In [1]:
```
from google.colab import drive
drive.mount('/content/drive')
```
```
Mounted at /content/drive
```

In [2]:
```
cd /content/drive/MyDrive/
```
```
/content/drive/MyDrive
```

In [3]:
```
%pwd
```
Out[3]:
```
'/content/drive/MyDrive'
```

In [4]:
```
!git  clone https://github.com/UCSD-AI4H/COVID-CT.git
```
```
Cloning into 'COVID-CT'...
remote: Enumerating objects: 5463, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 5463 (delta 0), reused 0 (delta 0), pack-reused 5459
Receiving objects: 100% (5463/5463), 1.09 GiB | 14.21 MiB/s, done.
Resolving deltas: 100% (360/360), done.
Checking out files: 100% (1048/1048), done.
```

In [5]:
```
## unzip archives  q - quite, n - don't overwrite
!unzip -q -n '/content/drive/My Drive/COVID-CT/Images-processed/CT_COVID.zip'
!unzip -q -n '/content/drive/My Drive/COVID-CT/Images-processed/CT_NonCOVID.zip'
```

In [6]:
```
import keras
import tensorflow as tf
import os,cv2
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.utils import shuffle
from sklearn.model_selection import train_test_split
from keras import backend as K
keras.backend.image_data_format()
from keras.utils import np_utils
from keras.models import Sequential
from keras.layers.core import Dense, Dropout, Activation, Flatten
from keras.layers.convolutional import Conv2D, MaxPooling2D
from keras.layers import BatchNormalization,InputLayer
from tensorflow.keras.optimizers import Adam,RMSprop,Adadelta,Adagrad,Adamax
import glob
from keras.preprocessing.image import ImageDataGenerator
from keras.models import Model
from keras.utils.vis_utils import plot_model
from keras.callbacks import EarlyStopping,ModelCheckpoint
print('Keras version is {}'.format(keras.__version__))
print ('Tensorflow verson is {}'.format(tf.__version__))
print ('OpenCv version is {}'.format(cv2.__version__))
```
```
Keras version is 2.6.0
Tensorflow verson is 2.6.0
OpenCv version is 4.1.2
```

In [7]:
```
path='/content/drive/My Drive/CT_COVID/'
for count, filename in enumerate(os.listdir(path)):
    dst ="covid-" + str(count) + ".png"
    src =path+ filename
    dst =path+ dst

    # rename() function will
    # rename all the files
    os.rename(src, dst)
print("Covid Positive cases: ", count)

path='/content/drive/MyDrive/CT_NonCOVID/'
for count, filename in enumerate(os.listdir(path)):
    dst ="noncovid-" + str(count) + ".png"
    src =path+ filename
    dst =path+ dst

    # rename() function will
    # rename all the files
    os.rename(src, dst)

print("Covid Negative cases: ", count)
```
```
Covid Positive cases:  348
Covid Negative cases:  396
```

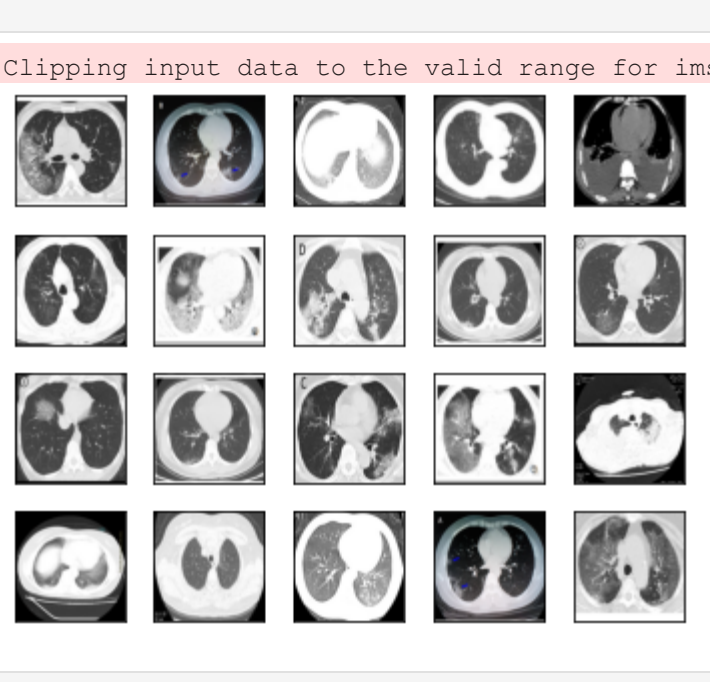
In [8]:
```
from PIL import Image
data_path = '/content/drive/MyDrive/'
#data_dir_list = os.listdir(data_path)
data_dir_list = ['CT_COVID','CT_NonCOVID']
odd_size_image = 0
for dataset in data_dir_list:
    img_list=os.listdir(data_path+'/'+ dataset)
    print ('Loaded the images of dataset-'+'{}\n'.format(dataset))
    for img in img_list:
        filename = os.path.join(data_path + '/'+ dataset + '/'+ img)
        image_png = Image.open(filename)
        if (image_png.size[0] or image_png.size[1])<224 :
          #print (img, image_png.size)
          odd_size_image += 1
    print("number of odd sized images are {}\n".format(odd_size_image))
```
```
Loaded the images of dataset-CT_COVID

number of odd sized images are 57

Loaded the images of dataset-CT_NonCOVID

number of odd sized images are 77
```

In [9]:
```
!rm -rf /content/drive/MyDrive/COVID-CT
```

In [10]:
```
data_path = '/content/drive/My Drive/'
data_dir_list = ['CT_COVID','CT_NonCOVID']
img_rows=224
img_cols=224
num_channel=3
num_epoch=100
no_images=0
for dataset in data_dir_list:
    img_list = os.listdir(data_path + '/' + dataset)
    no_images = no_images+len(img_list)
print(no_images)
# Define the number of classes
labels = np.ones((no_images,),dtype='int64')
num_classes = 2
label_index=0
img_data_list=[]
img=0

for dataset in data_dir_list:
    img_list=os.listdir(data_path+'/'+ dataset)
    print ('Loaded the images of dataset-'+'{}\n'.format(dataset))
    for img in img_list:
        input_img=cv2.imread(data_path + '/'+ dataset + '/'+ img)
        #input_img=cv2.cvtColor(input_img, cv2.COLOR_BGR2GRAY)
        input_img_resize=cv2.resize(input_img,(224,224))
        input_img_resize = cv2.normalize(input_img_resize, None, alpha=0, beta=1, norm_type=cv2.NORM_MINMAX, dt

        img_data_list.append(input_img_resize)
        if dataset[0:4]==  'CT_C':
            labels[label_index] =  1
            #print(dataset[0])
        if dataset[0:4] == 'CT_N':
            labels[label_index] = 0
        label_index = label_index+1
```
```
746
Loaded the images of dataset-CT_COVID

Loaded the images of dataset-CT_NonCOVID
```

In [11]:
```
img_data  = np.array(img_data_list)
print(img_data.shape)
```
```
(746, 224, 224, 3)
```

In [12]:
```
plt.figure(figsize=(5, 4))
for i in range(20):
  plt.subplot(4, 5, i+1)
  plt.imshow(img_data[i], cmap='gray')
  plt.xticks([])
  plt.yticks([])
plt.tight_layout()
plt.show()
```
```
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).
```



In [13]:
```
if num_channel==1:
    if K.image_data_format()=='th':
        img_data= np.expand_dims(img_data, axis=1)
        #print (img_data.shape)
    else:
        img_data= np.expand_dims(img_data, axis=3)
        #print (img_data.shape)

else:
    if K.image_data_format()=='th':
        img_data=np.rollaxis(img_data,3,1)
        #print (img_data.shape)
        labels[0:347] = 1
        labels[348:745] = 0

labels[0:745]
```
Out[13]:
```
array([1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0])
```

