```
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
import seaborn as sns
import tensorflow as tf
import datetime,os
from tensorflow import keras
from keras.models import Model


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

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mour
```

```
#!gdown --id 1Z4TyI7FcFVEx8qdl4jO9qxvxaqLSqoEu


#!wget --header="Host: storage.googleapis.com" --header="User-Agent: Mozilla/5.0 (Windows NT


#get_ipython().system_raw("unrar x rvl-cdip.rar")


!unrar x "/content/drive/My Drive/rvl-cdip.rar" "/content/drive/My Drive/image_data_TL"
```

```
Creating    /content/drive/My Drive/image_data_TL/data_final/imagesy/y/x/x/yxx65c00  O
Extracting  /content/drive/My Drive/image_data_TL/data_final/imagesy/y/x/x/yxx65c00/20
Creating    /content/drive/My Drive/image_data_TL/data_final/imagesy/y/x/x/yxx86e00  O
Extracting  /content/drive/My Drive/image_data_TL/data_final/imagesy/y/x/x/yxx86e00/20
Creating    /content/drive/My Drive/image_data_TL/data_final/imagesy/y/x/y  OK
Creating    /content/drive/My Drive/image_data_TL/data_final/imagesy/y/x/y/yxy13a00  O
Extracting  /content/drive/My Drive/image_data_TL/data_final/imagesy/y/x/y/yxy13a00/50
Creating    /content/drive/My Drive/image_data_TL/data_final/imagesy/y/x/y/yxy58e00  O
Extracting  /content/drive/My Drive/image_data_TL/data_final/imagesy/y/x/y/yxy58e00/20
Creating    /content/drive/My Drive/image_data_TL/data_final/imagesy/y/x/z  OK
Creating    /content/drive/My Drive/image_data_TL/data_final/imagesy/y/x/z/yxz48e00  O
Extracting  /content/drive/My Drive/image_data_TL/data_final/imagesy/y/x/z/yxz48e00/20
Creating    /content/drive/My Drive/image_data_TL/data_final/imagesy/y/x/z/yxz71f00  O
Extracting  /content/drive/My Drive/image_data_TL/data_final/imagesy/y/x/z/yxz71f00/20
Saved successfully!
rive/image_data_TL/data_final/imagesy/y/x/z/yxz79e00  O
rive/image_data_TL/data_final/imagesy/y/x/z/yxz79e00/00
Creating    /content/drive/My Drive/image_data_TL/data_final/imagesy/y/x/z/yxz85c00  O
Extracting  /content/drive/My Drive/image_data_TL/data_final/imagesy/y/x/z/yxz85c00/20
Creating    /content/drive/My Drive/image_data_TL/data_final/imagesy/y/x/z/yxz95e00  O
Extracting  /content/drive/My Drive/image_data_TL/data_final/imagesy/y/x/z/yxz95e00/20
Creating    /content/drive/My Drive/image_data_TL/data_final/imagesy/y/y  OK
Creating    /content/drive/My Drive/image_data_TL/data_final/imagesy/y/y/a  OK
Creating    /content/drive/My Drive/image_data_TL/data_final/imagesy/y/y/a/yya60d00  O
Extracting  /content/drive/My Drive/image_data_TL/data_final/imagesy/y/y/a/yya60d00/5
Creating    /content/drive/My Drive/image_data_TL/data_final/imagesy/y/y/b  OK
Creating    /content/drive/My Drive/image_data_TL/data_final/imagesy/y/y/b/yyb10e00  O
Extracting  /content/drive/My Drive/image_data_TL/data_final/imagesy/y/y/b/yyb10e00/80
```

```
Creating    /content/drive/My Drive/image_data_TL/data_final/imagesy/y/y/b/yyb44a00  O
Extracting  /content/drive/My Drive/image_data_TL/data_final/imagesy/y/y/b/yyb44a00/9
Creating    /content/drive/My Drive/image_data_TL/data_final/imagesy/y/y/b/yyb64c00  O
Extracting  /content/drive/My Drive/image_data_TL/data_final/imagesy/y/y/b/yyb64c00/8
Creating    /content/drive/My Drive/image_data_TL/data_final/imagesy/y/y/b/yyb80c00  O
Extracting  /content/drive/My Drive/image_data_TL/data_final/imagesy/y/y/b/yyb80c00/2O
Creating    /content/drive/My Drive/image_data_TL/data_final/imagesy/y/y/b/yyb92f00  O
Extracting  /content/drive/My Drive/image_data_TL/data_final/imagesy/y/y/b/yyb92f00/tO
Creating    /content/drive/My Drive/image_data_TL/data_final/imagesy/y/y/c  OK
Creating    /content/drive/My Drive/image_data_TL/data_final/imagesy/y/y/c/yyc55f00  O
Extracting  /content/drive/My Drive/image_data_TL/data_final/imagesy/y/y/c/yyc55f00/0O
Creating    /content/drive/My Drive/image_data_TL/data_final/imagesy/y/y/c/yyc58d00  O
Extracting  /content/drive/My Drive/image_data_TL/data_final/imagesy/y/y/c/yyc58d00/2O
Creating    /content/drive/My Drive/image_data_TL/data_final/imagesy/y/y/c/yyc88e00  O
Extracting  /content/drive/My Drive/image_data_TL/data_final/imagesy/y/y/c/yyc88e00/2O
Creating    /content/drive/My Drive/image_data_TL/data_final/imagesy/y/y/d  OK
Creating    /content/drive/My Drive/image_data_TL/data_final/imagesy/y/y/d/yyd01e00  O
Extracting  /content/drive/My Drive/image_data_TL/data_final/imagesy/y/y/d/yyd01e00/8
Creating    /content/drive/My Drive/image_data_TL/data_final/imagesy/y/y/d/yyd02f00  O
Extracting  /content/drive/My Drive/image_data_TL/data_final/imagesy/y/y/d/yyd02f00/o
Creating    /content/drive/My Drive/image_data_TL/data_final/imagesy/y/y/d/yyd15c00  O
Extracting  /content/drive/My Drive/image_data_TL/data_final/imagesy/y/y/d/yyd15c00/2O
Creating    /content/drive/My Drive/image_data_TL/data_final/imagesy/y/y/e  OK
Creating    /content/drive/My Drive/image_data_TL/data_final/imagesy/y/y/e/yye52f00  O
Extracting  /content/drive/My Drive/image_data_TL/data_final/imagesy/y/y/e/yye52f00/tO
Creating    /content/drive/My Drive/image_data_TL/data_final/imagesy/y/y/e/yye67c00  O
Extracting  /content/drive/My Drive/image_data_TL/data_final/imagesy/y/y/e/yye67c00/2O
Creating    /content/drive/My Drive/image_data_TL/data_final/imagesy/y/y/f  OK
Creating    /content/drive/My Drive/image_data_TL/data_final/imagesy/y/y/f/yyf89d00  O
Extracting  /content/drive/My Drive/image_data_TL/data_final/imagesy/y/y/f/yyf89d00/5O
Creating    /content/drive/My Drive/image_data_TL/data_final/imagesy/y/y/g  OK
```

```
#now we load all the  labels
df = pd.read_csv('/content/drive/My Drive/labels_final.csv')
df.head()
```

