ML_Caltech101_single_VGG16-Data_Agumentation-SR

March 28, 2020

1 Final project Caltech101

– Network used VGG16

- **Pretrained VGG model (using weights of imageNet)**

Student Name/ID : Aloukik aditya(1115290) , Sarthak Rawat(1101124)

[1]:
```
from __future__ import absolute_import, division, print_function,␣
 ↪unicode_literals

import tensorflow as tf
import glob
from tensorflow.keras.preprocessing import image
from tensorflow.keras.models import Model
from tensorflow.keras.layers import Flatten, Dense,␣
 ↪Dropout,concatenate,Activation
import numpy as np
from PIL import Image
from os import listdir
import sys
import matplotlib.pyplot as plt
from tensorflow.keras import optimizers
from tensorflow.keras.callbacks import EarlyStopping
from tensorflow.keras.callbacks import CSVLogger
from sklearn.preprocessing import OneHotEncoder
import timeit
import gc
import random
from tensorflow.python.keras.callbacks import TensorBoard
from tensorflow.keras.applications.vgg16 import VGG16
from tensorflow.keras.regularizers import l1
from tensorflow.keras.callbacks import LearningRateScheduler
from time import time
from keras.preprocessing.image import ImageDataGenerator
print("Setup Done..")
```

```
Setup Done..
```

```
Using TensorFlow backend.
```

1.1 Loading test and train data

- we have split the data into 30 images of each category for Training
- And the rest for testing

[2]:
```
path = './101_ObjectCategories/'
random.seed(28)
img_size = 224
test_dict = {}
train_dict = {}

train_imgs = []
test_imgs = []
train_labels = []
test_labels = []

ctr=0

start = timeit.default_timer()
categories = listdir(path)
for category in categories:
    image_files_list = glob.glob(path+category+ '/*.jpg')
    train_list = random.sample(image_files_list, k=30)
    test_list = [x for x in image_files_list if x not in train_list]
    test_dict[category] = test_list
    train_dict[category] = train_list


for key in test_dict:
    for img_name in test_dict[key]:
        file =  img_name
        test_imgs.append(np.array(image.load_img(file, target_size=(img_size,␣
 ↪img_size))))
        test_labels.append(key)


for key in train_dict:
    for img_name in train_dict[key]:
        file =  img_name
        train_imgs.append(np.array(image.load_img(file, target_size=(img_size,␣
 ↪img_size))))
        train_labels.append(key)


stop = timeit.default_timer()
```

```
print("Loading dataset Done")
print('Time taken to load data : ', stop - start)
test_imgs = np.array(test_imgs)
train_imgs = np.array(train_imgs)
test_labels = np.array(test_labels)
train_labels = np.array(train_labels)
```

```
Loading dataset Done
Time taken to load data :  13.727304715999935
```

1.2 Checking data shape and size

- We can see Train shape is of 7710 images (257*30)
- And rest for testing

[3]:
```
gc.collect()
print("Shape of train data: ",train_imgs.shape)
print("Shape of test data: ",test_imgs.shape)
print("Shape of train labels: ",train_labels.shape)
print("Shape of test labels: ",test_labels.shape)
print("Total instances: ",train_labels.shape[0]+test_labels.shape[0])
```

```
Shape of train data:  (3060, 224, 224, 3)
Shape of test data:  (6084, 224, 224, 3)
Shape of train labels:  (3060,)
Shape of test labels:  (6084,)
Total instances:  9144
```

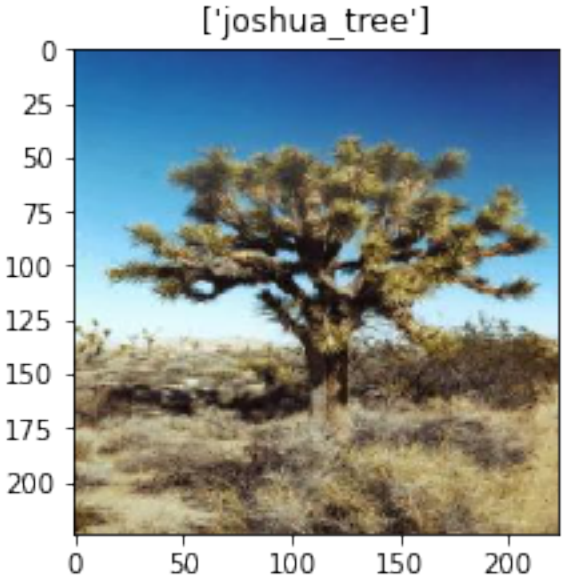
1.3 Flipping images for image augmentation

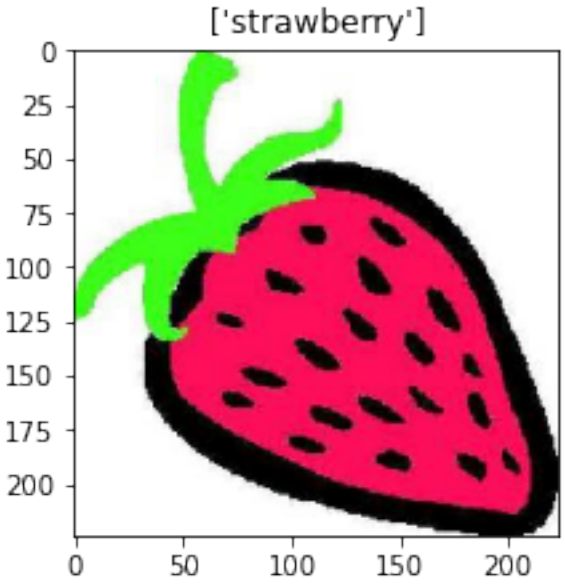
[4]:
```
train_imgs_flip_horizontal = np.flip(train_imgs,axis=2)

train_imgs = np.concatenate((train_imgs,train_imgs_flip_horizontal),axis=0)
train_labels = np.concatenate((train_labels,train_labels),axis=0)
```

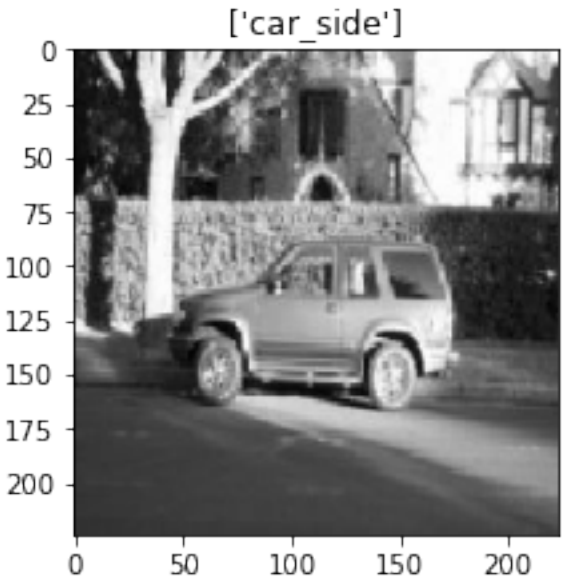
1.4 Plotting some random images

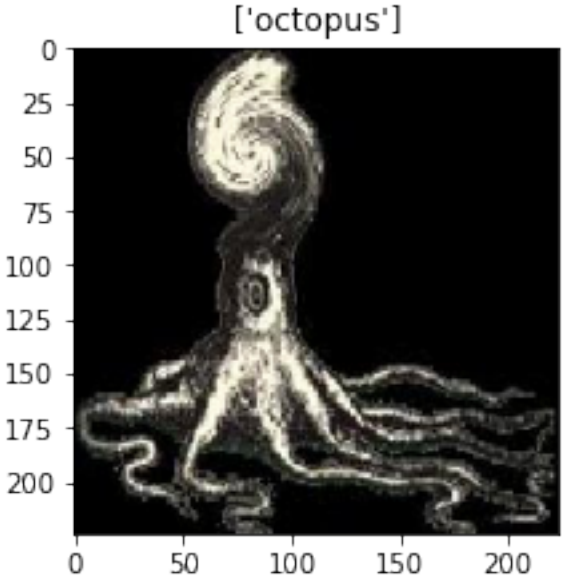
[5]:
```
val_index = []
random.seed(22)
for i in range(4):
    value = random.randint(0,3582)
    val_index.append(value)
    plt.figure( figsize=(15, 15))
    plt.subplot(1, 4, i+1)
    plt.title([train_labels[value]])
    plt.imshow(train_imgs[value])
```

```
# plt.tight_layout()
plt.show()
```









2 Preprocess image using VGG function image

[6]:
```
from tensorflow.keras.applications.vgg16 import preprocess_input
start = timeit.default_timer()
train_imgs = preprocess_input(train_imgs)
test_imgs = preprocess_input(test_imgs)
```

```
stop = timeit.default_timer()
print('Time taken to preprocess/normalize data : ', stop - start)
gc.collect()
```

```
Time taken to preprocess/normalize data :  5.590738644999988
```

[6]:
```
9803
```

[7]:
```
#check shape of  image
#number of train images are now doubled due to image augmentation(we have only␣
 ↪used flip image for augmentation)
print(train_imgs.shape)
test_imgs.shape
```

```
(6120, 224, 224, 3)
```

[7]:
```
(6084, 224, 224, 3)
```

2.1 Applying one hot encoding on our labels

[8]:
```
onehot_encoder = OneHotEncoder(sparse=False)

train_labels = train_labels.reshape(-1,1)
test_labels = test_labels.reshape(-1,1)

train_labels = onehot_encoder.fit_transform(train_labels)
test_labels = onehot_encoder.transform(test_labels)
```

[9]:
```
# test_labels1 = onehot_encoder.inverse_transform(test_labels)
```

[10]:
```
print("Shape of train labels: ",train_labels.shape)
print("Shape of test labels: ",test_labels.shape)
```

```
Shape of train labels:  (6120, 102)
Shape of test labels:  (6084, 102)
```

[11]:
```
# Function to free up keras memory

from keras.backend.tensorflow_backend import set_session
from keras.backend.tensorflow_backend import clear_session
from keras.backend.tensorflow_backend import get_session
import tensorflow

# Reset Keras Session
def reset_keras():
    sess = get_session()
    clear_session()
```

```
    sess.close()
    sess = get_session()

    try:
        del classifier # this is from global space - change this as you need
    except:
        pass

    print(gc.collect()) # if it's done something you should see a number being␣
↪outputted

    # use the same config as you used to create the session
    config = tensorflow.ConfigProto()
    config.gpu_options.per_process_gpu_memory_fraction = 1
    config.gpu_options.visible_device_list = "0"
    set_session(tensorflow.Session(config=config))
```

2.2 Using model from keras–> VGG16

- we have used pretrained weights for imagenet and removed the top fully connected layers.

[12]:
```
#define variable learning rates

def learning_rate_schedule(epoch):
    if epoch <= 3:
        return 1e-4 # 0.00001
    elif epoch <= 8:
        return 1e-5
    elif epoch <= 13:
        return 1e-6
    else:
        return 1e-7
    return LR
```

[13]:
```
#Initilize vgg net
vgg_net = VGG16(include_top=True, weights='imagenet', input_shape=(224,224,3))

#Define our model
x = vgg_net.get_layer('fc2').output
x = Dense(900, activity_regularizer=l1(0.001), name='my_fc3')(x)
x = Activation('relu')(x)
x = Dropout(0.5)(x)
x = Dense(102, activation='softmax', name='predictions')(x)
model_updated = Model(inputs=vgg_net.input, outputs=x)
```

```
## save our model weights
model_updated.save_weights('CALTECH101_VGG16_model_updated_initial.h5')
print("Model_Updated saved")

history_fc_list = []
history_full_list = []
```

```
WARNING:tensorflow:From /opt/conda/lib/python3.7/site-
packages/tensorflow_core/python/ops/resource_variable_ops.py:1630: calling
BaseResourceVariable.__init__ (from tensorflow.python.ops.resource_variable_ops)
with constraint is deprecated and will be removed in a future version.
Instructions for updating:
If using Keras pass *_constraint arguments to layers.
Model_Updated saved
```

[14]:
```
EPOCHS = 2
BATCH_SIZE = 128

train_generator = ImageDataGenerator(
                                        rotation_range = 20,
                                             horizontal_flip = True,
                                             zoom_range = .1 ,\
                                    fill_mode='nearest').flow(train_imgs,␣
 ↪train_labels, batch_size=BATCH_SIZE)
validation_generator = ImageDataGenerator().flow(test_imgs, test_labels,␣
 ↪batch_size=BATCH_SIZE)

validation_steps = test_imgs.shape[0]//BATCH_SIZE

steps_per_epoch = train_imgs.shape[0]//BATCH_SIZE
```

2.3 Run 1

[15]:
```
i=0
#Initilize vgg net
vgg_net = VGG16(include_top=True, weights='imagenet', input_shape=(224,224,3))

#Define our model
x = vgg_net.get_layer('fc2').output
x = Dense(900, activity_regularizer=l1(0.001), name='my_fc3')(x)
x = Activation('relu')(x)
x = Dropout(0.5)(x)
x = Dense(102, activation='softmax', name='predictions')(x)
model_updated = Model(inputs=vgg_net.input, outputs=x)

gc.collect()
```

```
model_updated.load_weights('CALTECH101_VGG16_model_updated_initial.h5')

print('------------------------New run started--------------------------------')
print('Initial model weights loaded , Started training run number ', i+1)
print('-----------------------------------------------------------------------')

# Make the last Fully connected layers trainable and freezing the rest of VGG␣
 ↪model
for layer in model_updated.layers:
    layer.trainable = False
for layer in model_updated.layers[-4:]:
    layer.trainable = True

tensorboard = TensorBoard(log_dir="logs\{}".format('CALTECH101\FC_Layer'+str(i)))

model_updated.compile(loss='categorical_crossentropy', optimizer=optimizers.
 ↪Adam(lr=0.0001), metrics=['accuracy'])


print('--------------------------Starting Fully connected layer␣
 ↪training-------------------------------------')

history_fc = model_updated.fit_generator(train_generator,␣
 ↪steps_per_epoch=steps_per_epoch, epochs=2,
                    validation_data=validation_generator,␣
 ↪validation_steps=validation_steps,
                    shuffle=True, callbacks=[tensorboard])


gc.collect()
##Unfreeze weights ------ Now full model will be trained
for layer in model_updated.layers:
    layer.trainable = True


no_epochs = 20
tensorboard = TensorBoard(log_dir="logs\{}".
 ↪format('CALTECH101\FULL_Model'+str(i)))

opt2 = optimizers.Adam(lr=0.00001)

### Variable learning rate ----------------------
lrate = LearningRateScheduler(learning_rate_schedule)
callbacks_list = [lrate,tensorboard]
```

```
model_updated.compile(loss='categorical_crossentropy', optimizer=opt2,␣
 ↪metrics=['accuracy'])


print('-------------------------Starting Full model training -->after␣
 ↪unfreez-------------------------------------')

## Training full model
history = model_updated.fit_generator(train_generator,␣
 ↪steps_per_epoch=steps_per_epoch, epochs=no_epochs,
                    validation_data=validation_generator,␣
 ↪validation_steps=validation_steps,
                    shuffle=True, callbacks=[tensorboard])



## Timer for checking time taken
stop = timeit.default_timer()
print('Time taken for one run is : ', stop - start)

model_updated.save_weights('CALTECH101_VGG16_model_updated_fin.h5')
print("Model_Updated weights saved")

## saving history for plots
history_fc_list.append(history_fc)
history_full_list.append(history)
```

```
-----------------------New run started---------------------------------
Initial model weights loaded , Started training run number  1
-----------------------------------------------------------------------
-------------------------Starting Fully connected layer
training-------------------------------------
Epoch 1/2
46/47 [============================>.] - ETA: 1s - loss: 4.7552 - acc:
0.2541Epoch 1/2
47/47 [==============================] - 14s 299ms/step - loss: 2.4395 - acc:
0.7392
47/47 [==============================] - 72s 2s/step - loss: 4.7206 - acc:
0.2600 - val_loss: 2.4395 - val_acc: 0.7392
Epoch 2/2
46/47 [============================>.] - ETA: 1s - loss: 2.4137 - acc:
0.6661Epoch 1/2
47/47 [==============================] - 14s 296ms/step - loss: 1.9018 - acc:
0.8153
```

```
47/47 [==============================] - 70s 1s/step - loss: 2.4118 - acc:
0.6664 - val_loss: 1.9018 - val_acc: 0.8153
-------------------------Starting Full model training -->after
unfreez-----------------------------------
Epoch 1/20
46/47 [============================>.] - ETA: 1s - loss: 1.6225 - acc:
0.8245Epoch 1/20
47/47 [==============================] - 14s 300ms/step - loss: 1.4356 - acc:
0.8725
47/47 [==============================] - 73s 2s/step - loss: 1.6179 - acc:
0.8248 - val_loss: 1.4356 - val_acc: 0.8725
Epoch 2/20
46/47 [============================>.] - ETA: 1s - loss: 1.1532 - acc:
0.9296Epoch 1/20
47/47 [==============================] - 14s 296ms/step - loss: 1.2630 - acc:
0.8833
47/47 [==============================] - 70s 1s/step - loss: 1.1515 - acc:
0.9296 - val_loss: 1.2630 - val_acc: 0.8833
Epoch 3/20
46/47 [============================>.] - ETA: 1s - loss: 0.9408 - acc:
0.9662Epoch 1/20
47/47 [==============================] - 14s 297ms/step - loss: 1.1567 - acc:
0.8988
47/47 [==============================] - 71s 2s/step - loss: 0.9410 - acc:
0.9654 - val_loss: 1.1567 - val_acc: 0.8988
Epoch 4/20
46/47 [============================>.] - ETA: 1s - loss: 0.8160 - acc:
0.9803Epoch 1/20
47/47 [==============================] - 14s 296ms/step - loss: 1.0776 - acc:
0.9028
47/47 [==============================] - 70s 1s/step - loss: 0.8149 - acc:
0.9806 - val_loss: 1.0776 - val_acc: 0.9028
Epoch 5/20
46/47 [============================>.] - ETA: 1s - loss: 0.7315 - acc:
0.9923Epoch 1/20
47/47 [==============================] - 14s 296ms/step - loss: 1.0310 - acc:
0.9094
47/47 [==============================] - 69s 1s/step - loss: 0.7302 - acc:
0.9922 - val_loss: 1.0310 - val_acc: 0.9094
Epoch 6/20
46/47 [============================>.] - ETA: 1s - loss: 0.6662 - acc:
0.9939Epoch 1/20
47/47 [==============================] - 14s 297ms/step - loss: 0.9918 - acc:
0.9109
47/47 [==============================] - 69s 1s/step - loss: 0.6660 - acc:
0.9938 - val_loss: 0.9918 - val_acc: 0.9109
Epoch 7/20
46/47 [============================>.] - ETA: 1s - loss: 0.6174 - acc:
```

```
0.9969Epoch 1/20
47/47 [==============================] - 14s 296ms/step - loss: 0.9635 - acc:
0.9104
47/47 [==============================] - 69s 1s/step - loss: 0.6170 - acc:
0.9968 - val_loss: 0.9635 - val_acc: 0.9104
Epoch 8/20
46/47 [============================>.] - ETA: 1s - loss: 0.5785 - acc:
0.9986Epoch 1/20
47/47 [==============================] - 14s 296ms/step - loss: 0.9331 - acc:
0.9139
47/47 [==============================] - 69s 1s/step - loss: 0.5780 - acc:
0.9987 - val_loss: 0.9331 - val_acc: 0.9139
Epoch 9/20
46/47 [============================>.] - ETA: 1s - loss: 0.5428 - acc:
0.9992Epoch 1/20
47/47 [==============================] - 14s 296ms/step - loss: 0.9054 - acc:
0.9161
47/47 [==============================] - 69s 1s/step - loss: 0.5425 - acc:
0.9992 - val_loss: 0.9054 - val_acc: 0.9161
Epoch 10/20
46/47 [============================>.] - ETA: 1s - loss: 0.5184 - acc:
0.9997Epoch 1/20
47/47 [==============================] - 14s 296ms/step - loss: 0.8931 - acc:
0.9151
47/47 [==============================] - 68s 1s/step - loss: 0.5182 - acc:
0.9995 - val_loss: 0.8931 - val_acc: 0.9151
Epoch 11/20
46/47 [============================>.] - ETA: 1s - loss: 0.4922 - acc:
0.9997Epoch 1/20
47/47 [==============================] - 14s 297ms/step - loss: 0.8753 - acc:
0.9147
47/47 [==============================] - 68s 1s/step - loss: 0.4919 - acc:
0.9997 - val_loss: 0.8753 - val_acc: 0.9147
Epoch 12/20
46/47 [============================>.] - ETA: 1s - loss: 0.4705 - acc:
1.0000Epoch 1/20
47/47 [==============================] - 14s 297ms/step - loss: 0.8513 - acc:
0.9184
47/47 [==============================] - 67s 1s/step - loss: 0.4705 - acc:
1.0000 - val_loss: 0.8513 - val_acc: 0.9184
Epoch 13/20
46/47 [============================>.] - ETA: 1s - loss: 0.4520 - acc:
0.9998Epoch 1/20
47/47 [==============================] - 14s 297ms/step - loss: 0.8345 - acc:
0.9192
47/47 [==============================] - 66s 1s/step - loss: 0.4518 - acc:
0.9998 - val_loss: 0.8345 - val_acc: 0.9192
Epoch 14/20
```

```
46/47 [============================>.] - ETA: 1s - loss: 0.4354 - acc:
1.0000Epoch 1/20
47/47 [==============================] - 14s 297ms/step - loss: 0.8216 - acc:
0.9194
47/47 [==============================] - 66s 1s/step - loss: 0.4354 - acc:
1.0000 - val_loss: 0.8216 - val_acc: 0.9194
Epoch 15/20
46/47 [============================>.] - ETA: 1s - loss: 0.4211 - acc:
0.9997Epoch 1/20
47/47 [==============================] - 14s 296ms/step - loss: 0.8146 - acc:
0.9199
47/47 [==============================] - 65s 1s/step - loss: 0.4211 - acc:
0.9997 - val_loss: 0.8146 - val_acc: 0.9199
Epoch 16/20
46/47 [============================>.] - ETA: 1s - loss: 0.4089 - acc:
1.0000Epoch 1/20
47/47 [==============================] - 14s 297ms/step - loss: 0.8006 - acc:
0.9202
47/47 [==============================] - 66s 1s/step - loss: 0.4087 - acc:
1.0000 - val_loss: 0.8006 - val_acc: 0.9202
Epoch 17/20
46/47 [============================>.] - ETA: 1s - loss: 0.3942 - acc:
0.9998Epoch 1/20
47/47 [==============================] - 14s 297ms/step - loss: 0.7929 - acc:
0.9212
47/47 [==============================] - 65s 1s/step - loss: 0.3941 - acc:
0.9998 - val_loss: 0.7929 - val_acc: 0.9212
Epoch 18/20
46/47 [============================>.] - ETA: 1s - loss: 0.3827 - acc:
1.0000Epoch 1/20
47/47 [==============================] - 14s 297ms/step - loss: 0.7769 - acc:
0.9220
47/47 [==============================] - 64s 1s/step - loss: 0.3827 - acc:
1.0000 - val_loss: 0.7769 - val_acc: 0.9220
Epoch 19/20
46/47 [============================>.] - ETA: 1s - loss: 0.3726 - acc:
1.0000Epoch 1/20
47/47 [==============================] - 14s 296ms/step - loss: 0.7691 - acc:
0.9225
47/47 [==============================] - 65s 1s/step - loss: 0.3725 - acc:
1.0000 - val_loss: 0.7691 - val_acc: 0.9225
Epoch 20/20
46/47 [============================>.] - ETA: 1s - loss: 0.3634 - acc:
1.0000Epoch 1/20
47/47 [==============================] - 14s 297ms/step - loss: 0.7655 - acc:
0.9199
47/47 [==============================] - 63s 1s/step - loss: 0.3634 - acc:
1.0000 - val_loss: 0.7655 - val_acc: 0.9199
```

```
Time taken for one run is :  1513.413336813
Model_Updated weights saved
```

2.4 Run 2

[16]:
```
i=1
#Initilize vgg net
vgg_net = VGG16(include_top=True, weights='imagenet', input_shape=(224,224,3))

#Define our model
x = vgg_net.get_layer('fc2').output
x = Dense(900, activity_regularizer=l1(0.001), name='my_fc3')(x)
x = Activation('relu')(x)
x = Dropout(0.5)(x)
x = Dense(102, activation='softmax', name='predictions')(x)
model_updated = Model(inputs=vgg_net.input, outputs=x)

gc.collect()


model_updated.load_weights('CALTECH101_VGG16_model_updated_initial.h5')

print('------------------------New run started--------------------------------')
print('Initial model weights loaded , Started training run number ', i+1)
print('-----------------------------------------------------------------------')

# Make the last Fully connected layers trainable and freezing the rest of VGG␣
 ↪model
for layer in model_updated.layers:
    layer.trainable = False
for layer in model_updated.layers[-4:]:
    layer.trainable = True

tensorboard = TensorBoard(log_dir="logs\{}".format('CALTECH101\FC_Layer'+str(i)))

model_updated.compile(loss='categorical_crossentropy', optimizer=optimizers.
 ↪Adam(lr=0.0001), metrics=['accuracy'])


print('-------------------------Starting Fully connected layer␣
 ↪training-------------------------------------')

history_fc = model_updated.fit_generator(train_generator,␣
 ↪steps_per_epoch=steps_per_epoch, epochs=2,
                    validation_data=validation_generator,␣
 ↪validation_steps=validation_steps,
                    shuffle=True, callbacks=[tensorboard])
```

```
gc.collect()
##Unfreeze weights ------ Now full model will be trained
for layer in model_updated.layers:
    layer.trainable = True


no_epochs = 20
tensorboard = TensorBoard(log_dir="logs\{}".
 ↪format('CALTECH101\FULL_Model'+str(i)))

opt2 = optimizers.Adam(lr=0.00001)

### Variable learning rate -----------------------
lrate = LearningRateScheduler(learning_rate_schedule)
callbacks_list = [lrate,tensorboard]




model_updated.compile(loss='categorical_crossentropy', optimizer=opt2,␣
 ↪metrics=['accuracy'])


print('--------------------------Starting Full model training -->after␣
 ↪unfreez-----------------------------------')

## Training full model
history = model_updated.fit_generator(train_generator,␣
 ↪steps_per_epoch=steps_per_epoch, epochs=no_epochs,
                    validation_data=validation_generator,␣
 ↪validation_steps=validation_steps,
                    shuffle=True, callbacks=[tensorboard])



## Timer for checking time taken
stop = timeit.default_timer()
print('Time taken for one run is : ', stop - start)

model_updated.save_weights('CALTECH101_VGG16_model_updated_fin.h5')
print("Model_Updated weights saved")

## saving history for plots
history_fc_list.append(history_fc)
history_full_list.append(history)
```

```
-----------------------New run started---------------------------------
Initial model weights loaded , Started training run number  2
-----------------------------------------------------------------------
-------------------------Starting Fully connected layer
training-----------------------------------
Epoch 1/2
46/47 [============================>.] - ETA: 1s - loss: 4.7248 - acc:
0.2527Epoch 1/2
47/47 [==============================] - 16s 347ms/step - loss: 2.4536 - acc:
0.7443
47/47 [==============================] - 74s 2s/step - loss: 4.6954 - acc:
0.2573 - val_loss: 2.4536 - val_acc: 0.7443
Epoch 2/2
46/47 [============================>.] - ETA: 1s - loss: 2.4028 - acc:
0.6673Epoch 1/2
47/47 [==============================] - 16s 344ms/step - loss: 1.8553 - acc:
0.8243
47/47 [==============================] - 73s 2s/step - loss: 2.3972 - acc:
0.6687 - val_loss: 1.8553 - val_acc: 0.8243
-------------------------Starting Full model training -->after
unfreez-----------------------------------
Epoch 1/20
46/47 [============================>.] - ETA: 1s - loss: 1.6225 - acc:
0.8204Epoch 1/20
47/47 [==============================] - 16s 349ms/step - loss: 1.4482 - acc:
0.8564
47/47 [==============================] - 76s 2s/step - loss: 1.6159 - acc:
0.8216 - val_loss: 1.4482 - val_acc: 0.8564
Epoch 2/20
46/47 [============================>.] - ETA: 1s - loss: 1.1511 - acc:
0.9313Epoch 1/20
47/47 [==============================] - 16s 344ms/step - loss: 1.2658 - acc:
0.8855
47/47 [==============================] - 74s 2s/step - loss: 1.1476 - acc:
0.9319 - val_loss: 1.2658 - val_acc: 0.8855
Epoch 3/20
46/47 [============================>.] - ETA: 1s - loss: 0.9334 - acc:
0.9693Epoch 1/20
47/47 [==============================] - 16s 344ms/step - loss: 1.1400 - acc:
0.8988
47/47 [==============================] - 74s 2s/step - loss: 0.9318 - acc:
0.9696 - val_loss: 1.1400 - val_acc: 0.8988
Epoch 4/20
46/47 [============================>.] - ETA: 1s - loss: 0.8177 - acc:
0.9831Epoch 1/20
47/47 [==============================] - 16s 344ms/step - loss: 1.0794 - acc:
0.9033
47/47 [==============================] - 74s 2s/step - loss: 0.8159 - acc:
```

```
0.9833 - val_loss: 1.0794 - val_acc: 0.9033
Epoch 5/20
46/47 [============================>.] - ETA: 1s - loss: 0.7310 - acc:
0.9906Epoch 1/20
47/47 [==============================] - 16s 343ms/step - loss: 1.0298 - acc:
0.9082
47/47 [==============================] - 74s 2s/step - loss: 0.7308 - acc:
0.9905 - val_loss: 1.0298 - val_acc: 0.9082
Epoch 6/20
46/47 [============================>.] - ETA: 1s - loss: 0.6677 - acc:
0.9932Epoch 1/20
47/47 [==============================] - 16s 344ms/step - loss: 0.9854 - acc:
0.9117
47/47 [==============================] - 73s 2s/step - loss: 0.6677 - acc:
0.9930 - val_loss: 0.9854 - val_acc: 0.9117
Epoch 7/20
46/47 [============================>.] - ETA: 1s - loss: 0.6146 - acc:
0.9971Epoch 1/20
47/47 [==============================] - 16s 344ms/step - loss: 0.9508 - acc:
0.9152
47/47 [==============================] - 74s 2s/step - loss: 0.6151 - acc:
0.9972 - val_loss: 0.9508 - val_acc: 0.9152
Epoch 8/20
46/47 [============================>.] - ETA: 1s - loss: 0.5764 - acc:
0.9981Epoch 1/20
47/47 [==============================] - 16s 345ms/step - loss: 0.9336 - acc:
0.9142
47/47 [==============================] - 73s 2s/step - loss: 0.5762 - acc:
0.9982 - val_loss: 0.9336 - val_acc: 0.9142
Epoch 9/20
46/47 [============================>.] - ETA: 1s - loss: 0.5430 - acc:
0.9990Epoch 1/20
47/47 [==============================] - 16s 345ms/step - loss: 0.9049 - acc:
0.9179
47/47 [==============================] - 73s 2s/step - loss: 0.5427 - acc:
0.9988 - val_loss: 0.9049 - val_acc: 0.9179
Epoch 10/20
46/47 [============================>.] - ETA: 1s - loss: 0.5150 - acc:
0.9995Epoch 1/20
47/47 [==============================] - 16s 345ms/step - loss: 0.8870 - acc:
0.9184
47/47 [==============================] - 73s 2s/step - loss: 0.5150 - acc:
0.9995 - val_loss: 0.8870 - val_acc: 0.9184
Epoch 11/20
46/47 [============================>.] - ETA: 1s - loss: 0.4911 - acc:
0.9997Epoch 1/20
47/47 [==============================] - 16s 346ms/step - loss: 0.8710 - acc:
0.9177
```

```
47/47 [==============================] - 73s 2s/step - loss: 0.4909 - acc:
0.9997 - val_loss: 0.8710 - val_acc: 0.9177
Epoch 12/20
46/47 [============================>.] - ETA: 1s - loss: 0.4728 - acc:
0.9993Epoch 1/20
47/47 [==============================] - 16s 345ms/step - loss: 0.8551 - acc:
0.9195
47/47 [==============================] - 72s 2s/step - loss: 0.4726 - acc:
0.9993 - val_loss: 0.8551 - val_acc: 0.9195
Epoch 13/20
46/47 [============================>.] - ETA: 1s - loss: 0.4517 - acc:
0.9998Epoch 1/20
47/47 [==============================] - 16s 345ms/step - loss: 0.8383 - acc:
0.9182
47/47 [==============================] - 72s 2s/step - loss: 0.4516 - acc:
0.9998 - val_loss: 0.8383 - val_acc: 0.9182
Epoch 14/20
46/47 [============================>.] - ETA: 1s - loss: 0.4365 - acc:
0.9997Epoch 1/20
47/47 [==============================] - 16s 345ms/step - loss: 0.8227 - acc:
0.9212
47/47 [==============================] - 72s 2s/step - loss: 0.4365 - acc:
0.9997 - val_loss: 0.8227 - val_acc: 0.9212
Epoch 15/20
46/47 [============================>.] - ETA: 1s - loss: 0.4226 - acc:
1.0000Epoch 1/20
47/47 [==============================] - 16s 345ms/step - loss: 0.8171 - acc:
0.9195
47/47 [==============================] - 72s 2s/step - loss: 0.4223 - acc:
1.0000 - val_loss: 0.8171 - val_acc: 0.9195
Epoch 16/20
46/47 [============================>.] - ETA: 1s - loss: 0.4057 - acc:
0.9998Epoch 1/20
47/47 [==============================] - 16s 345ms/step - loss: 0.7990 - acc:
0.9200
47/47 [==============================] - 71s 2s/step - loss: 0.4055 - acc:
0.9998 - val_loss: 0.7990 - val_acc: 0.9200
Epoch 17/20
46/47 [============================>.] - ETA: 1s - loss: 0.3935 - acc:
1.0000Epoch 1/20
47/47 [==============================] - 16s 346ms/step - loss: 0.7945 - acc:
0.9212
47/47 [==============================] - 71s 2s/step - loss: 0.3933 - acc:
1.0000 - val_loss: 0.7945 - val_acc: 0.9212
Epoch 18/20
46/47 [============================>.] - ETA: 1s - loss: 0.3823 - acc:
1.0000Epoch 1/20
47/47 [==============================] - 16s 345ms/step - loss: 0.7834 - acc:
```

```
0.9214
47/47 [==============================] - 71s 2s/step - loss: 0.3824 - acc:
1.0000 - val_loss: 0.7834 - val_acc: 0.9214
Epoch 19/20
46/47 [============================>.] - ETA: 1s - loss: 0.3723 - acc:
1.0000Epoch 1/20
47/47 [==============================] - 16s 345ms/step - loss: 0.7722 - acc:
0.9214
47/47 [==============================] - 71s 2s/step - loss: 0.3723 - acc:
1.0000 - val_loss: 0.7722 - val_acc: 0.9214
Epoch 20/20
46/47 [============================>.] - ETA: 1s - loss: 0.3630 - acc:
1.0000Epoch 1/20
47/47 [==============================] - 16s 346ms/step - loss: 0.7663 - acc:
0.9204
47/47 [==============================] - 71s 2s/step - loss: 0.3628 - acc:
1.0000 - val_loss: 0.7663 - val_acc: 0.9204
Time taken for one run is :  3122.7724372909997
Model_Updated weights saved
```

2.5 Run 3

[17]:
```
reset_keras()
i=2
#Initilize vgg net
vgg_net = VGG16(include_top=True, weights='imagenet', input_shape=(224,224,3))

#Define our model
x = vgg_net.get_layer('fc2').output
x = Dense(900, activity_regularizer=l1(0.001), name='my_fc3')(x)
x = Activation('relu')(x)
x = Dropout(0.5)(x)
x = Dense(102, activation='softmax', name='predictions')(x)
model_updated = Model(inputs=vgg_net.input, outputs=x)

gc.collect()


model_updated.load_weights('CALTECH101_VGG16_model_updated_initial.h5')

print('------------------------New run started--------------------------------')
print('Initial model weights loaded , Started training run number ', i+1)
print('-----------------------------------------------------------------------')

# Make the last Fully connected layers trainable and freezing the rest of VGG␣
 ↪model
for layer in model_updated.layers:
```

```
    layer.trainable = False
for layer in model_updated.layers[-4:]:
    layer.trainable = True

tensorboard = TensorBoard(log_dir="logs\{}".format('CALTECH101\FC_Layer'+str(i)))

model_updated.compile(loss='categorical_crossentropy', optimizer=optimizers.
 ↪Adam(lr=0.0001), metrics=['accuracy'])


print('-------------------------Starting Fully connected layer␣
 ↪training-----------------------------------')

history_fc = model_updated.fit_generator(train_generator,␣
 ↪steps_per_epoch=steps_per_epoch, epochs=2,
                    validation_data=validation_generator,␣
 ↪validation_steps=validation_steps,
                    shuffle=True, callbacks=[tensorboard])


gc.collect()
##Unfreeze weights ------ Now full model will be trained
for layer in model_updated.layers:
    layer.trainable = True


no_epochs = 20
tensorboard = TensorBoard(log_dir="logs\{}".
 ↪format('CALTECH101\FULL_Model'+str(i)))

opt2 = optimizers.Adam(lr=0.00001)

### Variable learning rate ----------------------
lrate = LearningRateScheduler(learning_rate_schedule)
callbacks_list = [lrate,tensorboard]




model_updated.compile(loss='categorical_crossentropy', optimizer=opt2,␣
 ↪metrics=['accuracy'])


print('-------------------------Starting Full model training -->after␣
 ↪unfreez-------------------------------------')
```

```
## Training full model
history = model_updated.fit_generator(train_generator,␣
 ↪steps_per_epoch=steps_per_epoch, epochs=no_epochs,
                    validation_data=validation_generator,␣
 ↪validation_steps=validation_steps,
                    shuffle=True, callbacks=[tensorboard])



## Timer for checking time taken
stop = timeit.default_timer()
print('Time taken for one run is : ', stop - start)

model_updated.save_weights('CALTECH101_VGG16_model_updated_fin.h5')
print("Model_Updated weights saved")

## saving history for plots
history_fc_list.append(history_fc)
history_full_list.append(history)
```

```
14
------------------------New run started----------------------------------
Initial model weights loaded , Started training run number  3
------------------------------------------------------------------------
-------------------------Starting Fully connected layer
training-------------------------------------
Epoch 1/2
46/47 [============================>.] - ETA: 1s - loss: 4.6901 - acc:
0.2653Epoch 1/2
47/47 [==============================] - 16s 347ms/step - loss: 2.4197 - acc:
0.7480
47/47 [==============================] - 74s 2s/step - loss: 4.6567 - acc:
0.2712 - val_loss: 2.4197 - val_acc: 0.7480
Epoch 2/2
46/47 [============================>.] - ETA: 1s - loss: 2.3918 - acc:
0.6755Epoch 1/2
47/47 [==============================] - 16s 344ms/step - loss: 1.9146 - acc:
0.8075
47/47 [==============================] - 73s 2s/step - loss: 2.3835 - acc:
0.6769 - val_loss: 1.9146 - val_acc: 0.8075
-------------------------Starting Full model training -->after
unfreez-----------------------------------
Epoch 1/20
46/47 [============================>.] - ETA: 1s - loss: 1.6196 - acc:
0.8242Epoch 1/20
47/47 [==============================] - 16s 347ms/step - loss: 1.4294 - acc:
0.8669
```

```
47/47 [==============================] - 76s 2s/step - loss: 1.6133 - acc:
0.8261 - val_loss: 1.4294 - val_acc: 0.8669
Epoch 2/20
46/47 [============================>.] - ETA: 1s - loss: 1.1544 - acc:
0.9267Epoch 1/20
47/47 [==============================] - 16s 344ms/step - loss: 1.2519 - acc:
0.8860
47/47 [==============================] - 74s 2s/step - loss: 1.1517 - acc:
0.9271 - val_loss: 1.2519 - val_acc: 0.8860
Epoch 3/20
46/47 [============================>.] - ETA: 1s - loss: 0.9419 - acc:
0.9659Epoch 1/20
47/47 [==============================] - 16s 346ms/step - loss: 1.1690 - acc:
0.8926
47/47 [==============================] - 74s 2s/step - loss: 0.9415 - acc:
0.9656 - val_loss: 1.1690 - val_acc: 0.8926
Epoch 4/20
46/47 [============================>.] - ETA: 1s - loss: 0.8180 - acc:
0.9828Epoch 1/20
47/47 [==============================] - 16s 345ms/step - loss: 1.0960 - acc:
0.8981
47/47 [==============================] - 74s 2s/step - loss: 0.8170 - acc:
0.9829 - val_loss: 1.0960 - val_acc: 0.8981
Epoch 5/20
46/47 [============================>.] - ETA: 1s - loss: 0.7301 - acc:
0.9909Epoch 1/20
47/47 [==============================] - 16s 345ms/step - loss: 1.0365 - acc:
0.9059
47/47 [==============================] - 73s 2s/step - loss: 0.7298 - acc:
0.9906 - val_loss: 1.0365 - val_acc: 0.9059
Epoch 6/20
46/47 [============================>.] - ETA: 1s - loss: 0.6603 - acc:
0.9952Epoch 1/20
47/47 [==============================] - 16s 346ms/step - loss: 0.9894 - acc:
0.9077
47/47 [==============================] - 73s 2s/step - loss: 0.6602 - acc:
0.9952 - val_loss: 0.9894 - val_acc: 0.9077
Epoch 7/20
46/47 [============================>.] - ETA: 1s - loss: 0.6185 - acc:
0.9962Epoch 1/20
47/47 [==============================] - 16s 345ms/step - loss: 0.9504 - acc:
0.9127
47/47 [==============================] - 72s 2s/step - loss: 0.6192 - acc:
0.9960 - val_loss: 0.9504 - val_acc: 0.9127
Epoch 8/20
46/47 [============================>.] - ETA: 1s - loss: 0.5739 - acc:
0.9986Epoch 1/20
47/47 [==============================] - 16s 345ms/step - loss: 0.9324 - acc:
```

```
0.9146
47/47 [==============================] - 72s 2s/step - loss: 0.5736 - acc:
0.9987 - val_loss: 0.9324 - val_acc: 0.9146
Epoch 9/20
46/47 [============================>.] - ETA: 1s - loss: 0.5471 - acc:
0.9986Epoch 1/20
47/47 [==============================] - 16s 345ms/step - loss: 0.9031 - acc:
0.9171
47/47 [==============================] - 71s 2s/step - loss: 0.5468 - acc:
0.9987 - val_loss: 0.9031 - val_acc: 0.9171
Epoch 10/20
46/47 [============================>.] - ETA: 1s - loss: 0.5149 - acc:
0.9991Epoch 1/20
47/47 [==============================] - 16s 346ms/step - loss: 0.8899 - acc:
0.9171
47/47 [==============================] - 71s 2s/step - loss: 0.5143 - acc:
0.9992 - val_loss: 0.8899 - val_acc: 0.9171
Epoch 11/20
46/47 [============================>.] - ETA: 1s - loss: 0.4921 - acc:
0.9997Epoch 1/20
47/47 [==============================] - 16s 344ms/step - loss: 0.8661 - acc:
0.9186
47/47 [==============================] - 71s 2s/step - loss: 0.4917 - acc:
0.9997 - val_loss: 0.8661 - val_acc: 0.9186
Epoch 12/20
46/47 [============================>.] - ETA: 1s - loss: 0.4717 - acc:
0.9998Epoch 1/20
47/47 [==============================] - 16s 346ms/step - loss: 0.8503 - acc:
0.9204
47/47 [==============================] - 70s 1s/step - loss: 0.4714 - acc:
0.9998 - val_loss: 0.8503 - val_acc: 0.9204
Epoch 13/20
46/47 [============================>.] - ETA: 1s - loss: 0.4506 - acc:
1.0000Epoch 1/20
47/47 [==============================] - 16s 344ms/step - loss: 0.8386 - acc:
0.9182
47/47 [==============================] - 70s 1s/step - loss: 0.4508 - acc:
1.0000 - val_loss: 0.8386 - val_acc: 0.9182
Epoch 14/20
46/47 [============================>.] - ETA: 1s - loss: 0.4364 - acc:
0.9997Epoch 1/20
47/47 [==============================] - 16s 345ms/step - loss: 0.8278 - acc:
0.9192
47/47 [==============================] - 70s 1s/step - loss: 0.4363 - acc:
0.9997 - val_loss: 0.8278 - val_acc: 0.9192
Epoch 15/20
46/47 [============================>.] - ETA: 1s - loss: 0.4212 - acc:
1.0000Epoch 1/20
```

```
47/47 [==============================] - 16s 345ms/step - loss: 0.8106 - acc:
0.9229
47/47 [==============================] - 69s 1s/step - loss: 0.4210 - acc:
1.0000 - val_loss: 0.8106 - val_acc: 0.9229
Epoch 16/20
46/47 [============================>.] - ETA: 1s - loss: 0.4071 - acc:
1.0000Epoch 1/20
47/47 [==============================] - 16s 345ms/step - loss: 0.8035 - acc:
0.9215
47/47 [==============================] - 70s 1s/step - loss: 0.4070 - acc:
1.0000 - val_loss: 0.8035 - val_acc: 0.9215
Epoch 17/20
46/47 [============================>.] - ETA: 1s - loss: 0.3954 - acc:
1.0000Epoch 1/20
47/47 [==============================] - 16s 345ms/step - loss: 0.7894 - acc:
0.9220
47/47 [==============================] - 69s 1s/step - loss: 0.3950 - acc:
1.0000 - val_loss: 0.7894 - val_acc: 0.9220
Epoch 18/20
46/47 [============================>.] - ETA: 1s - loss: 0.3844 - acc:
1.0000Epoch 1/20
47/47 [==============================] - 16s 345ms/step - loss: 0.7968 - acc:
0.9202
47/47 [==============================] - 69s 1s/step - loss: 0.3842 - acc:
1.0000 - val_loss: 0.7968 - val_acc: 0.9202
Epoch 19/20
46/47 [============================>.] - ETA: 1s - loss: 0.3732 - acc:
1.0000Epoch 1/20
47/47 [==============================] - 16s 346ms/step - loss: 0.7808 - acc:
0.9224
47/47 [==============================] - 69s 1s/step - loss: 0.3730 - acc:
1.0000 - val_loss: 0.7808 - val_acc: 0.9224
Epoch 20/20
46/47 [============================>.] - ETA: 1s - loss: 0.3629 - acc:
1.0000Epoch 1/20
47/47 [==============================] - 16s 345ms/step - loss: 0.7730 - acc:
0.9229
47/47 [==============================] - 69s 1s/step - loss: 0.3628 - acc:
1.0000 - val_loss: 0.7730 - val_acc: 0.9229
Time taken for one run is :  4703.706491549
Model_Updated weights saved
```

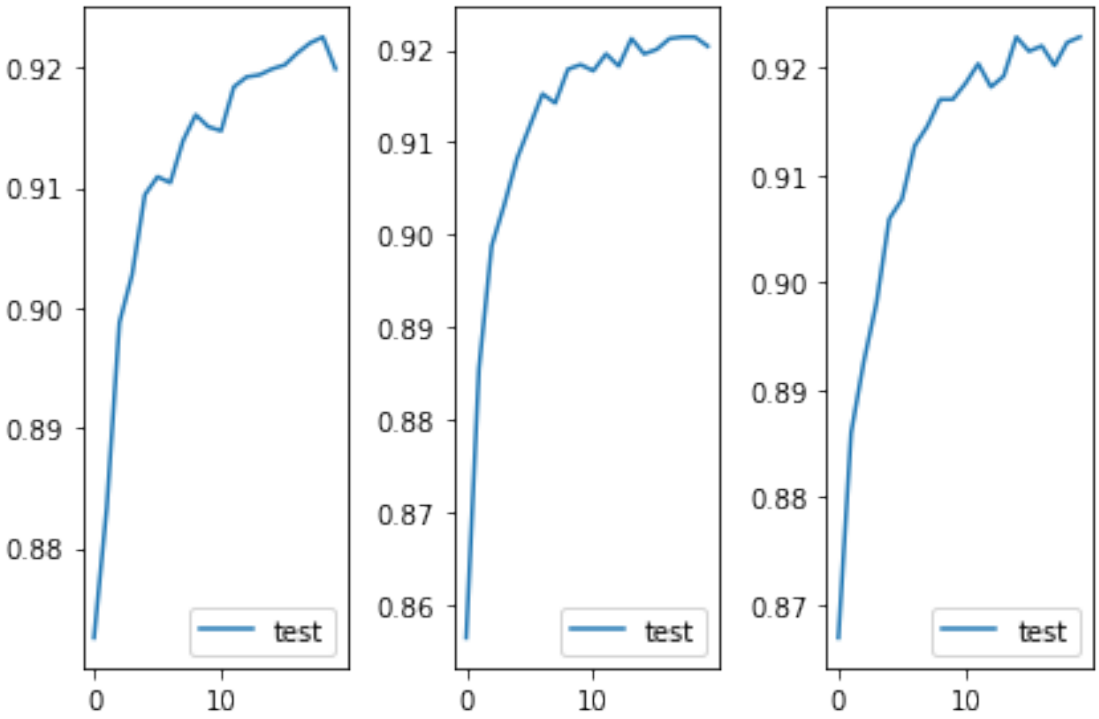
2.6 Tensorboard logs have been save to logs folder

- where "*_Layer" folder means logs where we train only our FC layers
- where "*_Model" folder means logs where we train all layers in our model

2.7 Plotting the graphs of testing accuracy

- Test accuracy
- Plotted for only FC_layer training
- Plotted for full model training

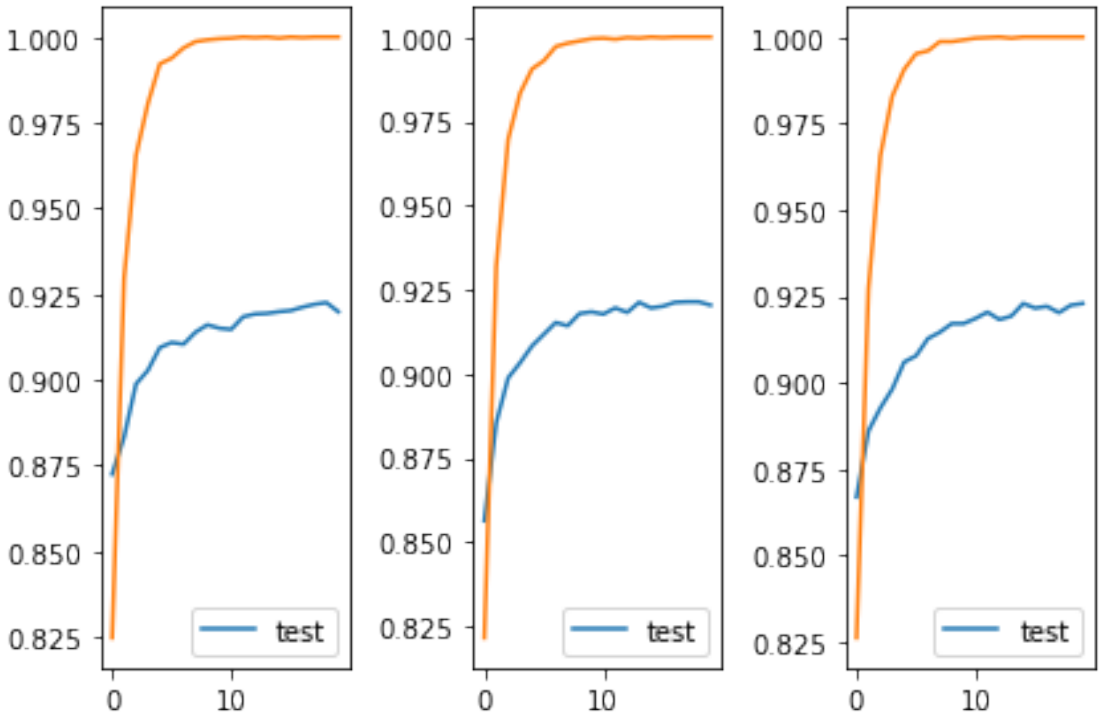
[18]:
```
plt.subplot(1, 3, 1)
plt.plot(history_full_list[0].history['val_acc'])
plt.legend(['test'], loc='lower right')

plt.subplot(1, 3, 2)
plt.plot(history_full_list[1].history['val_acc'])
plt.legend(['test'], loc='lower right')

plt.subplot(1, 3, 3)
plt.plot(history_full_list[2].history['val_acc'])
plt.legend(['test'], loc='lower right')
plt.tight_layout()

plt.show()

avg_train_acc = 0
avg_test_acc = 0
final_train_loss = []
final_test_loss = []

for his in history_full_list:
    final_train_loss.append(his.history['loss'][-1])
    final_test_loss.append(his.history['val_loss'][-1])
    avg_train_acc = avg_train_acc + his.history['acc'][-1]
    avg_test_acc = avg_test_acc + his.history['val_acc'][-1]

avg_train_acc = avg_train_acc/len(history_fc_list)
avg_test_acc = avg_test_acc/len(history_fc_list)


print("Average testing accuracy: {}".format(avg_test_acc))
```




```
Average testing accuracy: 0.9210438927014669
```

2.8 Train acc vs Test acc

[19]:
```
plt.subplot(1, 3, 1)
plt.plot(history_full_list[0].history['val_acc'])
plt.plot(history_full_list[0].history['acc'])
plt.legend(['test'], loc='lower right')

plt.subplot(1, 3, 2)
plt.plot(history_full_list[1].history['val_acc'])
plt.plot(history_full_list[1].history['acc'])
plt.legend(['test'], loc='lower right')

plt.subplot(1, 3, 3)
plt.plot(history_full_list[2].history['val_acc'])
plt.plot(history_full_list[2].history['acc'])
plt.legend(['test'], loc='lower right')
plt.tight_layout()

plt.show()
print("Average train accuracy: {}".format(avg_train_acc))
```



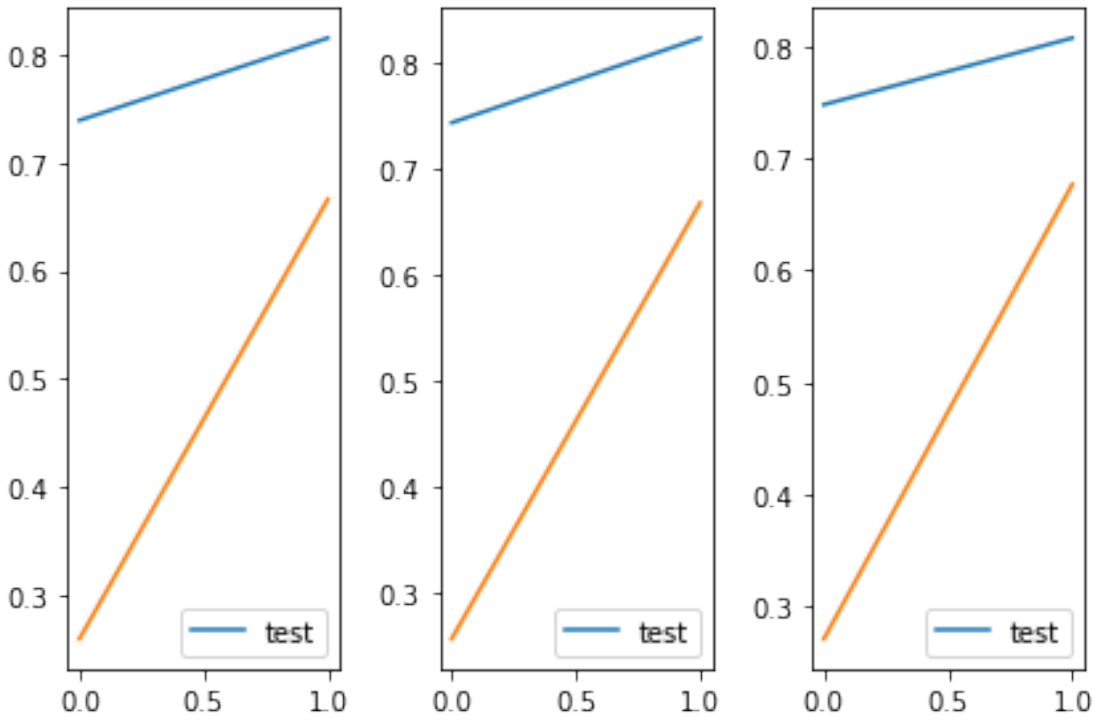

```
Average train accuracy: 1.0
```

2.9 FC layers test and train accuracy

- Fc layers training accuracy
- graphs look linear since we have only 2 epoch
- accuracy graph of full model training is plotted above

[20]:
```
plt.subplot(1, 3, 1)
plt.plot(history_fc_list[0].history['val_acc'])
plt.plot(history_fc_list[0].history['acc'])
plt.legend(['test'], loc='lower right')

plt.subplot(1, 3, 2)
plt.plot(history_fc_list[1].history['val_acc'])
plt.plot(history_fc_list[1].history['acc'])
plt.legend(['test'], loc='lower right')

plt.subplot(1, 3, 3)
plt.plot(history_fc_list[2].history['val_acc'])
plt.plot(history_fc_list[2].history['acc'])
plt.legend(['test'], loc='lower right')
plt.tight_layout()
```

```
plt.show()
```



3 Final model average accuracy on test set

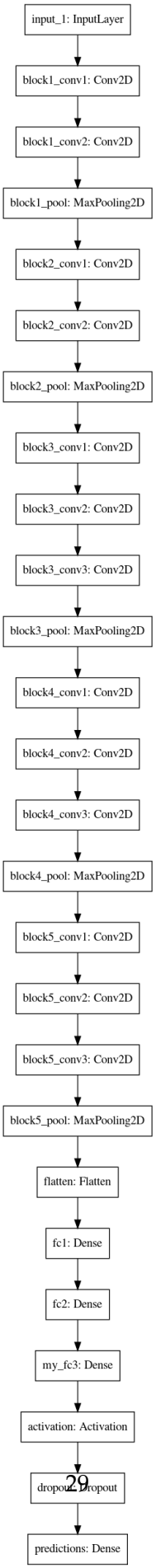
[21]:
```
print("Average Model Train accuracy is  : ",avg_test_acc*100,"%")
```

```
Average Model Train accuracy is  :  92.1043892701467 %
```

3.0.1 Model structure is saved to an image.

[22]:
```
from tensorflow.keras.utils import plot_model
plot_model(model_updated, to_file='model_CALTECH101_VGG16.png')
```

[22]:


input_1: InputLayer
block1_conv1: Conv2D
block1_conv2: Conv2D
block1_pool: MaxPooling2D
block2_conv1: Conv2D
block2_conv2: Conv2D
block2_pool: MaxPooling2D
block3_conv1: Conv2D
block3_conv2: Conv2D
block3_conv3: Conv2D
block3_pool: MaxPooling2D
block4_conv1: Conv2D
block4_conv2: Conv2D
block4_conv3: Conv2D
block4_pool: MaxPooling2D
block5_conv1: Conv2D
block5_conv2: Conv2D
block5_conv3: Conv2D
block5_pool: MaxPooling2D
flatten: Flatten
fc1: Dense
fc2: Dense
my_fc3: Dense
activation: Activation
dropout: Dropout
predictions: Dense