In [14]:
```
# Define the number of classes
num_classes = 2
names = ['COVID','NonCOVID']
# convert class labels to on-hot encoding
Y = np_utils.to_categorical(labels, num_classes)
X,y = shuffle(img_data, Y, random_state=42)
```

In [22]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42,stratify=y)
print("X_train size is {} and y_train size is {}".format(X_train.shape,y_train.shape))
print("X_test size is {} and y_test size is {}".format(X_test.shape,y_test.shape))
```
```
X_train size is (596, 224, 224, 3) and y_train size is (596, 2)
X_test size is (150, 224, 224, 3) and y_test size is (150, 2)
```

Feature Extraction using VGG16

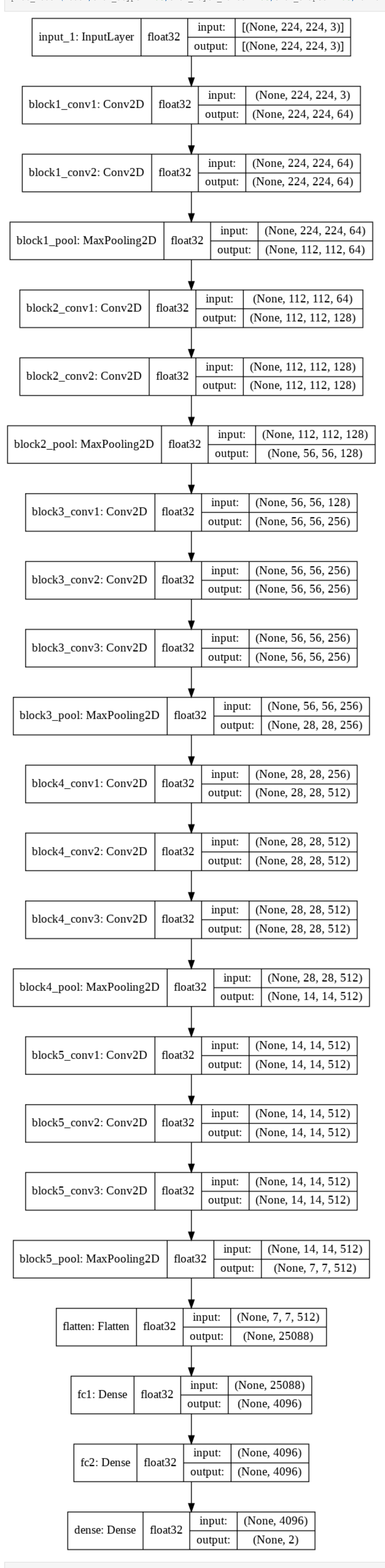
In [30]:
```
from keras.applications import vgg16
vgg = vgg16.VGG16(include_top=True, weights='imagenet',input_shape=(224,224,3))
base_model = vgg
base_model.summary()
```
```
Model: "vgg16"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_2 (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
_________________________________________________________________
flatten (Flatten)            (None, 25088)             0
_________________________________________________________________
fc1 (Dense)                  (None, 4096)              102764544
_________________________________________________________________
fc2 (Dense)                  (None, 4096)              16781312
_________________________________________________________________
predictions (Dense)          (None, 1000)              4097000
=================================================================
Total params: 138,357,544
Trainable params: 138,357,544
Non-trainable params: 0
_________________________________________________________________
```

In [31]:
```
# Extract the last layer before predciton of  vgg16 model
last = base_model.layers[-2].output
pred = Dense(2, activation='softmax')(last)
model = Model(base_model.input, pred)
for layer in base_model.layers:
    layer.trainable = False
```

In [32]:
```
model.summary()
```
```
Model: "model_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_2 (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
_________________________________________________________________
flatten (Flatten)            (None, 25088)             0
_________________________________________________________________
fc1 (Dense)                  (None, 4096)              102764544
_________________________________________________________________
fc2 (Dense)                  (None, 4096)              16781312
_________________________________________________________________
dense_1 (Dense)              (None, 2)                 8194
=================================================================
Total params: 134,268,738
Trainable params: 8,194
Non-trainable params: 134,260,544
_________________________________________________________________
```

In [33]:
```
model.compile(optimizer='Adam',loss='binary_crossentropy',metrics=['accuracy'])
```

In [25]:
```
plot_model(model,show_dtype=True,show_layer_names=True,show_shapes=True,rankdir='TB')
```
Out[25]:



In [26]:
```
custom_callbacks = [
    EarlyStopping(monitor="val_loss", patience=20, restore_best_weights=True),
    ModelCheckpoint(filepath='vgg16_model-fext.h5', save_best_only=True,verbose=1)
]
```

In [35]:
```
#del history
!rm vgg16_model-fext.h5
```

In [36]:
```
history = model.fit(X_train,y_train,validation_split=0.2,epochs=100,batch_size=10,verbose=1,callbacks=custom_ca
```

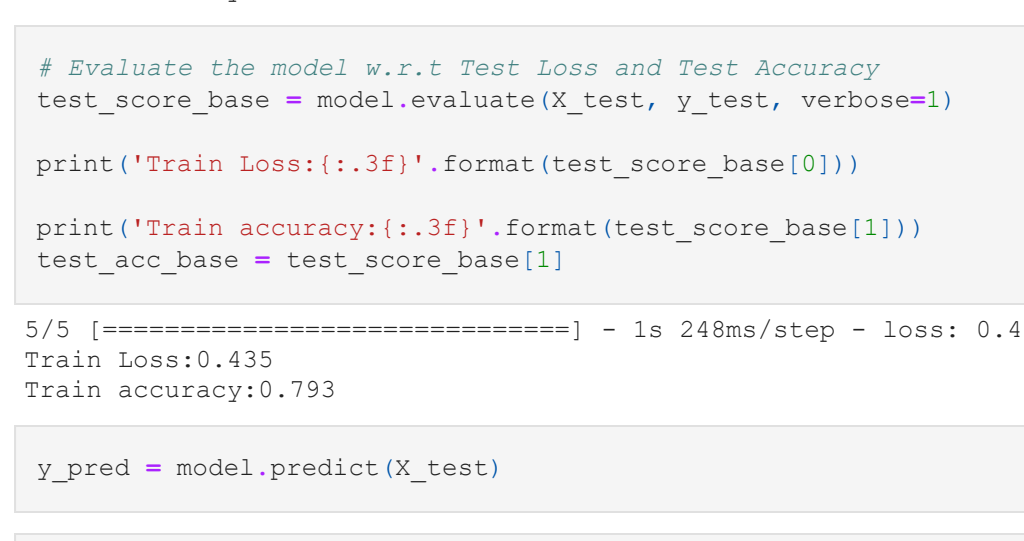
```
Epoch 1/100
48/48 [==============================] - 10s 190ms/step - loss: 0.7124 - accuracy: 0.5903 - val_loss: 0.6383 -
val_accuracy: 0.6583

Epoch 00001: val_loss did not improve from 0.37361
Epoch 2/100
48/48 [==============================] - 8s 162ms/step - loss: 0.6094 - accuracy: 0.6597 - val_loss: 0.6234 - v
al_accuracy: 0.5917

Epoch 00002: val_loss did not improve from 0.37361
Epoch 3/100
48/48 [==============================] - 9s 184ms/step - loss: 0.6419 - accuracy: 0.6218 - val_loss: 0.6401 - v
al_accuracy: 0.5667

Epoch 00003: val_loss did not improve from 0.37361
Epoch 4/100
48/48 [==============================] - 9s 184ms/step - loss: 0.5705 - accuracy: 0.7038 - val_loss: 0.7216 - v
al_accuracy: 0.5667

