```
import tensorflow as tf
import os
import numpy as np
import pandas as pd

from keras.layers import Input, Lambda, Dense, Flatten
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, Conv2D, MaxPool2D , Flatten
from keras.callbacks import Callback
from keras.callbacks import TensorBoard
import datetime
from tensorflow.keras.utils import plot_model

!gdown --id 1Z4TyI7FcFVEx8qdl4jO9qxvxaqLSqoEu

/usr/local/lib/python3.8/dist-packages/gdown/cli.py:127:
FutureWarning: Option `--id` was deprecated in version 4.3.1 and will
be removed in 5.0. You don't need to pass it anymore to use a file ID.
  warnings.warn(
Access denied with the following error:

     Cannot retrieve the public link of the file. You may need to
change
     the permission to 'Anyone with the link', or have had many
accesses.

You may still be able to access the file from the browser:

      https://drive.google.com/uc?id=1Z4TyI7FcFVEx8qdl4jO9qxvxaqLSqoEu



!wget --header="Host: doc-10-58-docs.googleusercontent.com" --
header="User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/109.0.0.0 Safari/537.36"
--header="Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image
/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9" --
header="Accept-Language: en-US,en;q=0.9" --header="Referer:
https://drive.google.com/" --header="Cookie:
AUTH_csgqhf4p22bquld8f6s95mu6pd817erv_nonce=q22o5hnasoge2" --
header="Connection: keep-alive" "https://doc-10-58-
docs.googleusercontent.com/docs/securesc/ka55uvoaagpve0kkmfos6gnjgr7s6
tea/lu2lpn9jj7baqgopd776arvajqn3jj9j/
1675136025000/00484516897554883881/00389752900367068778/1Z4TyI7FcFVEx8
qdl4jO9qxvxaqLSqoEu?e=download&ax=AB85Z1Ch6Ny_pkChc2waWK-3Q9-Q0clF-
Se1JmTx6lVAME1A6Bv6JHnvowFyI9BGI_-lp-Z1nsV04u-
```

```
yT5CSJsucdXKk1MWiansc7De4gX10WzZAmNvd0jq0hLyau9U6wWKr07QyiWs9FqThw7ag7
L-BIVhdfYFFxhP4tfDXURCJ9UI95vBGeE4HPTNjYZ-
js1hxeSzcLigKiROjDMgmJv4eue_dGlzxxtz2cKHNpseRruCp3bA-
BqX2VLvFGFJgMOkzSm99jzHH3ieOAvAhmuvnFZecBe30bpvIh2G9TmnpDcVI-
edoAEnq_j-SA5oj8dbiLMbt6i7wujHPgvGEV3g1pQyfqiwDcf_13VpOvNMMG-
sntYh_TitFYKy7hdtWTnCJfmn9Ud_VqU1Kz4p6E9nM2B8sApbK2nD9lW47gnWHQRPIouwA
B8AlgARZAjQsPym8C3ytxPu5la-
_5ilf1iwrBLnzdiZz5kymgVSWdjd48g_9GveTCddZFicK64rktETIRAqfDGiCjuEkU81dr
6oGeworByXAWGvqzaAzcL85i1D1YP69RbVdcEzQ2_kZioPhiTnnVGBllbnYCqJcl_j-
U6nSOjg5lxUarhH2P3v1SV8mH40tekwFxd2RgMqcOhUoiNDhFZi2cKI9IosTvwc-
F9YKatRRs8eCCNmfNyBTbBmK0phQpGk48CuHM0RkiRq95guFUSyXCZ2UhM9ckygT3o8SXg
cZ-doyX4SXsT-iZgAcvM0iQuxw-IbjFWKJqvFTNy-6ZXKTIWqVoWq5K4yP-
zn64R4Mv96BY64-d9l-vFY-
r7zUIvoLPBm3tS_2ktWxVjS_MWWQ3ktSKYLSVyMM4zIED95G0mOx5GlLtqZ3nKxyfJq7fi
8XH-2TrbFr4xw&uuid=d1144c7b-aca1-4755-a75b-
80324461cd83&authuser=0&nonce=q22o5hnasoge2&user=00389752900367068778&
hash=g1qhcu6ltvnv2qeqk8k4l8sild5u3bvk" -c -O 'rvl-cdip.rar'

--2023-01-31 03:34:43--  https://doc-10-58-
docs.googleusercontent.com/docs/securesc/ka55uvoaagpve0kkmfos6gnjgr7s6
tea/lu2lpn9jj7baqgopd776arvajqn3jj9j/
1675136025000/00484516897554883881/00389752900367068778/1Z4TyI7FcFVEx8
qdl4jO9qxvxaqLSqoEu?e=download&ax=AB85Z1Ch6Ny_pkChc2waWK-3Q9-Q0clF-
Se1JmTx6lVAME1A6Bv6JHnvowFyI9BGI_-lp-Z1nsV04u-
yT5CSJsucdXKk1MWiansc7De4gX10WzZAmNvd0jq0hLyau9U6wWKr07QyiWs9FqThw7ag7
L-BIVhdfYFFxhP4tfDXURCJ9UI95vBGeE4HPTNjYZ-
js1hxeSzcLigKiROjDMgmJv4eue_dGlzxxtz2cKHNpseRruCp3bA-
BqX2VLvFGFJgMOkzSm99jzHH3ieOAvAhmuvnFZecBe30bpvIh2G9TmnpDcVI-
edoAEnq_j-SA5oj8dbiLMbt6i7wujHPgvGEV3g1pQyfqiwDcf_13VpOvNMMG-
sntYh_TitFYKy7hdtWTnCJfmn9Ud_VqU1Kz4p6E9nM2B8sApbK2nD9lW47gnWHQRPIouwA
B8AlgARZAjQsPym8C3ytxPu5la-
_5ilf1iwrBLnzdiZz5kymgVSWdjd48g_9GveTCddZFicK64rktETIRAqfDGiCjuEkU81dr
6oGeworByXAWGvqzaAzcL85i1D1YP69RbVdcEzQ2_kZioPhiTnnVGBllbnYCqJcl_j-
U6nSOjg5lxUarhH2P3v1SV8mH40tekwFxd2RgMqcOhUoiNDhFZi2cKI9IosTvwc-
F9YKatRRs8eCCNmfNyBTbBmK0phQpGk48CuHM0RkiRq95guFUSyXCZ2UhM9ckygT3o8SXg
cZ-doyX4SXsT-iZgAcvM0iQuxw-IbjFWKJqvFTNy-6ZXKTIWqVoWq5K4yP-
zn64R4Mv96BY64-d9l-vFY-
r7zUIvoLPBm3tS_2ktWxVjS_MWWQ3ktSKYLSVyMM4zIED95G0mOx5GlLtqZ3nKxyfJq7fi
8XH-2TrbFr4xw&uuid=d1144c7b-aca1-4755-a75b-
80324461cd83&authuser=0&nonce=q22o5hnasoge2&user=00389752900367068778&
hash=g1qhcu6ltvnv2qeqk8k4l8sild5u3bvk
Resolving doc-10-58-docs.googleusercontent.com (doc-10-58-
docs.googleusercontent.com)... 142.251.111.132, 2607:f8b0:4004:c19::84
Connecting to doc-10-58-docs.googleusercontent.com (doc-10-58-
docs.googleusercontent.com)|142.251.111.132|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 4660541790 (4.3G) [application/rar]
Saving to: ‘rvl-cdip.rar’

rvl-cdip.rar        100%[===================>]   4.34G   191MB/s    in
```

```
23s

2023-01-31 03:35:07 (190 MB/s) - ‘rvl-cdip.rar’ saved
[4660541790/4660541790]


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

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

Mounted at /content/drive

df = pd.read_csv('labels_final.csv',dtype=str)
df.head()

                                          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

from keras_preprocessing.image import ImageDataGenerator
datagen = ImageDataGenerator(rescale=1/255., validation_split=0.2)
#Help in Image augmentation

dir_p = "/content/data_final"

from keras_preprocessing.image import ImageDataGenerator
# Help in Image augmentation
datagen=tf.keras.preprocessing.image.ImageDataGenerator(rescale=1/255,
validation_split=0.2)

# flow_from_dataframe used to loads the image dataset in memory and
generates batches of augmented data.
train_generator = datagen.flow_from_dataframe(dataframe= df,
directory=dir_p, x_col='path',y_col='label',

target_size=(128,128),class_mode='categorical',batch_size=128,subset='
training',seed=7)

Found 38400 validated image filenames belonging to 16 classes.

valid_generator = datagen.flow_from_dataframe(dataframe = df,
directory = dir_p,x_col='path',y_col='label',

target_size=(128,128),class_mode='categorical',batch_size=128,subset='
validation',seed=7)

Found 9600 validated image filenames belonging to 16 classes.

%reload_ext tensorboard
```

