Task - 2

In [2]:
```
from google.colab import drive
drive.mount('/gdrive')
%cd /gdrive
```

```
Go to this URL in a browser: https://accounts.google.com/o/oauth2/auth?client_id=947318989803-6bn6qk8qdgf4n4g3pfee6491hc0brc4i.apps.googleusercontent.com&redirect_uri=urn%3aietf%3awg%3aoauth%3a2.0%3aoob&scope=email%20https%3a%2f%2fwww.googleapis.com%2fauth%2fdocs.test%20https%3a%2f%2fwww.googleapis.com%2fauth%2fdrive%20https%3a%2f%2fwww.googleapis.com%2fauth%2fdrive.photos.readonly%20https%3a%2f%2fwww.googleapis.com%2fauth%2fpeopleapi.readonly&response_type=code (https://accounts.google.com/o/oauth2/auth?client_id=947318989803-6bn6qk8qdgf4n4g3pfee6491hc0brc4i.apps.googleusercontent.com&redirect_uri=urn%3aietf%3awg%3aoauth%3a2.0%3aoob&scope=email%20https%3a%2f%2fwww.googleapis.com%2fauth%2fdocs.test%20https%3a%2f%2fwww.googleapis.com%2fauth%2fdrive%20https%3a%2f%2fwww.googleapis.com%2fauth%2fdrive.photos.readonly%20https%3a%2f%2fwww.googleapis.com%2fauth%2fpeopleapi.readonly&response_type=code)

Enter your authorization code:
··········
Mounted at /gdrive
/gdrive
```

In [3]:
```
%tensorflow_version 2.x
```

In [4]:
```
import pandas as pd
import shutil
import os
from tqdm import tqdm
import tensorflow as tf
#importing tensorflow
from tensorflow.keras.layers import Dense,Input,Conv2D,MaxPool2D,Activation,Dropout,Flatten
from tensorflow.keras.models import Model
import random as rn
from tensorflow.keras import applications
from tensorflow.keras.models import Sequential
import numpy as np
from tensorflow.keras.callbacks import TensorBoard
from tensorflow.keras.callbacks import ModelCheckpoint
from tensorflow.keras.callbacks import ReduceLROnPlateau, EarlyStopping
import datetime
#importing tensorflow
from tensorflow.keras.layers import Dense,Input,Conv2D,MaxPool2D,Activation,Dropout,Flatten
from tensorflow.keras.models import Model
import random as rn
```

In [5]:
```
tf.__version__
```

Out[5]:
```
'2.3.0'
```

In [6]:
```
tf.test.gpu_device_name()
```

Out[6]:
```
'/device:GPU:0'
```

In [7]:
```
dir_path = '/gdrive/My Drive/Data_Final_main/train'
```

In [7]:
```
# Train folder classes count
for i in os.listdir(dir_path):

    print("No of Images in ",i," category is ",len(os.listdir(os.path.join(dir_path,i))))
```

```
No of Images in  news_article  category is  2101
No of Images in  presentation  category is  2106
No of Images in  questionnaire  category is  2106
No of Images in  resume  category is  2104
No of Images in  scientific_publication  category is  2085
No of Images in  scientific_report  category is  2099
No of Images in  specification  category is  2100
No of Images in  advertisement  category is  2094
No of Images in  budget  category is  2102
No of Images in  email  category is  2093
No of Images in  file_folder  category is  2103
No of Images in  form  category is  2094
No of Images in  handwritten  category is  2105
No of Images in  invoice  category is  2092
No of Images in  letter  category is  2431
No of Images in  memo  category is  2096
```

In [8]:
```
dir_path_test = '/gdrive/My Drive/Data_Final_main/val_test'
```

In [9]:
```
for i in os.listdir(dir_path_test):

    print("No of Images in ",i," category is ",len(os.listdir(os.path.join(dir_path_test,i))))
```

```
No of Images in  advertisement  category is  900
No of Images in  budget  category is  900
No of Images in  email  category is  900
No of Images in  file_folder  category is  900
No of Images in  form  category is  900
No of Images in  handwritten  category is  900
No of Images in  invoice  category is  900
No of Images in  letter  category is  900
No of Images in  memo  category is  900
No of Images in  news_article  category is  900
No of Images in  presentation  category is  900
No of Images in  questionnaire  category is  900
No of Images in  resume  category is  900
No of Images in  scientific_publication  category is  900
No of Images in  scientific_report  category is  900
No of Images in  specification  category is  900
```