Epoch 00004: val_loss did not improve from 0.37361
Epoch 5/100
48/48 [==============================] - 9s 185ms/step - loss: 0.5486 - accuracy: 0.7122 - val_loss: 0.6277 - v
al_accuracy: 0.6000

Epoch 00005: val_loss did not improve from 0.37361
Epoch 6/100
48/48 [==============================] - 8s 162ms/step - loss: 0.5594 - accuracy: 0.7101 - val_loss: 0.5643 - v
al_accuracy: 0.7000

Epoch 00006: val_loss did not improve from 0.37361
Epoch 7/100
48/48 [==============================] - 8s 162ms/step - loss: 0.6293 - accuracy: 0.6597 - val_loss: 0.6161 - v
al_accuracy: 0.7000

Epoch 00007: val_loss did not improve from 0.37361
Epoch 8/100
48/48 [==============================] - 8s 162ms/step - loss: 0.5367 - accuracy: 0.7290 - val_loss: 0.6053 - v
al_accuracy: 0.6167

Epoch 00008: val_loss did not improve from 0.37361
Epoch 9/100
48/48 [==============================] - 9s 184ms/step - loss: 0.4831 - accuracy: 0.7794 - val_loss: 0.5881 - v
al_accuracy: 0.7167

Epoch 00009: val_loss did not improve from 0.37361
Epoch 10/100
48/48 [==============================] - 8s 162ms/step - loss: 0.5176 - accuracy: 0.7185 - val_loss: 0.5394 - v
al_accuracy: 0.7167

Epoch 00010: val_loss did not improve from 0.37361
Epoch 11/100
48/48 [==============================] - 9s 184ms/step - loss: 0.5026 - accuracy: 0.7584 - val_loss: 0.5454 - v
al_accuracy: 0.6750

Epoch 00011: val_loss did not improve from 0.37361
Epoch 12/100
48/48 [==============================] - 8s 161ms/step - loss: 0.5140 - accuracy: 0.7374 - val_loss: 0.5620 - v
al_accuracy: 0.6500

Epoch 00012: val_loss did not improve from 0.37361
Epoch 13/100
48/48 [==============================] - 8s 162ms/step - loss: 0.4823 - accuracy: 0.7941 - val_loss: 0.7561 - v
al_accuracy: 0.5667

Epoch 00013: val_loss did not improve from 0.37361
Epoch 14/100
48/48 [==============================] - 9s 184ms/step - loss: 0.4974 - accuracy: 0.7563 - val_loss: 0.5559 - v
al_accuracy: 0.7333

Epoch 00014: val_loss did not improve from 0.37361
Epoch 15/100
48/48 [==============================] - 8s 162ms/step - loss: 0.4614 - accuracy: 0.7668 - val_loss: 0.7447 - v
al_accuracy: 0.5750

Epoch 00015: val_loss did not improve from 0.37361
Epoch 16/100
48/48 [==============================] - 8s 162ms/step - loss: 0.5034 - accuracy: 0.7647 - val_loss: 0.5426 - v
al_accuracy: 0.6667

Epoch 00016: val_loss did not improve from 0.37361
Epoch 17/100
48/48 [==============================] - 8s 162ms/step - loss: 0.4744 - accuracy: 0.7710 - val_loss: 0.4826 - v
al_accuracy: 0.7833

Epoch 00017: val_loss did not improve from 0.37361
Epoch 18/100
48/48 [==============================] - 8s 163ms/step - loss: 0.4869 - accuracy: 0.7794 - val_loss: 0.4810 - v
al_accuracy: 0.7667

Epoch 00018: val_loss did not improve from 0.37361
Epoch 19/100
48/48 [==============================] - 8s 161ms/step - loss: 0.4923 - accuracy: 0.7542 - val_loss: 0.5640 - v
al_accuracy: 0.6750

Epoch 00019: val_loss did not improve from 0.37361
Epoch 20/100
48/48 [==============================] - 9s 185ms/step - loss: 0.4467 - accuracy: 0.7710 - val_loss: 0.4756 - v
al_accuracy: 0.7833

Epoch 00020: val_loss did not improve from 0.37361
Epoch 21/100
48/48 [==============================] - 8s 161ms/step - loss: 0.4388 - accuracy: 0.7815 - val_loss: 0.5007 - v
al_accuracy: 0.7417

Epoch 00021: val_loss did not improve from 0.37361
Epoch 22/100
48/48 [==============================] - 9s 185ms/step - loss: 0.4275 - accuracy: 0.7983 - val_loss: 0.6636 - v
al_accuracy: 0.6333

Epoch 00022: val_loss did not improve from 0.37361
Epoch 23/100
48/48 [==============================] - 9s 184ms/step - loss: 0.4356 - accuracy: 0.8025 - val_loss: 0.4620 - v
al_accuracy: 0.7833

Epoch 00023: val_loss did not improve from 0.37361
Epoch 24/100
48/48 [==============================] - 8s 162ms/step - loss: 0.4196 - accuracy: 0.8151 - val_loss: 0.5051 - v
al_accuracy: 0.7417

Epoch 00024: val_loss did not improve from 0.37361
Epoch 25/100
48/48 [==============================] - 8s 162ms/step - loss: 0.4257 - accuracy: 0.7983 - val_loss: 0.5468 - v
al_accuracy: 0.7083

Epoch 00025: val_loss did not improve from 0.37361
Epoch 26/100
48/48 [==============================] - 9s 185ms/step - loss: 0.4226 - accuracy: 0.8067 - val_loss: 0.6898 - v
al_accuracy: 0.6167

Epoch 00026: val_loss did not improve from 0.37361
Epoch 27/100
48/48 [==============================] - 9s 185ms/step - loss: 0.4512 - accuracy: 0.7731 - val_loss: 0.4632 - v
al_accuracy: 0.7500

Epoch 00027: val_loss did not improve from 0.37361
Epoch 28/100
48/48 [==============================] - 9s 185ms/step - loss: 0.4554 - accuracy: 0.7878 - val_loss: 0.5823 - v
al_accuracy: 0.6667

Epoch 00028: val_loss did not improve from 0.37361
Epoch 29/100
48/48 [==============================] - 8s 162ms/step - loss: 0.4130 - accuracy: 0.7983 - val_loss: 0.4428 - v
al_accuracy: 0.7750

Epoch 00029: val_loss did not improve from 0.37361
Epoch 30/100
48/48 [==============================] - 8s 162ms/step - loss: 0.4252 - accuracy: 0.7941 - val_loss: 0.4416 - v
al_accuracy: 0.7667

Epoch 00030: val_loss did not improve from 0.37361
Epoch 31/100
48/48 [==============================] - 9s 185ms/step - loss: 0.4024 - accuracy: 0.8130 - val_loss: 0.5443 - v
al_accuracy: 0.7000

Epoch 00031: val_loss did not improve from 0.37361
Epoch 32/100
48/48 [==============================] - 9s 184ms/step - loss: 0.4079 - accuracy: 0.8256 - val_loss: 0.4367 - v
al_accuracy: 0.7750

Epoch 00032: val_loss did not improve from 0.37361
Epoch 33/100
48/48 [==============================] - 8s 162ms/step - loss: 0.3941 - accuracy: 0.8235 - val_loss: 0.4877 - v
al_accuracy: 0.7667

Epoch 00033: val_loss did not improve from 0.37361
Epoch 34/100
48/48 [==============================] - 8s 162ms/step - loss: 0.3869 - accuracy: 0.8214 - val_loss: 0.4306 - v
al_accuracy: 0.7583

Epoch 00034: val_loss did not improve from 0.37361
Epoch 35/100
48/48 [==============================] - 9s 185ms/step - loss: 0.3973 - accuracy: 0.8214 - val_loss: 0.5190 - v
al_accuracy: 0.7333