	path	label
0	imagesv/v/o/h/voh71d00/509132755+-2755.tif	3
1	imagesl/l/x/t/lxt19d00/502213303.tif	3
2	75325674.tif	2
3	imageso/o/j/b/ojb60d00/517511301+-1301.tif	3
4	imagesq/q/z/k/qzk17e00/2031320195.tif	7

Saved successfully!

```
#from above we can see that the labels of the images are in numbers. So, with the info in ref
#convert them understandable categories through their names
dict_of_labels = {
    0: 'letter',
    1: 'form',
```

```
    2: 'email',
    3: 'handwritten',
    4: 'advertisement',
    5: 'scientific report',
    6: 'scientific publication',
    7: 'specification',
    8: 'file folder',
    9: 'news article',
    10: 'budget',
    11: 'invoice',
    12: 'presentation',
    13: 'questionnaire',
    14: 'resume',
    15: 'memo'

}

df['label']=df['label'].apply(lambda x:dict_of_labels[x])
df.head()
```

	path	label
0	imagesv/v/o/h/voh71d00/509132755+-2755.tif	handwritten
1	imagesl/l/x/t/lxt19d00/502213303.tif	handwritten
2	imagesx/x/e/d/xed05a00/2075325674.tif	email
3	imageso/o/j/b/ojb60d00/517511301+-1301.tif	handwritten
4	imagesq/q/z/k/qzk17e00/2031320195.tif	specification

```
from  keras_preprocessing.image import ImageDataGenerator #import the module
imgdatagen = ImageDataGenerator(rescale=1/255,validation_split=0.2)

import os
os.getcwd()
```

```
'/content'
```

