3猫狗分类-迁移学习

一、目的

对已经在ImageNet数据集上训练好的模型进行微调以实现猫狗分类任务。利用Slim工具包中预定义好的网络结构并进行微调的方法实现，具体是基于nets文件夹中预定义好的Inception V3、V4进行微调。

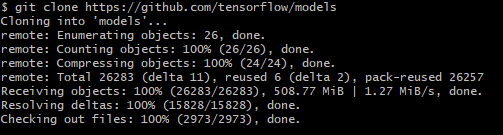
二、步骤

**!!把中文路径改成英文，不然会有各种坑**

1.下载TF-slim image models library

为了使用tF-Slim来进行图像分类，需要下载TF-Slim image models library，这个包不是核心的TF library的一部分。

在工作环境下打开git bush，输入git clone <https://github.com/tensorflow/models/>



下载完成之后，TF-Slim image models library就在$HOME/workspace/models/research/slim

1. 准备数据集

数据下载地址https://pan.baidu.com/s/1\_O2lGNmEoYqRFKhQpoo4hQ

数据放到models/research/slim/下，创建文件夹dogVScat,文件夹组织为

dogVScat

--dataSet

--train //包含RGB训练数据和验证数据，25000，20000：5000

--cat

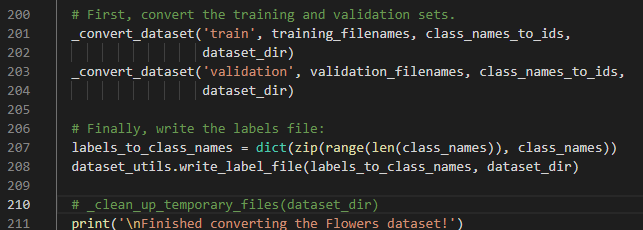
--dog

2.1 将RGB数据集转换为tfrecord格式以便训练模型

基于TensorFlow提供的API函数将RGB图转化为tfrecord格式以便模型训练。

①在download\_and\_convert\_flowers.py同目录下新建download\_and\_convert\_dogVScat.py

注释掉第201行清除原始数据的代码

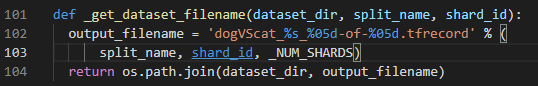


注释掉190行dataset\_utils.download\_and\_uncompress\_tarball(\_DATA\_URL, dataset\_dir)，不再下载数据。

43行，验证集的图片数量设置为5000，\_NUM\_VALIDATION = 5000

83行，文件夹名称。替换为自己的训练数据文件夹dataSet/train

102行，输出的文件名：



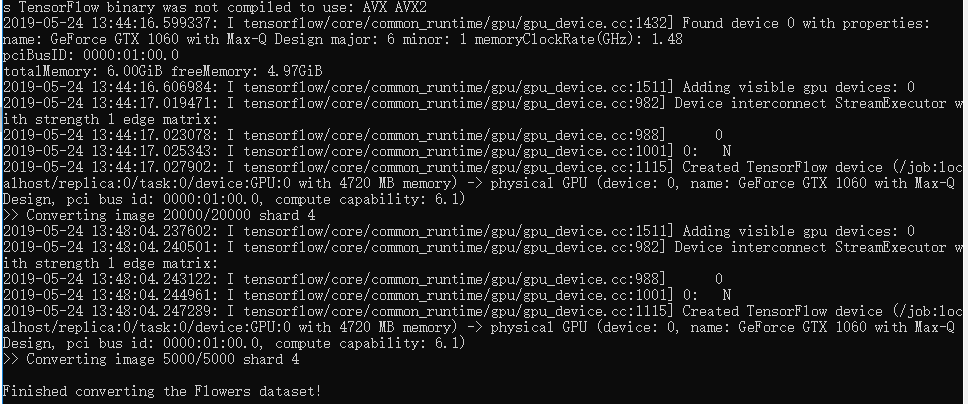
②在down\_and\_convert\_data.py中注册dogVScat的转换函数。