In [9]:
```
train_data_dir = dir_path
validation_data_dir = dir_path_test
```

In [10]:
```
epochs = 20
batch_size = 32
#batch_size = 128
img_width, img_height = 150, 150
#img_width, img_height = 224,224
```

In [11]:
```
# prepare data augmentation configuration
train_datagen = tf.keras.preprocessing.image.ImageDataGenerator(
    rescale=1. / 255,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True)



test_datagen = tf.keras.preprocessing.image.ImageDataGenerator(rescale=1. / 255)



train_generator = train_datagen.flow_from_directory(
    train_data_dir,
    target_size=(150,150),
    batch_size=batch_size,
    class_mode='categorical')


validation_generator = test_datagen.flow_from_directory(
    validation_data_dir,
    target_size=(150,150),
    batch_size=batch_size,
    class_mode='categorical')
```

```
Found 33911 images belonging to 16 classes.
Found 14400 images belonging to 16 classes.
```

In [14]:
```
# Create Model
os.environ['PYTHONHASHSEED'] = '0'

##https://keras.io/getting-started/faq/#how-can-i-obtain-reproducible-results-using-keras-during-development
## Have to clear the session. If you are not clearing, Graph will create again and again and graph size will increses.
## Varibles will also set to some value from before session
tf.keras.backend.clear_session()

## Set the random seed values to regenerate the model.
np.random.seed(0)
rn.seed(0)



#Get back the convolutional part of a VGG network trained on ImageNet
model_vgg16_conv = applications.VGG16(weights='imagenet', include_top=False,input_shape=(150,150,3))


# Freezing No trainable layer
for layer in model_vgg16_conv.layers:
    layer.trainable = False
#model_vgg16_conv.summary()

#Input layer - Create your own input format (here 150,150,3)
input_layer = Input(shape=(150,150,3),name='Input_Layer')

#Use the generated model
output_vgg16_conv = model_vgg16_conv(input_layer)

#Conv Layer
Conv1 = Conv2D(filters=64,kernel_size=(4,4),strides=(1,1),padding='valid',data_format='channels_last',
              activation='relu',kernel_initializer=tf.keras.initializers.he_normal(seed=0),name='Conv1')(output_vgg16_conv)


Conv2 = Conv2D(filters=32,kernel_size=(1,1),strides=(1,1),padding='valid',data_format='channels_last',
              activation='relu',kernel_initializer=tf.keras.initializers.he_normal(seed=0),name='Conv2')(Conv1)




output_flat = Flatten(data_format='channels_last',name='Flatten')(Conv2)
```

```
#output layer
Out = Dense(units=16,activation='softmax',kernel_initializer=tf.keras.initializers.he_normal(seed=3),name='Output')(output_flat)


model = Model(inputs=input_layer,outputs=Out)


model.summary()
```

```
Model: "functional_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
Input_Layer (InputLayer)     [(None, 150, 150, 3)]     0
_________________________________________________________________
vgg16 (Functional)           (None, 4, 4, 512)         14714688
_________________________________________________________________
Conv1 (Conv2D)               (None, 1, 1, 64)          524352
_________________________________________________________________
Conv2 (Conv2D)               (None, 1, 1, 32)          2080
_________________________________________________________________
Flatten (Flatten)            (None, 32)                0
_________________________________________________________________
Output (Dense)               (None, 16)                528
=================================================================
Total params: 15,241,648
Trainable params: 526,960
Non-trainable params: 14,714,688
_________________________________________________________________
```

In [15]:
```
# eARLY sTOOPING
earlystop = EarlyStopping(monitor='val_loss', patience=5, verbose=1)


reduce_lr = ReduceLROnPlateau(monitor='val_loss', factor=0.2,
                              patience=3, min_lr=0.001)

##Callbacks
#file path, it saves the model in the 'model_save' folder and we are naming model with epoch number
#and val acc to differtiate with other models
#you have to create model_save folder before running the code.
filepath="model_save/weights-{epoch:02d}.hdf5"
checkpoint = ModelCheckpoint(filepath=filepath, monitor='val_loss',  verbose=1, save_best_only=True, mode='auto')
```