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

Model 1

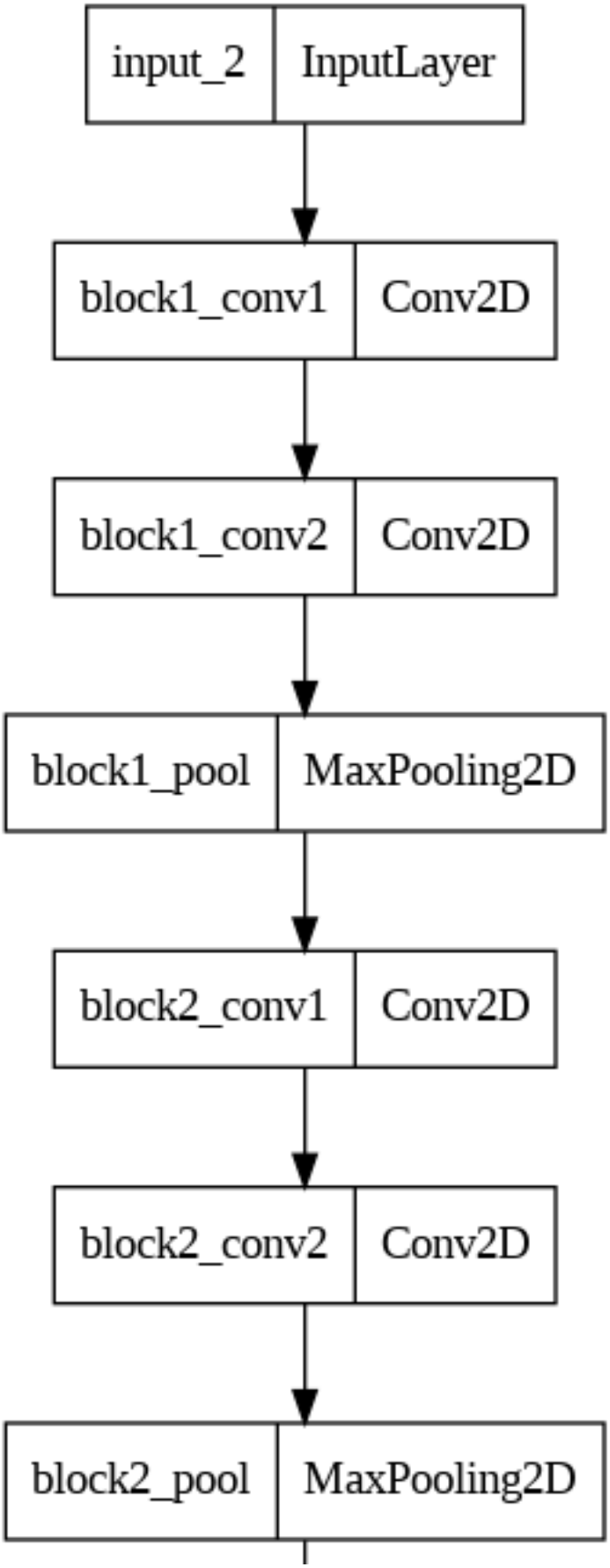
```
from tensorflow.keras.applications.vgg16 import VGG16
from tensorflow.keras.layers import
Dense,Input,Conv2D,MaxPooling2D,Flatten,Dropout
from tensorflow.keras.models import Model

# initilizing all the weights with Imagenet trained weights
# adding VGG-16 network without FC layers
vgg16_model = VGG16(weights='imagenet', include_top=False,
input_shape=(*(128, 128), 3))
# adding a new Conv block ( 1 Conv layer and 1 Maxpooling )
for layer in vgg16_model.layers:
  layer.trainable = False  #parameters from trainable goes into
non_trainable parameters

conv_layer=
Conv2D(512,kernel_size=(3,3),activation="relu",name="conv")
(vgg16_model.output)
max_pool_layer =  MaxPooling2D(name='Max_Pool')(conv_layer)
flatten_layer = Flatten(name="Flatten_layer")(max_pool_layer)
# 2 FC layers
fully_connected1=
Dense(1054,activation="relu",name="fully_connected1")(flatten_layer)
fully_connected2= Dense(256,activation="relu",name="fully_connected2")
(fully_connected1)
# output layer to classify 16 classes
op= Dense(16,activation="softmax")(fully_connected2)

#https://www.tensorflow.org/api_docs/python/tf/keras/Model
model1= Model(inputs=vgg16_model.input,outputs= op,name="model_1")
model1.compile(optimizer="Adam",
loss="categorical_crossentropy",metrics=["accuracy"])

model1.summary()
```

```
Model: "model_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_2 (InputLayer)        [(None, 128, 128, 3)]     0

 block1_conv1 (Conv2D)       (None, 128, 128, 64)      1792

 block1_conv2 (Conv2D)       (None, 128, 128, 64)      36928
```