③在命令行中运行

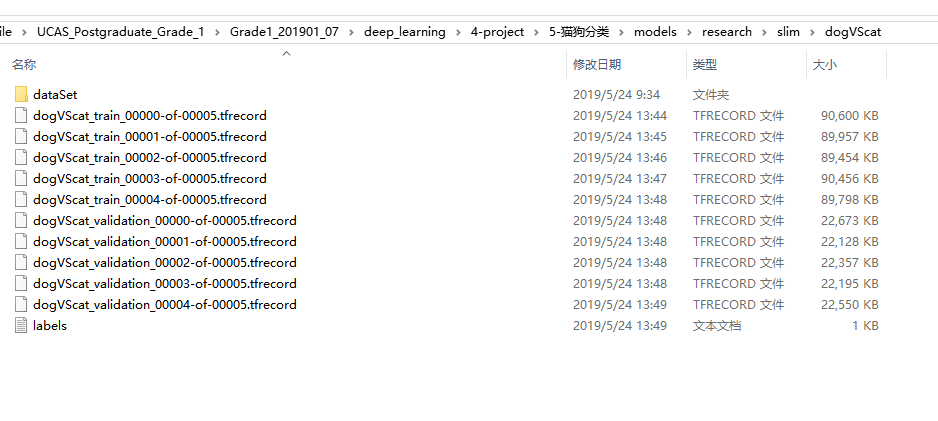
--dataset\_dir参数指向训练数据集的父文件夹训练数据文件夹dogVScat/dataSet/train

python download\_and\_convert\_data.py --dataset\_name=dogVScat --dataset\_dir=D:/file/UCAS\_Postgraduate\_Grade\_1/Grade1\_201901\_07/deep\_learning/4-project/5\_dogVScat\_class/models/research/slim/dogVScat

运行截图如下：



④在dogVScat文件夹下生成了tfrecord格式的验证数据集和训练数据集



3 迁移Inception-V3，进行训练

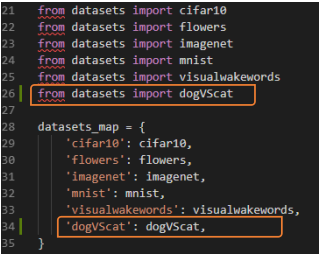
①在slim/datasets下新建一个dogsVScat.py，复制flowers.py的内容并改变其中的

\_FILE\_PATTERN=’dogsVScats\_%s\_\*.tfrecord’

SPLITS\_TO\_SIZES=(‘train’:20000,’validation’:5000)

\_NUM\_CLASSES=2

②在同目录的dataset\_factory.py中对dogsVScat数据集进行注册。添加两行代码



③在dogsVScat文件夹下：

a.新建一个data 目录，并将之前的11个转换好格式的训练数据复制进去。

b.新建一个空的train\_dir目录，用来保存训练过程中的日志和模型。

c.新建一个pretrained目录，在slim的GitHub页面找到Inception V3 模型的下载地址http://download.tensorflow.org/models/inception\_v3\_2016\_08\_28.tar.gz 下载并解压后，会得到一个inception\_v3.ckpt 文件，将该文件复制到pretrained 目录下。

d.在训练文件夹准备完成后，即可利用slim工具包中的训练模型入口代码train\_image\_clssifier.py进行训练：

python train\_image\_classifier.py

--train\_dir=dogVScat/train\_dir

--dataset\_name=dogVScat

--dataset\_split\_name=train

--dataset\_dir=dogVScat/data

--model\_name=inception\_v3

--checkpoint\_path=dogsVScat/pretrained/inception\_v3.ckpt

--checkpoint\_exclude\_scopes=InceptionV3/Logits,InceptionV3/AuxLogits

//在恢复预训练模型时，不恢复这两层。这两层是 Inception V3 模型的末端层，对应着 ImageNet 数据集的 1000 类，和相当前的数据集不符，随机初始化这两层的权重。

--trainable\_scopes=InceptionV3/Logits,InceptionV3/AuxLogits

--max\_number\_of\_steps=100000 //迭代次数

--batch\_size=32 //批次大小

--learning\_rate=0.001 //学习率

--learning\_rate\_decay\_type=fixed

--save\_interval\_secs=300 //每隔 300s，程序会把当前模型保存到train\_dir中

--save\_summaries\_secs=2 //每隔 2s，就会将日志写入到 train\_dir 中。可以用 TensorBoard 查看该日志。此处为了方便观察，设定的时间间隔较多，实际训练时,为了性能考虑，可以设定较长的时间间隔。

--log\_every\_n\_steps=10

--optimizer=rmsprop //优化器

--weight\_decay=0.00004 //权重衰减系数

在命令行中输入：

④在训练过程中，打开tensorboard观察指标变化情况

python train\_image\_classifier.py --train\_dir=dogVScat/train\_dir --dataset\_name=dogVScat --dataset\_split\_name=train --dataset\_dir=dogVScat/data --model\_name=inception\_v3 --checkpoint\_path=dogVScat/pretrained/inception\_v3.ckpt --checkpoint\_exclude\_scopes=InceptionV3/Logits,InceptionV3/AuxLogits --trainable\_scopes=InceptionV3/Logits,InceptionV3/AuxLogits --max\_number\_of\_steps=100000 --batch\_size=32 --learning\_rate=0.001 --learning\_rate\_decay\_type=fixed --save\_interval\_secs=300 --save\_summaries\_secs=2 --log\_every\_n\_steps=10 --optimizer=rmsprop --weight\_decay=0.00004

当生成了日志文件后，在命令行中使用tensorboard --logdir=日志文件目录启动一个服务，在浏览器中查看可视化结果。

4.模型验证

在模型训练完成后，即可利用slim工具包中的验证模型入口代码eval\_image\_clssifier.py进行验证：

python eval\_image\_classifier.py

--checkpoint\_path=dogVScat/train\_dir //保存模型权重的位置

--eval\_dir=dogVScat/eval\_dir //存放验证结果

--dataset\_name=dogVScat

--dataset\_split\_name=validation

--dataset\_dir=dogVScat/data

--model\_name=inception\_v3

1. 结果

1.Inceptionv3

1.1训练细节

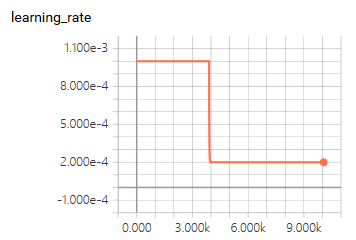
迭代次数10k step

学习率0-4k，0.001；4k-10k，0.0002

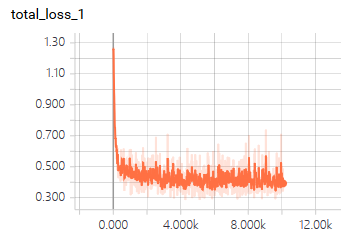
Batch size 64

1.2 实验结果

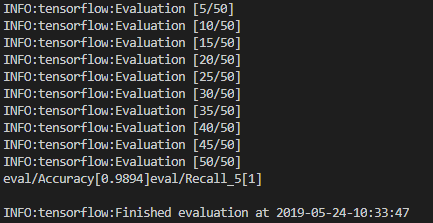
①学习率的设置



②损失函数的变化情况。刚开始下降速度很快，但是在后期损失逐渐不变。



③验证集上的精度0.9894，召回率1



2 Inceptionv4

2.1训练细节

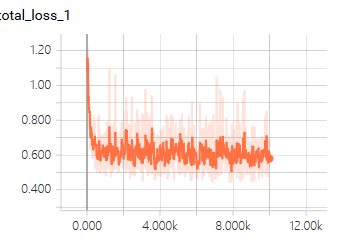
Batch size 32

Training step 10k

Learning rate 0-2k，0.001；2k-10k，0.0001

2.2 结果

损失函数的变化情况



验证集的精度0.9834和召回率1