In [16]:
```
# TensorBoard Creation
%load_ext tensorboard
folder_name = datetime.datetime.now().strftime("%Y%m%d-%H%M%S")
```

In [17]:
```
# Create log folder - TensorBoard
log_dir="/gdrive/My Drive/logs/fit/" + folder_name
tensorboard_callback =TensorBoard(log_dir=log_dir,histogram_freq=0, write_graph=True)
```

In [18]:
```
folder_name
```

Out[18]:
```
'20200907-072154'
```

In [20]:
```
#compiling
model.compile(optimizer=tf.keras.optimizers.Adam(lr=0.001),loss='categorical_crossentropy',metrics=['accuracy'])
```

In [21]:
```
# test Sep 07

##fitting generator
model.fit(train_generator,steps_per_epoch=1060,epochs=25,
           validation_data=validation_generator,
validation_steps=450,
callbacks=[reduce_lr,earlystop,tensorboard_callback])
```

```
Epoch 1/25
   1/1060 [..............................] - ETA: 0s - loss: 3.2627 - accuracy: 0.0000e+00WARNING:tensorflow:From /usr/local/lib/python3.6/dist-pack
ages/tensorflow/python/ops/summary_ops_v2.py:1277: stop (from tensorflow.python.eager.profiler) is deprecated and will be removed after 2020-07-01.
Instructions for updating:
use `tf.profiler.experimental.stop` instead.
1060/1060 [==============================] - 14350s 14s/step - loss: 1.7877 - accuracy: 0.4406 - val_loss: 1.4689 - val_accuracy: 0.5433
Epoch 2/25
1060/1060 [==============================] - 321s 303ms/step - loss: 1.4789 - accuracy: 0.5416 - val_loss: 1.3967 - val_accuracy: 0.5742
Epoch 3/25
1060/1060 [==============================] - 315s 297ms/step - loss: 1.3937 - accuracy: 0.5684 - val_loss: 1.3419 - val_accuracy: 0.5890
Epoch 4/25
1060/1060 [==============================] - 326s 307ms/step - loss: 1.3291 - accuracy: 0.5905 - val_loss: 1.3239 - val_accuracy: 0.6079
Epoch 5/25
1060/1060 [==============================] - 323s 304ms/step - loss: 1.2931 - accuracy: 0.6015 - val_loss: 1.2972 - val_accuracy: 0.6111
Epoch 6/25
1060/1060 [==============================] - 320s 302ms/step - loss: 1.2707 - accuracy: 0.6082 - val_loss: 1.3668 - val_accuracy: 0.5792
Epoch 7/25
1060/1060 [==============================] - 318s 300ms/step - loss: 1.2398 - accuracy: 0.6173 - val_loss: 1.2795 - val_accuracy: 0.6138
Epoch 8/25
1060/1060 [==============================] - 317s 299ms/step - loss: 1.2185 - accuracy: 0.6265 - val_loss: 1.2459 - val_accuracy: 0.6301
Epoch 9/25
1060/1060 [==============================] - 317s 299ms/step - loss: 1.1990 - accuracy: 0.6308 - val_loss: 1.2823 - val_accuracy: 0.6163
Epoch 10/25
1060/1060 [==============================] - 311s 293ms/step - loss: 1.1852 - accuracy: 0.6358 - val_loss: 1.2396 - val_accuracy: 0.6286
Epoch 11/25
1060/1060 [==============================] - 316s 298ms/step - loss: 1.1737 - accuracy: 0.6385 - val_loss: 1.2446 - val_accuracy: 0.6297
Epoch 12/25
1060/1060 [==============================] - 317s 299ms/step - loss: 1.1583 - accuracy: 0.6434 - val_loss: 1.2393 - val_accuracy: 0.6285
Epoch 13/25
1060/1060 [==============================] - 323s 304ms/step - loss: 1.1353 - accuracy: 0.6494 - val_loss: 1.2737 - val_accuracy: 0.6260
Epoch 14/25
1060/1060 [==============================] - 322s 304ms/step - loss: 1.1280 - accuracy: 0.6504 - val_loss: 1.2294 - val_accuracy: 0.6335
Epoch 15/25
1060/1060 [==============================] - 324s 306ms/step - loss: 1.1118 - accuracy: 0.6586 - val_loss: 1.2534 - val_accuracy: 0.6275
Epoch 16/25
```

```
1060/1060 [==============================] - 329s 310ms/step - loss: 1.1152 - accuracy: 0.6550 - val_loss: 1.2705 - val_accuracy: 0.6226
Epoch 17/25
1060/1060 [==============================] - 330s 311ms/step - loss: 1.1024 - accuracy: 0.6617 - val_loss: 1.2071 - val_accuracy: 0.6422
Epoch 18/25
1060/1060 [==============================] - 330s 311ms/step - loss: 1.0927 - accuracy: 0.6609 - val_loss: 1.2052 - val_accuracy: 0.6422
Epoch 19/25
1060/1060 [==============================] - 328s 309ms/step - loss: 1.0898 - accuracy: 0.6618 - val_loss: 1.2358 - val_accuracy: 0.6369
Epoch 20/25
1060/1060 [==============================] - 325s 306ms/step - loss: 1.0758 - accuracy: 0.6682 - val_loss: 1.2633 - val_accuracy: 0.6278
Epoch 21/25
1060/1060 [==============================] - 327s 308ms/step - loss: 1.0682 - accuracy: 0.6700 - val_loss: 1.1957 - val_accuracy: 0.6451
Epoch 22/25
1060/1060 [==============================] - 330s 311ms/step - loss: 1.0668 - accuracy: 0.6676 - val_loss: 1.2409 - val_accuracy: 0.6367
Epoch 23/25
1060/1060 [==============================] - 334s 315ms/step - loss: 1.0586 - accuracy: 0.6712 - val_loss: 1.1965 - val_accuracy: 0.6535
Epoch 24/25
1060/1060 [==============================] - 327s 309ms/step - loss: 1.0514 - accuracy: 0.6721 - val_loss: 1.2843 - val_accuracy: 0.6260
Epoch 25/25
1060/1060 [==============================] - 327s 309ms/step - loss: 1.0410 - accuracy: 0.6782 - val_loss: 1.2379 - val_accuracy: 0.6333
```