block1_pool (MaxPooling2D)	(None, 64, 64, 64)	0
block2_conv1 (Conv2D)	(None, 64, 64, 128)	73856
block2_conv2 (Conv2D)	(None, 64, 64, 128)	147584
block2_pool (MaxPooling2D)	(None, 32, 32, 128)	0
block3_conv1 (Conv2D)	(None, 32, 32, 256)	295168
block3_conv2 (Conv2D)	(None, 32, 32, 256)	590080
block3_conv3 (Conv2D)	(None, 32, 32, 256)	590080
block3_pool (MaxPooling2D)	(None, 16, 16, 256)	0
block4_conv1 (Conv2D)	(None, 16, 16, 512)	1180160
block4_conv2 (Conv2D)	(None, 16, 16, 512)	2359808
block4_conv3 (Conv2D)	(None, 16, 16, 512)	2359808
block4_pool (MaxPooling2D)	(None, 8, 8, 512)	0
block5_conv1 (Conv2D)	(None, 8, 8, 512)	2359808
block5_conv2 (Conv2D)	(None, 8, 8, 512)	2359808
block5_conv3 (Conv2D)	(None, 8, 8, 512)	2359808
block5_pool (MaxPooling2D)	(None, 4, 4, 512)	0
conv (Conv2D)	(None, 2, 2, 512)	2359808
Max_Pool (MaxPooling2D)	(None, 1, 1, 512)	0
Flatten_layer (Flatten)	(None, 512)	0
fully_connected1 (Dense)	(None, 1054)	540702
fully_connected2 (Dense)	(None, 256)	270080
dense (Dense)	(None, 16)	4112

==========
Total params: 17,889,390
Trainable params: 3,174,702
Non-trainable params: 14,714,688
_____

```
plot_model(model1)
```


input_2
InputLayer
block1_conv1
Conv2D
block1_conv2
Conv2D
block1_pool
MaxPooling2D
block2_conv1
Conv2D
block2_conv2
Conv2D
block2_pool
MaxPooling2D

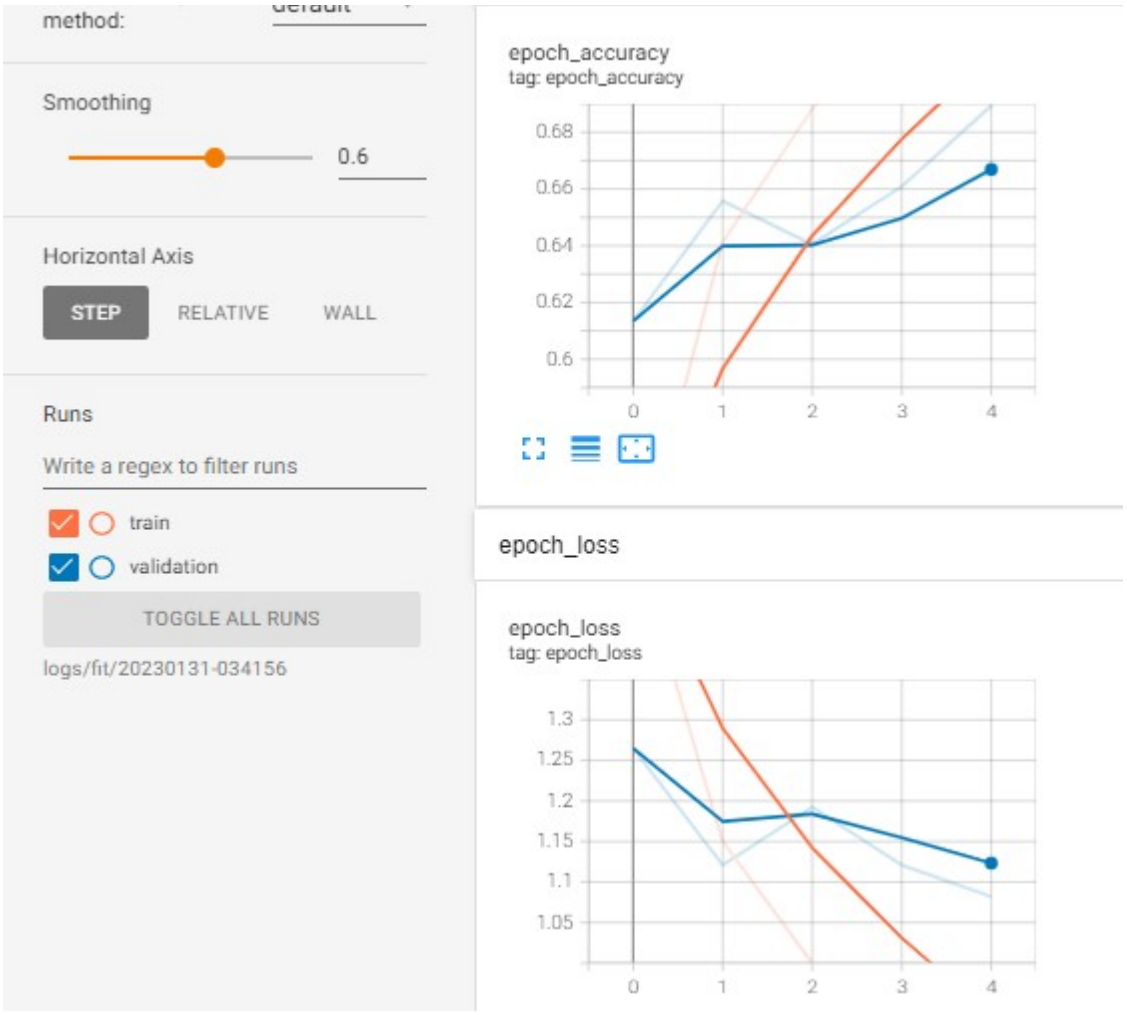
```
steps = len(train_generator.labels) // train_generator.batch_size
early_stoping = tf.keras.callbacks.EarlyStopping(patience=2)

log_dir="logs/fit/" + datetime.datetime.now().strftime("%Y%m%d-%H%M
%S")
tensorboard=
tf.keras.callbacks.TensorBoard(log_dir=log_dir,histogram_freq=1,
write_graph=True)
#callbacks_list = [early_stoping,tensorboard]

history = model1.fit(train_generator,
validation_data=valid_generator,epochs=5,
steps_per_epoch=steps,callbacks=tensorboard)
```

```
Epoch 1/5
300/300 [==============================] - 224s 706ms/step - loss:
1.5222 - accuracy: 0.5223 - val_loss: 1.2649 - val_accuracy: 0.6134
Epoch 2/5
300/300 [==============================] - 200s 666ms/step - loss:
1.1499 - accuracy: 0.6414 - val_loss: 1.1208 - val_accuracy: 0.6558
Epoch 3/5
300/300 [==============================] - 194s 647ms/step - loss:
0.9998 - accuracy: 0.6883 - val_loss: 1.1926 - val_accuracy: 0.6405
Epoch 4/5
300/300 [==============================] - 191s 638ms/step - loss:
0.9013 - accuracy: 0.7177 - val_loss: 1.1208 - val_accuracy: 0.6608
Epoch 5/5
300/300 [==============================] - 188s 628ms/step - loss:
0.8103 - accuracy: 0.7458 - val_loss: 1.0818 - val_accuracy: 0.6894
```