Saved successfully!

```
train_gen = imgdatagen.flow_from_dataframe(dataframe=df,directory='/content/drive/My Drive/im
#directory: string,path to the target directory that contains     all the images mapped in th
```

```
Found 9600 non-validated image filenames belonging to 16 classes.
```

```
val_gen = imgdatagen.flow_from_dataframe(dataframe=df,directory='content/drive/My Drive/image
```

```
Found 9600 non-validated image filenames belonging to 16 classes.
```

```
from keras.layers import Input,Dense,Lambda,Flatten  #The Lambda layer exists so that arbitra
from keras.models import Model
from keras.applications.vgg16 import VGG16
from keras.applications.vgg16 import preprocess_input
from keras.preprocessing import image
from keras.layers import Dense,MaxPool2D,Conv2D,Flatten
from keras.callbacks import Callback
from keras.callbacks import TensorBoard


%load_ext tensorboard


logdir="logs/fit/" + datetime.datetime.now().strftime("%Y%m%d=%H%M%S")
tensorboard_callback = TensorBoard(log_dir = logdir,histogram_freq=1)


image_size=[256,256]
model = VGG16(input_shape=image_size + [3],weights='imagenet',include_top=False) #input_shape
model.summary()
```

```
Model: "vgg16"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_1 (InputLayer)        [(None, 256, 256, 3)]     0

 block1_conv1 (Conv2D)       (None, 256, 256, 64)      1792

 block1_conv2 (Conv2D)       (None, 256, 256, 64)      36928

 block1_pool (MaxPooling2D)  (None, 128, 128, 64)      0

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

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

 block2_pool (MaxPooling2D)  (None, 64, 64, 128)       0

 block3_conv1 (Conv2D)       (None, 64, 64, 256)       295168

                             one, 64, 64, 256)         590080

                             one, 64, 64, 256)         590080

 block3_pool (MaxPooling2D)  (None, 32, 32, 256)       0

 block4_conv1 (Conv2D)       (None, 32, 32, 512)       1180160

 block4_conv2 (Conv2D)       (None, 32, 32, 512)       2359808

 block4_conv3 (Conv2D)       (None, 32, 32, 512)       2359808

 block4_pool (MaxPooling2D)  (None, 16, 16, 512)       0
```

Saved successfully!

```
 block5_conv1 (Conv2D)       (None, 16, 16, 512)       2359808

 block5_conv2 (Conv2D)       (None, 16, 16, 512)       2359808

 block5_conv3 (Conv2D)       (None, 16, 16, 512)       2359808

 block5_pool (MaxPooling2D)  (None, 8, 8, 512)         0

=================================================================
Total params: 14,714,688
Trainable params: 14,714,688
Non-trainable params: 0
_________________________________________________________________
```

Model 1

```
for layer in model.layers:
  layer.trainable = False
x= model.output
x=Conv2D(filters=512,kernel_size=(3,3),padding='same',activation='relu')(x)
x=MaxPool2D(2,2)(x)
x=Flatten()(x)
x=Dense(256,activation='relu')(x)
x=Dense(128,activation='relu')(x)
output=Dense(16,activation='softmax')(x)
model_1 = Model(inputs=model.input,outputs=output)
model_1.compile(loss='categorical_crossentropy',optimizer='Adam',metrics=['accuracy'])
model_1.summary()
```

```
Model: "model"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_1 (InputLayer)        [(None, 256, 256, 3)]     0

 block1_conv1 (Conv2D)       (None, 256, 256, 64)      1792

 block1_conv2 (Conv2D)       (None, 256, 256, 64)      36928

 block1_pool (MaxPooling2D)  (None, 128, 128, 64)      0

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

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

 block2_pool (MaxPooling2D)  (None, 64, 64, 128)       0

 block3_conv1 (Conv2D)       (None, 64, 64, 256)       295168

 block3_conv2 (Conv2D)       (None, 64, 64, 256)       590080

 block3_conv3 (Conv2D)       (None, 64, 64, 256)       590080
```

Saved successfully!

```
 block3_pool (MaxPooling2D)  (None, 32, 32, 256)       0

 block4_conv1 (Conv2D)       (None, 32, 32, 512)       1180160

 block4_conv2 (Conv2D)       (None, 32, 32, 512)       2359808

 block4_conv3 (Conv2D)       (None, 32, 32, 512)       2359808

 block4_pool (MaxPooling2D)  (None, 16, 16, 512)       0

 block5_conv1 (Conv2D)       (None, 16, 16, 512)       2359808

 block5_conv2 (Conv2D)       (None, 16, 16, 512)       2359808

 block5_conv3 (Conv2D)       (None, 16, 16, 512)       2359808

 block5_pool (MaxPooling2D)  (None, 8, 8, 512)         0

 conv2d (Conv2D)             (None, 8, 8, 512)         2359808

 max_pooling2d (MaxPooling2D  (None, 4, 4, 512)        0
 )

 flatten (Flatten)           (None, 8192)              0

 dense (Dense)               (None, 256)               2097408

 dense_1 (Dense)             (None, 128)               32896

 dense_2 (Dense)             (None, 16)                2064

=================================================================
Total params: 19,206,864
Trainable params: 4,492,176
```

```
train_steps = train_gen.n//train_gen.batch_size
val_steps = val_gen.n//val_gen.batch_size
print(train_steps,val_steps)
```

```
300 300
```

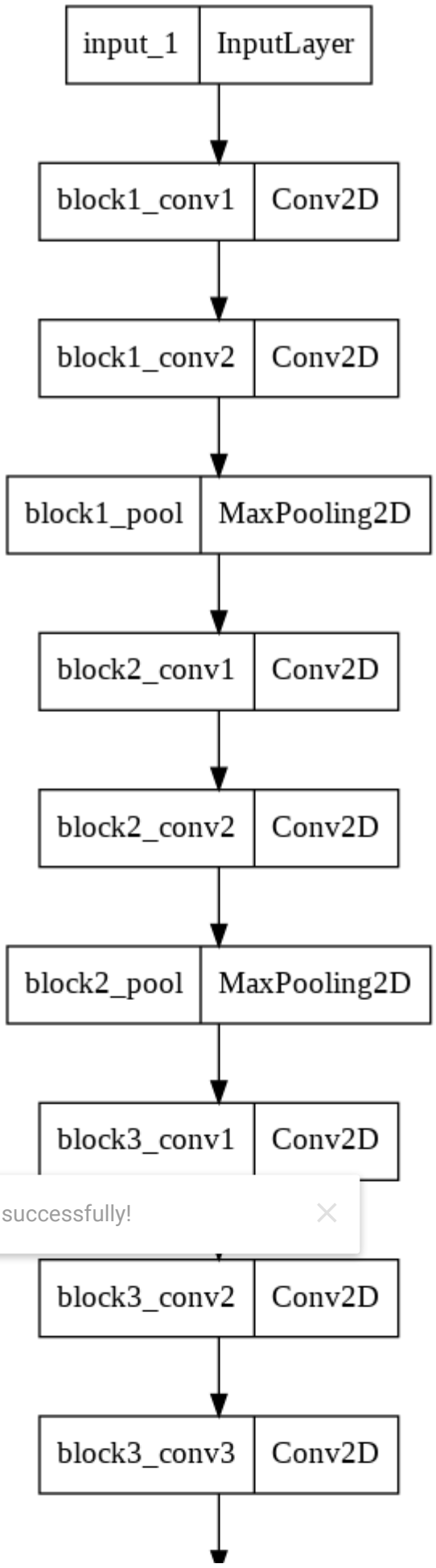
Saved successfully!

```
eps_per_epoch=train_steps,epochs=3,validation_data=val_ge
  pass
```

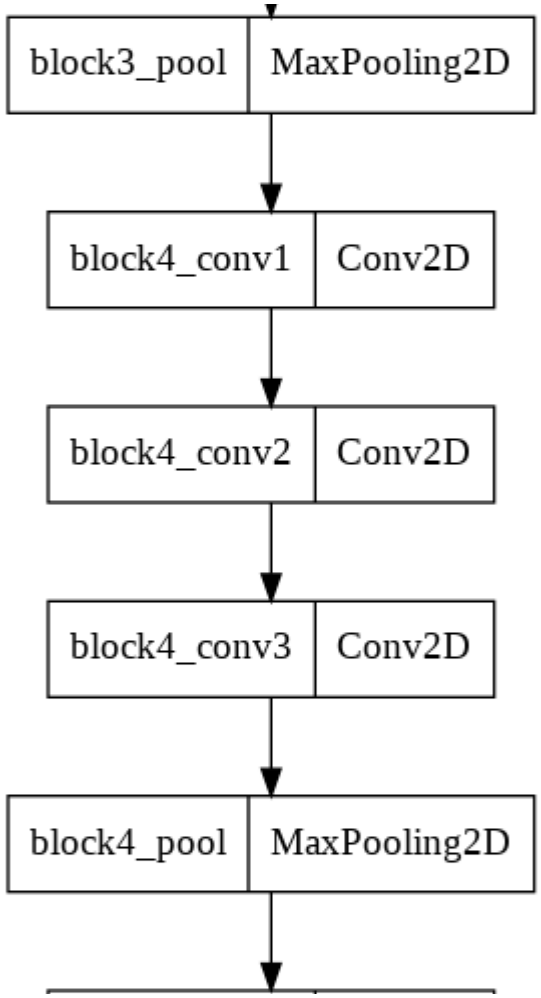
```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:2: UserWarning: `Model.fit_

Epoch 1/3
300/300 [==============================] - ETA: 0s - loss: 1.1655 - accuracy: 0.6402
```

```
#graph
```

```
tf.keras.utils.plot_model(model_1,to_file='model_1.png',show_shapes=False,show_layer_names=Tr
```

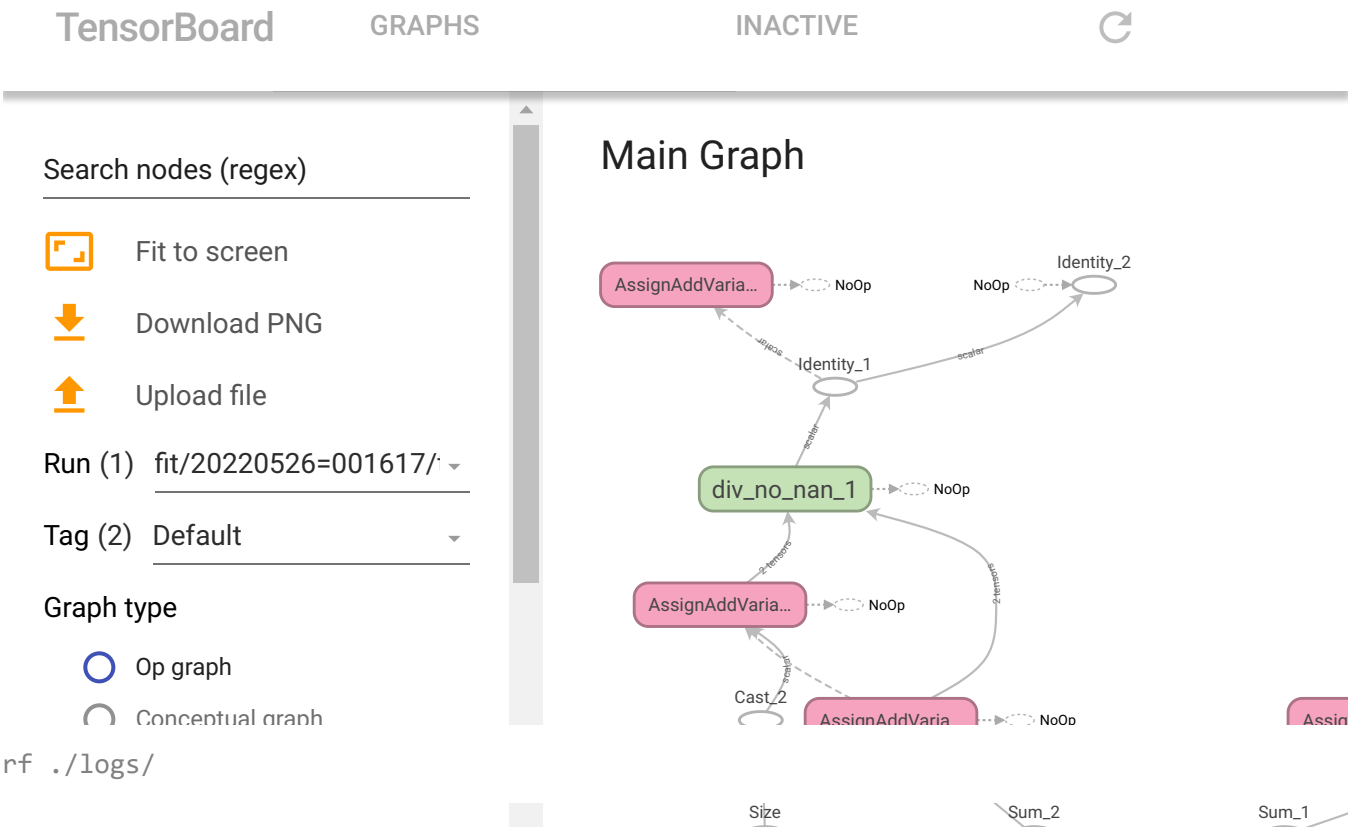


Saved successfully!




```
%tensorboard --logdir logs
```

Saved successfully!




```
!rm -rf ./logs/
```

Key takeaway:

1. The total parameters to trainable parameter ratio is around 4.3.
2. For that combination, we get an accuracy of 64%

Model 2

```
for layer in model.layers:
  layer.trainable = False
x= model.output
x=Conv2D(filters=4096,kernel_size=8,strides=1,activation='relu')(x)
x=Conv2D(filters=4096,kernel_size=1,strides=1,activation='relu')(x)
```

Saved successfully!

```
)(x)
model_2=Model(inputs=model.input,outputs=output)
model_2.compile(loss='categorical_crossentropy',optimizer='Adam',metrics=['accuracy'])
model_2.summary()
```

```
Model: "model"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_1 (InputLayer)        [(None, 256, 256, 3)]     0

 block1_conv1 (Conv2D)       (None, 256, 256, 64)      1792
```

```
block1_conv2 (Conv2D)       (None, 256, 256, 64)      36928

block1_pool (MaxPooling2D)  (None, 128, 128, 64)      0

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

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

block2_pool (MaxPooling2D)  (None, 64, 64, 128)       0

block3_conv1 (Conv2D)       (None, 64, 64, 256)       295168

block3_conv2 (Conv2D)       (None, 64, 64, 256)       590080

block3_conv3 (Conv2D)       (None, 64, 64, 256)       590080

block3_pool (MaxPooling2D)  (None, 32, 32, 256)       0

block4_conv1 (Conv2D)       (None, 32, 32, 512)       1180160

block4_conv2 (Conv2D)       (None, 32, 32, 512)       2359808

block4_conv3 (Conv2D)       (None, 32, 32, 512)       2359808

block4_pool (MaxPooling2D)  (None, 16, 16, 512)       0

block5_conv1 (Conv2D)       (None, 16, 16, 512)       2359808

block5_conv2 (Conv2D)       (None, 16, 16, 512)       2359808

block5_conv3 (Conv2D)       (None, 16, 16, 512)       2359808

block5_pool (MaxPooling2D)  (None, 8, 8, 512)         0

conv2d (Conv2D)             (None, 1, 1, 4096)        134221824

conv2d_1 (Conv2D)           (None, 1, 1, 4096)        16781312

flatten (Flatten)           (None, 4096)              0

dense (Dense)               (None, 16)                65552

=================================================================
Trainable params: 151,068,688
Non-trainable params: 14,714,688
_________________________________________________________________
```

Saved successfully!

```
try:
  model_2.fit_generator(train_gen,steps_per_epoch=train_steps,epochs=3,verbose=1,validation_d
except FileNotFoundError:
  pass
```

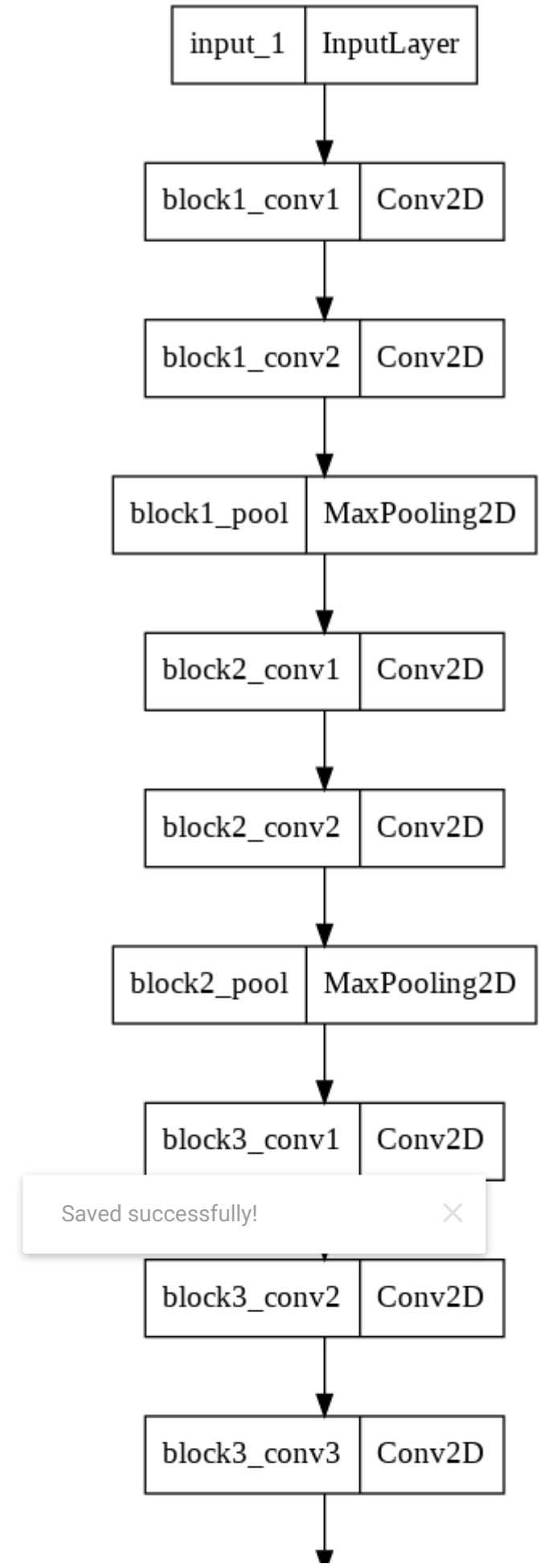
```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:2: UserWarning: `Model.fit_

Epoch 1/3
 13/300 [>.............................] - ETA: 15:53:10 - loss: 1.0707 - accuracy: 0.65
```

```
'''try:
  model_2.fit_generator(train_gen,steps_per_epoch=train_steps,epochs=3,verbose=1,validation_d
except FileNotFoundError:
  pass  '''
```

