In [1]:
```python
import os
import tensorflow as tf
import numpy as np
import pandas as pd
import pickle
import datetime
#import openCV
import cv2
import pandas as pd
from sklearn.model_selection import train_test_split
from tensorflow.keras.applications import VGG16
from tensorflow.keras.layers import Dense,Input,Conv2D,MaxPool2D,Activation,Dropout,Flatten
from tensorflow.keras.layers import Activation, Dropout, Flatten, Dense,Input
from tensorflow.keras.models import Model
```

In [2]:
```
!wget --header="Host: doc-0o-84-docs.googleusercontent.com" --header="User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64
```

```
--2021-05-10 20:16:12--  https://doc-0o-84-docs.googleusercontent.com/docs/securesc/n5tequvs1gm3hik1bov42ncmvls592ur/
vuaqjmtht4p3tcr4cic08bm76j35gnc0/1620677700000/00484516897554883881/06982383997589471526/1Z4TyI7FcFVEx8qdl4jO9qxvxaqL
SqoEu?e=download&authuser=0&nonce=4fmumi779fons&user=06982383997589471526&hash=jnjdck722h1vebajedgqe8ubr2ipntp2
Resolving doc-0o-84-docs.googleusercontent.com (doc-0o-84-docs.googleusercontent.com)... 142.251.33.193, 2607:f8b0:40
04:837::2001
Connecting to doc-0o-84-docs.googleusercontent.com (doc-0o-84-docs.googleusercontent.com)|142.251.33.193|:443... conn
ected.
HTTP request sent, awaiting response... 200 OK
Length: unspecified [application/rar]
Saving to: ‘rvl-cdip.rar’

rvl-cdip.rar            [          <=>        ]   4.34G  27.3MB/s    in 92s

2021-05-10 20:17:44 (48.3 MB/s) - ‘rvl-cdip.rar’ saved [4660541790]
```

In [3]:
```python
get_ipython().system_raw("unrar x rvl-cdip.rar")
```

In [4]:
```python
df=pd.read_csv(r"/content/labels_final.csv",dtype=str)
```

In [5]:
```python
df.head()
```

Out[5]:

	path	label

	path	label
0	imagesv/v/o/h/voh71d00/509132755+-2755.tif	3
1	imagesl/l/x/t/lxt19d00/502213303.tif	3
2	imagesx/x/e/d/xed05a00/2075325674.tif	2
3	imageso/o/j/b/ojb60d00/517511301+-1301.tif	3
4	imagesq/q/z/k/qzk17e00/2031320195.tif	7

In [6]:
```
dir_path = 'data_final'
```

In [7]:
```
from sklearn.model_selection import train_test_split
```

In [8]:
```
x_train,x_valid=train_test_split(df,test_size=0.30, random_state=42)
```

In [9]:
```
datagen=tf.keras.preprocessing.image.ImageDataGenerator(rescale=1./255)
```

In [10]:
```
# https://vijayabhaskar96.medium.com/tutorial-on-keras-flow-from-dataframe-1fd4493d237c
train_generator=datagen.flow_from_dataframe(
dataframe=x_train,
directory="/content/data_final",
x_col="path",
y_col="label",
batch_size=50,
seed=42,
shuffle=True,
class_mode="categorical",
target_size=(224,224))
```

```
Found 33600 validated image filenames belonging to 16 classes.
```

In [11]:
```
# https://vijayabhaskar96.medium.com/tutorial-on-keras-flow-from-dataframe-1fd4493d237c
valid_generator=datagen.flow_from_dataframe(
dataframe=x_valid,
directory="/content/data_final",
x_col="path",
y_col="label",
batch_size=50,
seed=42,
shuffle=True,
class_mode="categorical",
target_size=(224,224))
```

```
Found 14400 validated image filenames belonging to 16 classes.
```

Model 1

In [12]:

```
from keras.models import Sequential
from tensorflow.keras.models import Model
from keras.layers import Dense, Activation, Flatten, Dropout, Input
from keras.layers import Conv2D, MaxPooling2D
from keras import regularizers, optimizers
from keras.applications.vgg16 import VGG16
from tensorflow.keras.callbacks import ReduceLROnPlateau, EarlyStopping, ModelCheckpoint
import datetime
```

In [ ]:

```
#Use VGG-16 pretrained network without Fully Connected layers and initilize all the weights with Imagenet trained we
vgg_=VGG16(weights='imagenet',include_top=False,input_shape=(224,224,3))
```

```
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/vgg16/vgg16_weights_tf_dim_orderin
g_tf_kernels_notop.h5
58892288/58889256 [==============================] - 1s 0us/step
```