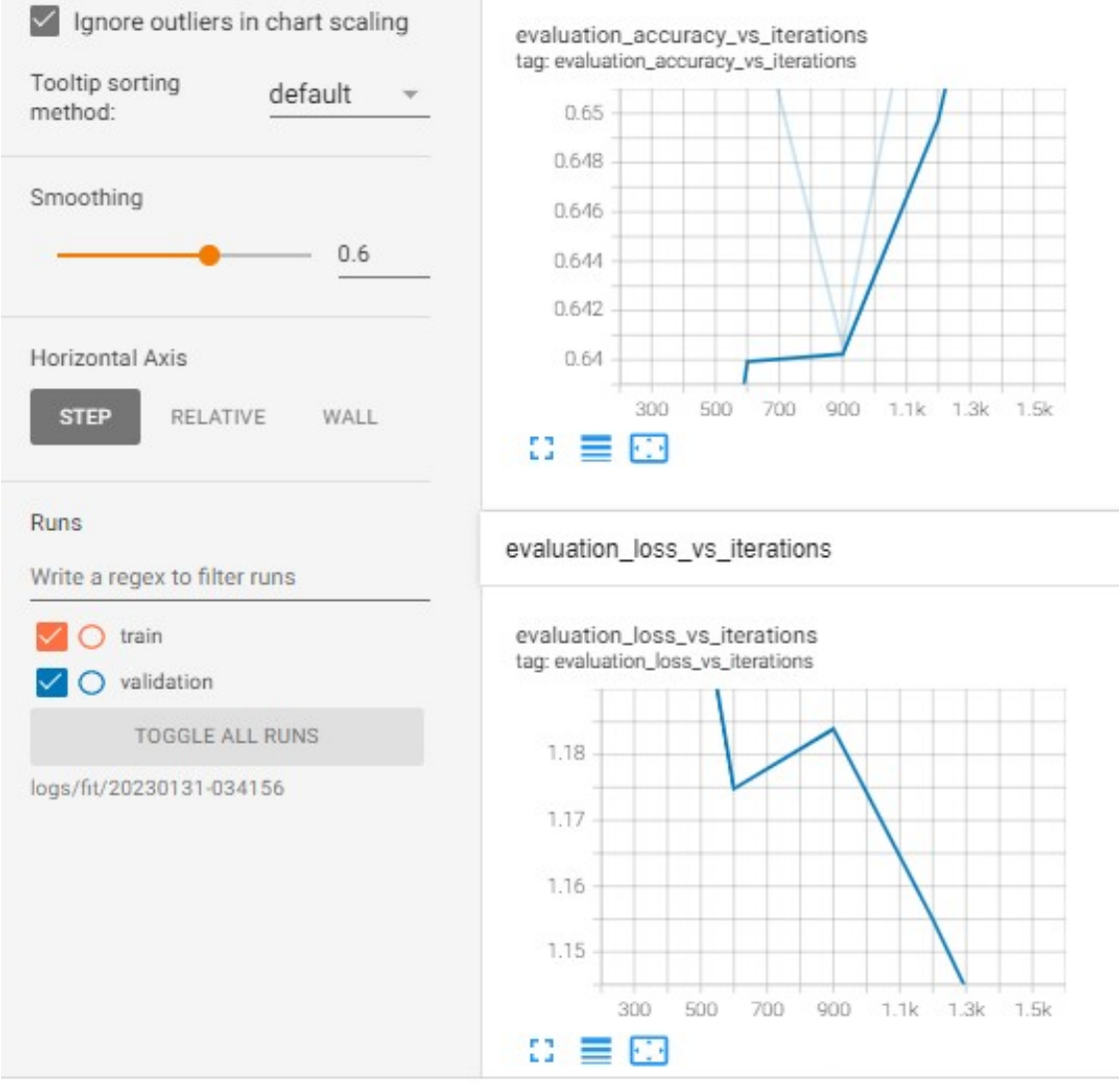
```
%reload_ext tensorboard
%tensorboard --logdir {log_dir}
```

```
<IPython.core.display.Javascript object>
```

```
import IPython.display as display
from PIL import Image
display.display(Image.open('t1.png'))
```




```
display.display(Image.open('t2.png'))
```