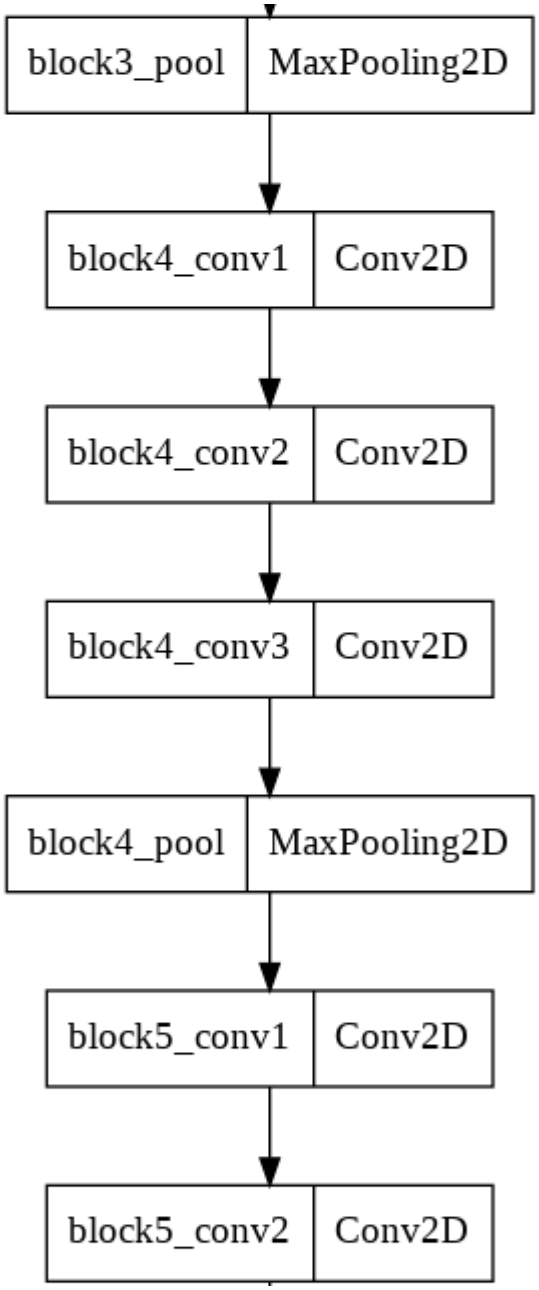
```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:2: UserWarning: `Model.fit_

Epoch 1/3
300/300 [==============================] - ETA: 0s - loss: 2.1016 - accuracy: 0.4876
```

```
#graph
tf.keras.utils.plot_model(model_2,to_file='model_2.png',show_shapes=False,show_layer_names=Tr
```

Saved successfully!



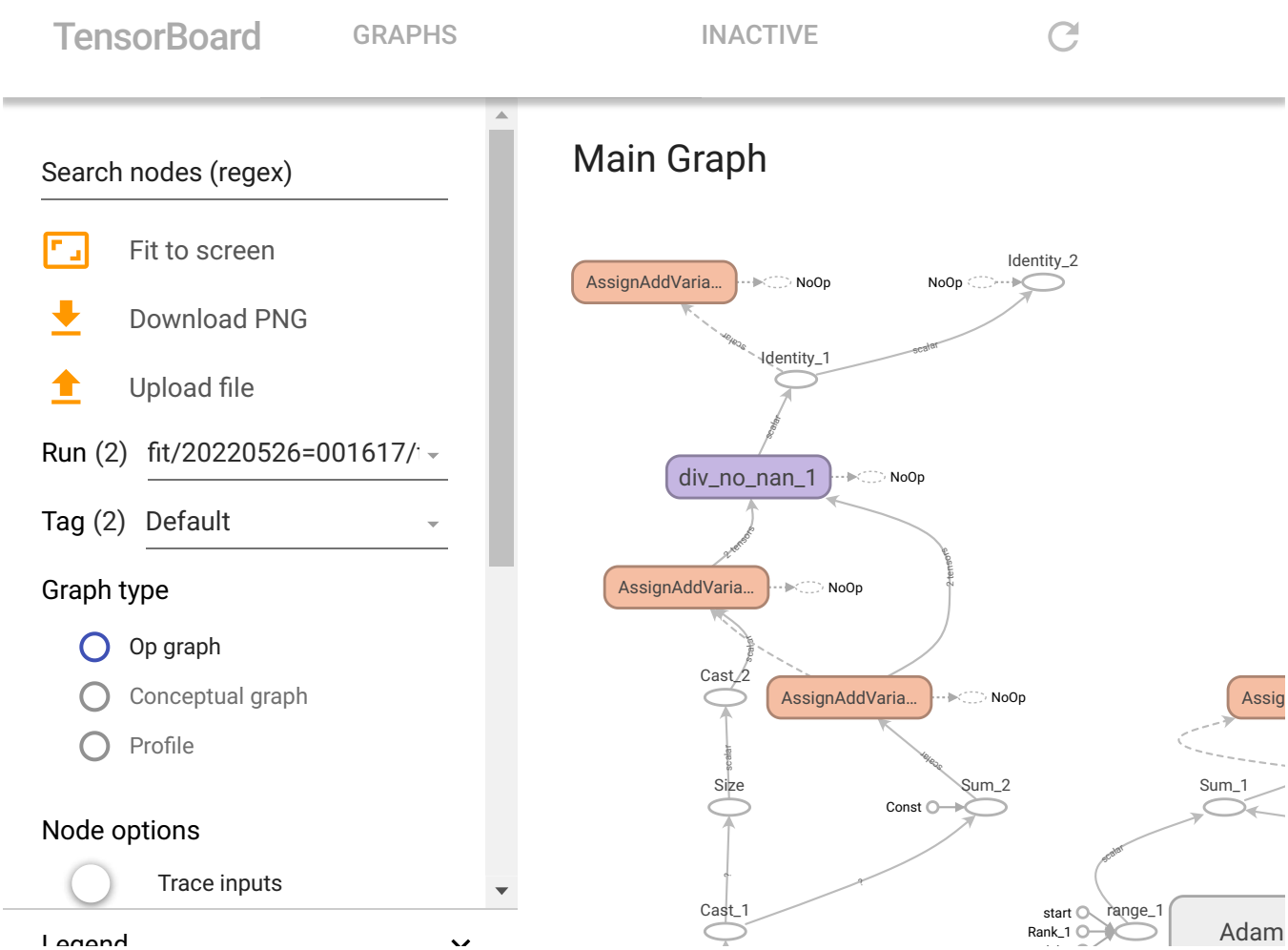
Saved successfully!




```
%tensorboard --logdir logs
```