In [ ]:

```
vgg_.trainable=False
```

In [ ]:

```
vgg_.summary()
```

```
Model: "vgg16"
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
input_1 (InputLayer)         [(None, 224, 224, 3)]     0         
_________________________________________________________________
block1_conv1 (Conv2D)        (None, 224, 224, 64)      1792      
_________________________________________________________________
block1_conv2 (Conv2D)        (None, 224, 224, 64)      36928     
_________________________________________________________________
block1_pool (MaxPooling2D)   (None, 112, 112, 64)      0         
_________________________________________________________________
```

```
block2_conv1 (Conv2D)        (None, 112, 112, 128)     73856     
_________________________________________________________________
block2_conv2 (Conv2D)        (None, 112, 112, 128)     147584    
_________________________________________________________________
block2_pool (MaxPooling2D)   (None, 56, 56, 128)       0         
_________________________________________________________________
block3_conv1 (Conv2D)        (None, 56, 56, 256)       295168    
_________________________________________________________________
block3_conv2 (Conv2D)        (None, 56, 56, 256)       590080    
_________________________________________________________________
block3_conv3 (Conv2D)        (None, 56, 56, 256)       590080    
_________________________________________________________________
block3_pool (MaxPooling2D)   (None, 28, 28, 256)       0         
_________________________________________________________________
block4_conv1 (Conv2D)        (None, 28, 28, 512)       1180160   
_________________________________________________________________
block4_conv2 (Conv2D)        (None, 28, 28, 512)       2359808   
_________________________________________________________________
block4_conv3 (Conv2D)        (None, 28, 28, 512)       2359808   
_________________________________________________________________
block4_pool (MaxPooling2D)   (None, 14, 14, 512)       0         
_________________________________________________________________
block5_conv1 (Conv2D)        (None, 14, 14, 512)       2359808   
_________________________________________________________________
block5_conv2 (Conv2D)        (None, 14, 14, 512)       2359808   
_________________________________________________________________
block5_conv3 (Conv2D)        (None, 14, 14, 512)       2359808   
_________________________________________________________________
block5_pool (MaxPooling2D)   (None, 7, 7, 512)         0         
=================================================================
Total params: 14,714,688
Trainable params: 0
Non-trainable params: 14,714,688
_________________________________________________________________
```

In [ ]:

```
#After VGG-16 network without FC layers, add a new Conv block ( 1 Conv layer and 1 Maxpooling ), 2 FC layers and a ou
# You are free to choose any hyperparameters/parameters of conv block, FC layers, output layer

#Input layer
input_layer = Input(shape=(224,224,3),name='Input_Layer')
vgg16_layer= vgg_(input_layer)


#vgg16_layer1 =vgg16_layer.layers[-1].output
```

```
#Conv Layer
Conv1 = Conv2D(filters=256,kernel_size=(3,3),strides=(1,1),padding='valid',
              activation='relu',kernel_initializer=tf.keras.initializers.he_normal(seed=23),name='Conv1')(vgg16_laye

#MaxPool Layer
Pool1 = MaxPool2D(pool_size=(2,2),strides=(2,2),padding='valid',name='Pool1')(Conv1)

#Flatten
flatten = Flatten(name='Flatten')(Pool1)
drop=Dropout(0.5)(flatten)

#FC layer
FC1 = Dense(units=512,activation='relu',kernel_initializer=tf.keras.initializers.he_normal(seed=25),name='FC1')(drop

#FC layer
FC2 = Dense(units=32,activation='relu',kernel_initializer=tf.keras.initializers.he_normal(seed=18),name='FC2')(FC1)

#output layer
Out = Dense(units=16,activation='softmax',kernel_initializer=tf.keras.initializers.glorot_normal(seed=29),name='Outpu

#Creating a model
model = Model(inputs=input_layer,outputs=Out)
#model1 = Model(inputs = vgg16_layer, outputs = Out)
```

In [ ]:

```
#compiling
model.compile(optimizer=tf.keras.optimizers.Adam(lr=0.0001),loss='categorical_crossentropy',metrics=['accuracy'])
```

In [ ]:

```
model.summary()
```

```
Model: "model"
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
Input_Layer (InputLayer)     [(None, 224, 224, 3)]     0         
_________________________________________________________________
vgg16 (Functional)           (None, 7, 7, 512)         14714688  
_________________________________________________________________
Conv1 (Conv2D)               (None, 5, 5, 256)         1179904   
_________________________________________________________________
Pool1 (MaxPooling2D)         (None, 2, 2, 256)         0         
_________________________________________________________________
Flatten (Flatten)            (None, 1024)              0         
```

```
_________________________________________________________________
dropout (Dropout)            (None, 1024)              0
_________________________________________________________________
FC1 (Dense)                  (None, 512)               524800
_________________________________________________________________
FC2 (Dense)                  (None, 32)                16416
_________________________________________________________________
Output (Dense)               (None, 16)                528
=================================================================
Total params: 16,436,336
Trainable params: 1,721,648
Non-trainable params: 14,714,688
_________________________________________________________________
```