Epoch 00035: val_loss did not improve from 0.37361
Epoch 36/100
48/48 [==============================] - 8s 162ms/step - loss: 0.3928 - accuracy: 0.8361 - val_loss: 0.6148 - v
al_accuracy: 0.6583

Epoch 00036: val_loss did not improve from 0.37361
Epoch 37/100
48/48 [==============================] - 8s 162ms/step - loss: 0.4279 - accuracy: 0.7815 - val_loss: 0.4231 - v
al_accuracy: 0.7667

Epoch 00037: val_loss did not improve from 0.37361
Epoch 38/100
48/48 [==============================] - 8s 162ms/step - loss: 0.4036 - accuracy: 0.8088 - val_loss: 0.4358 - v
al_accuracy: 0.7833

Epoch 00038: val_loss did not improve from 0.37361
Epoch 39/100
48/48 [==============================] - 8s 162ms/step - loss: 0.3849 - accuracy: 0.8277 - val_loss: 0.4179 - v
al_accuracy: 0.7833

Epoch 00039: val_loss did not improve from 0.37361
Epoch 40/100
48/48 [==============================] - 9s 184ms/step - loss: 0.3686 - accuracy: 0.8340 - val_loss: 0.5572 - v
al_accuracy: 0.6917

Epoch 00040: val_loss did not improve from 0.37361
Epoch 41/100
48/48 [==============================] - 9s 184ms/step - loss: 0.4013 - accuracy: 0.8109 - val_loss: 0.4928 - v
al_accuracy: 0.7667

Epoch 00041: val_loss did not improve from 0.37361
Epoch 42/100
48/48 [==============================] - 8s 162ms/step - loss: 0.4226 - accuracy: 0.7815 - val_loss: 0.5017 - v
al_accuracy: 0.7583

Epoch 00042: val_loss did not improve from 0.37361
Epoch 43/100
48/48 [==============================] - 9s 185ms/step - loss: 0.3540 - accuracy: 0.8466 - val_loss: 0.4505 - v
al_accuracy: 0.7917

Epoch 00043: val_loss did not improve from 0.37361
Epoch 44/100
48/48 [==============================] - 9s 185ms/step - loss: 0.3571 - accuracy: 0.8529 - val_loss: 0.4265 - v
al_accuracy: 0.8000

Epoch 00044: val_loss did not improve from 0.37361
Epoch 45/100
48/48 [==============================] - 9s 184ms/step - loss: 0.3640 - accuracy: 0.8508 - val_loss: 0.4775 - v
al_accuracy: 0.7667

Epoch 00045: val_loss did not improve from 0.37361
Epoch 46/100
48/48 [==============================] - 8s 162ms/step - loss: 0.3436 - accuracy: 0.8529 - val_loss: 0.4448 - v
al_accuracy: 0.7917

Epoch 00046: val_loss did not improve from 0.37361
Epoch 47/100
48/48 [==============================] - 8s 162ms/step - loss: 0.3911 - accuracy: 0.8277 - val_loss: 0.4851 - v
al_accuracy: 0.7583

Epoch 00047: val_loss did not improve from 0.37361
Epoch 48/100
48/48 [==============================] - 8s 162ms/step - loss: 0.3529 - accuracy: 0.8529 - val_loss: 0.4039 - v
al_accuracy: 0.7833

Epoch 00048: val_loss did not improve from 0.37361
Epoch 49/100
48/48 [==============================] - 9s 185ms/step - loss: 0.3744 - accuracy: 0.8382 - val_loss: 0.4971 - v
al_accuracy: 0.7417

Epoch 00049: val_loss did not improve from 0.37361
Epoch 50/100
48/48 [==============================] - 8s 162ms/step - loss: 0.3637 - accuracy: 0.8487 - val_loss: 0.4556 - v
al_accuracy: 0.7917

Epoch 00050: val_loss did not improve from 0.37361
Epoch 51/100
48/48 [==============================] - 8s 162ms/step - loss: 0.3561 - accuracy: 0.8361 - val_loss: 0.4260 - v
al_accuracy: 0.8167

Epoch 00051: val_loss did not improve from 0.37361
Epoch 52/100
48/48 [==============================] - 8s 162ms/step - loss: 0.3465 - accuracy: 0.8403 - val_loss: 0.4258 - v
al_accuracy: 0.8000

Epoch 00052: val_loss did not improve from 0.37361
Epoch 53/100
48/48 [==============================] - 8s 162ms/step - loss: 0.3454 - accuracy: 0.8424 - val_loss: 0.4010 - v
al_accuracy: 0.8000

Epoch 00053: val_loss did not improve from 0.37361
Epoch 54/100
48/48 [==============================] - 8s 162ms/step - loss: 0.3772 - accuracy: 0.8067 - val_loss: 0.4245 - v
al_accuracy: 0.8000

Epoch 00054: val_loss did not improve from 0.37361
Epoch 55/100
48/48 [==============================] - 8s 162ms/step - loss: 0.4240 - accuracy: 0.7920 - val_loss: 0.4634 - v
al_accuracy: 0.7750

Epoch 00055: val_loss did not improve from 0.37361
Epoch 56/100
48/48 [==============================] - 8s 162ms/step - loss: 0.3625 - accuracy: 0.8424 - val_loss: 0.3909 - v
al_accuracy: 0.8250

Epoch 00056: val_loss did not improve from 0.37361
Epoch 57/100
48/48 [==============================] - 9s 185ms/step - loss: 0.3596 - accuracy: 0.8403 - val_loss: 0.3882 - v
al_accuracy: 0.8250

Epoch 00057: val_loss did not improve from 0.37361
Epoch 58/100
48/48 [==============================] - 9s 184ms/step - loss: 0.3451 - accuracy: 0.8340 - val_loss: 0.4161 - v
al_accuracy: 0.8167

Epoch 00058: val_loss did not improve from 0.37361
Epoch 59/100
48/48 [==============================] - 8s 162ms/step - loss: 0.3367 - accuracy: 0.8613 - val_loss: 0.4111 - v
al_accuracy: 0.8167

Epoch 00059: val_loss did not improve from 0.37361
Epoch 60/100
48/48 [==============================] - 9s 185ms/step - loss: 0.3384 - accuracy: 0.8529 - val_loss: 0.4578 - v
al_accuracy: 0.7833

Epoch 00060: val_loss did not improve from 0.37361
Epoch 61/100
48/48 [==============================] - 8s 162ms/step - loss: 0.3779 - accuracy: 0.8361 - val_loss: 0.9685 - v
al_accuracy: 0.5583

Epoch 00061: val_loss did not improve from 0.37361
Epoch 62/100
48/48 [==============================] - 9s 185ms/step - loss: 0.3793 - accuracy: 0.8424 - val_loss: 0.3856 - v
al_accuracy: 0.8167

Epoch 00062: val_loss did not improve from 0.37361
Epoch 63/100
48/48 [==============================] - 9s 185ms/step - loss: 0.3203 - accuracy: 0.8676 - val_loss: 0.4001 - v
al_accuracy: 0.8000

Epoch 00063: val_loss did not improve from 0.37361
Epoch 64/100
48/48 [==============================] - 9s 185ms/step - loss: 0.3370 - accuracy: 0.8571 - val_loss: 0.3874 - v
al_accuracy: 0.8000

Epoch 00064: val_loss did not improve from 0.37361
Epoch 65/100
48/48 [==============================] - 8s 162ms/step - loss: 0.3333 - accuracy: 0.8571 - val_loss: 0.4008 - v
al_accuracy: 0.8083

Epoch 00065: val_loss did not improve from 0.37361
Epoch 66/100
48/48 [==============================] - 8s 163ms/step - loss: 0.3011 - accuracy: 0.8782 - val_loss: 0.4539 - v
al_accuracy: 0.7833

Epoch 00066: val_loss did not improve from 0.37361
Epoch 67/100
48/48 [==============================] - 8s 162ms/step - loss: 0.3148 - accuracy: 0.8739 - val_loss: 0.3790 - v
al_accuracy: 0.8167

Epoch 00067: val_loss did not improve from 0.37361
Epoch 68/100
48/48 [==============================] - 9s 186ms/step - loss: 0.3149 - accuracy: 0.8845 - val_loss: 0.4408 - v
al_accuracy: 0.7833

Epoch 00068: val_loss did not improve from 0.37361
Epoch 69/100
48/48 [==============================] - 8s 162ms/step - loss: 0.3300 - accuracy: 0.8466 - val_loss: 0.4212 - v
al_accuracy: 0.8083

Epoch 00069: val_loss did not improve from 0.37361
Epoch 70/100
48/48 [==============================] - 9s 186ms/step - loss: 0.3018 - accuracy: 0.8803 - val_loss: 0.3756 - v
al_accuracy: 0.8083

Epoch 00070: val_loss did not improve from 0.37361
Epoch 71/100
48/48 [==============================] - 9s 185ms/step - loss: 0.3102 - accuracy: 0.8866 - val_loss: 0.3851 - v
al_accuracy: 0.8083

Epoch 00071: val_loss did not improve from 0.37361
Epoch 72/100
48/48 [==============================] - 9s 186ms/step - loss: 0.2976 - accuracy: 0.8718 - val_loss: 0.3767 - v
al_accuracy: 0.8167

Epoch 00072: val_loss did not improve from 0.37361
Epoch 73/100
48/48 [==============================] - 8s 162ms/step - loss: 0.3078 - accuracy: 0.8697 - val_loss: 0.4300 - v
al_accuracy: 0.7833

Epoch 00073: val_loss did not improve from 0.37361
Epoch 74/100
48/48 [==============================] - 9s 185ms/step - loss: 0.3231 - accuracy: 0.8655 - val_loss: 0.4770 - v
al_accuracy: 0.7667

Epoch 00074: val_loss did not improve from 0.37361
Epoch 75/100
48/48 [==============================] - 8s 162ms/step - loss: 0.3422 - accuracy: 0.8487 - val_loss: 0.8296 - v
al_accuracy: 0.5833

Epoch 00075: val_loss did not improve from 0.37361
Epoch 76/100
48/48 [==============================] - 8s 163ms/step - loss: 0.3250 - accuracy: 0.8676 - val_loss: 0.3724 - v
al_accuracy: 0.8167

Epoch 00076: val_loss improved from 0.37361 to 0.37242, saving model to vgg16_model-fext.h5
Epoch 77/100
48/48 [==============================] - 9s 185ms/step - loss: 0.2941 - accuracy: 0.8887 - val_loss: 0.3782 - v
al_accuracy: 0.8250

Epoch 00077: val_loss did not improve from 0.37242
Epoch 78/100
48/48 [==============================] - 9s 187ms/step - loss: 0.2930 - accuracy: 0.8887 - val_loss: 0.3795 - v
al_accuracy: 0.8167

Epoch 00078: val_loss did not improve from 0.37242
Epoch 79/100
48/48 [==============================] - 8s 163ms/step - loss: 0.3258 - accuracy: 0.8613 - val_loss: 0.3688 - v
al_accuracy: 0.8167

Epoch 00079: val_loss improved from 0.37242 to 0.36877, saving model to vgg16_model-fext.h5
Epoch 80/100
48/48 [==============================] - 9s 186ms/step - loss: 0.3947 - accuracy: 0.8256 - val_loss: 0.3694 - v
al_accuracy: 0.8333

Epoch 00080: val_loss did not improve from 0.36877
Epoch 81/100
48/48 [==============================] - 8s 163ms/step - loss: 0.2835 - accuracy: 0.8950 - val_loss: 0.4080 - v
al_accuracy: 0.8000

Epoch 00081: val_loss did not improve from 0.36877
Epoch 82/100
48/48 [==============================] - 9s 186ms/step - loss: 0.2933 - accuracy: 0.8866 - val_loss: 0.3671 - v
al_accuracy: 0.8333

Epoch 00082: val_loss improved from 0.36877 to 0.36714, saving model to vgg16_model-fext.h5
Epoch 83/100
48/48 [==============================] - 8s 162ms/step - loss: 0.3041 - accuracy: 0.8803 - val_loss: 0.3764 - v
al_accuracy: 0.8167

Epoch 00083: val_loss did not improve from 0.36714
Epoch 84/100
48/48 [==============================] - 8s 163ms/step - loss: 0.3391 - accuracy: 0.8571 - val_loss: 0.3690 - v
al_accuracy: 0.8250

Epoch 00084: val_loss did not improve from 0.36714
Epoch 85/100
48/48 [==============================] - 9s 186ms/step - loss: 0.2896 - accuracy: 0.8739 - val_loss: 0.3809 - v
al_accuracy: 0.8250

Epoch 00085: val_loss did not improve from 0.36714
Epoch 86/100
48/48 [==============================] - 8s 163ms/step - loss: 0.2883 - accuracy: 0.8761 - val_loss: 0.3685 - v
al_accuracy: 0.8167

Epoch 00086: val_loss did not improve from 0.36714
Epoch 87/100
48/48 [==============================] - 9s 185ms/step - loss: 0.3759 - accuracy: 0.8256 - val_loss: 0.3976 - v
al_accuracy: 0.8083

Epoch 00087: val_loss did not improve from 0.36714
Epoch 88/100
48/48 [==============================] - 9s 185ms/step - loss: 0.3023 - accuracy: 0.8782 - val_loss: 0.4481 - v
al_accuracy: 0.7667

Epoch 00088: val_loss did not improve from 0.36714
Epoch 89/100
48/48 [==============================] - 8s 162ms/step - loss: 0.3382 - accuracy: 0.8319 - val_loss: 0.4253 - v
al_accuracy: 0.7833

Epoch 00089: val_loss did not improve from 0.36714
Epoch 90/100
48/48 [==============================] - 8s 162ms/step - loss: 0.3607 - accuracy: 0.8424 - val_loss: 0.3748 - v
al_accuracy: 0.8250

Epoch 00090: val_loss did not improve from 0.36714
Epoch 91/100
48/48 [==============================] - 8s 162ms/step - loss: 0.2930 - accuracy: 0.8739 - val_loss: 0.4375 - v
al_accuracy: 0.7750

Epoch 00091: val_loss did not improve from 0.36714
Epoch 92/100
48/48 [==============================] - 9s 185ms/step - loss: 0.2878 - accuracy: 0.8676 - val_loss: 0.3928 - v
al_accuracy: 0.8083

Epoch 00092: val_loss did not improve from 0.36714
Epoch 93/100
48/48 [==============================] - 8s 163ms/step - loss: 0.2847 - accuracy: 0.8992 - val_loss: 0.5733 - v
al_accuracy: 0.7167

Epoch 00093: val_loss did not improve from 0.36714
Epoch 94/100
48/48 [==============================] - 8s 163ms/step - loss: 0.2884 - accuracy: 0.8824 - val_loss: 0.5693 - v
al_accuracy: 0.7167

Epoch 00094: val_loss did not improve from 0.36714
Epoch 95/100
48/48 [==============================] - 8s 163ms/step - loss: 0.2871 - accuracy: 0.8761 - val_loss: 0.3619 - v
al_accuracy: 0.8583

Epoch 00095: val_loss improved from 0.36714 to 0.36195, saving model to vgg16_model-fext.h5
Epoch 96/100
48/48 [==============================] - 9s 186ms/step - loss: 0.2837 - accuracy: 0.8824 - val_loss: 0.3768 - v
al_accuracy: 0.8250

Epoch 00096: val_loss did not improve from 0.36195
Epoch 97/100
48/48 [==============================] - 8s 164ms/step - loss: 0.2628 - accuracy: 0.8887 - val_loss: 0.3618 - v
al_accuracy: 0.8333

Epoch 00097: val_loss improved from 0.36195 to 0.36176, saving model to vgg16_model-fext.h5
Epoch 98/100
48/48 [==============================] - 8s 163ms/step - loss: 0.2699 - accuracy: 0.8950 - val_loss: 0.3584 - v
al_accuracy: 0.8500

Epoch 00098: val_loss improved from 0.36176 to 0.35839, saving model to vgg16_model-fext.h5
Epoch 99/100
48/48 [==============================] - 8s 163ms/step - loss: 0.2575 - accuracy: 0.9097 - val_loss: 0.4175 - v
al_accuracy: 0.7667

Epoch 00099: val_loss did not improve from 0.35839
Epoch 100/100
48/48 [==============================] - 9s 186ms/step - loss: 0.2568 - accuracy: 0.9013 - val_loss: 0.3589 - v
al_accuracy: 0.8417

Epoch 00100: val_loss did not improve from 0.35839
```

In [37]:

```
history_df = pd.DataFrame.from_dict(history.history)
history_df['epoch'] = range(1,history_df.shape[0]+1)
```

In [38]:

```
plt.style.use("ggplot")
fig,ax = plt.subplots()
ax.figure.set_size_inches([8,6])
ax.plot(history_df['epoch'],history_df['loss'],label='Training loss',marker='o',ms=5)
ax.plot(history_df['epoch'],history_df['accuracy'],label='Training accuracy',marker='+',ms=5)
ax.plot(history_df['epoch'],history_df['val_loss'],label='Validation loss',marker='v',ms=5)
ax.plot(history_df['epoch'],history_df['val_accuracy'],label='Validation accuracy',marker='*',ms=5)

plt.title("Training Loss and Accuracy on Dataset")
plt.xlabel("Epoch #")
plt.ylabel("Loss/Accuracy")
ax.legend()
plt.show()
```



In [52]:

```
# Evaluate the model w.r.t Train Loss and Train Accuracy
train_score_base = model.evaluate(X_train, y_train, verbose=1)

print('Train Loss:{:.3f}'.format(train_score_base[0]))

print('Train accuracy:{:.3f}'.format(train_score_base[1]))
train_acc_base = train_score_base[1]
```

```
19/19 [==============================] - 5s 251ms/step - loss: 0.2788 - accuracy: 0.8859
Train Loss:0.279
Train accuracy:0.886
```

In [53]:

```
# Evaluate the model w.r.t Test Loss and Test Accuracy
test_score_base = model.evaluate(X_test, y_test, verbose=1)

print('Train Loss:{:.3f}'.format(test_score_base[0]))

print('Train accuracy:{:.3f}'.format(test_score_base[1]))
test_acc_base = test_score_base[1]
```

```
5/5 [==============================] - 1s 248ms/step - loss: 0.4350 - accuracy: 0.7933
Train Loss:0.435
Train accuracy:0.793
```

In [42]:

```
y_pred = model.predict(X_test)
```

In [43]:

```
from sklearn.metrics import classification_report,confusion_matrix,roc_auc_score,roc_curve,recall_score
print(classification_report(np.argmax(y_test,axis=-1),np.argmax(y_pred,axis=-1),target_names=names))
```

```
              precision    recall  f1-score   support

       COVID       0.87      0.72      0.79        80
    NonCOVID       0.73      0.87      0.80        70

    accuracy                           0.79       150
   macro avg       0.80      0.80      0.79       150
weighted avg       0.80      0.79      0.79       150
```

In [44]:

```
AUC_base = roc_auc_score(y_test,y_pred)
print("Area Under the Curve is {:.3f}".format(AUC_base))
```

```
Area Under the Curve is 0.891
```

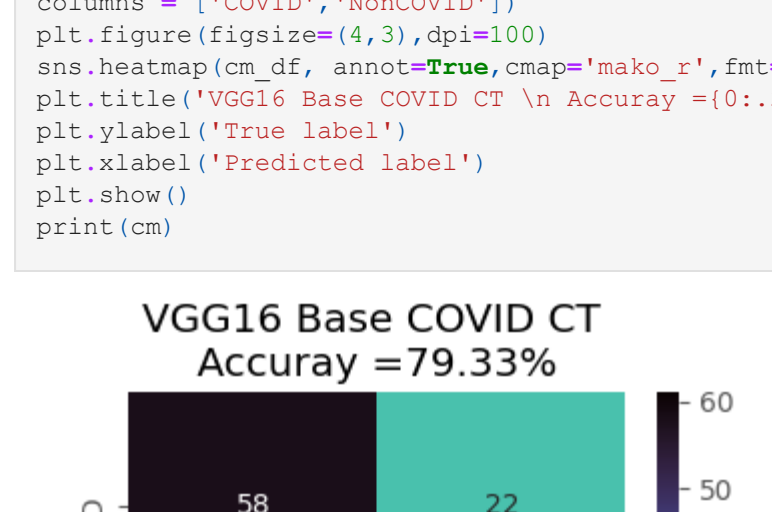
In [45]:

```
val_accuracy_base = np.max(history.history['val_accuracy'])
print('Val accuracy:{:.3f}'.format(val_accuracy_base))
```

```
Val accuracy:0.858
```

In [46]:

```
import pandas as pd
import seaborn as sns
cm = confusion_matrix(np.argmax(y_test,axis=-1),np.argmax(y_pred,axis=-1))
cm_df = pd.DataFrame(cm,
index = ['COVID','NonCOVID'],
columns = ['COVID','NonCOVID'])
plt.figure(figsize=(4,3),dpi=100)
sns.heatmap(cm_df, annot=True,cmap='mako_r',fmt='g')
plt.title('VGG16 Base COVID CT \n Accuray ={0:.2f}% '.format(test_score_base[1]*100))
plt.ylabel('True label')
plt.xlabel('Predicted label')
plt.show()
print(cm)
```




```
[[58 22]
 [ 9 61]]
```

In [62]:

```
comparison=pd.DataFrame({"Model":["Train Accuracy", "Validation Accuracy", "Test Accuracy", "Precision", "Recal
            "VGG Base Model":[train_acc_base, val_accuracy_base, test_acc_base, 0.8,0.79, 0.795,AUC_base]})
```

In [59]:

```
comparison
```

Out[59]:

	Model	VGG Base Model
0	Train Accuracy	0.885906
1	Validation Accuracy	0.858333
2	Test Accuracy	0.793333
3	Precision	0.800000
4	Recall	0.790000
5	F1 Score	0.795000
6	AUC	0.891071

VGG16 Fine Tuning

In [63]:

```
vgg = vgg16.VGG16(include_top=False, weights='imagenet',input_shape=(224,224,3))

output = vgg.layers[-1].output
output = keras.layers.Flatten()(output)

vgg_model = Model(vgg.input, output)
vgg_model.trainable = False

for layer in vgg_model.layers:
    layer.trainable = False
```

```
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/vgg16/vgg16_weights_tf_dim_o
rdering_tf_kernels_notop.h5
58892288/58889256 [==============================] - 1s 0us/step
58900480/58889256 [==============================] - 1s 0us/step
```

In [64]:

```
input_shape = vgg_model.output_shape[1]
model = Sequential()
model.add(vgg_model)
model.add(Dense(128, activation='relu', input_dim=input_shape))
model.add(Dense(64, activation='relu'))
model.add(Dense(2, activation='softmax'))
learning_rate = 0.0001
opt = RMSprop(learning_rate=learning_rate)
model.compile(optimizer=opt,loss='binary_crossentropy',metrics=['accuracy'])

model.summary()
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
model_2 (Functional)         (None, 25088)             14714688
_________________________________________________________________
dense_2 (Dense)              (None, 128)               3211392
_________________________________________________________________
dense_3 (Dense)              (None, 64)                8256
_________________________________________________________________
dense_4 (Dense)              (None, 2)                 130
=================================================================
Total params: 17,934,466
Trainable params: 3,219,778
Non-trainable params: 14,714,688
_________________________________________________________________
```

In [65]:

```
custom_callbacks = [
    EarlyStopping(monitor="val_loss", patience=20, restore_best_weights=True),
    ModelCheckpoint(filepath='vgg16_model-1.h5', save_best_only=True,verbose=1)
]
```

In [66]:

```
history1 = model.fit(X_train,y_train,validation_split=0.2,epochs=100,batch_size=20,verbose=1,callbacks=custom_c
```

```
Epoch 1/100
24/24 [==============================] - 13s 522ms/step - loss: 0.6450 - accuracy: 0.6513 - val_loss: 0.5346 -
val_accuracy: 0.7167

Epoch 00001: val_loss improved from inf to 0.53459, saving model to vgg16_model-1.h5
Epoch 2/100
24/24 [==============================] - 7s 273ms/step - loss: 0.4928 - accuracy: 0.7689 - val_loss: 0.6045 - v
al_accuracy: 0.6250

Epoch 00002: val_loss did not improve from 0.53459
Epoch 3/100
24/24 [==============================] - 7s 276ms/step - loss: 0.3694 - accuracy: 0.8361 - val_loss: 0.5463 - v
al_accuracy: 0.6500

Epoch 00003: val_loss did not improve from 0.53459
Epoch 4/100
24/24 [==============================] - 7s 275ms/step - loss: 0.3366 - accuracy: 0.8529 - val_loss: 0.3779 - v
al_accuracy: 0.8250

Epoch 00004: val_loss improved from 0.53459 to 0.37790, saving model to vgg16_model-1.h5
Epoch 5/100
24/24 [==============================] - 7s 275ms/step - loss: 0.2605 - accuracy: 0.9076 - val_loss: 0.8507 - v
al_accuracy: 0.6000

Epoch 00005: val_loss did not improve from 0.37790
Epoch 6/100
24/24 [==============================] - 7s 275ms/step - loss: 0.2387 - accuracy: 0.9118 - val_loss: 0.5485 - v
al_accuracy: 0.6917

Epoch 00006: val_loss did not improve from 0.37790
Epoch 7/100
24/24 [==============================] - 7s 274ms/step - loss: 0.2041 - accuracy: 0.9118 - val_loss: 0.4045 - v
al_accuracy: 0.8250

Epoch 00007: val_loss did not improve from 0.37790
Epoch 8/100
24/24 [==============================] - 8s 329ms/step - loss: 0.1789 - accuracy: 0.9517 - val_loss: 0.3234 - v
al_accuracy: 0.8333

Epoch 00008: val_loss improved from 0.37790 to 0.32345, saving model to vgg16_model-1.h5
Epoch 9/100
24/24 [==============================] - 8s 331ms/step - loss: 0.1587 - accuracy: 0.9328 - val_loss: 0.3429 - v
al_accuracy: 0.8250

Epoch 00009: val_loss did not improve from 0.32345
Epoch 10/100
24/24 [==============================] - 8s 329ms/step - loss: 0.1174 - accuracy: 0.9748 - val_loss: 0.4705 - v
al_accuracy: 0.7500

Epoch 00010: val_loss did not improve from 0.32345
Epoch 11/100
24/24 [==============================] - 7s 274ms/step - loss: 0.1057 - accuracy: 0.9601 - val_loss: 0.3183 - v
al_accuracy: 0.8333

Epoch 00011: val_loss improved from 0.32345 to 0.31833, saving model to vgg16_model-1.h5
Epoch 12/100
24/24 [==============================] - 6s 273ms/step - loss: 0.0802 - accuracy: 0.9748 - val_loss: 0.6296 - v
al_accuracy: 0.6833

Epoch 00012: val_loss did not improve from 0.31833
Epoch 13/100
24/24 [==============================] - 7s 274ms/step - loss: 0.0831 - accuracy: 0.9853 - val_loss: 0.4303 - v
al_accuracy: 0.7833

Epoch 00013: val_loss did not improve from 0.31833
Epoch 14/100
24/24 [==============================] - 7s 274ms/step - loss: 0.0662 - accuracy: 0.9811 - val_loss: 0.3529 - v
al_accuracy: 0.8167

Epoch 00014: val_loss did not improve from 0.31833
Epoch 15/100
24/24 [==============================] - 7s 274ms/step - loss: 0.0536 - accuracy: 0.9874 - val_loss: 0.4046 - v
al_accuracy: 0.7833

Epoch 00015: val_loss did not improve from 0.31833
Epoch 16/100
24/24 [==============================] - 7s 274ms/step - loss: 0.0715 - accuracy: 0.9769 - val_loss: 0.2749 - v
al_accuracy: 0.9000

Epoch 00016: val_loss improved from 0.31833 to 0.27494, saving model to vgg16_model-1.h5
Epoch 17/100
24/24 [==============================] - 8s 328ms/step - loss: 0.0319 - accuracy: 0.9937 - val_loss: 0.2706 - v
al_accuracy: 0.9000

Epoch 00017: val_loss improved from 0.27494 to 0.27062, saving model to vgg16_model-1.h5
Epoch 18/100
24/24 [==============================] - 7s 273ms/step - loss: 0.0376 - accuracy: 0.9937 - val_loss: 0.2643 - v
al_accuracy: 0.8917

Epoch 00018: val_loss improved from 0.27062 to 0.26429, saving model to vgg16_model-1.h5
Epoch 19/100
24/24 [==============================] - 8s 328ms/step - loss: 0.0357 - accuracy: 0.9916 - val_loss: 0.2915 - v
al_accuracy: 0.8667

Epoch 00019: val_loss did not improve from 0.26429
Epoch 20/100
24/24 [==============================] - 8s 330ms/step - loss: 0.0180 - accuracy: 0.9979 - val_loss: 0.2749 - v
al_accuracy: 0.8667

Epoch 00020: val_loss did not improve from 0.26429
Epoch 21/100
24/24 [==============================] - 8s 328ms/step - loss: 0.0487 - accuracy: 0.9832 - val_loss: 0.4116 - v
al_accuracy: 0.7917

Epoch 00021: val_loss did not improve from 0.26429
Epoch 22/100
24/24 [==============================] - 7s 273ms/step - loss: 0.0159 - accuracy: 0.9979 - val_loss: 0.2714 - v
al_accuracy: 0.8917

Epoch 00022: val_loss did not improve from 0.26429
Epoch 23/100
24/24 [==============================] - 8s 329ms/step - loss: 0.0111 - accuracy: 1.0000 - val_loss: 0.3036 - v
al_accuracy: 0.8500

Epoch 00023: val_loss did not improve from 0.26429
Epoch 24/100
24/24 [==============================] - 7s 274ms/step - loss: 0.0177 - accuracy: 0.9937 - val_loss: 0.3017 - v
al_accuracy: 0.8417

Epoch 00024: val_loss did not improve from 0.26429
Epoch 25/100
24/24 [==============================] - 7s 274ms/step - loss: 0.0183 - accuracy: 0.9958 - val_loss: 1.0005 - v
al_accuracy: 0.6583

Epoch 00025: val_loss did not improve from 0.26429
Epoch 26/100
24/24 [==============================] - 7s 274ms/step - loss: 0.0101 - accuracy: 0.9979 - val_loss: 0.2994 - v
al_accuracy: 0.8667

Epoch 00026: val_loss did not improve from 0.26429
Epoch 27/100
24/24 [==============================] - 7s 274ms/step - loss: 0.0222 - accuracy: 0.9916 - val_loss: 0.2877 - v
al_accuracy: 0.8750

Epoch 00027: val_loss did not improve from 0.26429
Epoch 28/100
24/24 [==============================] - 7s 274ms/step - loss: 0.0091 - accuracy: 1.0000 - val_loss: 0.2822 - v
al_accuracy: 0.8833

Epoch 00028: val_loss did not improve from 0.26429
Epoch 29/100
24/24 [==============================] - 6s 272ms/step - loss: 0.0222 - accuracy: 0.9916 - val_loss: 0.2801 - v
al_accuracy: 0.9000

Epoch 00029: val_loss did not improve from 0.26429
Epoch 30/100
24/24 [==============================] - 7s 274ms/step - loss: 0.0039 - accuracy: 1.0000 - val_loss: 0.2956 - v
al_accuracy: 0.8750

Epoch 00030: val_loss did not improve from 0.26429
Epoch 31/100
24/24 [==============================] - 6s 273ms/step - loss: 0.0183 - accuracy: 0.9916 - val_loss: 0.2914 - v
al_accuracy: 0.9083

Epoch 00031: val_loss did not improve from 0.26429
Epoch 32/100
24/24 [==============================] - 7s 274ms/step - loss: 0.0029 - accuracy: 1.0000 - val_loss: 0.2934 - v
al_accuracy: 0.8833

Epoch 00032: val_loss did not improve from 0.26429
Epoch 33/100
24/24 [==============================] - 8s 331ms/step - loss: 0.0103 - accuracy: 0.9958 - val_loss: 0.3001 - v
al_accuracy: 0.8917

Epoch 00033: val_loss did not improve from 0.26429
Epoch 34/100
24/24 [==============================] - 7s 273ms/step - loss: 0.0163 - accuracy: 0.9958 - val_loss: 0.3491 - v
al_accuracy: 0.8500

Epoch 00034: val_loss did not improve from 0.26429
Epoch 35/100
24/24 [==============================] - 6s 273ms/step - loss: 0.0029 - accuracy: 1.0000 - val_loss: 0.3236 - v
al_accuracy: 0.8583

Epoch 00035: val_loss did not improve from 0.26429
Epoch 36/100
24/24 [==============================] - 6s 272ms/step - loss: 0.0026 - accuracy: 1.0000 - val_loss: 0.3260 - v
al_accuracy: 0.8583

Epoch 00036: val_loss did not improve from 0.26429
Epoch 37/100
24/24 [==============================] - 7s 275ms/step - loss: 0.0099 - accuracy: 0.9979 - val_loss: 0.3845 - v
al_accuracy: 0.8333

Epoch 00037: val_loss did not improve from 0.26429
Epoch 38/100
24/24 [==============================] - 7s 274ms/step - loss: 0.0039 - accuracy: 1.0000 - val_loss: 0.7354 - v
al_accuracy: 0.8083

Epoch 00038: val_loss did not improve from 0.26429
```

In [67]:

```
history_df = pd.DataFrame.from_dict(history1.history)
```

In [68]:

```
history_df['epoch'] = range(1,history_df.shape[0]+1)
```

In [69]:

```
plt.style.use("ggplot")
fig,ax = plt.subplots()
ax.figure.set_size_inches([8,6])
ax.plot(history_df['epoch'],history_df['loss'],label='Training loss',marker='o',ms=5)
ax.plot(history_df['epoch'],history_df['accuracy'],label='Training accuracy',marker='+',ms=5)
ax.plot(history_df['epoch'],history_df['val_loss'],label='Validation loss',marker='v',ms=5)
ax.plot(history_df['epoch'],history_df['val_accuracy'],label='Validation accuracy',marker='*',ms=5)

plt.title("Training Loss and Accuracy on Dataset")
plt.xlabel("Epoch #")
plt.ylabel("Loss/Accuracy")
ax.legend()
plt.show()
```

In [78]:
```
# Evaluate the model w.r.t Train Loss and Train Accuracy
train_score_vgg16 = model.evaluate(X_train, y_train, verbose=1)

print('Train Loss:{:.3f}'.format(train_score_vgg16[0]))

print('Train accuracy:{:.3f}'.format(train_score_vgg16[1]))
train_acc_fine = train_score_vgg16[1]
```

```
19/19 [==============================] - 5s 244ms/step - loss: 0.0667 - accuracy: 0.9765
Train Loss:0.067
Train accuracy:0.977
```

In [71]:
```
# Evaluate the model w.r.t Test Loss and Test Accuracy
test_score_vgg16 = model.evaluate(X_test, y_test, verbose=1)
print('Test Loss:{:.3f}'.format(test_score_vgg16[0]))

print('Test accuracy:{:.3f}'.format(test_score_vgg16[1]))
test_acc_fine = test_score_vgg16[1]
```

```
5/5 [==============================] - 1s 236ms/step - loss: 0.2691 - accuracy: 0.9200
Test Loss:0.269
Test accuracy:0.920
```

In [72]:
```
y_pred = model.predict(X_test)
```

In [73]:
```
from sklearn.metrics import classification_report,confusion_matrix,roc_auc_score,roc_curve,recall_score
print(classification_report(np.argmax(y_test,axis=-1),np.argmax(y_pred,axis=-1),target_names=names))
```

```
              precision    recall  f1-score   support

       COVID       0.91      0.94      0.93        80
    NonCOVID       0.93      0.90      0.91        70

    accuracy                           0.92       150
   macro avg       0.92      0.92      0.92       150
weighted avg       0.92      0.92      0.92       150
```

In [74]:
```
AUC_vgg16 = roc_auc_score(y_test,y_pred)
print("Area Under the Curve is {:.3f}".format(AUC_vgg16))
```

```
Area Under the Curve is 0.956
```

In [75]:
```
val_accuracy_vgg16 = np.max(history.history['val_accuracy'])
print('Val accuracy:{:.3f}'.format(val_accuracy_vgg16))
```

```
Val accuracy:0.858
```

In [76]:
```
import pandas as pd
import seaborn as sns
cm = confusion_matrix(np.argmax(y_test,axis=-1),np.argmax(y_pred,axis=-1))
cm_df = pd.DataFrame(cm,
index = ['COVID','NonCOVID'],
columns = ['COVID','NonCOVID'])
plt.figure(figsize=(4,3),dpi=100)
sns.heatmap(cm_df, annot=True,cmap='mako_r',fmt='g')
plt.title('VGG16 Fine Tune COVID CT \n Accuray ={0:.2f}% '.format(test_score_vgg16[1]*100))
plt.ylabel('True label')
plt.xlabel('Predicted label')
plt.show()
print(cm)
```



```
[[75  5]
 [ 7 63]]
```

In [79]:
```
comparison['VGG Fine Model'] = [train_acc_fine, val_accuracy_vgg16, test_acc_fine, 0.92,0.92,0.92,AUC_vgg16]
```

In [80]:
```
comparison
```

Out[80]:

	Model	VGG Base Model	VGG Fine Model
0	Train Accuracy	0.885906	0.976510
1	Validation Accuracy	0.858333	0.858333
2	Test Accuracy	0.793333	0.920000
3	Precision	0.800000	0.920000
4	Recall	0.790000	0.920000
5	F1 Score	0.795000	0.920000
6	AUC	0.891071	0.956429