)         0
['block4h_activation[0][0]']
 agePooling2D)

 block4h_se_reshape (Reshape)   (None, 1, 1, 864)    0
['block4h_se_squeeze[0][0]']

 block4h_se_reduce (Conv2D)     (None, 1, 1, 36)     31140
['block4h_se_reshape[0][0]']

 block4h_se_expand (Conv2D)     (None, 1, 1, 864)    31968
['block4h_se_reduce[0][0]']

 block4h_se_excite (Multiply)   (None, 10, 10, 864)  0
['block4h_activation[0][0]',
'block4h_se_expand[0][0]']

 block4h_project_conv (Conv2D)  (None, 10, 10, 144)  124416
['block4h_se_excite[0][0]']

 block4h_project_bn (BatchNorma  (None, 10, 10, 144)  576
['block4h_project_conv[0][0]']
 lization)

 block4h_drop (Dropout)         (None, 10, 10, 144)  0
['block4h_project_bn[0][0]']

 block4h_add (Add)              (None, 10, 10, 144)  0
['block4h_drop[0][0]',
'block4g_add[0][0]']

 block5a_expand_conv (Conv2D)   (None, 10, 10, 864)  124416
['block4h_add[0][0]']

 block5a_expand_bn (BatchNormal  (None, 10, 10, 864)  3456
['block5a_expand_conv[0][0]']
 ization)

 block5a_expand_activation (Act  (None, 10, 10, 864)  0
['block5a_expand_bn[0][0]']
 ivation)

 block5a_dwconv (DepthwiseConv2  (None, 10, 10, 864)  21600
['block5a_expand_activation[0][0]
 D)                                                              ']

 block5a_bn (BatchNormalization  (None, 10, 10, 864)  3456
['block5a_dwconv[0][0]']
 )

 block5a_activation (Activation  (None, 10, 10, 864)  0
['block5a_bn[0][0]']
 )

 block5a_se_squeeze (GlobalAver  (None, 864)         0
['block5a_activation[0][0]']
 agePooling2D)

 block5a_se_reshape (Reshape)   (None, 1, 1, 864)    0
['block5a_se_squeeze[0][0]']

 block5a_se_reduce (Conv2D)     (None, 1, 1, 36)     31140
['block5a_se_reshape[0][0]']

 block5a_se_expand (Conv2D)     (None, 1, 1, 864)    31968
['block5a_se_reduce[0][0]']

 block5a_se_excite (Multiply)   (None, 10, 10, 864)  0
['block5a_activation[0][0]',
'block5a_se_expand[0][0]']

 block5a_project_conv (Conv2D)  (None, 10, 10, 200)  172800
['block5a_se_excite[0][0]']

 block5a_project_bn (BatchNorma  (None, 10, 10, 200)  800
['block5a_project_conv[0][0]']
 lization)
 block5b_expand_conv (Conv2D)   (None, 10, 10, 1200  240000
['block5a_project_bn[0][0]']
                                )

 block5b_expand_bn (BatchNormal  (None, 10, 10, 1200  4800
['block5b_expand_conv[0][0]']
 ization)                       )

 block5b_expand_activation (Act  (None, 10, 10, 1200  0
['block5b_expand_bn[0][0]']
 ivation)                       )

 block5b_dwconv (DepthwiseConv2  (None, 10, 10, 1200  30000
['block5b_expand_activation[0][0]
 D)                             )                                 ']

 block5b_bn (BatchNormalization  (None, 10, 10, 1200  4800
['block5b_dwconv[0][0]']
 )                              )

 block5b_activation (Activation  (None, 10, 10, 1200  0
['block5b_bn[0][0]']
 )                              )

 block5b_se_squeeze (GlobalAver  (None, 1200)        0
['block5b_activation[0][0]']
 agePooling2D)

 block5b_se_reshape (Reshape)   (None, 1, 1, 1200)   0
['block5b_se_squeeze[0][0]']

 block5b_se_reduce (Conv2D)     (None, 1, 1, 50)     60050
['block5b_se_reshape[0][0]']

 block5b_se_expand (Conv2D)     (None, 1, 1, 1200)   61200
['block5b_se_reduce[0][0]']

 block5b_se_excite (Multiply)   (None, 10, 10, 1200  0
['block5b_activation[0][0]',
                                )
'block5b_se_expand[0][0]']

 block5b_project_conv (Conv2D)  (None, 10, 10, 200)  240000
['block5b_se_excite[0][0]']

 block5b_project_bn (BatchNorma  (None, 10, 10, 200)  800
['block5b_project_conv[0][0]']
```

```
 lization)

 block5b_drop (Dropout)         (None, 10, 10, 200)  0
['block5b_project_bn[0][0]']

 block5b_add (Add)              (None, 10, 10, 200)  0
['block5b_drop[0][0]',
'block5a_project_bn[0][0]']

 block5c_expand_conv (Conv2D)   (None, 10, 10, 1200  240000
['block5b_add[0][0]']
                                )

 block5c_expand_bn (BatchNormal  (None, 10, 10, 1200  4800
['block5c_expand_conv[0][0]']
 ization)                       )

 block5c_expand_activation (Act  (None, 10, 10, 1200  0
['block5c_expand_bn[0][0]']
 ivation)                       )

 block5c_dwconv (DepthwiseConv2  (None, 10, 10, 1200  30000
['block5c_expand_activation[0][0]
 D)                             )                                 ']

 block5c_bn (BatchNormalization  (None, 10, 10, 1200  4800
['block5c_dwconv[0][0]']
 )                              )

 block5c_activation (Activation  (None, 10, 10, 1200  0
['block5c_bn[0][0]']
 )                              )

 block5c_se_squeeze (GlobalAver  (None, 1200)        0
['block5c_activation[0][0]']
 agePooling2D)

 block5c_se_reshape (Reshape)   (None, 1, 1, 1200)   0
['block5c_se_squeeze[0][0]']

 block5c_se_reduce (Conv2D)     (None, 1, 1, 50)     60050
['block5c_se_reshape[0][0]']

 block5c_se_expand (Conv2D)     (None, 1, 1, 1200)   61200
['block5c_se_reduce[0][0]']

 block5c_se_excite (Multiply)   (None, 10, 10, 1200  0
['block5c_activation[0][0]',
                                )
'block5c_se_expand[0][0]']

 block5c_project_conv (Conv2D)  (None, 10, 10, 200)  240000
['block5c_se_excite[0][0]']

 block5c_project_bn (BatchNorma  (None, 10, 10, 200)  800
['block5c_project_conv[0][0]']
 lization)