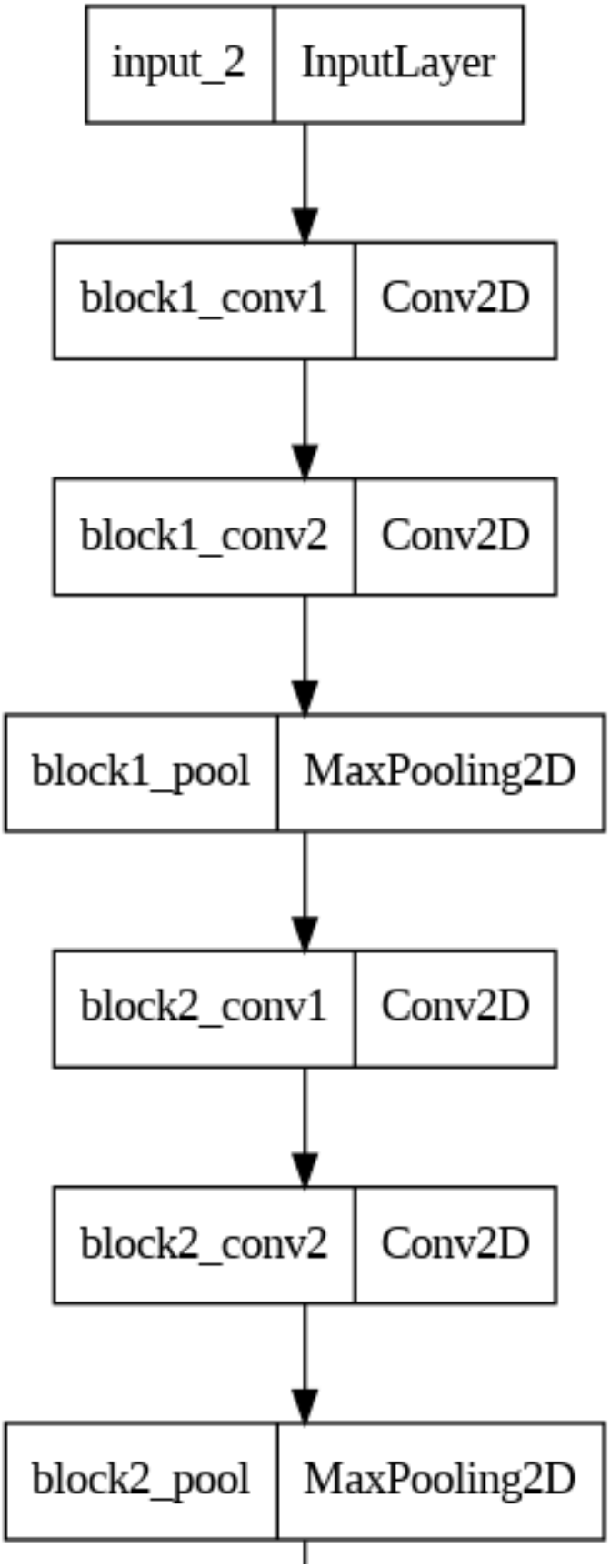
Model 2

```
vgg16_model2 = VGG16(weights='imagenet', include_top=False,
input_shape=(*(128,128), 3))

for layer in vgg16_model2.layers:
  layer.trainable = False #from book 14-61.17e ,as we see we freeze
the layers

# using conv layers only as Fully connected layer
# This conversion will reduce the No of Trainable parameters in FC
layers
conv_layer1=
Conv2D(512,kernel_size=(3,3),activation="relu",padding="same",name="co
nv1")(vgg16_model2.output)
conv_layer2=
Conv2D(128,kernel_size=(3,3),activation="relu",padding="same",name="co
```

```
nv2")(conv_layer1)
flatten= Flatten(name="Flatten_layer")(conv_layer2)
# output layer for 16 class classification
op= Dense(16,activation="softmax")(flatten)
#https://www.tensorflow.org/api_docs/python/tf/keras/Model
model2= Model(inputs=vgg16_model2.input,outputs=op,name="model_2")
model2.compile(optimizer="Adam",
loss="categorical_crossentropy",metrics=["accuracy"])

model2.summary()
```

```
Model: "model_2"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_2 (InputLayer)        [(None, 128, 128, 3)]     0

 block1_conv1 (Conv2D)       (None, 128, 128, 64)      1792

 block1_conv2 (Conv2D)       (None, 128, 128, 64)      36928

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

 block2_conv1 (Conv2D)       (None, 64, 64, 128)       73856

 block2_conv2 (Conv2D)       (None, 64, 64, 128)       147584

 block2_pool (MaxPooling2D)  (None, 32, 32, 128)       0

 block3_conv1 (Conv2D)       (None, 32, 32, 256)       295168

 block3_conv2 (Conv2D)       (None, 32, 32, 256)       590080

 block3_conv3 (Conv2D)       (None, 32, 32, 256)       590080

 block3_pool (MaxPooling2D)  (None, 16, 16, 256)       0

 block4_conv1 (Conv2D)       (None, 16, 16, 512)       1180160

 block4_conv2 (Conv2D)       (None, 16, 16, 512)       2359808

 block4_conv3 (Conv2D)       (None, 16, 16, 512)       2359808

 block4_pool (MaxPooling2D)  (None, 8, 8, 512)         0

 block5_conv1 (Conv2D)       (None, 8, 8, 512)         2359808

 block5_conv2 (Conv2D)       (None, 8, 8, 512)         2359808

 block5_conv3 (Conv2D)       (None, 8, 8, 512)         2359808
```

```
 block5_pool (MaxPooling2D)  (None, 4, 4, 512)         0

 conv1 (Conv2D)              (None, 4, 4, 512)         2359808

 conv2 (Conv2D)              (None, 4, 4, 128)         589952

 Flatten_layer (Flatten)     (None, 2048)              0

 dense_1 (Dense)             (None, 16)                32784

=================================================================
Total params: 17,697,232
Trainable params: 2,982,544
Non-trainable params: 14,714,688
_________________________________________________________________

plot_model(model2)
```


input_2
InputLayer
block1_conv1
Conv2D
block1_conv2
Conv2D
block1_pool
MaxPooling2D
block2_conv1
Conv2D
block2_conv2
Conv2D
block2_pool
MaxPooling2D

```
steps = len(train_generator.labels) // train_generator.batch_size
log_dir="logs/fit/" + datetime.datetime.now().strftime("%Y%m%d-%H%M
%S")
tensorboard=
tf.keras.callbacks.TensorBoard(log_dir=log_dir,histogram_freq=1,
write_graph=True)

history = model2.fit(train_generator,
validation_data=valid_generator,epochs=5,
steps_per_epoch=steps,callbacks=tensorboard)
```

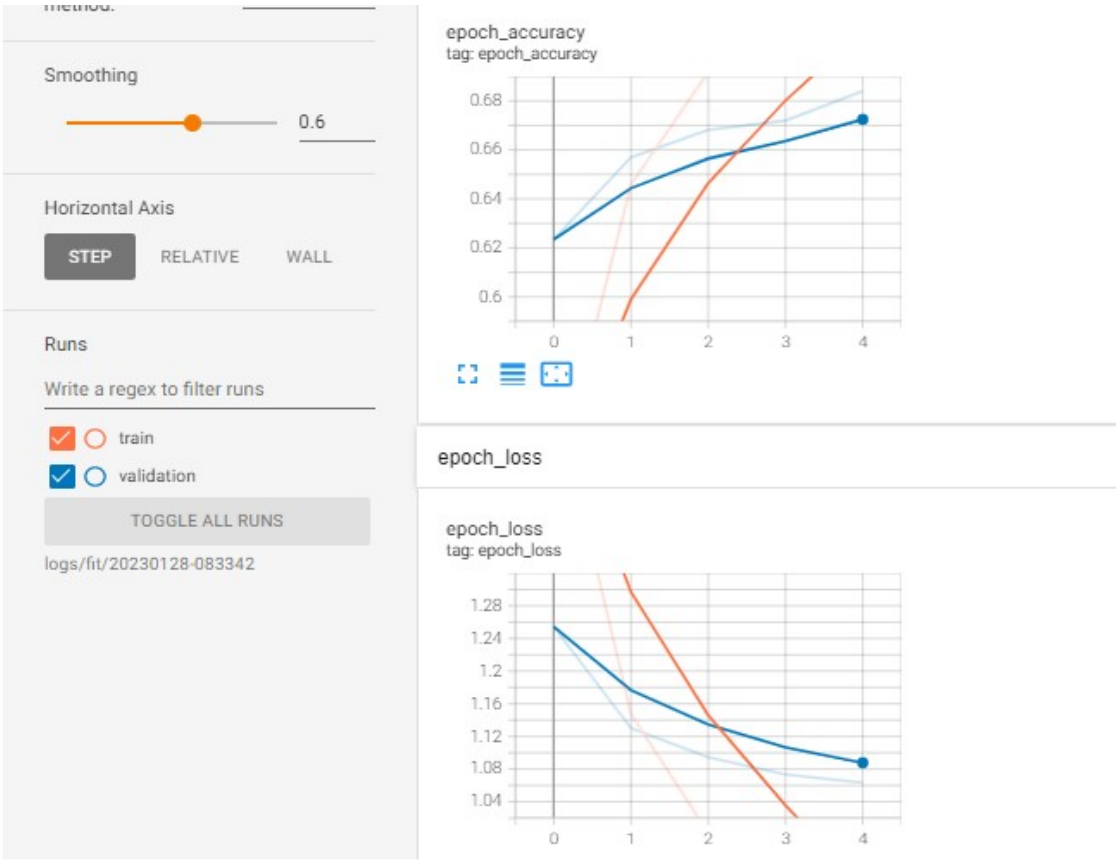
```
Epoch 1/5
300/300 [==============================] - 218s 724ms/step - loss:
1.5175 - accuracy: 0.5300 - val_loss: 1.3092 - val_accuracy: 0.5885
Epoch 2/5
300/300 [==============================] - 206s 688ms/step - loss:
1.1280 - accuracy: 0.6524 - val_loss: 1.1621 - val_accuracy: 0.6427
Epoch 3/5
300/300 [==============================] - 205s 683ms/step - loss:
0.9902 - accuracy: 0.6936 - val_loss: 1.0685 - val_accuracy: 0.6791
Epoch 4/5
300/300 [==============================] - 205s 682ms/step - loss:
0.8818 - accuracy: 0.7253 - val_loss: 1.1086 - val_accuracy: 0.6673
Epoch 5/5
300/300 [==============================] - 211s 705ms/step - loss:
0.7889 - accuracy: 0.7537 - val_loss: 1.0636 - val_accuracy: 0.6970
```

```
%reload_ext tensorboard
%tensorboard --logdir {log_dir}
```