In [ ]:
```
%load_ext tensorboard
```

In [ ]:
```
!rm -rf ./logs/
import datetime
```

In [ ]:
```
import os
logdir = os.path.join("logs", datetime.datetime.now().strftime("%Y%m%d-%H%M%S"))
tensorboard_callback = tf.keras.callbacks.TensorBoard(logdir, histogram_freq=1,write_graph=True,write_grads=True)
```

```
WARNING:tensorflow:`write_grads` will be ignored in TensorFlow 2.0 for the `TensorBoard` Callback.
```

In [ ]:
```
logdir
```

Out[ ]:
```
'logs/20210510-181438'
```

In [ ]:
```
STEP_SIZE_TRAIN=train_generator.n//train_generator.batch_size
STEP_SIZE_VALID=valid_generator.n//valid_generator.batch_size

model.fit(train_generator,
          steps_per_epoch=STEP_SIZE_TRAIN,
          validation_data=valid_generator,
          validation_steps=STEP_SIZE_VALID,
          epochs=6,
          callbacks=[tensorboard_callback])
```

```
Epoch 1/6
672/672 [==============================] - 314s 407ms/step - loss: 2.3492 - accuracy: 0.2587 - val_loss: 1.4531 - val
_accuracy: 0.5648
```

```
Epoch 2/6
672/672 [==============================] - 265s 394ms/step - loss: 1.4684 - accuracy: 0.5512 - val_loss: 1.2340 - val
_accuracy: 0.6253
Epoch 3/6
672/672 [==============================] - 255s 379ms/step - loss: 1.2562 - accuracy: 0.6178 - val_loss: 1.1230 - val
_accuracy: 0.6621
Epoch 4/6
672/672 [==============================] - 238s 353ms/step - loss: 1.1442 - accuracy: 0.6536 - val_loss: 1.0753 - val
_accuracy: 0.6747
Epoch 5/6
672/672 [==============================] - 228s 339ms/step - loss: 1.0619 - accuracy: 0.6795 - val_loss: 1.0315 - val
_accuracy: 0.6878
Epoch 6/6
672/672 [==============================] - 223s 332ms/step - loss: 0.9936 - accuracy: 0.6944 - val_loss: 0.9940 - val
_accuracy: 0.7017
```

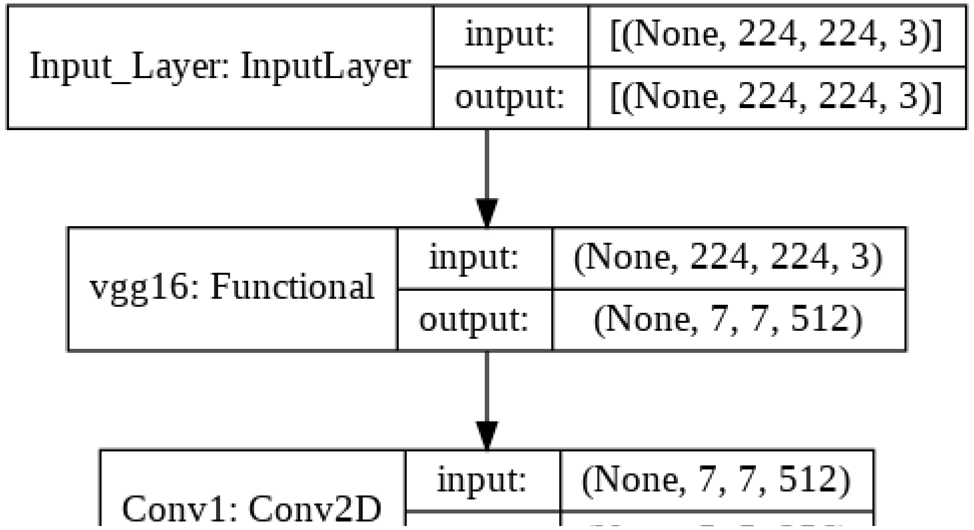
```
Out[ ]: <tensorflow.python.keras.callbacks.History at 0x7f09d4223a50>
```

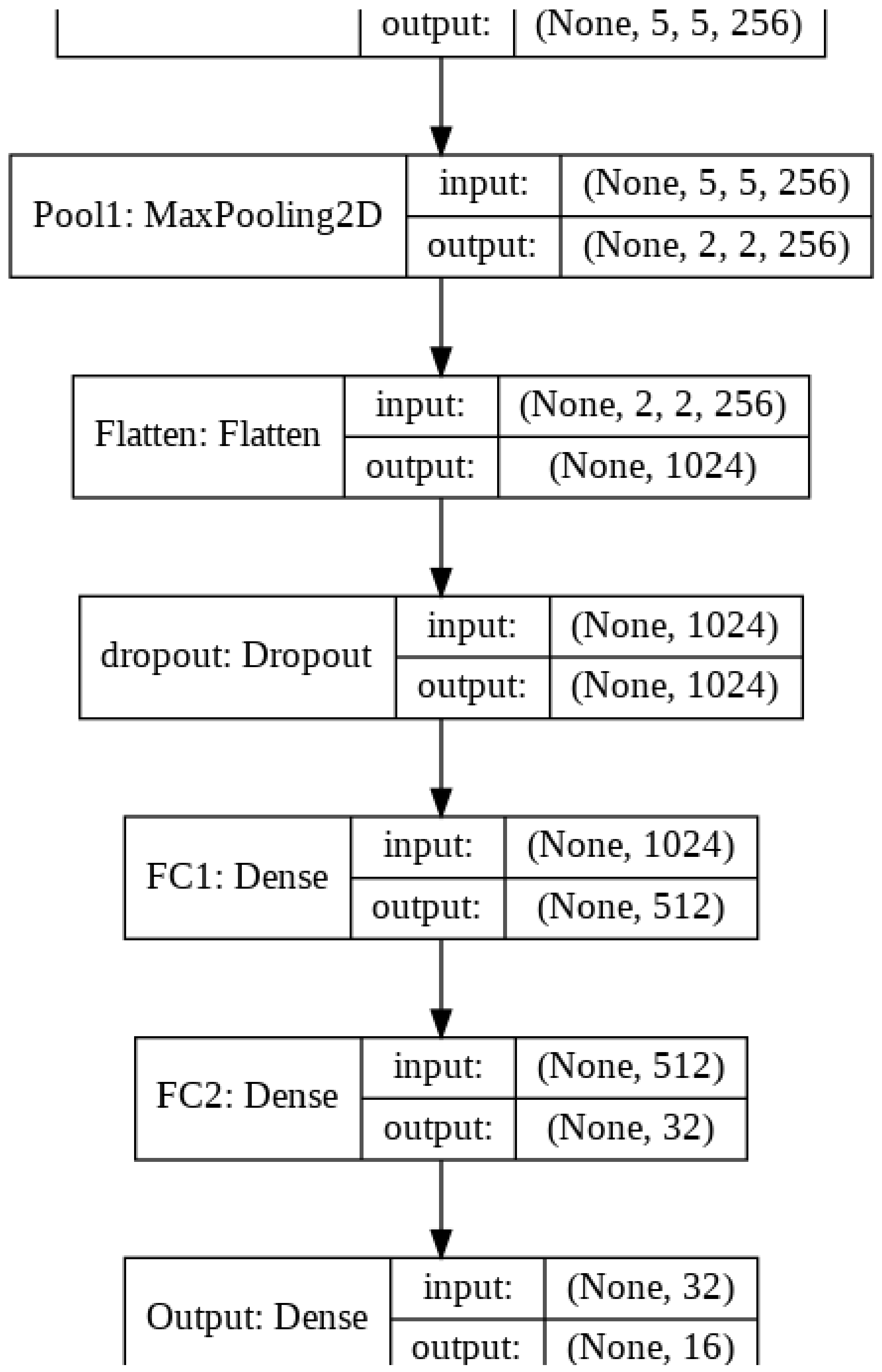
```
In [ ]: %load_ext tensorboard
        %tensorboard --logdir $logdir
```

```
The tensorboard extension is already loaded. To reload it, use:
  %reload_ext tensorboard
```

```
In [ ]: from tensorflow.keras.utils import plot_model
        plot_model(model, 'model.png', show_shapes=True)
```

Out[ ]:




output:
(None, 5, 5, 256)
Pool1: MaxPooling2D
input:
(None, 5, 5, 256)
output:
(None, 2, 2, 256)
Flatten: Flatten
input:
(None, 2, 2, 256)
output:
(None, 1024)
dropout: Dropout
input:
(None, 1024)
output:
(None, 1024)
FC1: Dense
input:
(None, 1024)
output:
(None, 512)
FC2: Dense
input:
(None, 512)
output:
(None, 32)
Output: Dense
input:
(None, 32)
output:
(None, 16)

Model 2

In [ ]:
```
tf.keras.backend.clear_session()
```

In [ ]:
```
import os
logdir2 = os.path.join("logs", datetime.datetime.now().strftime("%Y%m%d-%H%M%S"))
tensorboard_callback2 = tf.keras.callbacks.TensorBoard(logdir2, histogram_freq=1,write_graph=True,write_grads=True)
```

```
WARNING:tensorflow:`write_grads` will be ignored in TensorFlow 2.0 for the `TensorBoard` Callback.
```

In [ ]:
```
logdir2
```

Out[ ]:
```
'logs/20210510-184646'
```

In [ ]:
```
#After VGG-16 network without FC layers, add a new Conv block ( 1 Conv layer and 1 Maxpooling ), 2 FC layers and a ou
# You are free to choose any hyperparameters/parameters of conv block, FC layers, output layer

#Input layer
input_layer = Input(shape=(224,224,3),name='Input_Layer')
vgg16_layer= vgg_(input_layer)


#vgg16_layer1 =vgg16_layer.layers[-1].output
#Conv Layer
Conv1 = Conv2D(filters=4096,kernel_size=(7,7),strides=(1,1),padding='valid',
              activation='relu',kernel_initializer=tf.keras.initializers.he_normal(seed=25),name='Conv1')(vgg16_layer

#Conv Layer
Conv2 = Conv2D(filters=4096,kernel_size=(1,1),strides=(1,1),padding='valid',
              activation='relu',kernel_initializer=tf.keras.initializers.he_normal(seed=27),name='Conv2')(Conv1)

#Conv Layer
Conv3 = Conv2D(filters=4096,kernel_size=(1,1),strides=(1,1),padding='valid',
              activation='relu',kernel_initializer=tf.keras.initializers.he_normal(seed=27),name='Conv3')(Conv2)


#Flatten
flatten = Flatten(name='Flatten')(Conv3)
```

```
drop=Dropout(0.5)(flatten)


#output layer
Out = Dense(units=16,activation='softmax',kernel_initializer=tf.keras.initializers.glorot_normal(seed=29),name='Outpu

#Creating a model
model2 = Model(inputs=input_layer,outputs=Out)
#model1 = Model(inputs = vgg16_layer, outputs = Out)
```

In [ ]:
```
model2.summary()
```

```
Model: "model_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
Input_Layer (InputLayer)     [(None, 224, 224, 3)]     0         
_________________________________________________________________
vgg16 (Functional)           (None, 7, 7, 512)         14714688  
_________________________________________________________________
Conv1 (Conv2D)               (None, 1, 1, 4096)        102764544 
_________________________________________________________________
Conv2 (Conv2D)               (None, 1, 1, 4096)        16781312  
_________________________________________________________________
Conv3 (Conv2D)               (None, 1, 1, 4096)        16781312  
_________________________________________________________________
Flatten (Flatten)            (None, 4096)              0         
_________________________________________________________________
dropout_1 (Dropout)          (None, 4096)              0         
_________________________________________________________________
Output (Dense)               (None, 16)                65552     
=================================================================
Total params: 151,107,408
Trainable params: 136,392,720
Non-trainable params: 14,714,688
_________________________________________________________________
```

Tensorboard 2

In [ ]:
```
#compiling
model2.compile(optimizer=tf.keras.optimizers.Adam(lr=0.0001),loss='categorical_crossentropy',metrics=['accuracy'])
```

In [ ]:
```
STEP_SIZE_TRAIN=train_generator.n//train_generator.batch_size
STEP_SIZE_VALID=valid_generator.n//valid_generator.batch_size

model2.fit(train_generator,
          steps_per_epoch=STEP_SIZE_TRAIN,
          validation_data=valid_generator,
          validation_steps=STEP_SIZE_VALID,
          epochs=7,
          callbacks=[tensorboard_callback])
```

```
Epoch 1/7
672/672 [==============================] - 371s 546ms/step - loss: 1.7567 - accuracy: 0.4742 - val_loss: 1.0914 - val
_accuracy: 0.6680
Epoch 2/7
672/672 [==============================] - 366s 545ms/step - loss: 0.9776 - accuracy: 0.7012 - val_loss: 0.9414 - val
_accuracy: 0.7174
Epoch 3/7
672/672 [==============================] - 368s 547ms/step - loss: 0.7708 - accuracy: 0.7641 - val_loss: 0.9326 - val
_accuracy: 0.7181
Epoch 4/7
672/672 [==============================] - 366s 545ms/step - loss: 0.6365 - accuracy: 0.8045 - val_loss: 0.8730 - val
_accuracy: 0.7370
Epoch 5/7
672/672 [==============================] - 367s 545ms/step - loss: 0.5306 - accuracy: 0.8360 - val_loss: 0.8704 - val
_accuracy: 0.7505
Epoch 6/7
672/672 [==============================] - 366s 545ms/step - loss: 0.4436 - accuracy: 0.8607 - val_loss: 0.8555 - val
_accuracy: 0.7638
Epoch 7/7
672/672 [==============================] - 366s 545ms/step - loss: 0.3660 - accuracy: 0.8859 - val_loss: 1.0525 - val
_accuracy: 0.7383
```

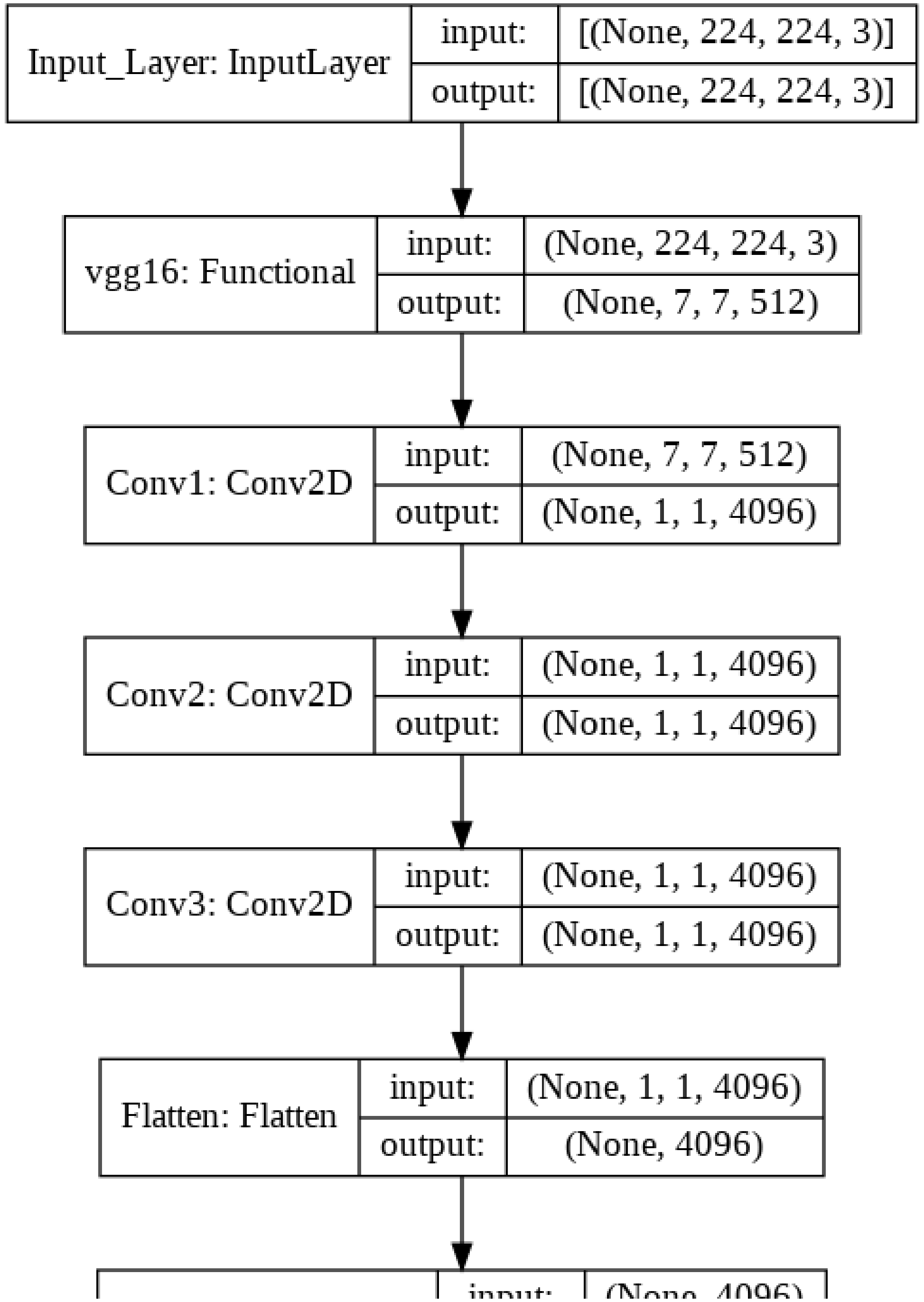
Out[ ]:
```
<tensorflow.python.keras.callbacks.History at 0x7f097fa20810>
```

In [ ]:
```
%load_ext tensorboard
%tensorboard --logdir $logdir
```

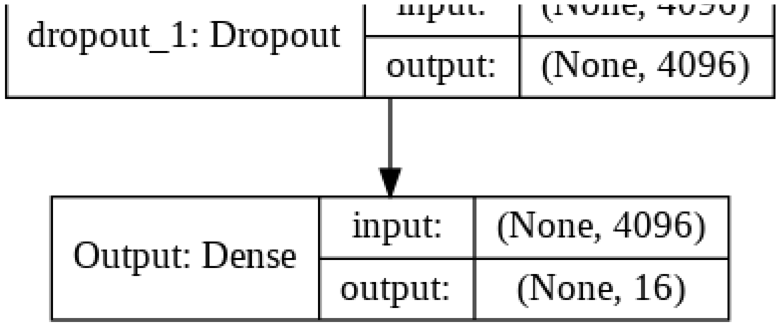
```
The tensorboard extension is already loaded. To reload it, use:
  %reload_ext tensorboard
Reusing TensorBoard on port 6007 (pid 525), started 0:59:59 ago. (Use '!kill 525' to kill it.)
```

In [ ]:
```
from tensorflow.keras.utils import plot_model
plot_model(model2, 'model2.png', show_shapes=True)
```

Out[ ]:



Input_Layer: InputLayer
input: [(None, 224, 224, 3)]
output: [(None, 224, 224, 3)]
vgg16: Functional
input: (None, 224, 224, 3)
output: (None, 7, 7, 512)
Conv1: Conv2D
input: (None, 7, 7, 512)
output: (None, 1, 1, 4096)
Conv2: Conv2D
input: (None, 1, 1, 4096)
output: (None, 1, 1, 4096)
Conv3: Conv2D
input: (None, 1, 1, 4096)
output: (None, 1, 1, 4096)
Flatten: Flatten
input: (None, 1, 1, 4096)
output: (None, 4096)



Model 3

In [ ]:
```
from keras.models import Sequential
from tensorflow.keras.models import Model
from keras.layers import Dense, Activation, Flatten, Dropout, Input
from keras.layers import Conv2D, MaxPooling2D
from keras import regularizers, optimizers
from keras.applications.vgg16 import VGG16
from tensorflow.keras.callbacks import ReduceLROnPlateau, EarlyStopping, ModelCheckpoint
import datetime
```

In [ ]:
```
tf.keras.backend.clear_session()
```

In [15]:
```
import os
logdir3 = os.path.join("logs", datetime.datetime.now().strftime("%Y%m%d-%H%M%S"))
tensorboard_callback3 = tf.keras.callbacks.TensorBoard(logdir3, histogram_freq=1,write_graph=True,write_grads=True)
```

```
WARNING:tensorflow:`write_grads` will be ignored in TensorFlow 2.0 for the `TensorBoard` Callback.
```

In [13]:
```
#Use VGG-16 pretrained network without Fully Connected layers and initilize all the weights with Imagenet trained wei
vgg_3=VGG16(weights='imagenet',include_top=False,input_shape=(224,224,3))
```

```
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/vgg16/vgg16_weights_tf_dim_orderin
g_tf_kernels_notop.h5
58892288/58889256 [==============================] - 0s 0us/step
```

In [14]:
```
vgg_3.trainable
```

Out[14]:
```
True
```

In [35]:
```
for layer in vgg_3.layers[:-6]: # only last 6 layers will be trainable
  layer.trainable=False
```

In [50]:
```
# for layer in vgg_3.layers:
#   print(layer.name , "==",layer.trainable)
```

In [45]:
```
#After VGG-16 network without FC layers, add a new Conv block ( 1 Conv layer and 1 Maxpooling ), 2 FC layers and a ou
# You are free to choose any hyperparameters/parameters of conv block, FC layers, output layer

#input layer
input=Input(shape=(224,224,3))
vgg_layers=vgg_3(input)

#conv layer
layer1=Conv2D(filters=4096 ,kernel_size=(7,7) ,strides=(1, 1),padding='valid',
              activation='relu',kernel_initializer=tf.keras.initializers.he_normal(seed=23))(vgg_layers)

#conv layer
layer2=Conv2D(filters=4096 ,kernel_size=(1,1) ,strides=(1, 1),padding='valid',
              activation='relu',kernel_initializer=tf.keras.initializers.he_normal(seed=28))(layer1)

#flatten
layer3=Flatten()(layer2)
#drop=Dropout(0.5)(flatten)

#output layer
output=Dense(16,activation='softmax',kernel_initializer=tf.keras.initializers.glorot_normal(seed=39))(layer3)

#creating a model
model3 = Model(inputs=input, outputs=output)
```

In [46]:
```
model3.compile(optimizer=tf.keras.optimizers.Adam(lr=0.001),loss='categorical_crossentropy',metrics=['accuracy'])
```

In [47]:
```
model3.summary()
```

```
Model: "model_12"
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
input_6 (InputLayer)         [(None, 224, 224, 3)]     0         
```

```
_________________________________________________________________
vgg16 (Functional)           (None, 7, 7, 512)         14714688
_________________________________________________________________
conv2d_11 (Conv2D)           (None, 1, 1, 4096)        102764544
_________________________________________________________________
conv2d_12 (Conv2D)           (None, 1, 1, 4096)        16781312
_________________________________________________________________
flatten_1 (Flatten)          (None, 4096)              0
_________________________________________________________________
dense_1 (Dense)              (None, 16)                65552
=================================================================
Total params: 134,326,096
Trainable params: 129,050,640
Non-trainable params: 5,275,456
_________________________________________________________________
```

In [49]:
```
#STEP_SIZE_TRAIN=train_generator.n//train_generator.batch_size
#STEP_SIZE_VALID=valid_generator.n//valid_generator.batch_size

model3.fit_generator(train_generator,
steps_per_epoch=33600/50,    #STEP_SIZE_TRAIN
validation_data=valid_generator,
validation_steps=14400/50,     #STEP_SIZE_VALID
epochs=4,
callbacks=[tensorboard_callback3])
```

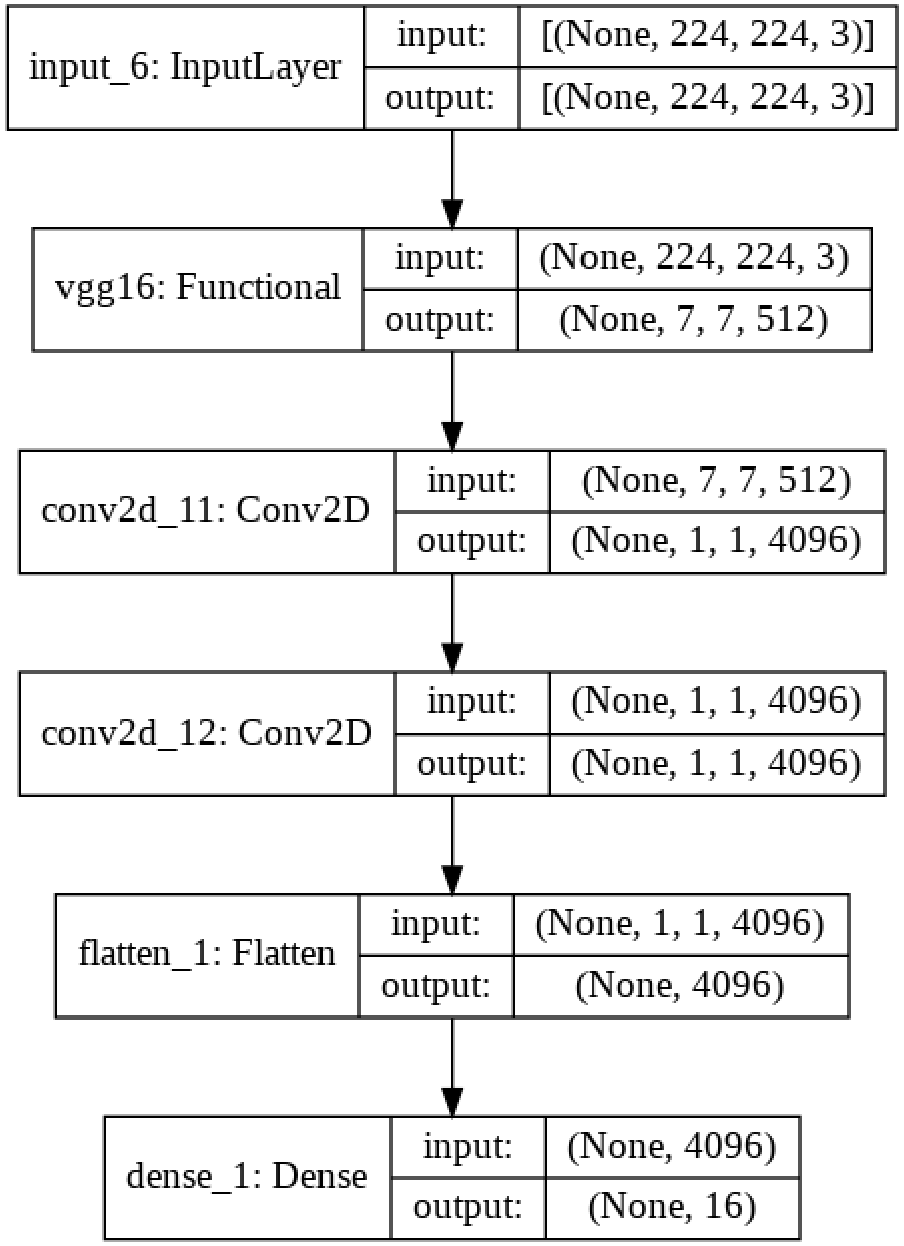
```
/usr/local/lib/python3.7/dist-packages/tensorflow/python/keras/engine/training.py:1844: UserWarning: `Model.fit_gener
ator` is deprecated and will be removed in a future version. Please use `Model.fit`, which supports generators.
  warnings.warn('`Model.fit_generator` is deprecated and '
Epoch 1/4
672/672 [==============================] - 505s 752ms/step - loss: 2.7728 - accuracy: 0.0629 - val_loss: 2.7729 - val
_accuracy: 0.0593
Epoch 2/4
672/672 [==============================] - 504s 750ms/step - loss: 2.7727 - accuracy: 0.0617 - val_loss: 2.7730 - val
_accuracy: 0.0593
Epoch 3/4
672/672 [==============================] - 506s 753ms/step - loss: 2.7727 - accuracy: 0.0643 - val_loss: 2.7731 - val
_accuracy: 0.0593
Epoch 4/4
672/672 [==============================] - 506s 753ms/step - loss: 2.7727 - accuracy: 0.0640 - val_loss: 2.7731 - val
_accuracy: 0.0593
```

Out[49]:
```
<tensorflow.python.keras.callbacks.History at 0x7f570a29fa50>
```

In [51]:
```
%load_ext tensorboard
%tensorboard --logdir $logdir3
```

In [52]:
```
from tensorflow.keras.utils import plot_model
plot_model(model3, 'model2.png', show_shapes=True)
```

Out[52]:


input_6: InputLayer
input:
[(None, 224, 224, 3)]
output:
[(None, 224, 224, 3)]
vgg16: Functional
input:
(None, 224, 224, 3)
output:
(None, 7, 7, 512)
conv2d_11: Conv2D
input:
(None, 7, 7, 512)
output:
(None, 1, 1, 4096)
conv2d_12: Conv2D
input:
(None, 1, 1, 4096)
output:
(None, 1, 1, 4096)
flatten_1: Flatten
input:
(None, 1, 1, 4096)
output:
(None, 4096)
dense_1: Dense
input:
(None, 4096)
output:
(None, 16)

```
In [ ]:
```