 block5c_drop (Dropout)         (None, 10, 10, 200)  0
['block5c_project_bn[0][0]']

 block5c_add (Add)              (None, 10, 10, 200)  0
['block5c_drop[0][0]',
'block5b_add[0][0]']

 block5d_expand_conv (Conv2D)   (None, 10, 10, 1200  240000
['block5c_add[0][0]']
                                )

 block5d_expand_bn (BatchNormal  (None, 10, 10, 1200  4800
['block5d_expand_conv[0][0]']
 ization)                       )

 block5d_expand_activation (Act  (None, 10, 10, 1200  0
['block5d_expand_bn[0][0]']
 ivation)                       )

 block5d_dwconv (DepthwiseConv2  (None, 10, 10, 1200  30000
['block5d_expand_activation[0][0]
 D)                             )                                ']

 block5d_bn (BatchNormalization  (None, 10, 10, 1200  4800
['block5d_dwconv[0][0]']
 )                              )

 block5d_activation (Activation  (None, 10, 10, 1200  0
['block5d_bn[0][0]']
 )                              )

 block5d_se_squeeze (GlobalAver  (None, 1200)        0
['block5d_activation[0][0]']
 agePooling2D)

 block5d_se_reshape (Reshape)   (None, 1, 1, 1200)   0
['block5d_se_squeeze[0][0]']
```

```
 block5d_se_reduce (Conv2D)     (None, 1, 1, 50)     60050
['block5d_se_reshape[0][0]']

 block5d_se_expand (Conv2D)     (None, 1, 1, 1200)   61200
['block5d_se_reduce[0][0]']

 block5d_se_excite (Multiply)   (None, 10, 10, 1200  0
['block5d_activation[0][0]',
                                )
'block5d_se_expand[0][0]']

 block5d_project_conv (Conv2D)  (None, 10, 10, 200)  240000
['block5d_se_excite[0][0]']

 block5d_project_bn (BatchNorma  (None, 10, 10, 200)  800
['block5d_project_conv[0][0]']
 lization)

 block5d_drop (Dropout)         (None, 10, 10, 200)  0
['block5d_project_bn[0][0]']

 block5d_add (Add)              (None, 10, 10, 200)  0
['block5d_drop[0][0]',
'block5c_add[0][0]']

 block5e_expand_conv (Conv2D)   (None, 10, 10, 1200  240000
['block5d_add[0][0]']
                                )

 block5e_expand_bn (BatchNormal  (None, 10, 10, 1200  4800
['block5e_expand_conv[0][0]']
 ization)                       )

 block5e_expand_activation (Act  (None, 10, 10, 1200  0
['block5e_expand_bn[0][0]']
 ivation)                       )

 block5e_dwconv (DepthwiseConv2  (None, 10, 10, 1200  30000
['block5e_expand_activation[0][0]
 D)                             )                                '] 

 block5e_bn (BatchNormalization  (None, 10, 10, 1200  4800
['block5e_dwconv[0][0]']
 )                              )

 block5e_activation (Activation  (None, 10, 10, 1200  0
['block5e_bn[0][0]']
 )                              )
 block5e_se_squeeze (GlobalAver  (None, 1200)         0
['block5e_activation[0][0]']
 agePooling2D)

 block5e_se_reshape (Reshape)   (None, 1, 1, 1200)   0
['block5e_se_squeeze[0][0]']

 block5e_se_reduce (Conv2D)     (None, 1, 1, 50)     60050
['block5e_se_reshape[0][0]']

 block5e_se_expand (Conv2D)     (None, 1, 1, 1200)   61200
['block5e_se_reduce[0][0]']

 block5e_se_excite (Multiply)   (None, 10, 10, 1200  0
['block5e_activation[0][0]',
                                )
'block5e_se_expand[0][0]']

 block5e_project_conv (Conv2D)  (None, 10, 10, 200)  240000
['block5e_se_excite[0][0]']

 block5e_project_bn (BatchNorma  (None, 10, 10, 200)  800
['block5e_project_conv[0][0]']
 lization)

 block5e_drop (Dropout)         (None, 10, 10, 200)  0
['block5e_project_bn[0][0]']

 block5e_add (Add)              (None, 10, 10, 200)  0
['block5e_drop[0][0]',
'block5d_add[0][0]']

 block5f_expand_conv (Conv2D)   (None, 10, 10, 1200  240000
['block5e_add[0][0]']
                                )

 block5f_expand_bn (BatchNormal  (None, 10, 10, 1200  4800
['block5f_expand_conv[0][0]']
 ization)                       )

 block5f_expand_activation (Act  (None, 10, 10, 1200  0
['block5f_expand_bn[0][0]']
 ivation)                       )

 block5f_dwconv (DepthwiseConv2  (None, 10, 10, 1200  30000
['block5f_expand_activation[0][0]
 D)                             )                                  ']
 block5f_bn (BatchNormalization  (None, 10, 10, 1200  4800
['block5f_dwconv[0][0]']
 )                              )

 block5f_activation (Activation  (None, 10, 10, 1200  0
['block5f_bn[0][0]']
 )                              )

 block5f_se_squeeze (GlobalAver  (None, 1200)        0
['block5f_activation[0][0]']
 agePooling2D)

 block5f_se_reshape (Reshape)   (None, 1, 1, 1200)   0
['block5f_se_squeeze[0][0]']

 block5f_se_reduce (Conv2D)     (None, 1, 1, 50)     60050
['block5f_se_reshape[0][0]']

 block5f_se_expand (Conv2D)     (None, 1, 1, 1200)   61200
['block5f_se_reduce[0][0]']

 block5f_se_excite (Multiply)   (None, 10, 10, 1200  0
['block5f_activation[0][0]',
                                )
'block5f_se_expand[0][0]']

 block5f_project_conv (Conv2D)  (None, 10, 10, 200)  240000
['block5f_se_excite[0][0]']

 block5f_project_bn (BatchNorma  (None, 10, 10, 200)  800
['block5f_project_conv[0][0]']
 lization)

 block5f_drop (Dropout)         (None, 10, 10, 200)  0
['block5f_project_bn[0][0]']

 block5f_add (Add)              (None, 10, 10, 200)  0
['block5f_drop[0][0]',
'block5e_add[0][0]']

 block5g_expand_conv (Conv2D)   (None, 10, 10, 1200  240000
['block5f_add[0][0]']
                                )

 block5g_expand_bn (BatchNormal  (None, 10, 10, 1200  4800
['block5g_expand_conv[0][0]']
 ization)                       )
 block5g_expand_activation (Act  (None, 10, 10, 1200  0
['block5g_expand_bn[0][0]']
 ivation)                       )