Saved successfully!

```
Reusing TensorBoard on port 6006 (pid 1029), started 1:52:34 ago. (Use '!kill 1029' to
kill it.)
```




```
!rm -rf ./logs/
```

colors same substructure

Key takeaway:

1. The total parameter to trainable parameter ratio is around 1.097 (very close).
2. For that ratio, we see the accuracy is around 65.14% (though the 2nd epoch was interrupted abruptly due to RAM insufficiency).
3. Here, what we need to take note is that we have paramters in this model than model_1. It's del is secondary to the fact that most parameters are
4. So, we can conclude that more trainable parameters have positive effect on the accuracy.

Saved successfully!

Model 3

```
for layer in model.layers[-6:]:
```

```
  layer.trainable = True

x = model.output
x = Conv2D(filters=4096,kernel_size = 8,strides=1,activation='relu')(x)
x = Conv2D(filters=4096,kernel_size = 1,strides=1,activation='relu')(x)
x = Flatten()(x)
output = Dense(16,activation='softmax')(x)
model_3 = Model(inputs=model.input,outputs=output)
model_3.compile(loss='categorical_crossentropy',optimizer='Adam',metrics=['accuracy'])


model_3.summary()
```

```
Model: "model_3"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_1 (InputLayer)        [(None, 256, 256, 3)]     0

 block1_conv1 (Conv2D)       (None, 256, 256, 64)      1792

 block1_conv2 (Conv2D)       (None, 256, 256, 64)      36928

 block1_pool (MaxPooling2D)  (None, 128, 128, 64)      0

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

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

 block2_pool (MaxPooling2D)  (None, 64, 64, 128)       0

 block3_conv1 (Conv2D)       (None, 64, 64, 256)       295168

 block3_conv2 (Conv2D)       (None, 64, 64, 256)       590080

 block3_conv3 (Conv2D)       (None, 64, 64, 256)       590080

 block3_pool (MaxPooling2D)  (None, 32, 32, 256)       0

 block4_conv1 (Conv2D)       (None, 32, 32, 512)       1180160

 block4_conv2 (Conv2D)       (None, 32, 32, 512)       2359808

 block4_conv3 (Conv2D)       (None, 32, 32, 512)       2359808

 block4_pool (MaxPooling2D)  (None, 16, 16, 512)       0

 block5_conv1 (Conv2D)       (None, 16, 16, 512)       2359808

 block5_conv2 (Conv2D)       (None, 16, 16, 512)       2359808

 block5_conv3 (Conv2D)       (None, 16, 16, 512)       2359808

 block5_pool (MaxPooling2D)  (None, 8, 8, 512)         0
```

```
 conv2d_6 (Conv2D)           (None, 1, 1, 4096)        134221824

 conv2d_7 (Conv2D)           (None, 1, 1, 4096)        16781312

 flatten_3 (Flatten)         (None, 4096)              0

 dense_3 (Dense)             (None, 16)                65552

=================================================================
Total params: 165,783,376
Trainable params: 160,507,920
Non-trainable params: 5,275,456
_________________________________________________________________
```

```
try:
  model_3.fit_generator(train_gen,steps_per_epoch = train_steps,epochs=3,validation_data=val_
except FileNotFoundError:
  pass
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:2: UserWarning: `Model.fit_

Epoch 1/3
300/300 [==============================] - ETA: 0s - loss: 2.7727 - accuracy: 0.0625
```

```
tf.keras.utils.plot_model(model_3,to_file='model_3.png',show_shapes=False,show_layer_names=Tr
```

Saved successfully!