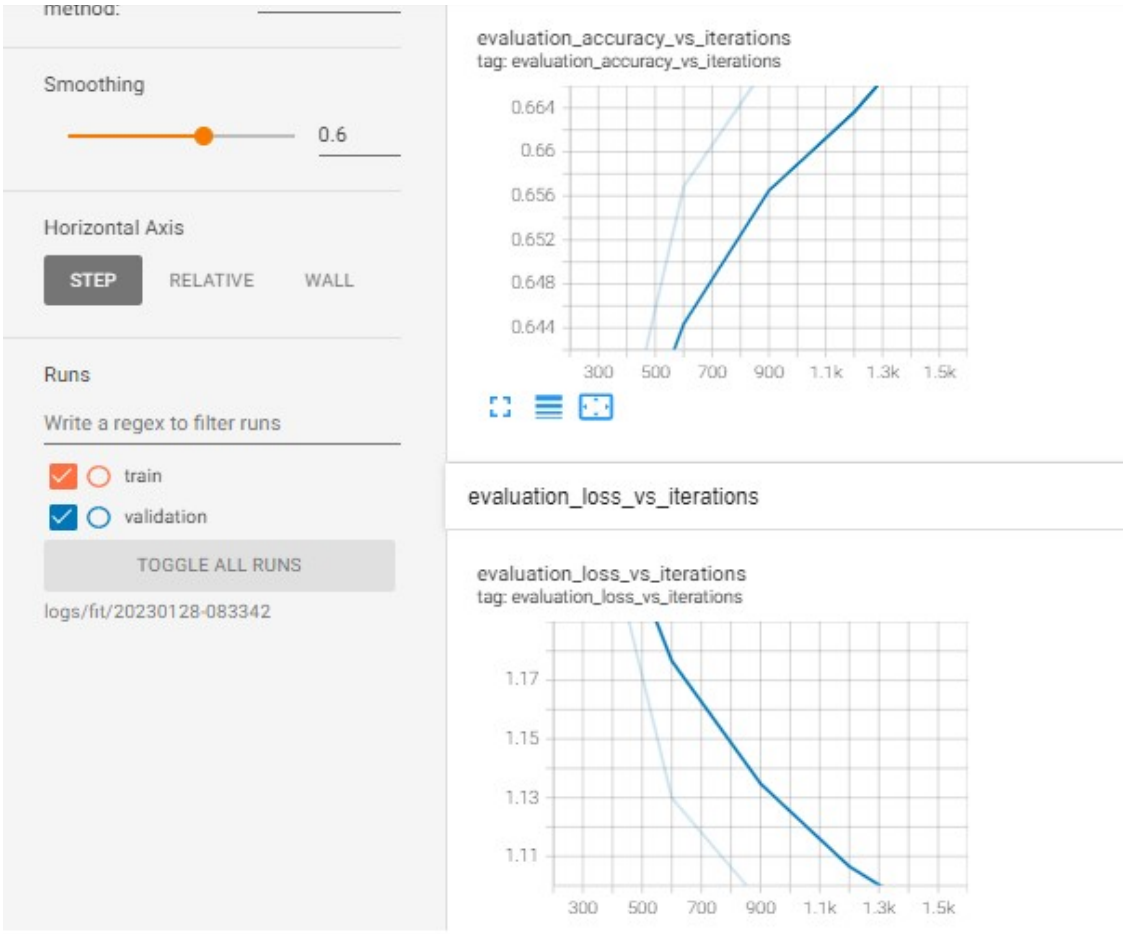
```
Reusing TensorBoard on port 6006 (pid 8578), started 0:01:23 ago. (Use
'!kill 8578' to kill it.)
```

```
<IPython.core.display.Javascript object>
```

```
import IPython.display as display
from PIL import Image
display.display(Image.open('ts.png'))
```




```
display.display(Image.open('ts2.png'))
```



Model 3

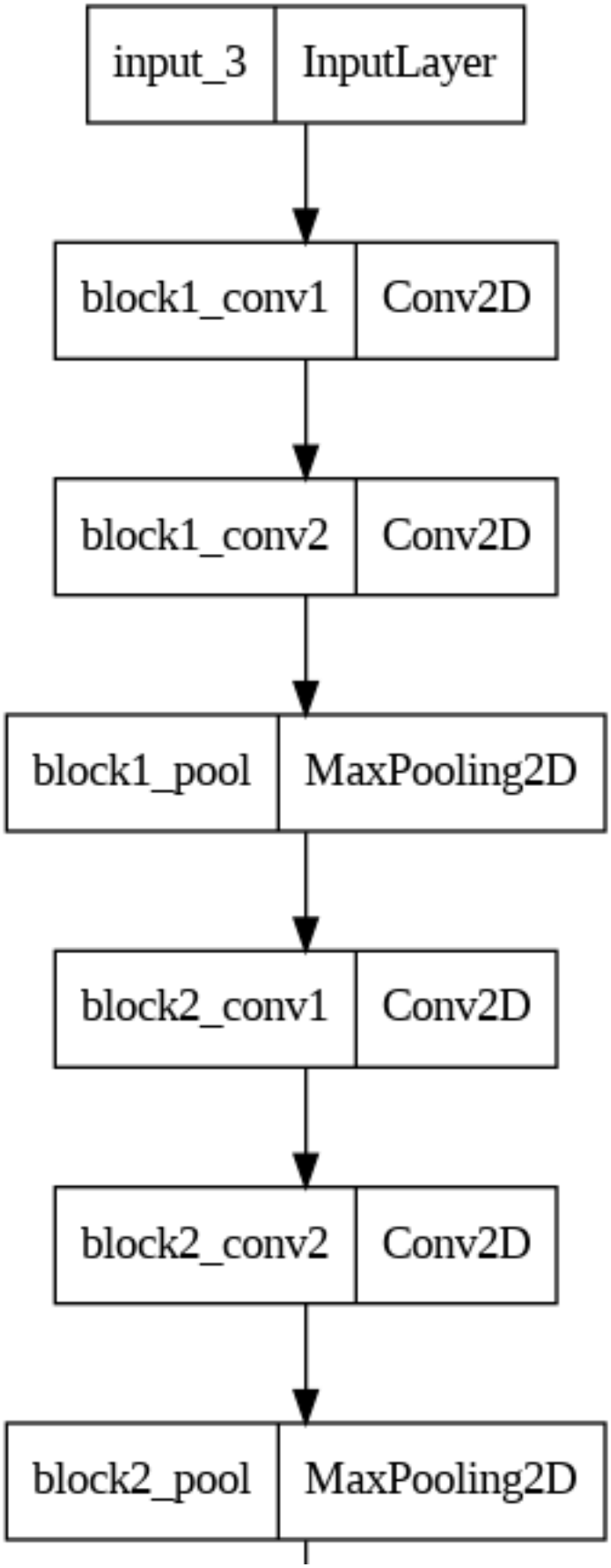
```
vgg16base_model3 = VGG16(weights='imagenet',
include_top=False,input_shape=(*(128,128), 3))

for layer in vgg16base_model3.layers[0:-6]:  #training only Last 6
Layers of VGG-16 network
  layer.trainable = False

conv_layer1=
Conv2D(512,kernel_size=(3,3),activation="relu",padding="same",name="co
nv1")(vgg16base_model3.output)
conv_layer2=
Conv2D(128,kernel_size=(3,3),activation="relu",padding="same",name="co
nv2")(conv_layer1)
flatten= Flatten(name="Flatten_layer")(conv_layer2)

otput= Dense(16,activation="softmax")(flatten)
#optm =  tf.keras.optimizers.SGD(learning_rate=0.001, momentum=0.8)
#https://www.tensorflow.org/api_docs/python/tf/keras/Model
model3=
```

```
Model(inputs=vgg16base_model3.input,outputs=otput,name="model_3")
model3.compile(optimizer="Adam",
loss="categorical_crossentropy",metrics=["accuracy"])

model3.summary()
```

Model: "model_3"

Layer (type)	Output Shape	Param #
input_6 (InputLayer)	[(None, 128, 128, 3)]	0
block1_conv1 (Conv2D)	(None, 128, 128, 64)	1792
block1_conv2 (Conv2D)	(None, 128, 128, 64)	36928
block1_pool (MaxPooling2D)	(None, 64, 64, 64)	0
block2_conv1 (Conv2D)	(None, 64, 64, 128)	73856
block2_conv2 (Conv2D)	(None, 64, 64, 128)	147584
block2_pool (MaxPooling2D)	(None, 32, 32, 128)	0
block3_conv1 (Conv2D)	(None, 32, 32, 256)	295168
block3_conv2 (Conv2D)	(None, 32, 32, 256)	590080
block3_conv3 (Conv2D)	(None, 32, 32, 256)	590080
block3_pool (MaxPooling2D)	(None, 16, 16, 256)	0
block4_conv1 (Conv2D)	(None, 16, 16, 512)	1180160
block4_conv2 (Conv2D)	(None, 16, 16, 512)	2359808
block4_conv3 (Conv2D)	(None, 16, 16, 512)	2359808
block4_pool (MaxPooling2D)	(None, 8, 8, 512)	0
block5_conv1 (Conv2D)	(None, 8, 8, 512)	2359808
block5_conv2 (Conv2D)	(None, 8, 8, 512)	2359808
block5_conv3 (Conv2D)	(None, 8, 8, 512)	2359808
block5_pool (MaxPooling2D)	(None, 4, 4, 512)	0
conv1 (Conv2D)	(None, 4, 4, 512)	2359808
conv2 (Conv2D)	(None, 4, 4, 128)	589952

```
 Flatten_layer (Flatten)     (None, 2048)              0

 dense_6 (Dense)             (None, 16)                32784

=================================================================
Total params: 17,697,232
Trainable params: 12,421,776
Non-trainable params: 5,275,456
_________________________________________________________________

plot_model(model3)
```


input_3
InputLayer
block1_conv1
Conv2D
block1_conv2
Conv2D
block1_pool
MaxPooling2D
block2_conv1
Conv2D
block2_conv2
Conv2D
block2_pool
MaxPooling2D

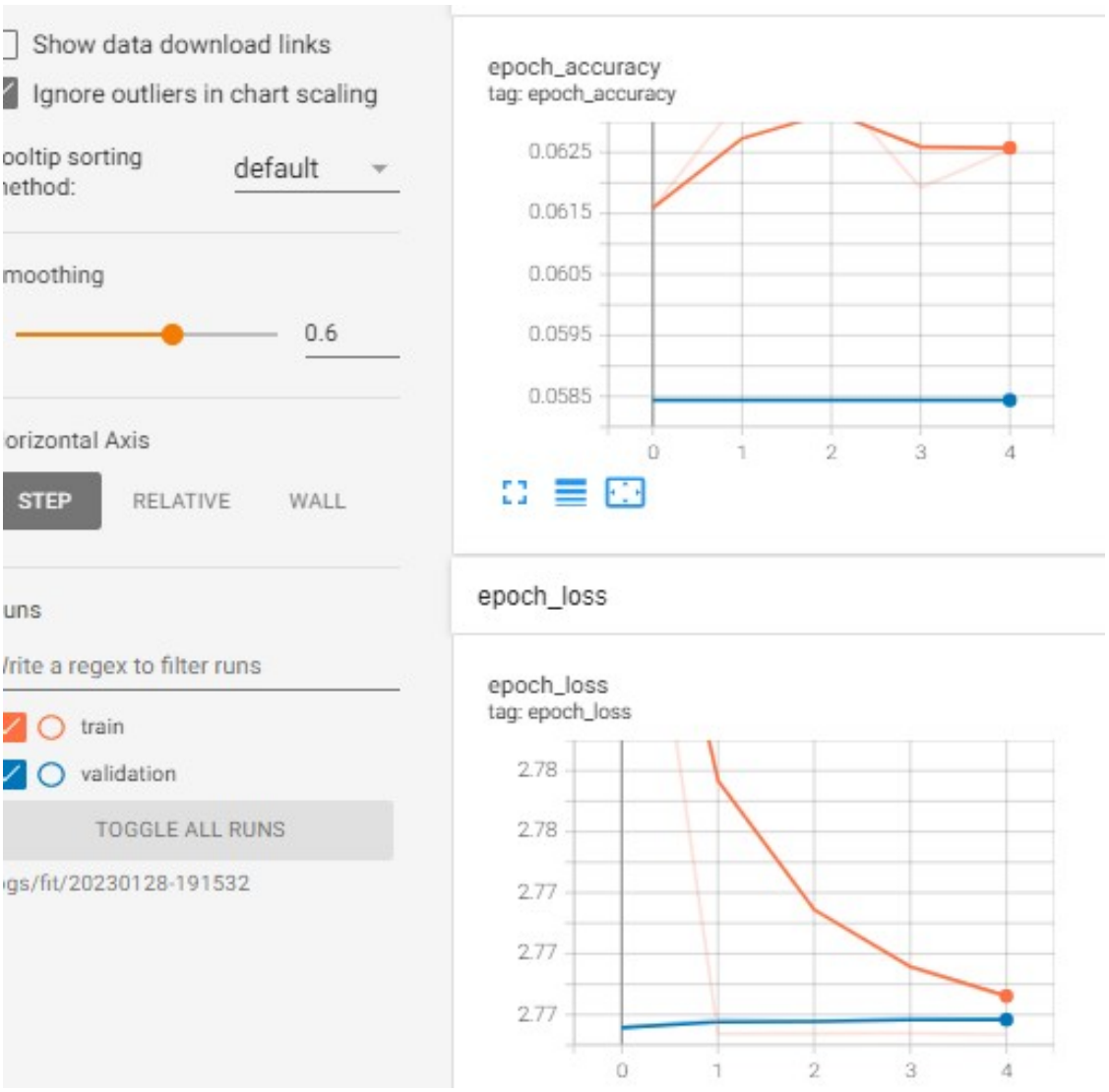
```
history = model3.fit(train_generator,
validation_data=valid_generator,epochs=5,
steps_per_epoch=steps,callbacks=tensorboard)
```

```
Epoch 1/5
300/300 [==============================] - 198s 657ms/step - loss:
2.7928 - accuracy: 0.0582 - val_loss: 2.7727 - val_accuracy: 0.0584
Epoch 2/5
300/300 [==============================] - 205s 685ms/step - loss:
2.7727 - accuracy: 0.0626 - val_loss: 2.7728 - val_accuracy: 0.0584
Epoch 3/5
300/300 [==============================] - 208s 694ms/step - loss:
2.7727 - accuracy: 0.0626 - val_loss: 2.7729 - val_accuracy: 0.0584
Epoch 4/5
300/300 [==============================] - 209s 697ms/step - loss:
2.7727 - accuracy: 0.0614 - val_loss: 2.7729 - val_accuracy: 0.0584
Epoch 5/5
300/300 [==============================] - 210s 701ms/step - loss:
2.7727 - accuracy: 0.0620 - val_loss: 2.7730 - val_accuracy: 0.0584
```