Out[21]:
```
<tensorflow.python.keras.callbacks.History at 0x7f61906506d8>
```

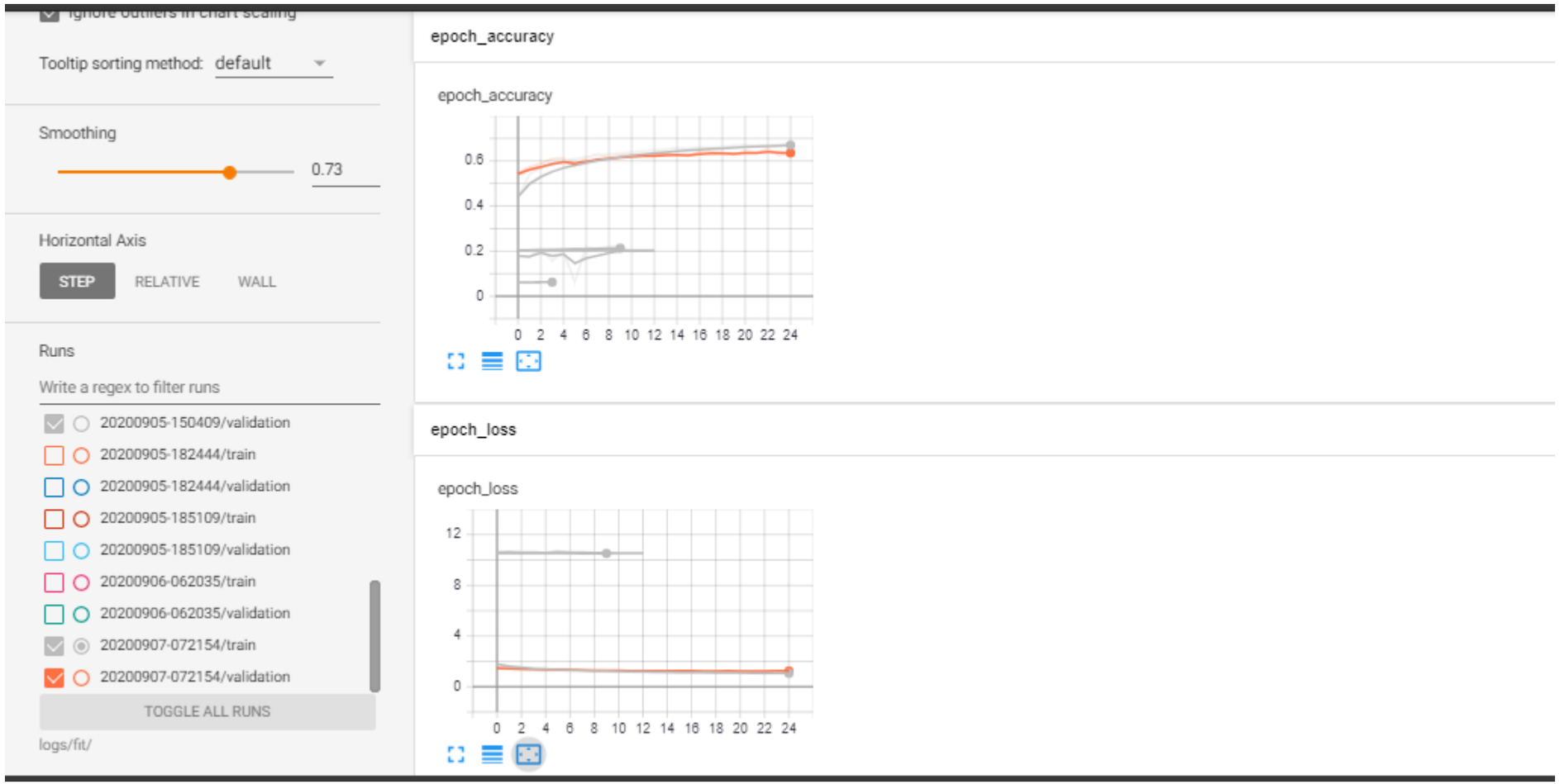
In [22]:
```
os.chdir('/gdrive/My Drive')
```

In [23]:
```
%tensorboard --logdir logs/fit/
```

```
Output hidden; open in https://colab.research.google.com (https://colab.research.google.com) to view.
```

In [28]:
```
#Model 1 - results
from IPython.display import Image
Image(filename='/gdrive/My Drive/Transfer_model2.PNG')
```

Out[28]:



Task - 3

In [23]:
```
# Create Model
os.environ['PYTHONHASHSEED'] = '0'

##https://keras.io/getting-started/faq/#how-can-i-obtain-reproducible-results-using-keras-during-development
## Have to clear the session. If you are not clearing, Graph will create again and again and graph size will increses.
## Varibles will also set to some value from before session
tf.keras.backend.clear_session()

## Set the random seed values to regenerate the model.
np.random.seed(0)
rn.seed(0)



#Get back the convolutional part of a VGG network trained on ImageNet
model_vgg16_conv = applications.VGG16(weights='imagenet', include_top=False,input_shape=(150,150,3))




# Freezing No trainable layer
for layer in model_vgg16_conv.layers:
    layer.trainable = False

for layer in model_vgg16_conv.layers[-6:]:
    layer.trainable = True

#model_vgg16_conv.summary()

#Input layer - Create your own input format (here 150,150,3)
input_layer = Input(shape=(150,150,3),name='Input_Layer')

#Use the generated model
output_vgg16_conv = model_vgg16_conv(input_layer)

#Conv Layer
Conv1 = Conv2D(filters=64,kernel_size=(4,4),strides=(1,1),padding='valid',data_format='channels_last',