 block5g_dwconv (DepthwiseConv2  (None, 10, 10, 1200  30000
['block5g_expand_activation[0][0]
 D)                             )                                ']

 block5g_bn (BatchNormalization  (None, 10, 10, 1200  4800
['block5g_dwconv[0][0]']
 )                              )

 block5g_activation (Activation  (None, 10, 10, 1200  0
['block5g_bn[0][0]']
 )                              )

 block5g_se_squeeze (GlobalAver  (None, 1200)        0
['block5g_activation[0][0]']
 agePooling2D)

 block5g_se_reshape (Reshape)   (None, 1, 1, 1200)   0
['block5g_se_squeeze[0][0]']

 block5g_se_reduce (Conv2D)     (None, 1, 1, 50)     60050
['block5g_se_reshape[0][0]']

 block5g_se_expand (Conv2D)     (None, 1, 1, 1200)   61200
['block5g_se_reduce[0][0]']

 block5g_se_excite (Multiply)   (None, 10, 10, 1200  0
['block5g_activation[0][0]',
                                )
'block5g_se_expand[0][0]']

 block5g_project_conv (Conv2D)  (None, 10, 10, 200)  240000
['block5g_se_excite[0][0]']

 block5g_project_bn (BatchNorma  (None, 10, 10, 200)  800
['block5g_project_conv[0][0]']
 lization)

 block5g_drop (Dropout)         (None, 10, 10, 200)  0
['block5g_project_bn[0][0]']

 block5g_add (Add)              (None, 10, 10, 200)  0
['block5g_drop[0][0]',
'block5f_add[0][0]']
```

```
 block5h_expand_conv (Conv2D)   (None, 10, 10, 1200  240000
['block5g_add[0][0]']
                                )

 block5h_expand_bn (BatchNormal  (None, 10, 10, 1200  4800
['block5h_expand_conv[0][0]']
 ization)                       )

 block5h_expand_activation (Act  (None, 10, 10, 1200  0
['block5h_expand_bn[0][0]']
 ivation)                       )

 block5h_dwconv (DepthwiseConv2  (None, 10, 10, 1200  30000
['block5h_expand_activation[0][0]
 D)                             )                                ']

 block5h_bn (BatchNormalization  (None, 10, 10, 1200  4800
['block5h_dwconv[0][0]']
 )                              )

 block5h_activation (Activation  (None, 10, 10, 1200  0
['block5h_bn[0][0]']
 )                              )

 block5h_se_squeeze (GlobalAver  (None, 1200)        0
['block5h_activation[0][0]']
 agePooling2D)

 block5h_se_reshape (Reshape)   (None, 1, 1, 1200)   0
['block5h_se_squeeze[0][0]']

 block5h_se_reduce (Conv2D)     (None, 1, 1, 50)     60050
['block5h_se_reshape[0][0]']

 block5h_se_expand (Conv2D)     (None, 1, 1, 1200)   61200
['block5h_se_reduce[0][0]']

 block5h_se_excite (Multiply)   (None, 10, 10, 1200  0
['block5h_activation[0][0]',
                                )
'block5h_se_expand[0][0]']

 block5h_project_conv (Conv2D)  (None, 10, 10, 200)  240000
['block5h_se_excite[0][0]']

 block5h_project_bn (BatchNorma  (None, 10, 10, 200)  800
['block5h_project_conv[0][0]']
```

```
 lization)

 block5h_drop (Dropout)         (None, 10, 10, 200)  0
['block5h_project_bn[0][0]']

 block5h_add (Add)              (None, 10, 10, 200)  0
['block5h_drop[0][0]',
'block5g_add[0][0]']

 block6a_expand_conv (Conv2D)   (None, 10, 10, 1200  240000
['block5h_add[0][0]']
                                )

 block6a_expand_bn (BatchNormal  (None, 10, 10, 1200  4800
['block6a_expand_conv[0][0]']
 ization)                       )

 block6a_expand_activation (Act  (None, 10, 10, 1200  0
['block6a_expand_bn[0][0]']
 ivation)                       )

 block6a_dwconv_pad (ZeroPaddin  (None, 13, 13, 1200  0
['block6a_expand_activation[0][0]
 g2D)                           )                                ']

 block6a_dwconv (DepthwiseConv2  (None, 5, 5, 1200)  30000
['block6a_dwconv_pad[0][0]']
 D)

 block6a_bn (BatchNormalization  (None, 5, 5, 1200)  4800
['block6a_dwconv[0][0]']
 )

 block6a_activation (Activation  (None, 5, 5, 1200)  0
['block6a_bn[0][0]']
 )

 block6a_se_squeeze (GlobalAver  (None, 1200)        0
['block6a_activation[0][0]']
 agePooling2D)

 block6a_se_reshape (Reshape)   (None, 1, 1, 1200)   0
['block6a_se_squeeze[0][0]']

 block6a_se_reduce (Conv2D)     (None, 1, 1, 50)     60050
['block6a_se_reshape[0][0]']

 block6a_se_expand (Conv2D)     (None, 1, 1, 1200)   61200
```

```
['block6a_se_reduce[0][0]']

 block6a_se_excite (Multiply)   (None, 5, 5, 1200)   0
['block6a_activation[0][0]',
'block6a_se_expand[0][0]']

 block6a_project_conv (Conv2D)  (None, 5, 5, 344)    412800
['block6a_se_excite[0][0]']

 block6a_project_bn (BatchNorma  (None, 5, 5, 344)   1376
['block6a_project_conv[0][0]']
 lization)

 block6b_expand_conv (Conv2D)   (None, 5, 5, 2064)   710016
['block6a_project_bn[0][0]']

 block6b_expand_bn (BatchNormal  (None, 5, 5, 2064)  8256
['block6b_expand_conv[0][0]']
 ization)

 block6b_expand_activation (Act  (None, 5, 5, 2064)  0
['block6b_expand_bn[0][0]']
 ivation)

 block6b_dwconv (DepthwiseConv2  (None, 5, 5, 2064)  51600
['block6b_expand_activation[0][0]
 D)                                                               ']

 block6b_bn (BatchNormalization  (None, 5, 5, 2064)  8256
['block6b_dwconv[0][0]']
 )

 block6b_activation (Activation  (None, 5, 5, 2064)  0
['block6b_bn[0][0]']
 )

 block6b_se_squeeze (GlobalAver  (None, 2064)        0
['block6b_activation[0][0]']
 agePooling2D)

 block6b_se_reshape (Reshape)   (None, 1, 1, 2064)   0
['block6b_se_squeeze[0][0]']

 block6b_se_reduce (Conv2D)     (None, 1, 1, 86)     177590
['block6b_se_reshape[0][0]']

 block6b_se_expand (Conv2D)     (None, 1, 1, 2064)   179568
['block6b_se_reduce[0][0]']
```

```
 block6b_se_excite (Multiply)   (None, 5, 5, 2064)   0
['block6b_activation[0][0]',
'block6b_se_expand[0][0]']

 block6b_project_conv (Conv2D)  (None, 5, 5, 344)    710016
['block6b_se_excite[0][0]']

 block6b_project_bn (BatchNorma  (None, 5, 5, 344)   1376
['block6b_project_conv[0][0]']
 lization)

 block6b_drop (Dropout)         (None, 5, 5, 344)    0
['block6b_project_bn[0][0]']

 block6b_add (Add)              (None, 5, 5, 344)    0
['block6b_drop[0][0]',
'block6a_project_bn[0][0]']

 block6c_expand_conv (Conv2D)   (None, 5, 5, 2064)   710016
['block6b_add[0][0]']

 block6c_expand_bn (BatchNormal  (None, 5, 5, 2064)  8256
['block6c_expand_conv[0][0]']
 ization)

 block6c_expand_activation (Act  (None, 5, 5, 2064)  0
['block6c_expand_bn[0][0]']
 ivation)

 block6c_dwconv (DepthwiseConv2  (None, 5, 5, 2064)  51600
['block6c_expand_activation[0][0]
 D)                                                                '] 

 block6c_bn (BatchNormalization  (None, 5, 5, 2064)  8256
['block6c_dwconv[0][0]']
 )

 block6c_activation (Activation  (None, 5, 5, 2064)  0
['block6c_bn[0][0]']
 )

 block6c_se_squeeze (GlobalAver  (None, 2064)        0
['block6c_activation[0][0]']
 agePooling2D)

 block6c_se_reshape (Reshape)   (None, 1, 1, 2064)   0
['block6c_se_squeeze[0][0]']
```

```
 block6c_se_reduce (Conv2D)     (None, 1, 1, 86)     177590
['block6c_se_reshape[0][0]']

 block6c_se_expand (Conv2D)     (None, 1, 1, 2064)   179568
['block6c_se_reduce[0][0]']

 block6c_se_excite (Multiply)   (None, 5, 5, 2064)   0
['block6c_activation[0][0]',
'block6c_se_expand[0][0]']

 block6c_project_conv (Conv2D)  (None, 5, 5, 344)    710016
['block6c_se_excite[0][0]']

 block6c_project_bn (BatchNorma  (None, 5, 5, 344)   1376
['block6c_project_conv[0][0]']
 lization)

 block6c_drop (Dropout)         (None, 5, 5, 344)    0
['block6c_project_bn[0][0]']

 block6c_add (Add)              (None, 5, 5, 344)    0
['block6c_drop[0][0]',
'block6b_add[0][0]']

 block6d_expand_conv (Conv2D)   (None, 5, 5, 2064)   710016
['block6c_add[0][0]']

 block6d_expand_bn (BatchNormal  (None, 5, 5, 2064)  8256
['block6d_expand_conv[0][0]']
 ization)

 block6d_expand_activation (Act  (None, 5, 5, 2064)  0
['block6d_expand_bn[0][0]']
 ivation)

 block6d_dwconv (DepthwiseConv2  (None, 5, 5, 2064)  51600
['block6d_expand_activation[0][0]
 D)                                                               ']

 block6d_bn (BatchNormalization  (None, 5, 5, 2064)  8256
['block6d_dwconv[0][0]']
 )

 block6d_activation (Activation  (None, 5, 5, 2064)  0
['block6d_bn[0][0]']
 )
 block6d_se_squeeze (GlobalAver  (None, 2064)        0
['block6d_activation[0][0]']
 agePooling2D)

 block6d_se_reshape (Reshape)   (None, 1, 1, 2064)   0
['block6d_se_squeeze[0][0]']

 block6d_se_reduce (Conv2D)     (None, 1, 1, 86)     177590
['block6d_se_reshape[0][0]']

 block6d_se_expand (Conv2D)     (None, 1, 1, 2064)   179568
['block6d_se_reduce[0][0]']

 block6d_se_excite (Multiply)   (None, 5, 5, 2064)   0
['block6d_activation[0][0]',
'block6d_se_expand[0][0]']

 block6d_project_conv (Conv2D)  (None, 5, 5, 344)    710016
['block6d_se_excite[0][0]']

 block6d_project_bn (BatchNorma  (None, 5, 5, 344)   1376
['block6d_project_conv[0][0]']
 lization)

 block6d_drop (Dropout)         (None, 5, 5, 344)    0
['block6d_project_bn[0][0]']

 block6d_add (Add)              (None, 5, 5, 344)    0
['block6d_drop[0][0]',
'block6c_add[0][0]']

 block6e_expand_conv (Conv2D)   (None, 5, 5, 2064)   710016
['block6d_add[0][0]']

 block6e_expand_bn (BatchNormal  (None, 5, 5, 2064)  8256
['block6e_expand_conv[0][0]']
 ization)

 block6e_expand_activation (Act  (None, 5, 5, 2064)  0
['block6e_expand_bn[0][0]']
 ivation)

 block6e_dwconv (DepthwiseConv2  (None, 5, 5, 2064)  51600
['block6e_expand_activation[0][0]
 D)                                                           ']

 block6e_bn (BatchNormalization  (None, 5, 5, 2064)  8256
['block6e_dwconv[0][0]']
```

```
 )

 block6e_activation (Activation  (None, 5, 5, 2064)  0
['block6e_bn[0][0]']
 )

 block6e_se_squeeze (GlobalAver  (None, 2064)        0
['block6e_activation[0][0]']
 agePooling2D)

 block6e_se_reshape (Reshape)   (None, 1, 1, 2064)   0
['block6e_se_squeeze[0][0]']

 block6e_se_reduce (Conv2D)     (None, 1, 1, 86)     177590
['block6e_se_reshape[0][0]']

 block6e_se_expand (Conv2D)     (None, 1, 1, 2064)   179568
['block6e_se_reduce[0][0]']

 block6e_se_excite (Multiply)   (None, 5, 5, 2064)   0
['block6e_activation[0][0]',
'block6e_se_expand[0][0]']

 block6e_project_conv (Conv2D)  (None, 5, 5, 344)    710016
['block6e_se_excite[0][0]']

 block6e_project_bn (BatchNorma  (None, 5, 5, 344)   1376
['block6e_project_conv[0][0]']
 lization)

 block6e_drop (Dropout)         (None, 5, 5, 344)    0
['block6e_project_bn[0][0]']

 block6e_add (Add)              (None, 5, 5, 344)    0
['block6e_drop[0][0]',
'block6d_add[0][0]']

 block6f_expand_conv (Conv2D)   (None, 5, 5, 2064)   710016
['block6e_add[0][0]']

 block6f_expand_bn (BatchNormal  (None, 5, 5, 2064)  8256
['block6f_expand_conv[0][0]']
 ization)

 block6f_expand_activation (Act  (None, 5, 5, 2064)  0
['block6f_expand_bn[0][0]']
 ivation)
 block6f_dwconv (DepthwiseConv2  (None, 5, 5, 2064)  51600
['block6f_expand_activation[0][0]
 D)                                                               ']

 block6f_bn (BatchNormalization  (None, 5, 5, 2064)  8256
['block6f_dwconv[0][0]']
 )

 block6f_activation (Activation  (None, 5, 5, 2064)  0
['block6f_bn[0][0]']
 )

 block6f_se_squeeze (GlobalAver  (None, 2064)        0
['block6f_activation[0][0]']
 agePooling2D)

 block6f_se_reshape (Reshape)   (None, 1, 1, 2064)   0
['block6f_se_squeeze[0][0]']

 block6f_se_reduce (Conv2D)     (None, 1, 1, 86)     177590
['block6f_se_reshape[0][0]']

 block6f_se_expand (Conv2D)     (None, 1, 1, 2064)   179568
['block6f_se_reduce[0][0]']

 block6f_se_excite (Multiply)   (None, 5, 5, 2064)   0
['block6f_activation[0][0]',
'block6f_se_expand[0][0]']

 block6f_project_conv (Conv2D)  (None, 5, 5, 344)    710016
['block6f_se_excite[0][0]']

 block6f_project_bn (BatchNorma  (None, 5, 5, 344)   1376
['block6f_project_conv[0][0]']
 lization)

 block6f_drop (Dropout)         (None, 5, 5, 344)    0
['block6f_project_bn[0][0]']

 block6f_add (Add)              (None, 5, 5, 344)    0
['block6f_drop[0][0]',
'block6e_add[0][0]']

 block6g_expand_conv (Conv2D)   (None, 5, 5, 2064)   710016
['block6f_add[0][0]']

 block6g_expand_bn (BatchNormal  (None, 5, 5, 2064)  8256
['block6g_expand_conv[0][0]']
```

```
 ization)

 block6g_expand_activation (Act  (None, 5, 5, 2064)  0
['block6g_expand_bn[0][0]']
 ivation)

 block6g_dwconv (DepthwiseConv2  (None, 5, 5, 2064)  51600
['block6g_expand_activation[0][0]
 D)                                                             ']

 block6g_bn (BatchNormalization  (None, 5, 5, 2064)  8256
['block6g_dwconv[0][0]']
 )

 block6g_activation (Activation  (None, 5, 5, 2064)  0
['block6g_bn[0][0]']
 )

 block6g_se_squeeze (GlobalAver  (None, 2064)        0
['block6g_activation[0][0]']
 agePooling2D)

 block6g_se_reshape (Reshape)   (None, 1, 1, 2064)   0
['block6g_se_squeeze[0][0]']

 block6g_se_reduce (Conv2D)     (None, 1, 1, 86)     177590
['block6g_se_reshape[0][0]']

 block6g_se_expand (Conv2D)     (None, 1, 1, 2064)   179568
['block6g_se_reduce[0][0]']

 block6g_se_excite (Multiply)   (None, 5, 5, 2064)   0
['block6g_activation[0][0]',
'block6g_se_expand[0][0]']

 block6g_project_conv (Conv2D)  (None, 5, 5, 344)    710016
['block6g_se_excite[0][0]']

 block6g_project_bn (BatchNorma  (None, 5, 5, 344)   1376
['block6g_project_conv[0][0]']
 lization)

 block6g_drop (Dropout)         (None, 5, 5, 344)    0
['block6g_project_bn[0][0]']

 block6g_add (Add)              (None, 5, 5, 344)    0
['block6g_drop[0][0]',
'block6f_add[0][0]']
```

```
 block6h_expand_conv (Conv2D)   (None, 5, 5, 2064)   710016
['block6g_add[0][0]']

 block6h_expand_bn (BatchNormal  (None, 5, 5, 2064)  8256
['block6h_expand_conv[0][0]']
 ization)

 block6h_expand_activation (Act  (None, 5, 5, 2064)  0
['block6h_expand_bn[0][0]']
 ivation)

 block6h_dwconv (DepthwiseConv2  (None, 5, 5, 2064)  51600
['block6h_expand_activation[0][0]
 D)                                                              ']

 block6h_bn (BatchNormalization  (None, 5, 5, 2064)  8256
['block6h_dwconv[0][0]']
 )

 block6h_activation (Activation  (None, 5, 5, 2064)  0
['block6h_bn[0][0]']
 )

 block6h_se_squeeze (GlobalAver  (None, 2064)        0
['block6h_activation[0][0]']
 agePooling2D)

 block6h_se_reshape (Reshape)   (None, 1, 1, 2064)   0
['block6h_se_squeeze[0][0]']

 block6h_se_reduce (Conv2D)     (None, 1, 1, 86)     177590
['block6h_se_reshape[0][0]']

 block6h_se_expand (Conv2D)     (None, 1, 1, 2064)   179568
['block6h_se_reduce[0][0]']

 block6h_se_excite (Multiply)   (None, 5, 5, 2064)   0
['block6h_activation[0][0]',
'block6h_se_expand[0][0]']

 block6h_project_conv (Conv2D)  (None, 5, 5, 344)    710016
['block6h_se_excite[0][0]']

 block6h_project_bn (BatchNorma  (None, 5, 5, 344)   1376
['block6h_project_conv[0][0]']
 lization)
 block6h_drop (Dropout)         (None, 5, 5, 344)    0
['block6h_project_bn[0][0]']

 block6h_add (Add)              (None, 5, 5, 344)    0
['block6h_drop[0][0]',
'block6g_add[0][0]']

 block6i_expand_conv (Conv2D)   (None, 5, 5, 2064)   710016
['block6h_add[0][0]']

 block6i_expand_bn (BatchNormal  (None, 5, 5, 2064)  8256
['block6i_expand_conv[0][0]']
 ization)

 block6i_expand_activation (Act  (None, 5, 5, 2064)  0
['block6i_expand_bn[0][0]']
 ivation)

 block6i_dwconv (DepthwiseConv2  (None, 5, 5, 2064)  51600
['block6i_expand_activation[0][0]
 D)                                                              ']

 block6i_bn (BatchNormalization  (None, 5, 5, 2064)  8256
['block6i_dwconv[0][0]']
 )

 block6i_activation (Activation  (None, 5, 5, 2064)  0
['block6i_bn[0][0]']
 )

 block6i_se_squeeze (GlobalAver  (None, 2064)        0
['block6i_activation[0][0]']
 agePooling2D)

 block6i_se_reshape (Reshape)   (None, 1, 1, 2064)   0
['block6i_se_squeeze[0][0]']

 block6i_se_reduce (Conv2D)     (None, 1, 1, 86)     177590
['block6i_se_reshape[0][0]']

 block6i_se_expand (Conv2D)     (None, 1, 1, 2064)   179568
['block6i_se_reduce[0][0]']

 block6i_se_excite (Multiply)   (None, 5, 5, 2064)   0
['block6i_activation[0][0]',
'block6i_se_expand[0][0]']

 block6i_project_conv (Conv2D)  (None, 5, 5, 344)    710016
```

```
['block6i_se_excite[0][0]']

 block6i_project_bn (BatchNorma  (None, 5, 5, 344)   1376
['block6i_project_conv[0][0]']
 lization)

 block6i_drop (Dropout)         (None, 5, 5, 344)    0
['block6i_project_bn[0][0]']

 block6i_add (Add)              (None, 5, 5, 344)    0
['block6i_drop[0][0]',
'block6h_add[0][0]']

 block6j_expand_conv (Conv2D)   (None, 5, 5, 2064)   710016
['block6i_add[0][0]']

 block6j_expand_bn (BatchNormal  (None, 5, 5, 2064)  8256
['block6j_expand_conv[0][0]']
 ization)

 block6j_expand_activation (Act  (None, 5, 5, 2064)  0
['block6j_expand_bn[0][0]']
 ivation)

 block6j_dwconv (DepthwiseConv2  (None, 5, 5, 2064)  51600
['block6j_expand_activation[0][0]
 D)                                                              ']

 block6j_bn (BatchNormalization  (None, 5, 5, 2064)  8256
['block6j_dwconv[0][0]']
 )

 block6j_activation (Activation  (None, 5, 5, 2064)  0
['block6j_bn[0][0]']
 )

 block6j_se_squeeze (GlobalAver  (None, 2064)        0
['block6j_activation[0][0]']
 agePooling2D)

 block6j_se_reshape (Reshape)   (None, 1, 1, 2064)   0
['block6j_se_squeeze[0][0]']

 block6j_se_reduce (Conv2D)     (None, 1, 1, 86)     177590
['block6j_se_reshape[0][0]']

 block6j_se_expand (Conv2D)     (None, 1, 1, 2064)   179568
['block6j_se_reduce[0][0]']
```

```
 block6j_se_excite (Multiply)   (None, 5, 5, 2064)   0
['block6j_activation[0][0]',
'block6j_se_expand[0][0]']

 block6j_project_conv (Conv2D)  (None, 5, 5, 344)    710016
['block6j_se_excite[0][0]']

 block6j_project_bn (BatchNorma  (None, 5, 5, 344)   1376
['block6j_project_conv[0][0]']
 lization)

 block6j_drop (Dropout)         (None, 5, 5, 344)    0
['block6j_project_bn[0][0]']

 block6j_add (Add)              (None, 5, 5, 344)    0
['block6j_drop[0][0]',
'block6i_add[0][0]']

 block6k_expand_conv (Conv2D)   (None, 5, 5, 2064)   710016
['block6j_add[0][0]']

 block6k_expand_bn (BatchNormal  (None, 5, 5, 2064)  8256
['block6k_expand_conv[0][0]']
 ization)

 block6k_expand_activation (Act  (None, 5, 5, 2064)  0
['block6k_expand_bn[0][0]']
 ivation)

 block6k_dwconv (DepthwiseConv2  (None, 5, 5, 2064)  51600
['block6k_expand_activation[0][0]
 D)                                                              ']

 block6k_bn (BatchNormalization  (None, 5, 5, 2064)  8256
['block6k_dwconv[0][0]']
 )

 block6k_activation (Activation  (None, 5, 5, 2064)  0
['block6k_bn[0][0]']
 )

 block6k_se_squeeze (GlobalAver  (None, 2064)        0
['block6k_activation[0][0]']
 agePooling2D)

 block6k_se_reshape (Reshape)   (None, 1, 1, 2064)   0
['block6k_se_squeeze[0][0]']
```

```
 block6k_se_reduce (Conv2D)     (None, 1, 1, 86)     177590
['block6k_se_reshape[0][0]']

 block6k_se_expand (Conv2D)     (None, 1, 1, 2064)   179568
['block6k_se_reduce[0][0]']

 block6k_se_excite (Multiply)   (None, 5, 5, 2064)   0
['block6k_activation[0][0]',
'block6k_se_expand[0][0]']

 block6k_project_conv (Conv2D)  (None, 5, 5, 344)    710016
['block6k_se_excite[0][0]']

 block6k_project_bn (BatchNorma  (None, 5, 5, 344)   1376
['block6k_project_conv[0][0]']
 lization)

 block6k_drop (Dropout)         (None, 5, 5, 344)    0
['block6k_project_bn[0][0]']

 block6k_add (Add)              (None, 5, 5, 344)    0
['block6k_drop[0][0]',
'block6j_add[0][0]']

 block7a_expand_conv (Conv2D)   (None, 5, 5, 2064)   710016
['block6k_add[0][0]']

 block7a_expand_bn (BatchNormal  (None, 5, 5, 2064)  8256
['block7a_expand_conv[0][0]']
 ization)

 block7a_expand_activation (Act  (None, 5, 5, 2064)  0
['block7a_expand_bn[0][0]']
 ivation)

 block7a_dwconv (DepthwiseConv2  (None, 5, 5, 2064)  18576
['block7a_expand_activation[0][0]
 D)                                                               ']

 block7a_bn (BatchNormalization  (None, 5, 5, 2064)  8256
['block7a_dwconv[0][0]']
 )

 block7a_activation (Activation  (None, 5, 5, 2064)  0
['block7a_bn[0][0]']
 )
 block7a_se_squeeze (GlobalAver  (None, 2064)        0
['block7a_activation[0][0]']
 agePooling2D)

 block7a_se_reshape (Reshape)   (None, 1, 1, 2064)   0
['block7a_se_squeeze[0][0]']

 block7a_se_reduce (Conv2D)     (None, 1, 1, 86)     177590
['block7a_se_reshape[0][0]']

 block7a_se_expand (Conv2D)     (None, 1, 1, 2064)   179568
['block7a_se_reduce[0][0]']

 block7a_se_excite (Multiply)   (None, 5, 5, 2064)   0
['block7a_activation[0][0]',
'block7a_se_expand[0][0]']

 block7a_project_conv (Conv2D)  (None, 5, 5, 576)    1188864
['block7a_se_excite[0][0]']

 block7a_project_bn (BatchNorma  (None, 5, 5, 576)   2304
['block7a_project_conv[0][0]']
 lization)

 block7b_expand_conv (Conv2D)   (None, 5, 5, 3456)   1990656
['block7a_project_bn[0][0]']

 block7b_expand_bn (BatchNormal  (None, 5, 5, 3456)  13824
['block7b_expand_conv[0][0]']
 ization)

 block7b_expand_activation (Act  (None, 5, 5, 3456)  0
['block7b_expand_bn[0][0]']
 ivation)

 block7b_dwconv (DepthwiseConv2  (None, 5, 5, 3456)  31104
['block7b_expand_activation[0][0]
 D)                                                               ']

 block7b_bn (BatchNormalization  (None, 5, 5, 3456)  13824
['block7b_dwconv[0][0]']
 )

 block7b_activation (Activation  (None, 5, 5, 3456)  0
['block7b_bn[0][0]']
 )

 block7b_se_squeeze (GlobalAver  (None, 3456)        0
['block7b_activation[0][0]']
 agePooling2D)

 block7b_se_reshape (Reshape)   (None, 1, 1, 3456)   0
['block7b_se_squeeze[0][0]']

 block7b_se_reduce (Conv2D)     (None, 1, 1, 144)    497808
['block7b_se_reshape[0][0]']

 block7b_se_expand (Conv2D)     (None, 1, 1, 3456)   501120
['block7b_se_reduce[0][0]']

 block7b_se_excite (Multiply)   (None, 5, 5, 3456)   0
['block7b_activation[0][0]',
'block7b_se_expand[0][0]']

 block7b_project_conv (Conv2D)  (None, 5, 5, 576)    1990656
['block7b_se_excite[0][0]']

 block7b_project_bn (BatchNorma  (None, 5, 5, 576)   2304
['block7b_project_conv[0][0]']
 lization)

 block7b_drop (Dropout)         (None, 5, 5, 576)    0
['block7b_project_bn[0][0]']

 block7b_add (Add)              (None, 5, 5, 576)    0
['block7b_drop[0][0]',
'block7a_project_bn[0][0]']

 block7c_expand_conv (Conv2D)   (None, 5, 5, 3456)   1990656
['block7b_add[0][0]']

 block7c_expand_bn (BatchNormal  (None, 5, 5, 3456)  13824
['block7c_expand_conv[0][0]']
 ization)

 block7c_expand_activation (Act  (None, 5, 5, 3456)  0
['block7c_expand_bn[0][0]']
 ivation)

 block7c_dwconv (DepthwiseConv2  (None, 5, 5, 3456)  31104
['block7c_expand_activation[0][0]
 D)                                                               ']

 block7c_bn (BatchNormalization  (None, 5, 5, 3456)  13824
['block7c_dwconv[0][0]']
 )
 block7c_activation (Activation  (None, 5, 5, 3456)  0
['block7c_bn[0][0]']
 )

 block7c_se_squeeze (GlobalAver  (None, 3456)        0
['block7c_activation[0][0]']
 agePooling2D)

 block7c_se_reshape (Reshape)   (None, 1, 1, 3456)   0
['block7c_se_squeeze[0][0]']

 block7c_se_reduce (Conv2D)     (None, 1, 1, 144)    497808
['block7c_se_reshape[0][0]']

 block7c_se_expand (Conv2D)     (None, 1, 1, 3456)   501120
['block7c_se_reduce[0][0]']

 block7c_se_excite (Multiply)   (None, 5, 5, 3456)   0
['block7c_activation[0][0]',
'block7c_se_expand[0][0]']

 block7c_project_conv (Conv2D)  (None, 5, 5, 576)    1990656
['block7c_se_excite[0][0]']

 block7c_project_bn (BatchNorma  (None, 5, 5, 576)   2304
['block7c_project_conv[0][0]']
 lization)

 block7c_drop (Dropout)         (None, 5, 5, 576)    0
['block7c_project_bn[0][0]']

 block7c_add (Add)              (None, 5, 5, 576)    0
['block7c_drop[0][0]',
'block7b_add[0][0]']

 top_conv (Conv2D)              (None, 5, 5, 2304)   1327104
['block7c_add[0][0]']

 top_bn (BatchNormalization)    (None, 5, 5, 2304)   9216
['top_conv[0][0]']

 top_activation (Activation)    (None, 5, 5, 2304)   0
['top_bn[0][0]']

 max_pool (GlobalMaxPooling2D)  (None, 2304)         0
['top_activation[0][0]']
```

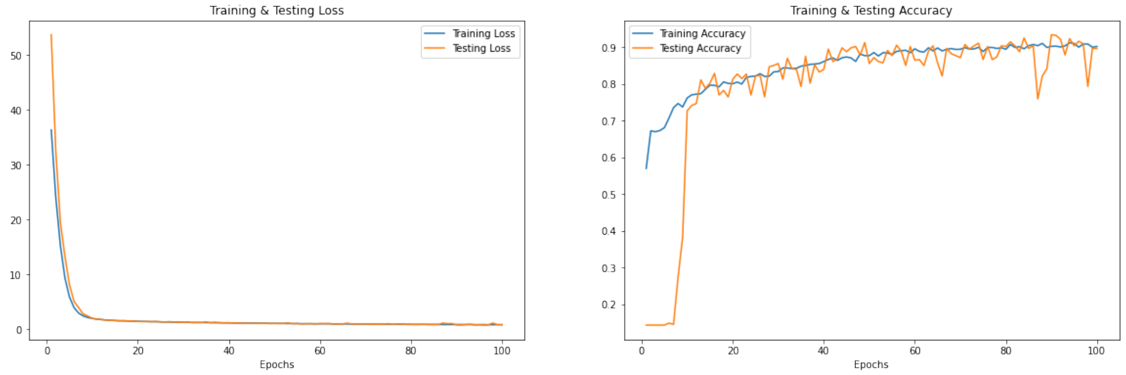
```
================================================================================
==================
Total params: 40,960,143
Trainable params: 40,735,704
Non-trainable params: 224,439
--------------------------------------------------------------------------------
------------------
```

[72]:

```
out_size = 2304
train_features_6, train_labels_6 = extract_features(train_dir, 2240,
                                                    out_size = out_size,
                                                    transfer = efficient_6)
test_features_6, test_labels_6 = extract_features(test_dir, 560,
                                                  out_size = out_size,
                                                  transfer = efficient_6)
print("DONE")

eff_model_6 = Sequential()
eff_model_6.add(tf.keras.layers.BatchNormalization(axis=-1, momentum=0.99,␣
 ↪epsilon=0.001))
eff_model_6.add(layers.Dense(512,
                       kernel_regularizer   = tf.keras.regularizers.l2(l = 0.
 ↪05),
                       activity_regularizer = tf.keras.regularizers.l1(0.05),
                       bias_regularizer     = tf.keras.regularizers.l1(0.05),
                       activation='relu',
                       input_dim = out_size))
# eff_model_6.add(Dropout(0.45))
# eff_model_6.add(tf.keras.layers.BatchNormalization(axis=-1, momentum=0.99,␣
 ↪epsilon=0.001))
# eff_model_6.add(layers.Dense(256, activation='relu'))
# eff_model_6.add(Dropout(0.45))
# eff_model_6.add(layers.Dense(128, activation='relu'))
# eff_model_6.add(Dropout(0.45))

eff_model_6.add(layers.Dense(7, activation='softmax'))

eff_model_6.compile(optimizer=tf.keras.optimizers.RMSprop(lr=1e-4),
              loss='categorical_crossentropy',
              metrics=['accuracy'])

eff_model_6.fit(train_features_6, train_labels_6,
                    epochs=100,
                    batch_size=30,
                    verbose = 0,
                    validation_data=(test_features_6, test_labels_6))
print("Fitting Done")
```

```
loss_acc(eff_model_6)
```

```
Found 2240 images belonging to 7 classes.
Found 560 images belonging to 7 classes.
DONE
Fitting Done
```



6.8.2 EfficientNetB7

[37]:

```
# All images will be rescaled by 1./255.
datagen = ImageDataGenerator( rescale = 1.0/255, dtype= tf.float64)
batch_size = 30
target_size = (150, 150)
input_shape = (target_size[0], target_size[1], 3)
```

[38]:

```
# All images will be rescaled by 1./255.
datagen = ImageDataGenerator( rescale = 1.0/255, dtype= tf.float64)
batch_size = 30
target_size = (150, 150)
input_shape = (target_size[0], target_size[1], 3)



model_name='EfficientNetB7'

efficient_7=tf.keras.applications.EfficientNetB7(
                                      include_top=False,
                                      weights="imagenet",
                                      input_shape=input_shape,
                                      classes=7,
                                      classifier_activation='softmax',
                                      pooling='max')
```

```
# efficient_7.summary()
```

```
Downloading data from https://storage.googleapis.com/keras-
applications/efficientnetb7_notop.h5
258080768/258076736 [==============================] - 1s 0us/step
258088960/258076736 [==============================] - 1s 0us/step
```

[40]:

```
# You should be able to divide sample_amount by batch_size
train_features, train_labels = extract_features(train_dir, 2240, out_size =␣
 ↪2560, transfer = efficient_7)
test_features, test_labels = extract_features(test_dir, 560, out_size = 2560,␣
 ↪transfer = efficient_7)
print("DONE")
```

```
Found 2240 images belonging to 7 classes.
Found 560 images belonging to 7 classes.
DONE
```

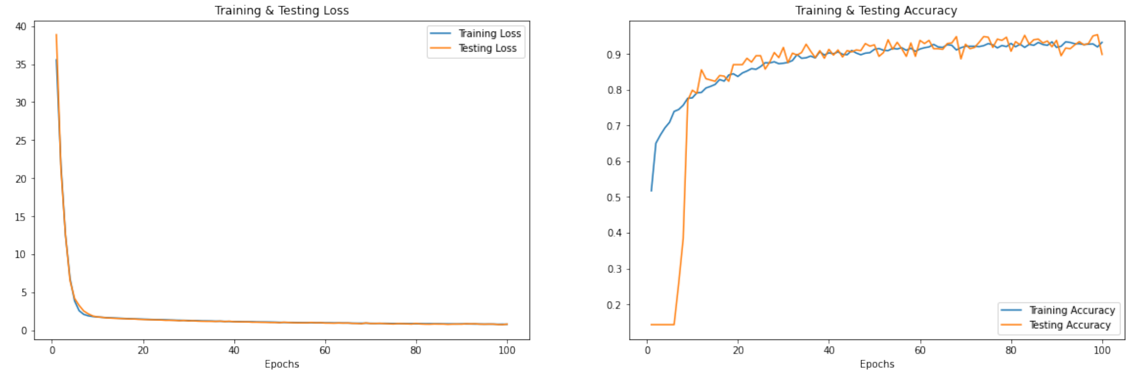
[47]:

```
eff_model_7 = Sequential()
eff_model_7.add(tf.keras.layers.BatchNormalization(axis=-1, momentum=0.99,␣
 ↪epsilon=0.001))
eff_model_7.add(layers.Dense(512,
                       kernel_regularizer   = tf.keras.regularizers.l2(l = 0.
 ↪05),
                       activity_regularizer = tf.keras.regularizers.l1(0.05),
                       bias_regularizer     = tf.keras.regularizers.l1(0.05),
                       activation='relu',
                       input_dim=2560))
# eff_model_7.add(Dropout(0.45))
# eff_model_7.add(tf.keras.layers.BatchNormalization(axis=-1, momentum=0.99,␣
 ↪epsilon=0.001))
# eff_model_7.add(layers.Dense(256, activation='relu'))
# eff_model_7.add(Dropout(0.45))
# eff_model_7.add(layers.Dense(128, activation='relu'))
# eff_model_7.add(Dropout(0.45))

eff_model_7.add(layers.Dense(7, activation='softmax'))

eff_model_7.compile(optimizer=tf.keras.optimizers.RMSprop(lr=1e-4),
              loss='categorical_crossentropy',
              metrics=['accuracy'])

eff_model_7.fit(train_features, train_labels,
                    epochs=100,
                    batch_size=30,
                    verbose = 0,
                              validation_data=(test_features, test_labels))
print("Fitting Done")
```

```
Fitting Done
```

[48]:

```
loss_acc(eff_model_7)
```



7 Results and Conclusion

In this notebook, we trained several deep learning models by using convolutional neural network as well as transfer learning. We can see that all models performed well. Therefore, we will use the structure of the model to choose the final model for this work. Since the first model has the simplest structure and it converges after only 2 epochs, we would recommend this model to be used for emotion detection in audio speech.

For future work, we would recommend obtaining more data for training purposes and also we would recommend using LSTM layers for numerical values obtained from audio files.