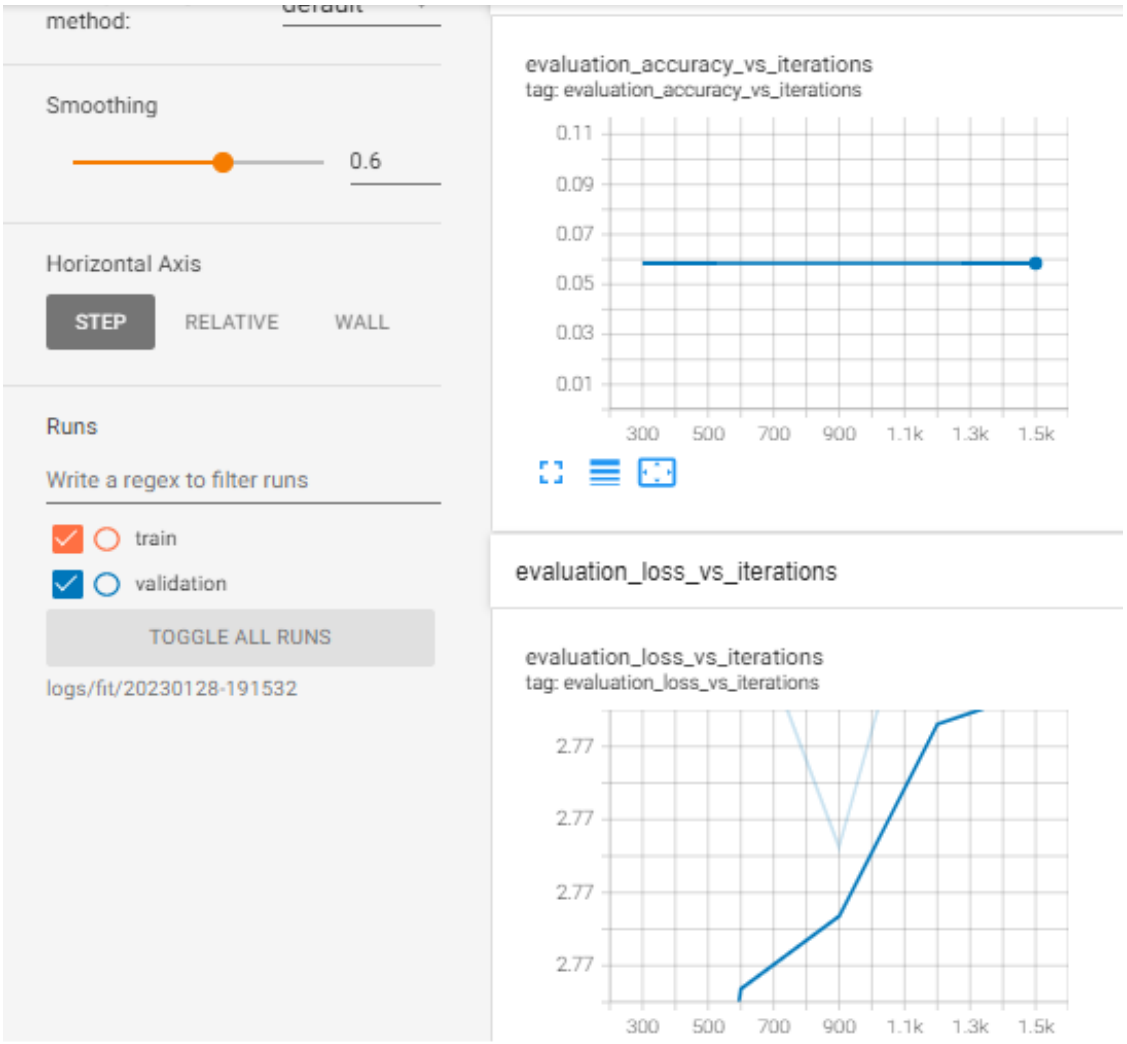
```
%tensorboard --logdir {log_dir}
```

```
<IPython.core.display.Javascript object>
```

```
display.display(Image.open('ts5.png'))
```




```
display.display(Image.open('ts6.png'))
```



Summary

```
from prettytable import PrettyTable
myTable = PrettyTable(['Model
','Max_Train_Accuracy','Max_Validation_Accuracy'])
myTable.add_row(['Model-1','0.7458','0.6894'])
myTable.add_row(['Model-2','0.7537','0.6970'])
myTable.add_row(['Model-3','0.0620','0.0584'])
print(myTable)
```

```
+---------+--------------------+-------------------------+
| Model   | Max_Train_Accuracy | Max_Validation_Accuracy |
+---------+--------------------+-------------------------+
| Model-1 |       0.7458       |          0.6894         |
| Model-2 |       0.7537       |          0.6970         |
| Model-3 |       0.0620       |          0.0584         |
+---------+--------------------+-------------------------+
```