             activation='relu',kernel_initializer=tf.keras.initializers.he_normal(seed=0),name='Conv1')(output_vgg16_conv)


Conv2 = Conv2D(filters=32,kernel_size=(1,1),strides=(1,1),padding='valid',data_format='channels_last',
             activation='relu',kernel_initializer=tf.keras.initializers.he_normal(seed=0),name='Conv2')(Conv1)
```

```
output_flat = Flatten(data_format='channels_last',name='Flatten')(Conv2)

#output layer
Out = Dense(units=16,activation='softmax',kernel_initializer=tf.keras.initializers.he_normal(seed=3),name='Output')(output_flat)


model = Model(inputs=input_layer,outputs=Out)


model.summary()
```

```
Model: "functional_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
Input_Layer (InputLayer)     [(None, 150, 150, 3)]     0
_________________________________________________________________
vgg16 (Functional)           (None, 4, 4, 512)         14714688
_________________________________________________________________
Conv1 (Conv2D)               (None, 1, 1, 64)          524352
_________________________________________________________________
Conv2 (Conv2D)               (None, 1, 1, 32)          2080
_________________________________________________________________
Flatten (Flatten)            (None, 32)                0
_________________________________________________________________
Output (Dense)               (None, 16)                528
=================================================================
Total params: 15,241,648
Trainable params: 9,966,192
Non-trainable params: 5,275,456
_________________________________________________________________
```

In [29]:
```
# eARLY sTOOPING
earlystop = EarlyStopping(monitor='val_loss', patience=2, verbose=1)


reduce_lr = ReduceLROnPlateau(monitor='val_loss', factor=0.2,
                              patience=3, min_lr=0.001)

##Callbacks
#file path, it saves the model in the 'model_save' folder and we are naming model with epoch number
#and val acc to differtiate with other models
#you have to create model_save folder before running the code.
filepath="model_save/weights-{epoch:02d}.hdf5"
checkpoint = ModelCheckpoint(filepath=filepath, monitor='val_loss',  verbose=1, save_best_only=True, mode='auto')
```

In [25]:
```
# TensorBoard Creation
%load_ext tensorboard
folder_name = datetime.datetime.now().strftime("%Y%m%d-%H%M%S")
```

In [26]:
```
# Create log folder - TensorBoard
log_dir="/gdrive/My Drive/logs/fit/" + folder_name
tensorboard_callback =TensorBoard(log_dir=log_dir,histogram_freq=0, write_graph=True)
```

In [27]:
```
folder_name
```

Out[27]:
```
'20200907-162234'
```

In [30]:
```
#compiling
model.compile(optimizer=tf.keras.optimizers.Adam(lr=0.001),loss='categorical_crossentropy',metrics=['accuracy'])
```

In [31]:
```
# test Sep 07

##fitting generator
model.fit(train_generator,steps_per_epoch=100,epochs=3,
            validation_data=validation_generator,
validation_steps=50,
callbacks=[reduce_lr,earlystop,tensorboard_callback])
```

```
Epoch 1/3
  1/100 [..............................] - ETA: 0s - loss: 3.2627 - accuracy: 0.0000e+00WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packag
es/tensorflow/python/ops/summary_ops_v2.py:1277: stop (from tensorflow.python.eager.profiler) is deprecated and will be removed after 2020-07-01.
Instructions for updating:
use `tf.profiler.experimental.stop` instead.
100/100 [==============================] - 985s 10s/step - loss: 2.8121 - accuracy: 0.0696 - val_loss: 2.7725 - val_accuracy: 0.0688
Epoch 2/3
100/100 [==============================] - 851s 9s/step - loss: 2.7749 - accuracy: 0.0609 - val_loss: 2.7725 - val_accuracy: 0.0644
Epoch 3/3
100/100 [==============================] - 800s 8s/step - loss: 2.7722 - accuracy: 0.0722 - val_loss: 2.7728 - val_accuracy: 0.0606
Epoch 00003: early stopping
```

Out[31]:
```
<tensorflow.python.keras.callbacks.History at 0x7f4b9847fe10>
```

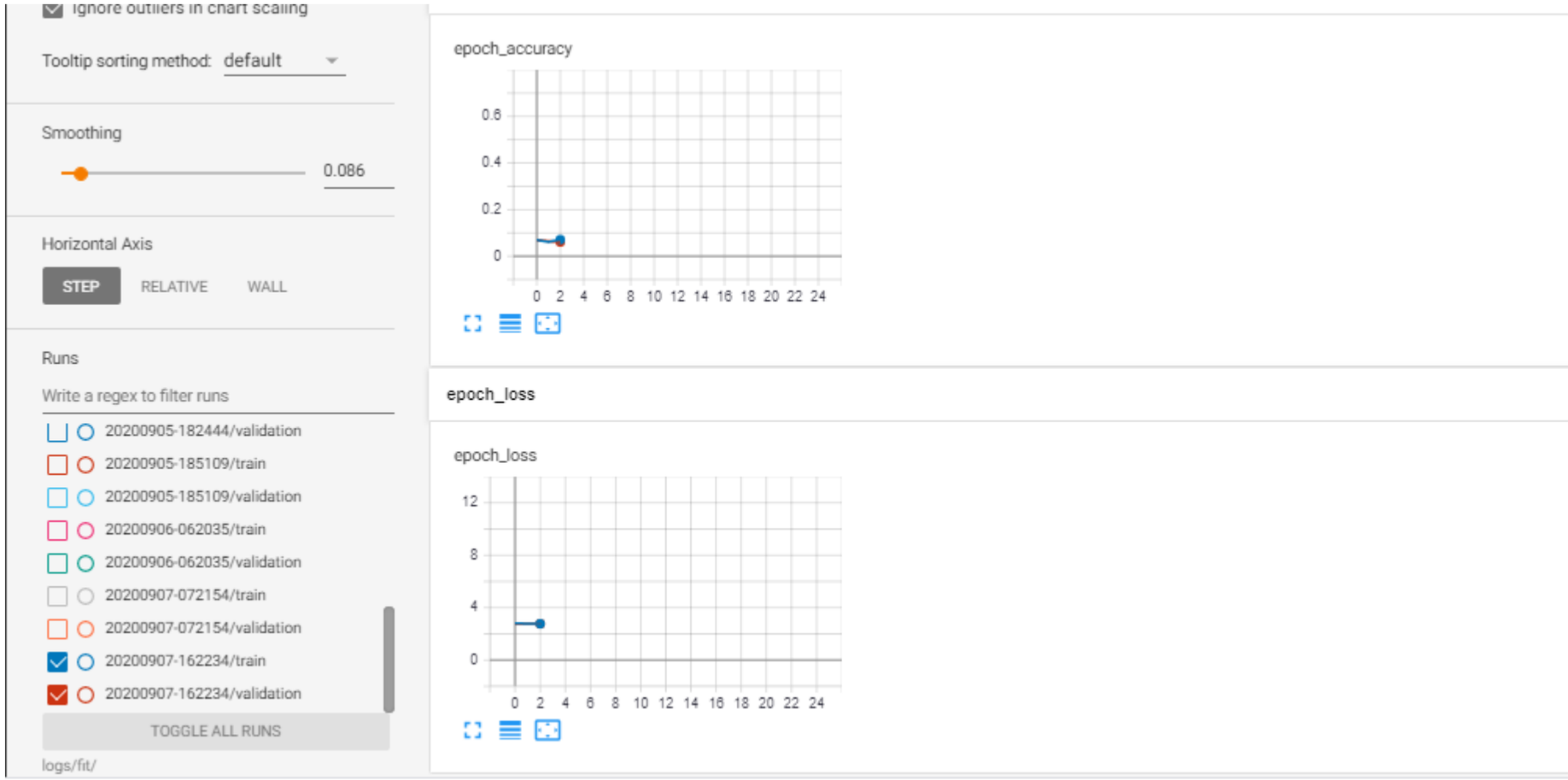
In [32]:
```
os.chdir('/gdrive/My Drive')
```

In [33]:
```
%tensorboard --logdir logs/fit/
```

```
Output hidden; open in https://colab.research.google.com (https://colab.research.google.com) to view.
```

In [35]:
```
#Model 1 - results
from IPython.display import Image
Image(filename='/gdrive/My Drive/Transfer_model3.PNG')
```

Out[35]:



In [ ]: